Energy Logserver Documentation

Release 8.1.0

October 26, 2026

CONTENTS

1 Get Started **3**

2 Core Platform **5**

3 Tools & Administration **7**

4 Resources **9**

4.1 Introduction . . . 9
4.2 Installation . . . 18
4.3 Upgrade . . . 39
4.4 Licensing . . . 56
4.5 User Interface . . . 62
4.6 Log Management . . . 91
4.7 Network Probe . . . 206
4.8 SIEM Module . . . 309
4.9 UEBA . . . 410
4.10 AI and Analytics . . . 414
4.11 Reporting . . . 453
4.12 CMDB . . . 459
4.13 E-doc . . . 462
4.14 System Administration . . . 463
4.15 System Configuration . . . 502
4.16 Integrations . . . 621
4.17 API and Integrations . . . 630
4.18 Changelog . . . 638

Enterprise-grade security information and event management platform that centralizes security events and operational data across your entire IT infrastructure.

CHAPTER

ONE

GET STARTED

Introduction Core components, architecture overview, and product portfolio.

Introduction Installation System requirements, step-by-step installation, and initial setup.

Installation Upgrade Upgrade procedures and migration guides between versions.

Upgrade

CHAPTER

TWO

CORE PLATFORM

User Interface Dashboards, Discovery, visualizations, and navigation.

User Interface Log Management Index management, SQL queries, PPL, and data lifecycle.

Log Management Network Probe Agent and agentless collection, IDS/IPS, NetFlow, packet capture, and NDR capabilities.

Network Probe SIEM Module Alerting, incidents, risk scoring, and automated response.

SIEM Module UEBA Analytics User and entity behavior analytics for insider threat detection.

UEBA AI & Analytics AI Assistant, machine learning, anomaly detection, and clustering.

AI and Analytics

CHAPTER

THREE

TOOLS & ADMINISTRATION

Reporting Scheduled reports, CSV and HTML data export, PDF, JPG and DOCX report export.

Reporting Index Management Index lifecycle management, rollover policies, and data retention.

Index Management Archive Long-term data archiving and retrieval.

Archive System Administration Backup, cluster management, audit, and troubleshooting.

System Administration System Configuration User management, RBAC, SSO, authentication, and system settings.

System Configuration Integrations Plugin catalog: install pipelines, dashboards, and alerts from the GUI.

Integrations API REST API and XLSX data import.

API and Integrations CMDB Asset inventory and infrastructure relationship mapping.

CMDB E-doc Knowledge base and wiki portal with SSO integration.

E-doc

CHAPTER

FOUR

RESOURCES

Licensing License types, activation, and management.

Licensing Release Notes Version history, new features, and changelog.

Changelog

4.1 Introduction

Energy Logserver is an enterprise-grade SIEM platform designed for centralized management of security events and operational data across your entire IT infrastructure. The solution enables immediate review, analysis, and reporting of system logs regardless of data volume.

Built to address the growing demand for efficient processing and storage of large-scale data from modern IT environments, Energy Logserver delivers powerful data storage and search capabilities. The platform includes comprehensive security features for information protection, user verification, data correlation and visualization, alerting, and reporting - providing organizations with the tools needed to effectively monitor and secure their IT systems.

4.1.1 Core Capabilities

Real-Time Security Monitoring - Threat detection with behavioral analytics and real-time event analysis **AI-Powered Analytics** - Machine learning for threat detection, anomaly detection, and clustering **SIEM** - Alert rules, risk scoring, and case management. SOAR integration available as a separate product. **Visualizations** - Interactive dashboards with drill-down capabilities and real-time updates **Network Monitoring** - IDS/IPS, full packet capture, NetFlow analysis, and NDR capabilities **Data Ingestion** - Scalable pipeline supporting petabytes of daily logs with advanced correlation using MITRE ATT&CK framework **Compliance** - Automated reporting, audit trails, retention policies for GDPR, PCI DSS, SOX, and other regulations

4.1.2 Key Benefits

- **Unified Security View**: Centralize security events from all IT domains
- **AI-Enhanced Detection**: Leverage machine learning for anomaly detection
- **Flexible Licensing**: Cost-effective scaling based on data volume
- **Streamlined Deployment**: Guided installation process with interactive setup
- **Enterprise Integration**: Native support for AD, LDAP, SSO, and a wide range of data sources

ELS Copyright

Energy Logserver® since 2015

Copyright © 2026 EMCA Software sp. z o.o.

All title and ownership rights to Energy Logserver, including its entire codebase and documentation, are owned by EMCA Software sp. z o.o. All software components and modules are maintained by EMCA Software and protected under vendor licensing agreements.

License Information: A valid license is required for production use. Evaluation licenses are available for testing purposes.

Support: For technical support, documentation updates, or license inquiries, visit our Knowledge Base or contact support@energylogserver.com.

Core Components

ELS Data Node

Purpose: Serves as the core data storage component, providing scalable NoSQL database capabilities for security and operational data.

Key Features:

- **Distributed Architecture**: Automatic sharding and replication across cluster nodes
- **Real-time Search**: Sub-second query response times across millions of events
- **Hot-Warm-Cold Storage**: Automatic data lifecycle management for cost optimization
- **Security Features**: Built-in authentication, authorization, and audit logging

Node Types:

- **Master-eligible nodes**: Cluster state management and coordination
- **Data nodes**: search execution, document storage and indexing operations

ELS Console

Purpose: Provides the visualization and management interface for security analysts and administrators.

Key Capabilities:

- **Security Dashboards**: Pre-built dashboards for MITRE ATT&CK, compliance, and threat hunting
- **Live Threat Map**: Real-time global threat visualization
- **Advanced Analytics**: Support for Vega-Lite custom visualizations
- **RBAC Integration**: Granular access control with Active Directory integration
- **Plugin Ecosystem**: Modular architecture supporting custom security apps

Enhanced Features:

- **Empowered AI Interface**: Direct access to AI model store and anomaly detection
- **Data Export Wizard**: Streamlined report generation from dedicated tab

ELS Network Node

Purpose: Ingests, processes, and enriches data from multiple sources before storing in ELS Data Node.

Processing Capabilities:

- **Multi-Protocol Ingestion**: Syslog, Beats, SNMP, APIs, databases, cloud services
- **Real-time Parsing**: Custom parsing rules for any log format or data structure
- **Threat Intelligence**: Automatic IOC enrichment from multiple threat feeds
- **GeoIP Enhancement**: Location-based analysis for network traffic
- **Custom Pipeline Creation**: GUI-based pipeline builder

Empowered AI

Purpose: Provides machine learning capabilities for threat detection, user behavior analysis, and predictive analytics.

AI Capabilities:

- **Online AI Store**: Download pre-trained models from https://energylogserver.com/ai-store/
- **Anomaly Detection**: Identifies unusual patterns in user behavior, network traffic, and system events
- **Automated Model Training**: Learns from your environment's normal patterns
- **UEBA Integration**: User and Entity Behavior Analytics for insider threat detection

Available Use Cases:

- **Netflow Traffic Analysis**: Detects network anomalies and data exfiltration
- **User Behavior Monitoring**: Identifies compromised accounts and privilege escalation
- **AI Assistant**: Prompt-based log analysis integrated into the Discover view. See *AI Assistant in Discover* for usage details.

Automation

Purpose: Integrates the n8n workflow automation engine into Energy Logserver, enabling automated responses, data enrichment, and integration with external systems.

Key Features:

- **Workflow Engine**: Visual workflow builder based on n8n with trigger, action, and logic nodes
- **SSO Integration**: Admin users are automatically authenticated via the Console proxy
- **Extensible**: Supports hundreds of n8n nodes for connecting to external services (email, ticketing, APIs, databases)

Automation is an opt-in module. It is disabled by default and requires a separately installed n8n instance.

Product Portfolio

Energy Logserver is available in three product plans. Each plan can be deployed independently or combined depending on the organization's requirements.

Product Plan

Product Plan	Core Purpose	Primary Use Cases	Target Organizations
Log Management Plan (LMP)	Foundation platform for comprehensive log management	Operational monitoring, compliance	All organizations requiring centralized logging
SIEM Plan	Advanced security analytics and threat detection	SOC operations, threat hunting, incident response	Security-focused organizations, regulated industries
Network Probe	Network traffic analysis and passive monitoring	Network security, advanced threat detection, forensics	Organizations with advanced network security requirements

Log Management Plan

Core Modules:

- **Discover**: Advanced search and filtering with ad-hoc query capabilities
- **Dashboards**: Interactive operational and security dashboards
- **Visualizations**: Comprehensive charting, tables, heatmaps, and custom visualizations
- **Agents**: Centralized management of log collection agents across all platforms
- **Archive**: Long-term data retention and compliance archiving
- **CMDB**: Configuration Management Database integration with event correlation
- **Integrations**: Native connectors for data sources and external systems
- **Network Probe**: Integrated data collection and processing (first instance included)
- **Reports**: Automated and ad-hoc reporting with scheduling capabilities
- **CRUD**: Document management interface for creating, viewing, editing, and deleting records in custom indices

Management and Administration Modules:

- **Config**: System-wide and user-specific configuration management
- **Sources**: Registration, configuration, and management of data sources
- **Dev Tools**: Testing and development environment for queries and transformations
- **Index Management**: Data lifecycle policies, retention management, and optimization

Key Business Benefits:

- **Unlimited Data Retention**: No arbitrary limits on log storage duration
- **Unlimited User Access**: Role-based access control without per-user licensing
- **Unlimited Data Sources**: Connect any number of systems and applications (applies to standard licensing; MSSP licensing may differ)
- **Included Network Analysis**: First Network Probe instance at no additional cost

SIEM Plan Enhancement

Security-Focused Modules:

- **Alerts**: Advanced correlation rules and real-time threat detection policies
- **Vulnerability Dashboard**: Endpoint Security vulnerability data visualization

- **FIM (File Integrity Monitoring)**: Critical file and system resource monitoring
- **Correlation Engine**: Advanced multi-source event correlation and attack chain analysis

Empowered AI Capabilities:

- **AI Cases**: Machine learning-driven incident analysis and recommendation engine
- **AI Assistant**: Prompt-based log analysis using configured LLM providers
- **Scheduled Analytics**: Automated execution of configured Empowered AI use cases on a defined schedule
- **Behavioral Baselines**: Automated establishment of normal behavior patterns

Enterprise Security Features:

- **MITRE ATT&CK Integration**: Native framework mapping and technique coverage analysis
- **Threat Intelligence Platform**: Integration with commercial and open-source threat feeds
- **SOAR Integration**: Integration with external SOAR platforms, including Energy SOAR
- **Advanced Compliance**: Automated compliance reporting for multiple frameworks
- **Globe Visualization**: 3D geolocation visualization for network traffic and security events (not available in Log Management Plan)

Network Probe

Network Analysis Capabilities:

- **Passive Traffic Monitoring**: Layer 2-7 protocol analysis without network impact
- **Flow Analysis**: NetFlow, sFlow, IPFIX support for network behavior analysis
- **Protocol Recognition**: Automatic identification of applications and services
- **Metadata Extraction**: Deep packet inspection for security-relevant information
- **Network Anomaly Detection**: Identification of unusual communication patterns

Performance Specifications:

- **Traffic Capacity**: Up to 10 Gbps sustained network traffic analysis
- **Event Processing**: 20,000+ Flows/Events Per Second (FPS/EPS)
- **Deployment Flexibility**: Physical appliance, virtual machine, or cloud deployment
- **Integration**: Seamless data forwarding to ELS Data Node and correlation engine

Deployment Scenarios

- *Small Office Deployment*
- *Medium Enterprise Deployment*
- *Large Enterprise Deployment*
- *MSSP/Service Provider Deployment*
- *Supported Environments*
- *Capacity Planning Guidelines*

- *Raw volume versus on-disk size*
- *Storage formula*
- *Worked example*
- *Memory and shards*
- *EPS - Events per second*
- *Quick reference*

The EPS figures quoted with the scenarios below refer to the hardware listed with each of them. See *EPS - Events per second* for what the rate depends on and how it scales.

Small Office Deployment

Single-node deployment for basic log management and security monitoring

Architecture:

- **Single ELS Data Node**: All-in-one deployment hosting all components
- **Licensing**: 1x Log Management Plan license
- **Capacity**: Up to 10 GB/day, 90-day retention
- **Use Cases**: Compliance logging, basic security monitoring, operational troubleshooting

Technical Specifications:

- **Hardware**: 8 CPU cores, 32 GB RAM, 1 TB SSD storage (storage requirements vary depending on log volume and retention period)
- **Components**: ELS Data Node, ELS Console, ELS Network Node (co-hosted)
- **Performance**: 2,000-5,000 EPS with the hardware above
- **Investment**: Lowest entry point, suitable for budget-conscious organizations

Medium Enterprise Deployment

Multi-node cluster with role separation and enhanced performance

Architecture:

- **3-Node Cluster**: Separated roles for optimal performance and reliability
  - 3x Data Nodes (ELS Data Node for storage and processing)
  - 1x Dedicated ELS Network Node
- **Licensing**: 3x Log Management Plan licenses + optional SIEM Plan
- **Capacity**: 10-100 GB/day, 1-year retention
- **Use Cases**: SOC operations, compliance automation, advanced threat detection

Technical Specifications:

- **Hardware per Data Node**: 16 CPU cores, 64 GB RAM, 5 TB NVMe storage (storage requirements vary depending on log volume and retention period)
- **Network Node**: 8 CPU cores, 32 GB RAM, 500 GB SSD (storage requirements vary depending on log volume and retention period)

- **Performance**: 15,000+ EPS with the hardware above
- **High Availability**: Node redundancy, automated failover

Large Enterprise Deployment

Distributed cluster with geographic redundancy and full SIEM capabilities

Architecture:

- **Multi-Site Cluster**: 5+ nodes with geographic distribution
  - 5x Data Nodes (horizontal scaling based on data volume)
  - 2+ Dedicated ELS Network Nodes (load balancing and redundancy)
  - Multiple Network Probes (regional deployment)
- **Licensing**: 5+ Log Management Plan + SIEM Plan + multiple Network Probes
- **Capacity**: 100+ GB/day, multi-year retention
- **Use Cases**: Enterprise SOC, advanced threat hunting, regulatory compliance

Technical Specifications:

- **Hardware per Data Node**: 32+ CPU cores, 128+ GB RAM, 20+ TB NVMe storage
- **Geographic Distribution**: Multi-region deployment for disaster recovery
- **Performance**: 50,000+ EPS with the hardware above
- **Enterprise Features**: Full SIEM capabilities, AI analytics, compliance automation

MSSP/Service Provider Deployment

Multi-tenant architecture for managed security service providers

Architecture:

- **Tenant Isolation**: Dedicated clusters or logical separation per customer
- **Centralized Management**: Unified console for multi-tenant operations
- **Scalable Infrastructure**: Scalable infrastructure with flexible deployment options
- **Service Automation**: Automated provisioning and customer onboarding

Licensing Considerations:

- **MSSP Pricing Model**: Specialized pricing for service providers
- **Multi-Tenant Support**: Built-in capabilities for customer separation
- **Flexible Scaling**: Pay-as-you-grow model for service provider economics
- **Partner Support**: Dedicated channel partner support and training

Supported Environments

Deployment Type	Use Case	Node Configuration	Storage Requirements
All-in-One	Small office, testing	Single node (all components)	500GB+ SSD
Distributed	Medium enterprise	3+ nodes, role separation	1TB+ per data node
Distributed: Large	Large enterprise, 24/7 SOC	5+ nodes, full redundancy	10TB+ distributed

Capacity Planning Guidelines

Two numbers drive every deployment: how much data arrives per day, and how long it stays searchable. Size the cluster to the on-disk footprint those two produce, not to the raw figure a customer quotes.

Raw volume versus on-disk size

The "GB per day" a customer reports is usually the raw or licensed figure. What lands on the data nodes differs, because the pipeline changes the size in both directions:

- Parsing and enrichment add fields (geoIP, tags, normalized names), which grows the record.
- Compression shrinks it. Live indices are compressed by default, and the *Archive* packs data further with zstd.
- Every replica stores a full second copy, so each replica doubles the disk the primary data uses.

These factors depend on the data and the pipeline, so measure the ratio instead of assuming one. Ingest one real day into a test index, read its on-disk size, and divide by the raw input. Apply that ratio to the daily volume and the rest of the calculation follows.

Storage formula

Once you know the on-disk size of one day:

```
required storage = on_disk_per_day × retention_days × (1 + replicas) × 1.2
```

The 1.2 factor reserves headroom for segment merges, indexing overhead, and growth. Keep total usage below the disk watermarks (low 95%, high 97%, flood 99%; see *Disk-based shard allocation*) and plan so normal operation stays under the high watermark rather than at it.

Worked example

A customer ingests 300 GB/day raw, needs 90 days searchable, and keeps one replica. A one-day pilot shows the data lands at roughly the same size on disk after parsing and compression:

- **Flat, single tier:** 300 GB × 90 × 2 × 1.2 gives about 65 TB. Spread over three data nodes that is more than 20 TB of fast storage each. All-SSD at that size is what makes the hardware expensive.
- **Tiered:** keep the recent, most-queried window on SSD and move the rest of the 90 days to cheaper high-capacity disk through shard allocation (see *Index Management*). The 90-day searchable requirement still holds, because the relocated data stays searchable; only the newest data needs sub-second speed. Retention is unchanged, the storage bill drops.

Retention beyond the online window belongs in the *Archive*, not on replicated fast disk. A one-year archive compressed with zstd is a fraction of the size of the same data held live.

Memory and shards

Storage is one axis; heap and shard count are the other. Right-size the JVM heap to the workload, cap it at 31 GB, reserve at least twice the heap in system RAM for the page cache, and keep the shard count in proportion to the heap. See *JVM Configuration* for the per-node tiers and the shards-per-heap ratio.

EPS - Events per second

EPS is not a licensing metric in Energy Logserver. The license covers the plans and the limits described in *Licensing*, among them the number of ELS Data Nodes, the document count and the index count. It is still the figure most Log Management and SIEM projects are specified in, so it is worth stating what it means here.

EPS capacity follows the architecture of the cluster. Each ELS Data Node contributes its own processing capacity, which depends on the CPU, memory and storage performance allocated to it, and on the complexity of the incoming data.

For that reason no single value describes every ELS Data Node. A node sized to the recommended resources handles thousands of events per second, and the figures quoted with the scenarios above refer to the hardware listed with them rather than to ELS Data Nodes in general.

The architecture imposes no fixed upper limit. A deployment starts at a single ELS Data Node and scales horizontally to tens or hundreds of nodes, so the same platform can be built for 1,000 EPS, 50,000 EPS, 500,000 EPS or more. Size it with the *storage formula* and the heap and shard tiers in *JVM Configuration*, then confirm the rate a running cluster reaches against the `Indexing Rate` metric in *Monitoring and Performance*.

Quick reference

Use this as a starting point, then refine it with the measured ratio and the formula above. The storage column assumes tiered storage at the higher volumes, not a flat all-SSD layout.

Daily Data Volume	Recommended Architecture	ELS Data Nodes	Storage Requirements
< 10 GB/day	Single node deployment	1 node	1 TB total storage
10-100 GB/day	Small cluster	3 nodes	5 TB per node
100-500 GB/day	Medium cluster	5 nodes	10 TB per node
500GB-2TB/day	Large cluster	8+ nodes	20 TB per node
> 2TB/day	Enterprise cluster	15+ nodes	Custom sizing

System Requirements

The requirements baseline lives with the installation instructions, in *System Requirements*. That page is the single source for:

- supported operating systems and web browsers,
- software dependencies, OpenJDK and NodeJS, both bundled with the installation package,
- network communication requirements, that is the port matrix between components,
- minimum and production hardware sizing per component.

For the deployment architectures behind those numbers, see *Deployment Scenarios*.

Getting Help & Support

AI Assistant

Energy Logserver includes an AI-powered assistant that works in the context of the **Discover** module. It uses configured LLM providers to help analyze log entries, interpret security events, and analyze log content using configured prompts. For details on configuration and usage, see the *AI Assistant* chapter.

Support Resources

- **Knowledge Base**: kb.energylogserver.com
- **Community**: community.energylogserver.com
- **Technical Support**: support@energylogserver.com
- **API Documentation**: *REST API Reference*

4.2 Installation

This chapter covers the installation of Energy Logserver SIEM, including system requirements, installation methods, and backup strategies. After installation, refer to *System Configuration* for initial setup and tuning.

4.2.1 System Requirements

Supported Operating Systems

Recommended:

- Oracle Linux 8.x, 9.x with Unbreakable Enterprise Kernel (UEK)

Supported:

- Red Hat Enterprise Linux 8.x, 9.x
- AlmaLinux/Rocky Linux 8.x, 9.x
- CentOS Stream 9.x

> **Note**
>
> CentOS Stream 8.x is not supported. Only CentOS Stream 9.x is a supported platform.

Supported Web Browsers

- Google Chrome 90+
- Mozilla Firefox 88+
- Opera 76+
- Microsoft Edge 90+
- Safari 14+

Software Dependencies

Component	Required Version	Notes
OpenJDK	21	64-bit required; bundled with installation package
NodeJS	22.x	ELS Console dependency; bundled with installation package

Network Communication Requirements

From	To	Port	Protocol	Description
SIEM Agent	ELS Data Node	1514	TCP	SIEM Agent connection service, default protocol
	ELS Data Node	1514	UDP	SIEM Agent connection service, alternative protocol (disabled by default)
	ELS Data Node	1515	TCP	SIEM Agent enrollment service
ELS Data Node	ELS Data Node	1516	TCP	SIEM cluster daemon
Syslog source	ELS Network Node	5514	UDP	Syslog collector, default protocol (installer forwards 514 to 5514)
	ELS Network Node	5514	TCP	Syslog collector, alternative protocol (installer forwards 514 to 5514)
Beats agent	ELS Network Node	5044	TCP	Beats collector (filebeat, winlogbeat, metricbeat, packetbeat, auditbeat, heartbeat)
ELS Console	ELS Data Node	5500(	TCP	SIEM server RESTful API
ELS components, integrations	ELS Data Node	9200	TCP	OpenSearch REST API (data, license verification, integrations)
ELS Data Node	Other ELS Data Node	9300	TCP	Data Node cluster transport (cross-cluster search uses the same port)
User browser	ELS Console	5601	TCP	Default GUI
	ELS Console	5602	TCP	Admin console
	ELS Console	5603	TCP	Wiki GUI
ELS Console	Every Network Node	9000	TCP	Manage files, services and pipelines
ELS Network Node	Kafka broker	9092	TCP	Message queue (data transport between probes and to Data Node)
ELS Network Node	Kafka broker	9093	TCP	Kafka broker SSL listener (when TLS is enabled)
Kafka broker	Zookeeper	2181	TCP	Kafka and Zookeeper coordination
Zookeeper node	Other Zookeeper node	2888	TCP	Zookeeper peer communication (multi-node)
Zookeeper node	Other Zookeeper node	3888	TCP	Zookeeper leader election (multi-node)
NetFlow exporter	ELS Network Node	2055	UDP	NetFlow collector (pmacct)
IPFIX exporter	ELS Network Node	4739	UDP, TCP	IPFIX collector
sFlow exporter	ELS Network Node	6343	UDP	sFlow collector (pmacct)
Windows event source	Event Collector	5986	TCP	WS-Man subscription manager (agentless Windows collection)
Event Collector	ELS Network Node	6514	TCP	Event Collector forwarding to the `syslog_wec` pipeline (TLS)
localhost	ELS Network Node	9600	TCP	Network Probe monitoring API (bound to localhost, not exposed externally)
Masteragent	ELS Network Node	8080	TCP	Masteragent → Network Probe communication
ELS Network Node	Masteragent server	8081	TCP	Network Probe → masteragent server callback

All ports above are defaults; each component allows the port to be changed in its configuration file. For the GUI port specifically, including the installer's 443→5601 redirect, see *GUI Port and HTTPS Redirect*.

> **Note**
>
> The table lists the ports the installer opens. Individual integrations bind additional listener ports of their own, and the installer does not open those. See *Connecting Data Sources* for how to find and open the port an integration needs.

Hardware Requirements

Minimum Requirements (Development/Testing)

Component	CPU	RAM	Storage	Network
All-in-One	8 cores	32 GB	500 GB SSD	1 Gbps
ELS Data Node	6 cores	16 GB	200 GB SSD	1 Gbps
ELS Console	4 cores	8 GB	50 GB	1 Gbps
ELS Network Node	6 cores	8 GB	50 GB	1 Gbps

Production Requirements (Recommended)

Component	CPU	RAM	Storage	Network
ELS Data Node	16+ cores	64 GB	1 TB NVMe SSD	10 Gbps
ELS Console	8 cores	32 GB	200 GB SSD	1 Gbps
ELS Network Node	8 cores	32 GB	200 GB SSD	10 Gbps

For deployment architecture options and sizing guidance, see *Deployment Scenarios*.

4.2.2 Installation

The Energy Logserver installer is delivered as:

- RPM packages
- Installation script: `install.sh`

- *Prerequisites and Environment Preparation*
  - *System Requirements Check*
  - *System Configuration*
  - *Download Installation Packages*
- *Installation Methods*
  - *Interactive Installation using "install.sh"*
  - *Non-interactive Installation Mode using "install.sh"*
- *Multi-node Cluster Installation*
  - *Installation Order*

    - *Prerequisites*
    - *Procedure*
- *Installation Verification*
    - *Check Cluster/Indices Status and ELS Data Node Information*
- *Post Installation Configuration*
    - *Configure ELS Data Node Cluster Settings*
    - *Start Services*
    - *Configure Data Sources and Agents*
    - *SIEM Agents Configuration*
    - *Blacklist and Threat Intelligence Setup*
- *Post-Installation Steps*
    - *Change Default Credentials*

Prerequisites and Environment Preparation

Before beginning installation, prepare your system environment.

System Requirements Check

Verify your system meets minimum requirements:

```
# Check CPU cores
nproc

# Check available memory
free -h

# Check disk space
df -h
```

System Configuration

The installer sets these parameters automatically via RPM post-install scripts, but applying them before running `install.sh` avoids startup failures if the environment is restricted.

Kernel parameters

Check current values:

```
sysctl vm.max_map_count vm.swappiness net.ipv4.tcp_retries2
```

Required values:

Parameter	Required value	Purpose
vm.max_map_count	262144	OpenSearch memory-mapped files limit
vm.swappiness	1	Minimise OS swapping of JVM heap
net.ipv4.tcp_retries2	5	Faster detection of failed cluster connections

If any value is wrong, apply all three and make them persistent:

```
cat > /etc/sysctl.d/90-logserver.conf << 'EOF'
vm.max_map_count=262144
vm.swappiness=1
net.ipv4.tcp_retries2=5
EOF
sysctl --system
```

File descriptor limits

```
ulimit -n   # Required: 262144
```

If the value is lower, create the limits file:

```
cat > /etc/security/limits.d/30-logserver.conf << 'EOF'
logserver -   memlock unlimited
logserver -   nproc   unlimited
logserver -   nofile  262144
logserver -   fsize   unlimited
EOF
```

Log out and back in for the new limits to take effect.

Download Installation Packages

Download Energy Logserver packages from the customer portal or the provided distribution link. Place all RPM packages and the `install.sh` script in a single directory on the target host.

Installation Methods

Interactive Installation using “install.sh”

Energy Logserver comes with a simple installation script called `install.sh`. It is designed to facilitate the installation and deployment process. After executing the script, it will detect the supported distribution and ask about the components to install. The script is located in the `install` directory.

Note

Before running `install.sh`, make sure you are located in the `install` directory. Failing to do so will cause the script to fail.

```
cd install
./install.sh
```

Installation process:

1. Unpack the archive containing the installer:

```
tar xjf energy-logserver-8.1.0.x86_64.tar.bz2
```

2. Unpack the archive containing the SIEM installer (only in SIEM plan):

```
tar xjf energy-logserver-siem-plan-8.1.0.x86_64.tar.bz2
```

3. Copy license to installation directory:

```
cp es_*.* install/
```

4. Navigate to the installation directory and run installation script:

```
cd install
./install.sh -i
```

During interactive installation you will be asked about:

- Install & configure Network Probe with custom Energy Logserver Configuration - including Beats, Syslog, Blacklist, Netflow, Winrm, Logtrail, OP5, etc.
- Install the ELS Console, as well as other console dependencies
- Install the ELS Data Node, as well as other data-node dependencies
- Load the Energy Logserver custom dashboards, alerts and configurations

Non-interactive Installation Mode using "install.sh"

The `-n|--noninteractive` flag answers **"Yes" to every component prompt** during installation, installing every available component on the target host. This mode is intended for automated all-in-one deployments:

```
./install.sh -i -n
```

> **Note**
>
> Component selection flags (`-d`, `-c`, `-p`, `-s`) do **not** restrict the installation when combined with `-n`. Every component prompt is still answered "Yes" regardless of which flags are passed. To install only selected components, run the installer in interactive mode and answer the remaining prompts manually.

Interactive mode with preselected components:

Without `-n`, component flags preselect the answer for a given component prompt; the installer still asks about the remaining components, where you can decline the ones you do not want:

```
./install.sh -i -d    # ELS Data Node preselected
./install.sh -i -p    # Network Probe preselected (standalone sensor␣
↪deployment)
```

Multi-node Cluster Installation

This section covers the order in which you install the nodes of a new cluster and adding a new node to an existing cluster. For a single-node deployment, skip to *Installation Verification*.

Installation Order

Start a new cluster from the ELS Data Node. On the first node, answer `y` when the installer asks about the ELS Data Node and `n` when it asks whether the node is a part of an existing cluster. The installer forms the cluster, creates the built-in users and writes `/etc/logserver-store/logserver.keystore`. The first node may also run the ELS Console. Add every other node, including client nodes, with the procedure below.

> **Warning**
>
> Do not start a new cluster from a client node. When you decline the ELS Data Node, the installer still installs the data-node package with `node.master: false` and `node.data: false`. Such a node cannot form a cluster on its own, so the installer waits at `Waiting for a Logserver initialization...` with no time limit and does not create the keystore.
>
> To recover, stop the installer with `Ctrl+C`, install the ELS Data Node on another host as the first node, and then join the client node with the procedure below, starting from step 3.

Prerequisites

- The first ELS Data Node of the cluster must already be installed and running.
- The new node must have the Energy Logserver packages unpacked and the `install.sh` script available.
- `/etc/logserver-store/logserver.keystore` must be present on the new node before running `./install.sh configure`. This file is created during the first node's installation. Copy it from the existing node if it is not present.

Procedure

1. Run the installer on the new node:

```
cd install
./install.sh -i
```

2. When the installer asks:

```
====> Is this node a part of an existing cluster? [y/n]
```

   Answer `Y`. The installer exits and prompts you to update the node configuration before continuing.

3. Edit the main configuration file:

```
vi /etc/logserver/logserver.yml
```

   Set the following parameters:

Parameter	Value
`cluster.name`	Same as the existing cluster
`node.name`	Unique name for this node
`node.master`	`true` or `false`
`node.data`	`true` or `false`
`discovery.seed_hosts`	IP addresses and ports of existing cluster nodes, e.g. `["10.0.0.4:9300", "10.0.0.5:9300"]`

> **Note**
>
> When binding `network.host` to a specific IP, include `127.0.0.1` or `_local_` in the list. Tools such as `install.sh` and `logserver-password-util.sh` connect to `http://127.0.0.1:9200` and fail with *Connection refused* if the loopback address is missing.

If the new node has the `data` role, clear the data directory before starting:

```
rm -rf /var/lib/logserver/*
```

4. Finalize the installation:

```
cd install
./install.sh configure
```

This command initializes the node and propagates credentials from the keystore. It requires `/etc/logserver-store/logserver.keystore` to be present.

For node role definitions, naming conventions, and a full configuration example, see *Cluster Configuration*.

Installation Verification

> **Note**
>
> In the commands below, `$CREDENTIAL` refers to the `logserver` user credentials in `user:password` format. The default credentials are set during installation. To change them, use the `logserver-password-util.sh` utility described in *Post-Installation Steps*.

Check Cluster/Indices Status and ELS Data Node Information

Verify installation by checking the status of ELS Data Node:

```
curl -s -u $CREDENTIAL localhost:9200/_cluster/health?pretty
```

Expected response showing cluster health:


```
{
  "cluster_name" : "logserver",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 1,
  "number_of_data_nodes" : 1,
  "active_primary_shards" : 25,
  "active_shards" : 25,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0,
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 100.0
}
```

Check ELS Data Node details:

```
curl -s -u $CREDENTIAL localhost:9200
```

If everything went correctly, you should see 100% allocated shards in cluster health.

Post Installation Configuration

Configure ELS Data Node Cluster Settings

Edit the main configuration file:

```
vi /etc/logserver/logserver.yml
```

Add all IPs of ELS Data Nodes in the discovery directive:

```
discovery.seed_hosts: ["172.10.0.1:9300", "172.10.0.2:9300"]
```

Note

When binding `network.host` to a specific IP, include `127.0.0.1` or `_local_` in the list:

```
network.host: ["10.x.x.x", "127.0.0.1"]
```

Tools such as `install.sh` and `logserver-password-util.sh` connect to `http://127.0.0.1:9200`. Without the loopback address they fail with *Connection refused*.

Start Services

Start Energy Logserver services. The services are grouped by function.

Core services (all deployments):

```
systemctl start logserver               # ELS Data Node
systemctl start logserver-gui           # ELS Console
systemctl start logserver-probe         # Network Probe
systemctl start license-service         # License verification
systemctl start cerebro                 # Cluster management UI
```

Security services (SIEM Plan):

```
systemctl start alert                   # Alert engine
systemctl start skimmer                 # Data processing
systemctl start intelligence            # Threat intelligence
systemctl start intelligence-scheduler  # Scheduled intelligence updates
systemctl start wazuh-manager           # SIEM agent management
```

Optional services:

```
systemctl start e-doc                   # Documentation service
systemctl start automation              # Task automation
```

> **Note**
>
> Not all services are required on every node. The `install.sh` script enables only the services relevant to the selected installation profile. In multi-node deployments, each node runs only its designated services.

Configure Data Sources and Agents

With the platform installed, the next step is connecting the sources you want to collect from. Both agent-based and agentless methods are available, and which one fits depends on the source. See *Connecting Data Sources* for the comparison and the procedure for each method.

Example agent configuration files are available in the Agents module of the Console.

SIEM Agents Configuration

Configure SIEM agents for security event collection:

Agent Connection Ports

- **Port 1514 (TCP)** - Primary agent connection service
- **Port 1514 (UDP)** - Optional agent connection service (disabled by default)
- **Port 1515 (TCP)** - Agent enrollment service
- **Port 1516 (TCP)** - SIEM cluster daemon communication
- **Port 55000 (TCP)** - SIEM server RESTful API

Agent Installation

Agent packages for all three platforms are in `install/agents/wazuh/client/` on the installation media, and are also served by the Console for download. Passing the manager address at install time enrolls the agent automatically, so no separate enrollment step is needed.

Red Hat family:

```
WAZUH_MANAGER="<manager address>" yum install ./wazuh-agent-*.rpm
```

Debian family:

```
WAZUH_MANAGER="<manager address>" apt-get install ./wazuh-agent_*.deb
```

Windows, from an elevated prompt:

```
wazuh-agent-4.14.6-1.msi /q WAZUH_MANAGER="<manager address>" WAZUH_
↪REGISTRATION_SERVER="<manager address>"
```

The `/q` switch makes the Windows installation unattended, which is what makes the agent deployable through Group Policy or a software distribution tool such as SCCM. Point the package at the MSI and pass the same parameters as above.

Verify the agent afterwards in the SIEM module of the Console. Enrollment traffic uses port 1515 and the agent then reports on 1514.

Blacklist and Threat Intelligence Setup

Configure automated threat intelligence updates for enhanced security monitoring.

Blacklist Creation Methods

1. **Crontab scheduling** - Traditional cron-based updates
2. **ELS Console scheduler** - GUI-based scheduling
3. **Network Probe input** - Preferred method for real-time updates

Instructions for Network Probe setup: *Network Probe Installation*

Scheduling Bad IP Lists Update

Requirements:

- Credentials configured in scripts: `misp_threat_lists.sh`

Option 1: Cron scheduling (host with Network Probe installed)

```
# Add to /etc/crontab
0 2 * * * logserver-probe /etc/logserver-probe/lists/bin/misp_threat_lists.sh
```

Option 2: ELS Console Scheduler (only if Network Probe runs on same host) Prepare script path:

```
/bin/ln -sfn /etc/logserver-probe/lists/bin /opt/ai/bin/lists
chown <user>:<group> /etc/logserver-probe/lists/ # Replace with appropriate␣
↪user and group
chmod g+w /etc/logserver-probe/lists/
```

Configure in ELS Console Scheduler app:

- **Name:** MispThreatList
- **Cron pattern:** `0 2 * * *`
- **Command:** `lists/misp_threat_lists.sh`
- **Category:** network-probe

> **Warning**
>
> The credentials shown are default values. Change them immediately after installation.

Verify blacklists index creation:

```
curl -sS -u $CREDENTIAL -XGET '127.0.0.1:9200/_cat/indices/.blacklists?
↪s=index&v'
```

Post-Installation Steps

Change Default Credentials

After installation, change the default credentials using the password utility:

```
/usr/share/logserver/utils/logserver-password-util.sh update_credentials
```

The utility supports the following commands:

Command	Description
`update_credentials`	Change the password for a selected user and update related services
`update_services`	Update service configuration files with credentials from the keystore
`list_users`	List users currently saved in the keystore
`get_password`	Retrieve the password for a single user
`verify`	Verify stored credentials against the database

Common options:

Option	Description
`-u, --users`	Comma-separated list of users to update (default: all)
`-f, --file-path`	Custom directory for the keystore
`-p, --els-url`	Custom Logserver database URL (if no local database node)
`-n, --no-check`	Skip credential verification against the database
`-y, --yes`	Automatically answer yes to all prompts
`-r, --no-restart`	Only update configuration files, do not restart services
`-v, --verbose`	Print debug messages
`-q, --quiet`	Print only error messages
`-h, --help`	Display help message

Each system component authenticates to the Logserver database using a dedicated internal account. The password utility manages these credentials. It stores them in a central keystore and propagates changes to the configuration files of the related services.

The following accounts are managed by the utility:

Account	Related service(s)
`logserver`	logserver-gui, cerebro, skimmer, logserver-probe
`alert`	alert
`e-doc`	e-doc
`intelligence`	intelligence, intelligence-scheduler
`license`	license-service
`probe`	logserver-probe
`logstash` *(deprecated)*	logserver-probe

Credentials are stored in the keystore at `/etc/logserver-store/logserver.keystore`.

> **Warning**
>
> Change the default credentials immediately after installation. Default credentials in a production environment pose a security risk.

4.2.3 Data Node Cluster Configuration

- *Encryption*
- *Browser layer encryption*
- *Building a cluster*
- *Node roles*
- *Naming convention*
- *Config files*
- *TLS Certificates*
- *Example setup*
- *Adding a new node to the existing cluster*

Encryption

Cipher configuration for the Network Probe, including the `java.properties` defaults and the procedure for releases older than 7.6.1, is described in *Network Probe Installation*.

Browser layer encryption

Browser layer encryption is available in the *TLS and Encryption* section.

Building a cluster

For the step-by-step procedure to install and join a new node to an existing cluster, see *Multi-node Cluster Installation* in the Installation chapter. The sections below cover node role definitions, configuration parameters, and a reference example.

Node roles

Every instance of the Data Node server is called a *node*. A collection of connected nodes is called a *cluster*. All nodes know about all the other nodes in the cluster and can forward client requests to the appropriate node.

Besides that, each node serves one or more purposes:

- Master-eligible node - A node that has a *node.master* set to true (default), which makes it eligible to be elected as the master node, which controls the cluster
- Data node - A node that has a *node.data* set to true (default). Data nodes hold data and perform data-related operations such as CRUD, search, and aggregations
- Client node - A client node has both *node.master* and *node.data* set to false. It can neither hold data nor become the master node. It behaves as a "*smart router*" and is used to forward cluster-level requests to the master node and data-related requests (such as search) to the appropriate data nodes

Naming convention

Data Node requires little configuration before going to work.

The following settings must be considered before going to production:

- **path.data** and **path.logs** - default locations of the data.
- **cluster.name** - A node can only join a cluster when it shares its `cluster.name` with all the other nodes in the cluster. The default name is "logserver", but you should change it to an appropriate name that describes the purpose of the cluster. You can do this in the `/etc/logserver/logserver.yml` file.
- **node.name** - By default, Data Node will use the first seven characters of the randomly generated UUID as the node ID. Node ID is persisted and does not change when a node restarts. It is worth configuring a more human-readable name: `node.name: prod-data-2` in file `/etc/logserver/logserver.yml`
- **network.host** - parameter specifying network interfaces to which Data Node can bind. The default is `network.host`: ["_local_", "_site_"].
- **discovery** - configuration for cluster node discovery:
  - `discovery.seed_hosts` - specify a list of other nodes in the cluster that are likely to be live and contactable;
  - `cluster.initial_master_nodes` - list of master-eligible nodes that should participate in the initial cluster bootstrap.
- **heap size** - By default, Data Node tells the JVM to use a heap with a minimum (Xms) and maximum (Xmx) size of 1 GB. When moving to production, it is important to configure heap size to ensure that Data Node has enough heap available

Config files

To configure the Data Node cluster you must specify some parameters in the Data Node main .yml configuration file on every node that will be connected to the cluster:

- `/etc/logserver/`:
  - `cluster.name:name_of_the_cluster` - same for every node;
  - `node.name:name_of_the_node` - uniq for every node;
  - `node.master:true_or_false`
  - `node.data:true_or_false`
  - `network.host:["_local_","_site_"]`
  - `discovery.seed_hosts`
  - `cluster.initial_master_nodes`
- `/etc/logserver/log4j2.properties`:
  - `logger: action: DEBUG` - for easier debugging.

TLS Certificates

To generate TLS certificates for each node of the cluster, you can check the `logserver-cluster.yml` config that is provided with the `tlstool.sh`.

```
cd /usr/share/logserver/utils/tlstool
bash tlstool.sh -c config/logserver-cluster.yml -ca -crt
```

Example setup

Example of the Data Node cluster configuration:

File `/etc/logserver/logserver.yml`:

```
cluster.name: tm-lab
node.name: "logserver"
node.master: true
node.data: true
network.host: 127.0.0.1,10.0.0.4
http.port: 9200
discovery.seed_hosts: ["10.0.0.4:9300","10.0.0.5:9300","10.0.0.6:9300"]
cluster.initial_master_nodes: ["logserver"]
```

Apply the configuration on the node:

```
systemctl restart logserver
```

Then check the cluster over the API:

```
curl -XGET '127.0.0.1:9200/_cat/nodes?v'

host            ip   heap.percent ram.percent load node.role master name
10.0.0.4   10.0.0.4     18           91      0.00 -         -       els01
10.0.0.5   10.0.0.5     66           91      0.00 d         *       els02
10.0.0.6   10.0.0.6     43           86      0.65 d         m       els03
10.0.0.7   10.0.0.7     45           77      0.26 d         m       els04
```

Or follow the cluster log:

```
tail -f /var/log/logserver/tm-lab.log
```

The log file is named after `cluster.name`, so replace `tm-lab` with the name used on the node.

Adding a new node to the existing cluster

For the full procedure, see *Multi-node Cluster Installation* in the Installation chapter.

In brief:

1. Run `./install.sh -i` on the new node. When asked "Is this node a part of an existing cluster?" answer `Y`.
2. Update `/etc/logserver/logserver.yml` with the cluster parameters: `cluster.name`, `node.name`, `node.master`, `node.data`, `discovery.seed_hosts`.
3. If the node has the `data` role, clear the data directory: `rm -rf /var/lib/logserver/*`
4. Finalize: `./install.sh configure`

4.2.4 Backup and Disaster Recovery

Energy Logserver provides two backup mechanisms:

- **Configuration backup**: the `configuration-backup.sh` script saves configuration files, system indices, and templates to a local archive. This script is included with the installation and is offered automatically before each upgrade. See *Backup and Restore* for details.
- **Snapshot backup**: the snapshot API for full index data backup and disaster recovery. This requires a configured snapshot repository (described below).

Configuration Backup

The built-in `configuration-backup.sh` script creates a timestamped archive in `/root/`:

```
bash /usr/share/logserver/utils/configuration-backup.sh
```

The archive includes: Logserver config, GUI config, Network Probe config, alert config, cerebro config, skimmer config, intelligence config, license-service config, e-doc config, SIEM manager config and agent keys, credential keystore, system indices data, and templates.

The installation already schedules the script daily at 01:00 (`0 1 * * *`) through `/etc/cron.d/configuration-backup`. Edit the schedule in that file; a second cron entry only produces duplicate archives.

Snapshot Backup

For full index data backup, use the snapshot API. This requires configuring a shared filesystem repository.

1. Configure snapshot repository path

Add the repository path to the Logserver configuration on **every data node**:

```
# /etc/logserver/logserver.yml
path.repo: ["/mnt/snapshots"]
```

Restart the data node service after changing the configuration:

```
systemctl restart logserver
```

2. Register the repository

```
curl -u $USER:$PASSWORD -X PUT "localhost:9200/_snapshot/backup_repo" \
  -H 'Content-Type: application/json' -d '{
  "type": "fs",
  "settings": {
    "location": "/mnt/snapshots"
  }
}'
```

3. Create a snapshot

```
curl -u $USER:$PASSWORD -X PUT "localhost:9200/_snapshot/backup_repo/snapshot_
↪$(date +%Y%m%d)" \
  -H 'Content-Type: application/json' -d '{
  "indices": "*",
  "ignore_unavailable": true,
  "include_global_state": true
}'
```

4. List existing snapshots

```
curl -u $USER:$PASSWORD "localhost:9200/_snapshot/backup_repo/_all?pretty"
```

Disaster Recovery from Snapshot

1. Stop services

```
systemctl stop logserver-gui logserver-probe alert cerebro e-doc skimmer \
  intelligence intelligence-scheduler license-service
```

2. Restore from snapshot

```
curl -u $USER:$PASSWORD -X POST \
  "localhost:9200/_snapshot/backup_repo/SNAPSHOT_NAME/_restore" \
  -H 'Content-Type: application/json' -d '{
  "indices": "*",
  "ignore_unavailable": true,
  "include_global_state": true
}'
```

3. Monitor restore progress

```
curl -u $USER:$PASSWORD "localhost:9200/_cat/recovery?v&active_only=true"
```

4. Start services

```
systemctl start logserver-gui alert cerebro e-doc skimmer \
  intelligence intelligence-scheduler license-service logserver-probe
```

Scheduling Snapshots

Run the snapshot call on a schedule with cron, the same way as the configuration backup. Escape the `%` characters for crontab:

```
0 2 * * * root curl -u $USER:$PASSWORD -X PUT "localhost:9200/_snapshot/
↪backup_repo/snapshot_$(date +\%Y\%m\%d)" -H 'Content-Type: application/json
↪' -d '{"indices":"*","ignore_unavailable":true,"include_global_state":true}'
```

Prune old snapshots in the same routine so the repository does not grow without bound.

High Availability within a Site

Keep every index replicated with at least one replica, and let the cluster place a primary and its replica on different physical hosts, racks, or rooms, so no single hardware failure loses data. Tag each node with a location attribute and enable allocation awareness:

```
# /etc/logserver/logserver.yml (set a distinct value per node group)
node.attr.zone: zone-a
cluster.routing.allocation.awareness.attributes: zone
```

With awareness enabled, the cluster keeps a primary and its replica in different zones and spreads shards across zones. Running the nodes as virtual machines on separate hypervisors with this setting keeps a primary and its replica off the same physical host.

Multi-Site Recovery

Energy Logserver recovers a second site from snapshots rather than replicating a live cluster between sites.

Shared or replicated repository. Place the snapshot repository on storage the recovery site can reach: a shared filesystem mounted on both sites, or a directory whose contents you replicate to the recovery site. Register the repository on the recovery cluster as read-only, so it can restore and search without writing to it:

```
curl -u $USER:$PASSWORD -X PUT "localhost:9200/_snapshot/backup_repo" \
  -H 'Content-Type: application/json' -d '{
  "type": "fs",
  "settings": {
    "location": "/mnt/snapshots",
    "readonly": true
  }
}'
```

Only one cluster writes to a repository. Registering every other cluster as `readonly` avoids repository corruption from two writers.

Querying both sites. To search data held on separate clusters in different data centers from a single place, use *Cross-cluster Search*.

Moving selected data. To copy or migrate specific indices between clusters, use elasticdump, the same tool the bundled `small_backup.sh` uses, which exports and imports index data over the API.

4.2.5 Offline Installation

Energy Logserver supports installation on servers that do not have access to the internet (security zones, air-gapped networks). The installer ships with helper scripts that build a local RPM repository on a staging server with internet access, package it into a single archive, and consume it on the offline target host.

This procedure applies both to fresh installations (`./install.sh -i --offline`) and upgrades (`./install.sh -u --offline`).

When to Use Offline Installation

Use this mode when the target host cannot reach public RPM repositories, typically:

- Production servers in a security zone with restricted outbound traffic
- Isolated laboratory environments
- Hosts behind a proxy that is not configured for the installer

In all other cases prefer the standard online installation described in *Installation*.

Prerequisites

- **Staging server** with internet access, running the **same OS and version** as the offline target. Supported: CentOS Stream 9, Oracle Linux, Red Hat Enterprise Linux, Rocky Linux, AlmaLinux, version 8 or 9. CentOS Stream 8 is not supported. A mismatch in OS version will cause the offline installation to fail because the prepared dependencies will not match the target's `dnf` module streams.
- **Free disk space** on the staging server for the dependency archive (typically several gigabytes).
- **The same Energy Logserver installation package** on both servers (staging and target).

Step 1: Prepare the Local Repository on the Staging Server

On the staging server (with internet access):

1. Extract the Energy Logserver installation package and enter the `install/` directory:

```
tar xjf energy-logserver-*.x86_64.tar.bz2
cd install
```

2. Run the repository preparation script as `root`:

```
./scripts/prepare_local_repository.sh
```

   The script:

   - Installs the tools required to build the repository: `createrepo`, `python3-dnf-plugin-modulesync`, `tar`.
   - Downloads every dependency (Energy Logserver uses Java, Python, mail, archive, reports, blacklist tooling and optional components like Suricata, Zeek, pmacct, GraphicsMagick) along with all transitive dependencies into `install/offline-dependencies/`.
   - Synchronises `dnf` module metadata into the same directory.
   - Packages the result into a single tarball: `install/local-logserver-repo.tar`.

3. When the script finishes, copy `install/local-logserver-repo.tar` to the offline target server and place it in the `install/` directory of the installation package on the target.

Step 2: Install on the Offline Target

On the offline target server, from the `install/` directory of the installation package:

```
./install.sh -i --offline
```

The `--offline` flag (short form `-o`) tells the installer to:

1. Create a local `dnf` repository configuration at `/etc/yum.repos.d/offline-logserver-dependencies.repo` pointing at `file://<install_dir>/offline-dependencies`.
2. Extract `local-logserver-repo.tar` into `install/offline-dependencies/`.
3. Rebuild the `dnf` cache against this local repository only (`--disablerepo=* --enablerepo=offline-logserver-dependencies`).
4. Install all Energy Logserver packages and their dependencies from the local repository. No network traffic is required.

All other installation options (`--data`, `--client`, `--siem`, `--network-probe`, `--automation`, `--noninteractive`) work with `--offline` exactly as in an online installation.

Upgrading an Offline Installation

Use the same `--offline` flag with the `-u` command and a matching `local-logserver-repo.tar` rebuilt from the installation package of the new version:

```
./install.sh -u --offline
```

Troubleshooting

"To configure local repository first copy and run following script..."

The installer prints this message when it cannot find a usable `local-logserver-repo.tar` or the archive is missing valid repodata (no `repodata/repomd.xml` inside). Re-run `prepare_local_repository.sh` on the staging server, then copy the refreshed `local-logserver-repo.tar` to `install/` on the target.

Unsupported OS detected

`prepare_local_repository.sh` aborts with this message when run on an OS other than CentOS Stream 9, Oracle/Red Hat/Rocky/AlmaLinux 8 or 9. Use a supported staging OS that matches the offline target.

OS mismatch between staging and target

If the staging and target servers run different minor versions or different distributions, the prepared RPMs and module metadata will not satisfy dependencies during installation. `dnf` will report unresolved packages. Rebuild the repository on a staging server whose OS version exactly matches the target.

4.2.6 Uninstallation

This procedure removes Energy Logserver from a RHEL-based system (RHEL, Rocky Linux, AlmaLinux).

> **Warning**
>
> Uninstallation permanently deletes all collected data, indices, dashboards, alert rules, and configuration. Back up anything you want to keep before proceeding. See *Backup and Recovery*.

Step 1: Stop all services

Stop every ELS component before removing packages:

```
for svc in logserver-probe alert intelligence intelligence-scheduler␣
↪automation skimmer cerebro e-doc license-service logserver-gui logserver; do
    systemctl is-active --quiet "$svc" && systemctl stop "$svc"
done
```

Step 2: Disable services

Prevent the components from restarting on boot:

```
for svc in logserver-probe alert intelligence intelligence-scheduler␣
↪automation skimmer cerebro e-doc license-service logserver-gui logserver; do
    systemctl is-enabled --quiet "$svc" 2>/dev/null && systemctl disable "$svc
↪"
done
```

Step 3: Remove packages

The product ships as several packages. The `energy-logserver-*` glob covers the platform packages (data node, client node, SIEM plan, Empowered AI, integrations, network probe, automation), while the license service, Skimmer, and the SIEM manager carry their own names:

```
yum remove -y 'energy-logserver-*' license-service skimmer wazuh-manager
```

If the host ran the optional Network Probe sensors, remove them as well:

```
yum remove -y zeek suricata pmacct
```

Verify nothing remains:

```
rpm -qa | grep -E 'energy-logserver|license-service|skimmer|wazuh'
```

The command should return no output.

> **Warning**
>
> Do not use `/usr/share/logserver/utils/uninstall.sh`. That script predates the 8.x path and service renaming: it stops services that no longer exist under those names, removes directories the current release does not use, and leaves the data directories in place. Follow the procedure on this page instead.

Step 4: Remove residual data and configuration

The package manager removes RPM-tracked files, but data directories, indices, and logs are preserved by design. Remove them manually. Target the real directories, not the product-branded paths: the branded paths (`/etc/logserver`, `/etc/logserver-gui`, `/etc/logserver-probe`, and their `/var/lib`, `/var/log`, `/usr/share` variants) are symlinks that Step 3 already removed with the packages, so deleting them now would leave the actual data behind.

```
# Data Node: config, indices, logs
rm -rf /etc/elasticsearch /var/lib/elasticsearch /var/log/elasticsearch /usr/
↪share/elasticsearch

# GUI: config and data
rm -rf /etc/kibana /var/lib/kibana /usr/share/kibana

# Network Probe: config, data, logs
rm -rf /etc/logstash /var/lib/logstash /var/log/logstash /usr/share/logstash

# Component directories
rm -rf /opt/alert /opt/cerebro /opt/e-doc /opt/intelligence /opt/license-
↪service /opt/skimmer /opt/automation /opt/ai /opt/plugins /opt/uba

# System tuning files installed by ELS
rm -f /etc/security/limits.d/30-logserver.conf
rm -f /etc/sysctl.d/90-logserver.conf /etc/sysctl.d/90-namespaces.conf

# Cron jobs
rm -f /etc/cron.d/configuration-backup /etc/cron.d/maxmind_geoip_db_job
```

Step 5: Reload systemd

```
systemctl daemon-reload
systemctl reset-failed
```

Step 6: Revert system settings

The sysctl and limits files were removed in Step 4, but the running kernel still holds the values set during installation. Revert them for the current session, or reboot:

```
sysctl -w vm.max_map_count=65530
sysctl -w vm.swappiness=60
sysctl -w net.ipv4.tcp_retries2=15
```

Or simply reboot. The defaults from `/etc/sysctl.conf` will apply.

4.3 Upgrade

This chapter covers upgrade procedures for Energy Logserver, including version verification, planning, and execution of upgrades with minimal downtime. It provides detailed upgrade paths for version-specific migrations, cluster upgrade procedures for multi-node environments, and rollback procedures to ensure safe upgrade execution.

4.3.1 Version Verification and Planning

Confirm what you are running and verify the supported upgrade path before you download the installer. This page shows how to read the installed version and license, check cluster health, and plan the upgrade to Energy Logserver 8.1.

- *Current Version Check*
- *System Health Pre-Check*
- *Pre-Upgrade Checklist*
- *Capacity and Requirements*
- *Compatibility Matrix*
- *Upgrade Downtime*

Current Version Check

The installed product version is the value you match against the *Compatibility Matrix*. Read it from the installed packages on the node:

```
rpm -qa | grep energy-logserver
```

Example output:

```
energy-logserver-data-node-8.1.0-1.x86_64
energy-logserver-client-node-8.1.0-1.x86_64
energy-logserver-siem-plan-8.1.0-1.x86_64
```

The version segment (`8.1.0-1`) is your current release.

> **Note**
>
> The cluster root endpoint reports the underlying OpenSearch engine version, not the Energy Logserver product version:

```
curl -u $USER:$PASSWORD -X GET "http://localhost:9200/"
```

The `version.number` it returns (for example `3.3.2`) is the search engine build. Use the installed package version above for upgrade planning.

License and Entitlements

Review the active license before planning. The upgrade reuses the existing license file, but the license also defines the node count and add-on features you are entitled to:

```
curl -u $USER:$PASSWORD -X GET "http://localhost:9200/_logserver/license"
```

The response lists the licensed node count (`nodes`), expiration (`expiration_date`, `days_left`), and the enabled add-ons (`siemPlan`, `networkProbe`, and others). The `version` field records the release the license was issued for, which can be older than the installed build: a license issued for any release from 7.4.0 onwards stays valid on 8.x, so treat this field as entitlement information, not as the running version. A new license is only required to enable add-on features introduced in a newer release. See *Licensing* for details.

System Health Pre-Check

Confirm the cluster is healthy and has resource headroom before you start:

```
curl -u $USER:$PASSWORD "http://localhost:9200/_cluster/health?pretty"

systemctl status logserver logserver-gui logserver-probe intelligence␣
↪intelligence-scheduler alert license-service skimmer e-doc --no-pager

df -h /usr/share/logserver /var/lib/logserver

free -h
```

A `green` or `yellow` cluster status, all services active, and free disk on the data path are the baseline for a safe upgrade.

Pre-Upgrade Checklist

Complete these steps before running `install.sh -u`:

1. **Back up the cluster.** Take a full snapshot so you can roll back if the upgrade fails. See *Backup and Recovery* and *Rollback Procedures*.
2. **Check index compatibility.** Indices created on versions older than 7.4.0 must be reindexed or deleted first. Run the compatibility check before you download the installer, as described in *Index compatibility check*.
3. **Preserve custom configuration.** The upgrade overwrites `jvm.options`; move custom JVM settings into `/etc/logserver/jvm.options.d/`. Review the full list in *Breaking and major changes*.
4. **Verify loopback access on multi-node clusters.** The installer connects to the data node at `127.0.0.1:9200`. Confirm `network.host` in `logserver.yml` includes the loopback address before upgrading, as described in *Multi-node upgrade order*.
5. **Confirm license entitlements.** Check that the active license covers the target deployment, and obtain a new license first if you plan to enable a new add-on feature.
6. **Plan the maintenance window.** See *Upgrade Downtime* for estimates.

Capacity and Requirements

An in-place upgrade keeps the existing hardware, so no additional CPU or memory is required beyond the running deployment. Confirm the node still meets the hardware and software baseline for the target release in *System Requirements*.

Reserve free disk headroom on the data path for the pre-upgrade snapshot, and for any reindexing the compatibility check flags. Both consume additional space on top of the existing indices.

Compatibility Matrix

The Compatibility Matrix defines the supported upgrade paths to Energy Logserver 8.1. Match your current version, read from `rpm -qa` above, against the table.

> **Warning**
>
> Upgrading from 7.x to 8.x requires a full cluster restart. Rolling upgrades are not supported, and an upgraded node will not join an existing 7.x cluster.

From Version	Direct Upgrade	Notes
8.0.x	Supported	Run `install.sh -u`, then follow the post-upgrade steps in *Upgrade to Version 8.1.0*.
7.4.0 - 7.9.x	Supported	Full cluster restart required. `install.sh -u` checks index compatibility; indices created on versions older than 7.4.0 must be reindexed or deleted before upgrading.
< 7.4.0	Multi-step	Requires an intermediate upgrade to a version between 7.4.0 and 7.9.0 first, then reindexing incompatible indices, then upgrading to 8.x. Alternatively: a fresh 8.x installation and data migration.
7.2.x and below	Unsupported	Complete reinstallation needed

Upgrade Downtime

Single-node deployment:

- Downtime: approximately 1 hour
- Process: full backup, stop services, run `install.sh -u`, verify

Multi-node cluster:

- Downtime: depends on cluster size and index count; plan for an extended maintenance window
- Process: stop all collectors, upgrade the client node first, then the data nodes one by one, answering **No** to the installer's restart prompt. Start services manually only after every node has been upgraded. See *Multi-node upgrade order* for the full procedure.

Rolling upgrades within 7.x:

- Downtime: 15-30 minutes
- Process: node-by-node upgrade with service continuity, available only for upgrades within the 7.x release line. See *Rolling Upgrades*.

4.3.2 Version-Specific Upgrades

Warning

The upgrade script must be executed from within the `install` directory. Running `./install.sh -u` from any other location will result in a failed upgrade.

```
cd install
./install.sh -u
```

Tip

If the cluster has been intentionally stopped before the upgrade, you can bypass the index compatibility check by passing the `--skip-check` flag:

```
./install.sh -u --skip-check
```

Use this only when you have already verified that all cluster indices are compatible with the target version.

- *Upgrade to Version 8.1.0*
- *Upgrade to Version 8.0.0*
- *Upgrade from Version 7.9.0*
- *Upgrade from Version 7.8.0*
- *Upgrade from Version 7.7.0*
- *Upgrade from Version 7.6.0*
- *Upgrade from Version 7.5.0*
- *Upgrade from Version 7.4.2 / 7.4.3*
- *Upgrade from Version 7.4.1*
- *Upgrade from Version 7.4.0*
- *Upgrade from Version 7.3.0*

Upgrade to Version 8.1.0

These steps cover an upgrade from 8.0.x. If you upgrade from 7.x directly to 8.1.0, follow *Upgrade to Version 8.0.0* first, then complete the post-upgrade steps below as well.

Breaking changes in 8.1.0

- **Microsoft Teams alerter removed:** The alert engine in 8.1.0 no longer ships the `ms_teams` alert type. After the upgrade the alert engine skips rules that use it, logs `Could not load rule ... Error initiating alert ['ms_teams']` in `/var/log/alert/alert.log`, and keeps running the other rules. Before you upgrade, list the affected rules and move them to another notification method, for example a webhook:

```
curl -u $USER:$PASSWORD "127.0.0.1:9200/.alertrules/_search?q=alertrule_
↪any:*ms_teams*&_source=alertrulename&pretty"
```

Running the 8.1.0 upgrade

```
cd install
./install.sh -u
```

At the end of the run the installer prints the post-upgrade steps below. It performs some of them on its own and reports which ones failed.

Steps after upgrading to 8.1.0

1. **SIEM Engine inventory and vulnerability detection.** On every node that runs the SIEM Engine manager (`wazuh-manager`), store the `probe` user credentials in the SIEM Engine keystore:

   ```
   /usr/share/logserver/utils/logserver-password-util.sh update_services -u␣
   ↪probe
   ```

   Confirm both prompts with `y`. The command also restarts `wazuh-manager`, `logserver-probe`, and `logserver-gui`.

   Then check that the `<indexer>` section in `/var/ossec/etc/ossec.conf` is enabled and that its `<host>` points at the data node. The installer does not overwrite your `ossec.conf`. It places the 8.1.0 default next to it as `/var/ossec/etc/ossec.conf.new`, which you can use as a reference:

   ```
   <indexer>
     <enabled>yes</enabled>
     <hosts>
       <host>http://127.0.0.1:9200</host>
     </hosts>
   </indexer>
   ```

   If the data node serves HTTPS, use an `https://` address and configure the `<ssl>` block with your CA certificate.

2. **GeoIP and threat list scripts.** The installer updates `/etc/logserver-probe/geoipdb/bin/maxmind_geoip_db.sh` and `/etc/logserver-probe/lists/bin/misp_threat_lists.sh` with the repository credentials stored in your license. If it reports *Failed to fetch repo credentials from a license*, update them by hand, from the `install` directory:

   ```
   [[ -f /etc/logserver-probe/geoipdb/bin/maxmind_geoip_db.sh.rpmnew ]] &&␣
   ↪mv -vf /etc/logserver-probe/geoipdb/bin/maxmind_geoip_db.sh.rpmnew /etc/
   ↪logserver-probe/geoipdb/bin/maxmind_geoip_db.sh
   [[ -f siem/logstash/logstash/lists/bin/misp_threat_lists.sh ]] && cp -vf␣
   ↪siem/logstash/logstash/lists/bin/misp_threat_lists.sh /etc/logserver-
   ↪probe/lists/bin/misp_threat_lists.sh
   [[ -f logstash/logstash/lists/bin/misp_threat_lists.sh ]] && cp -vf␣
   ↪logstash/logstash/lists/bin/misp_threat_lists.sh /etc/logserver-probe/
   ↪lists/bin/misp_threat_lists.sh
   ```

   Then set `GEOIP_USER` and `GEOIP_PASS` in `maxmind_geoip_db.sh`, and `IOC_USER` and `IOC_PASS` in `misp_threat_lists.sh` (SIEM only), to your repository credentials.

The update resets the data node connection settings in `misp_threat_lists.sh`, whether the installer or you replaced the file. Check `LOGSERVER_HOST`, `LOGSERVER_PORT`, and `LOGSERVER_SSL` in it. The script uses them to remove expired entries from the data node, and the defaults (`127.0.0.1`, `9200`, `false`) work only on a node with a local data node over HTTP.

3. **Probe user in the Network Probe keystore.** On **each** Network Probe node, check that the `logserver-probe` keystore holds the `probe` user:

```
cd /usr/share/logserver-probe && sudo -u logserver-probe /usr/share/
↪logserver-probe/bin/logstash-keystore --path.settings /etc/logserver-
↪probe/ list | grep 'probe_'
```

The output must contain `probe_user` and `probe_pass`. If they are missing, add them. On a node without a local data node, point `--els-url` at the data node; on a node that runs one, omit the option:

```
/etc/logserver-probe/utils/logserver-password-util.sh update_services -u␣
↪probe --els-url http://<data-node>:9200
```

The installer runs this check and the fix on its own when it upgrades the Network Probe. On a node that also runs the client node, it does so only if you let it restart the services at the end of the upgrade.

Verifying the 8.1.0 upgrade

```
systemctl status logserver logserver-gui logserver-probe intelligence␣
↪intelligence-scheduler alert license-service skimmer e-doc --no-pager
```

On SIEM nodes, also check `systemctl status wazuh-manager --no-pager`.

Upgrade to Version 8.0.0

> **Warning**
>
> Upgrading to 8.0.0 requires a full cluster restart. Rolling upgrades are not supported for this version. An upgraded node will not join an existing 7.x cluster.

Pre-upgrade steps

The Data Node update process will shut down the database service (`logserver.service`). Execute the following steps before running the upgrade:

1. Stop the collectors (Network Probe and others):

```
systemctl stop logserver-probe
```

2. Disable shard allocation:

```
curl -u $USER:$PASSWORD -X PUT "127.0.0.1:9200/_cluster/settings" \
  -H "Content-Type: application/json" -d '{
  "persistent": {
    "cluster.routing.allocation.enable": "none"
  }
}'
```

3. Perform a flush:

```
curl -u $USER:$PASSWORD -X POST "127.0.0.1:9200/_flush?pretty"
```

4. Run the upgrade:

```
./install.sh -u
```

Multi-node upgrade order

Warning

`install.sh -u` connects to the data node at `http://127.0.0.1:9200` (hardcoded default; the URL prompt is skipped in upgrade mode). If `network.host` in `logserver.yml` is set to a specific IP without including `127.0.0.1` or `_local_`, the installer cannot reach OpenSearch and fails with *Connection refused*.

Before upgrading each node, verify that `network.host` includes the loopback address:

```
# /etc/logserver/logserver.yml
network.host: ["10.x.x.x", "127.0.0.1"]
```

Restart `logserver.service` on that node if you change this setting.

For multi-node deployments, run the upgrade in this order:

1. Start with the **client node** (the node that serves the GUI and coordinates queries).
2. When `install.sh` asks whether to restart services, answer **No**. Services on each upgraded host stay stopped until every node has finished the upgrade.
3. Upgrade the remaining data nodes one by one using the same procedure.
4. Once every node has completed `install.sh -u`, start the services manually across the cluster - `logserver` first on each data node, then the remaining services.
5. After the data nodes are running and the cluster reports at least `yellow` status, finalize the built-in user initialization on the **client node**:

```
curl -u $USER:$PASSWORD -X POST "127.0.0.1:9200/_logserver/init"
```

Warning

This manual step is required when upgrading from 7.9.0 or older to 8.x. On 8.x the built-in users (`admin` and others) are created through `_logserver/init`. The installer runs this initialization automatically only when it restarts `logserver.service` itself, which happens during a data-node upgrade if you answer **Yes** to the *"Some system services require to be restarted"* prompt. In the multi-node procedure above you answer **No** to that prompt and restart the services by hand, so the automatic initialization is skipped and the cluster stays uninitialized until you run `_logserver/init` yourself.

To check whether a node still needs initialization, query its health endpoint - an uninitialized node responds with `Node uninitialized. To finalize init process please run ...`:

```
curl -X GET "127.0.0.1:9200/_logserver/health"
```

> **Warning**
>
> The data-node RPM stops `logserver.service` on the host being upgraded, so the cluster runs degraded from the first node onwards. Do not let the installer restart services mid-upgrade - restart them manually only after every node has been upgraded.

Index compatibility check

The upgrade requires reindexing for indices created in versions older than Energy Logserver 7.4.0. The installation process (`./install.sh -u`) performs a compatibility check automatically and outputs a list of incompatible indices. These indices must be either reindexed or deleted before the upgrade can proceed.

To check index compatibility manually before downloading the installer package:

```
ELS_URL="http://127.0.0.1:9200"
ELS_AUTH='user:pass'

indices_metadata="$(curl --fail -XGET --insecure -u "${ELS_AUTH}" \
  "${ELS_URL}/_cluster/state/metadata?filter_path=metadata.indices.**.index.
↪version.created&pretty")"

mapfile -t indices_to_upgrade < <(echo "${indices_metadata}" | tr '\n' ' ' \
  | sed -e 's/ //g' \
        -e 's/,"\([^ "]\+\)":{"settings":{"index":{"version":{"created":"\([0-
↪9]\+\)"}\+/ \1 \2/g' \
  | grep -o '\S\+ [0-9]\+' \
  | awk '$2 < 136217827 { print $1 }' \
  | sort)

printf "%s\n" "${indices_to_upgrade[@]}"
```

If any indices are listed, they must be reindexed or removed before upgrading.

Breaking and major changes

- **JVM options:** The file `jvm.options` is overwritten by the upgrade. User-defined JVM settings must be placed in `/etc/logserver/jvm.options.d/`. Specifically:
  - Heap size settings (`-Xms`, `-Xmx`) are now in `/etc/logserver/jvm.options.d/heap-size.options`
  - JVM temporary directory (`-Djava.io.tmpdir`) is now in `/etc/logserver/jvm.options.d/tmp-dir.options`

  If the Data Node had additional settings in `jvm.options`, move them from `/etc/logserver/jvm.options.rpmsave` to `/etc/logserver/jvm.options.d/`.

  > **Warning**
  >
  > `-Djava.io.tmpdir` cannot be set to a path inside `/etc/`, `/usr/`, or `/boot/`.

- **Java update:** OpenJDK 21 is now required (previously OpenJDK 17).

- **Service names change:** the services run as `logserver`, `logserver-gui`, and `logserver-probe`. The previous names remain systemd aliases, so `systemctl` still accepts them, but `journalctl -u` requires the new names.
- **License:** The existing license file is reused. No replacement is required to complete the upgrade. Licenses issued for any release from **7.4.0** onwards remain valid on 8.0. A new license is only needed if you want to enable add-on features introduced in 8.0 (for example, `AI Assistant`); in that case, finish the upgrade first and upload the new license from the GUI (**Config** → **License** → **Upload new license**).
- **Alert Discover link field renamed:** The field that injects the Discover link into an alert notification was renamed from `kibana_discovery_url` to `discovery_url` (changed in 7.9.0). Rule definitions that still reference `kibana_discovery_url` resolve to an empty value, so the affected alerts arrive without the link. This applies only to rules where the field was placed manually in the **Rule Definition** editor; alerts that use the **Generate Discover URL** toggle in the form are not affected. To fix an affected rule, edit the alert and rename the field. For many rules at once, replace the field name directly in the `.rules` index documents.

Post-upgrade steps

Once services on all nodes are running, re-enable shard allocation (it was set to `none` in the pre-upgrade steps):

```
curl -u $USER:$PASSWORD -X PUT "127.0.0.1:9200/_cluster/settings" \
  -H "Content-Type: application/json" -d '{
  "persistent": {
    "cluster.routing.allocation.enable": null
  }
}'
```

Setting the value to `null` restores the default (`all`). You can also set it explicitly to `"all"`.

Post-upgrade verification

```
systemctl status logserver logserver-gui logserver-probe intelligence␣
↪intelligence-scheduler alert license-service skimmer e-doc --no-pager
```

Upgrade from Version 7.9.0

Standard upgrade path with no special migration steps.

```
./install.sh -u
```

Post-upgrade verification:

- Verify all services are running:

```
systemctl status logserver logserver-gui logserver-probe intelligence␣
↪intelligence-scheduler alert license-service skimmer e-doc --no-pager
```

- Test new features accessibility
- Check log processing continuity

Upgrade from Version 7.8.0

Preferred Upgrade Steps for 7.8.0

1. Run upgrade script:

```
./install.sh -u
```

Required Post Upgrade from Version 7.8.0

Breaking and major changes

- **Keystore migration:** The default password store has been renamed from `passstore` to `logserver.keystore`. Run the password utility to migrate:

```
/usr/share/logserver/utils/logserver-password-util.sh update_services --yes --
↪no-restart --quiet
```

- **Java update:** OpenJDK 17 is now required. The upgrade script sets the correct Java version automatically. Verify after upgrade:

```
java -version
```

- **Backup scope expanded:** The `configuration-backup.sh` script now includes additional configuration files for `license-service`, `intelligence`, and `logserver-store`. Run a post-upgrade backup to capture the full scope:

```
bash /usr/share/logserver/utils/configuration-backup.sh
```

Post-upgrade verification:

```
systemctl status logserver logserver-gui logserver-probe intelligence␣
↪intelligence-scheduler alert license-service skimmer e-doc --no-pager
```

Upgrade from Version 7.7.0

Standard upgrade path with minimal changes.

```
./install.sh -u
```

Post-upgrade verification:

- Test new features accessibility
- Verify all services functionality
- Check log processing continuity

Upgrade from Version 7.6.0

Preferred Upgrade Steps for 7.6.0

1. Run upgrade script:

```
./install.sh -u
```

Required Post Upgrade from Version 7.6.0

Breaking and major changes

- Archive: Changed the default archives location to `/usr/share/logserver-gui/data/archive/archives/` - please adjust external storage resource to this path [if used]. To customize the `archive.archivefolderpath` directive edit the `/etc/logserver-gui/logserver-gui.yml` file and restart the logserver-gui service.

```
sed -i 's|archive.archivefolderpath:.*|archive.archivefolderpath: "/usr/share/
↪logserver-gui/data/archive/archives/"|' /etc/logserver-gui/logserver-gui.yml

systemctl restart logserver-gui
```

- Network-Probe: Move required directives from `/opt/license-service/license-service.conf` to `/opt/license-service/license-service.conf.rpmnew` and replace `license-service.conf` then restart the license-service.

Upgrade from Version 7.5.0

Preferred Upgrade Steps for 7.5.0

1. Run upgrade script:

```
./install.sh -u
```

Required Post Upgrade from Version 7.5.0

Breaking and major changes

- User "logserver" will no longer be able to log into GUI. Use "admin" user instead. The update process will print the generated password for this new user. In case you have missed it you can run the following command on your main client node to recover the password:

```
/usr/share/logserver/utils/logserver-password-util.sh get_password -u admin -q
```

Upgrade from Version 7.4.2 / 7.4.3

Preferred Upgrade Steps for 7.4.2 / 7.4.3

1. Run upgrade script:

```
./install.sh -u
```

Required Post Upgrade from Version 7.4.2 / 7.4.3

Breaking and major changes

- Network-Probe replaces Logserver-Probe: follow the steps below.

LOGSERVER-PROBE:

- Backup `/etc/logserver-probe`

```
cp -r /etc/logserver-probe /backup/logserver-probe-7.4-backup/
```

- Uninstall old version:

> **Warning**
>
> This command permanently removes Network Probe configuration and data. Ensure you have a backup before proceeding.

```
yum versionlock delete logserver-probe-oss-7.17.11-1
yum remove logserver-probe-oss
rm -rf /etc/logserver-probe /var/lib/logserver-probe /usr/share/logserver-
↪probe
```

- Install the current Input Layer from scratch using `sudo ./install.sh -i` - Network-Probe Section
- Restore from backup custom pipelines to `/etc/logserver-probe/conf.d/`

```
cp /backup/logserver-probe-7.4-backup/conf.d/* /etc/logserver-probe/conf.d/
```

ENERGYLOGSERVER

- `./install.sh` checks indexes compatibility before upgrading. If any problems exist, please contact product support to guide you through the upgrade process.
- Move required directives from `/etc/logserver/logserver.yml` to `/etc/logserver/logserver.yml.rpmnew` and replace `logserver.yml`.

```
mv /etc/logserver/logserver.yml /etc/logserver/logserver.yml.backup
mv /etc/logserver/logserver.yml.rpmnew /etc/logserver/logserver.yml
```

LOGSERVER-GUI

- Move required directives from `/etc/logserver-gui/logserver-gui.yml` to `/etc/logserver-gui/logserver-gui.yml.rpmnew` and replace `logserver-gui.yml`.

```
mv /etc/logserver-gui/logserver-gui.yml /etc/logserver-gui/logserver-gui.yml.
↪backup
mv /etc/logserver-gui/logserver-gui.yml.rpmnew /etc/logserver-gui/logserver-
↪gui.yml
```

- Clear browser cache on client side.

LICENSE-SERVICE

- If required, configure `logserver_connection` in `/opt/license-service/license-service.conf`.
- Old configuration should be in `/opt/license-service/license-service.conf.rpmsave`. Do not replace `license-service.conf` with `license-service.conf.rpmsave`

EMPOWERED-AI (7.4.3 only)

- Backup `.intelligence_models`, `.intelligence_rule_configuration` indices - if needed
- Stop services:

```
systemctl stop empowered-ai
```

- Delete old indices:

```
curl -XDELETE '127.0.0.1:9200/.intelligence_rule_configuration,.intelligence_
↪models,.intelligence_results' -u $USER:$PASSWORD
```

- Start services:

```
systemctl start empowered-ai
```

Upgrade from Version 7.4.1

Preferred Upgrade Steps for 7.4.1

1. Run upgrade script:

```
./install.sh -u
```

No additional post-upgrade steps are required for this version.

Upgrade from Version 7.4.0

Preferred Upgrade Steps for 7.4.0

1. Run upgrade script:

```
./install.sh -u
```

No additional post-upgrade steps are required for this version.

Upgrade from Version 7.3.0

Preferred Upgrade Steps for 7.3.0

1. Run upgrade script:

```
./install.sh -u
```

Required Post Upgrade from Version 7.3.0

ENERGYLOGSERVER

- `./install.sh` checks indexes compatibility before upgrading. If any problems exist, please contact product support to guide you through the upgrade process.
- Move required directives from `/etc/logserver/logserver.yml` to `/etc/logserver/logserver.yml.rpmnew` and replace `logserver.yml`.

LOGSERVER-GUI

- Move required directives from `/etc/logserver-gui/logserver-gui.yml` to `/etc/logserver-gui/logserver-gui.yml.rpmnew` and replace `logserver-gui.yml`.
- Clear browser cache on client side.

LOGSERVER-PROBE

- Backup `/etc/logserver-probe`
- Uninstall old version:

```
yum versionlock delete logserver-probe-oss-7.17.11-1
yum remove logserver-probe-oss
rm -rf /etc/logserver-probe /var/lib/logserver-probe /usr/share/logserver-
↪probe
```

- Install the current Input Layer from scratch using `sudo ./install.sh -i` - Network-Probe Section.
- Restore from backup custom pipelines to `/etc/logserver-probe/conf.d/`

LICENSE-SERVICE

- If required, configure `logserver_connection` in `/opt/license-service/license-service.conf`.
- Old configuration should be in `/opt/license-service/license-service.conf.rpmsave`. Do not replace `license-service.conf` with `license-service.conf.rpmsave`

EMPOWERED-AI Migration (7.3.0 only):

```
curl -X POST "localhost:9200/_snapshot/backup_repo/intelligence_backup" -H
↪'Content-Type: application/json' -u $USER:$PASSWORD -d'
{
  "indices": ".intelligence_models,.intelligence_rule_configuration"
}'

systemctl stop empowered-ai
yum remove empowered-ai

./install.sh -u
```

4.3.3 Cluster Upgrade

> **Warning**
>
> Rolling upgrades are **not supported** when upgrading to version 8.x. An upgraded node will not join an existing 7.x cluster. For upgrades to 8.x, all nodes must be stopped and upgraded together. See *Upgrade to Version 8.0.0* for the full procedure.

This section describes how to upgrade a **multi-node cluster** while maintaining availability during upgrades **within the same major version** (e.g., 7.8 to 7.9). For single-node installations, use the standard procedure described in *Version-Specific Upgrades*.

In a multi-node deployment, each node is upgraded individually using the same `./install.sh -u` script. The API commands in this procedure (shard allocation control, flush) prevent unnecessary shard rebalancing while nodes are being restarted.

Prerequisites

- Cluster health is **green**
- A current backup exists (see *Backup and Recovery*)
- All nodes are running the same Energy Logserver version

Verify cluster health before proceeding:

```
curl -u $USER:$PASSWORD "http://localhost:9200/_cluster/health?pretty"
```

Procedure

1. Disable shard reallocation

Run this once, from any node, before starting the upgrade process:

```
curl -u $USER:$PASSWORD -X PUT "localhost:9200/_cluster/settings" \
  -H "Content-Type: application/json" -d '{
  "persistent": {
    "cluster.routing.allocation.enable": "primaries"
  }
}'

curl -u $USER:$PASSWORD "localhost:9200/_flush?pretty"
```

2. Upgrade each node

Repeat the following steps for **each node** in the cluster.

> **Warning**
>
> Do not stop `logserver`, `logserver-gui`, or other services managed by the installer before running `./install.sh -u`. The script stops and replaces these services itself; pre-stopping the database (in particular) prevents the installer from completing the upgrade.

a) Stop the collector on the node so no new data is ingested during the upgrade:

```
systemctl stop logserver-probe
```

b) Run the upgrade: `install.sh` stops the services it manages, replaces the packages, and waits for confirmation before restarting them:

```
cd install
./install.sh -u
```

When the script asks whether to restart services, answer **no**: services will be started manually in the correct order.

c) Start the database service first and wait for the node to rejoin the cluster:

```
systemctl start logserver
```

Wait until the node appears in the cluster:

```
curl -u $USER:$PASSWORD "localhost:9200/_cat/nodes?v"
```

d) Wait for cluster health to reach at least **yellow** before moving to the next node:

```
curl -u $USER:$PASSWORD "http://localhost:9200/_cluster/health?pretty"
```

3. Re-enable shard allocation

After **all nodes** have been upgraded:

```
curl -u $USER:$PASSWORD -X PUT \
  "http://localhost:9200/_cluster/settings?pretty" \
  -H 'Content-Type: application/json' -d '{
  "persistent": {
    "cluster.routing.allocation.enable": "all"
  }
}'
```

4. Start remaining services

On each node that runs client services, start services in the following order: the database service (`logserver`) must be running and healthy before starting the remaining services:

```
systemctl start logserver-gui alert cerebro e-doc skimmer \
  intelligence intelligence-scheduler license-service logserver-probe
```

5. Verify

```
curl -u $USER:$PASSWORD "http://localhost:9200/_cluster/health?pretty"
curl -u $USER:$PASSWORD "localhost:9200/_cat/nodes?v"
```

After the upgrade, follow any version-specific post-upgrade steps described in *Version-Specific Upgrades*.

4.3.4 Rollback Procedures

When an upgrade encounters critical issues, a rollback to the previous version may be necessary. Energy Logserver does not provide an automated rollback mechanism. The process requires reinstalling the previous version and restoring configuration from a backup.

Prerequisites

Before every upgrade, the `./install.sh -u` script offers to run `configuration-backup.sh`, which creates a backup archive in `/root/`. This backup contains configuration files, Logserver indices data, templates, and cluster settings. Ensure this backup exists before attempting a rollback.

To verify that a backup archive exists:

```
ls -lt /root/backup-*.tar.gz | head -5
```

Procedure

1. Stop all services

```
systemctl stop logserver logserver-gui logserver-probe alert cerebro \
  e-doc skimmer intelligence intelligence-scheduler license-service
```

2. Reinstall previous version

Use the installer package of the previous version:

```
tar xjf energy-logserver-<previous-version>.x86_64.tar.bz2
cd install
./install.sh -u
```

3. Restore configuration from backup

Extract the backup archive created by `configuration-backup.sh`:

```
mkdir -p /tmp/restore
tar xzf /root/backup-<hostname>-<date>.tar.gz -C /tmp/restore
```

The archive contains the following configuration backups:

Archive file	Restore path
`elasticsearch_conf.tar.gz`	`/etc/elasticsearch/`
`kibana_conf.tar.gz`	`/etc/kibana/`
`logstash_conf.tar.gz`	`/etc/logstash/`
`alert_conf.tar.gz`	`/opt/alert/`
`cerebro_conf.tar.gz`	`/opt/cerebro/`
`skimmer_conf.tar.gz`	`/opt/skimmer/`
`intelligence_conf.tar.gz`	`/opt/intelligence/`
`license_conf.tar.gz`	`/opt/license-service/`
`e-doc_conf.tar.gz`	`/opt/e-doc/`
`logserver-store_conf.tar.gz`	`/etc/logserver-store/`
`wazuh_conf.tar.gz`	`/var/ossec/etc/` (only if Endpoint Security is installed)
`wazuh_gui_conf.tar.gz`	`/usr/share/kibana/data/wazuh/config/` (only if Endpoint Security GUI is installed)
`wiki_conf.tar.gz`	`/opt/wiki/` (only if Wiki.js is installed)

The last three files are produced only when the matching component is installed on the node, so they may be missing from the archive.

Restore only the configuration files that were affected by the upgrade. Each archive can be extracted individually, for example:

```
tar xzf /tmp/restore/<date>/elasticsearch_conf.tar.gz -C /
tar xzf /tmp/restore/<date>/kibana_conf.tar.gz -C /
```

4. Start services

Start the database service first and wait for it to become available:

```
systemctl start logserver
curl -u $USER:$PASSWORD "http://localhost:9200/_cluster/health?pretty"
```

Then start the remaining services:

```
systemctl start logserver-gui alert cerebro e-doc skimmer \
  intelligence intelligence-scheduler license-service logserver-probe
```

5. Verify

```
systemctl status logserver logserver-gui logserver-probe alert cerebro \
  e-doc skimmer intelligence intelligence-scheduler license-service \
  --no-pager
```

4.4 Licensing

Energy Logserver uses file-based licensing. Each license consists of two matching files (`.license` and `.info`) that must be placed in the installation directory during initial setup or updated together via the GUI on a running system.

- *License Plans*
- *License Files*
- *Initial License Installation*
- *Viewing License Information*
- *Updating the License*
- *Updating the License via CLI*
- *License Service*
- *License Renewal*
- *License Compatibility Across Versions*

4.4.1 License Plans

Energy Logserver is available in the following plans:

- **Log Management Plan (LMP)**: base platform for centralized log management and operational monitoring
- **SIEM Plan**: add-on package extending LMP with SIEM capabilities (requires additional license)
- **Network Probe**: add-on package for network traffic analysis, available in configurable node count (requires additional license)
- **AI Assistant**: add-on enabling LLM-based features in Discover (requires additional license)
- **MSSP**: mode for managed service providers enabling multi-tenant source management (requires additional license)

4.4.2 License Files

Each Energy Logserver license is delivered as two matching files that share the same `es_<timestamp>` prefix:

File	Purpose
`es_<timestamp>.license`	Encrypted license payload that the platform validates at runtime
`es_<timestamp>.info`	Metadata used by `install.sh` and the GUI to display license details (company, expiration date, enabled plans, limits)

Both files must be supplied together. Each file is at most 1024 bytes. After installation, the active files live in `/usr/share/logserver/license/`, owned by the Data Node service user, `logserver` on current releases.

4.4.3 Initial License Installation

During installation, copy both license files to the `install` directory before running the setup script:

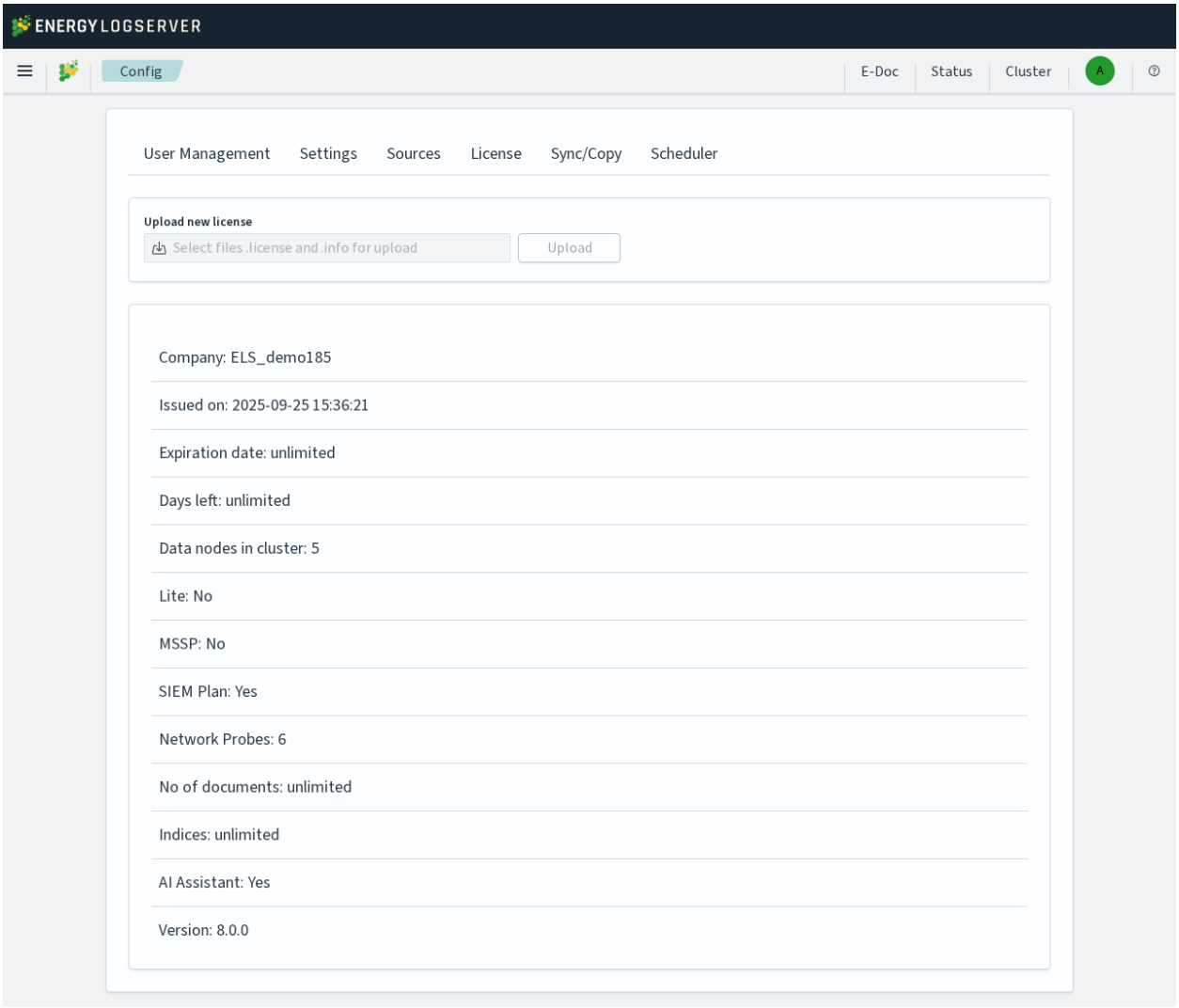
```
cp es_*.license es_*.info install/
cd install
./install.sh -i
```

Note

Both `.license` and `.info` files must be present before running `install.sh`. The installer aborts with `I cannot find the license file` if either file is missing. This applies to interactive (`-i`) and non-interactive (`-i -n`) installations.

4.4.4 Viewing License Information

License details are available in the GUI under **Config** → **License**.



The License tab displays the following read-only fields:

Field	Description
Company	Organization that owns the license
Issued on	License issue date
Expiration date	License validity period (`unlimited` for perpetual licenses)
Days left	Remaining days until expiration
Data nodes in cluster	Maximum number of ELS Data Nodes allowed in the cluster
Lite	Whether the SIEM Lite variant is active
MSSP	Whether MSSP mode is licensed
MSSP Sources	Number of licensed MSSP sources (shown when MSSP mode is enabled)
SIEM Plan	Whether the SIEM add-on is active
Network Probes	Number of licensed Network Probe instances
No of documents	Maximum document count (`unlimited` for unrestricted licenses)
Indices	Maximum number of indices (`unlimited` for unrestricted licenses)
AI Assistant	Whether the AI Assistant add-on is active
Version	Maximum supported platform version

4.4.5 Updating the License

To upload a new or renewed license file on a running system:

1. Navigate to **Config** → **License** tab.
2. In the **Upload new license** section, select both the `.license` and `.info` files. The files must match the naming pattern `es_<timestamp>.[license|info]` and must not exceed 1024 bytes each.
3. Click **Upload**. The server validates the files and returns a preview of the new license.
4. Review the side-by-side comparison of the current and new license parameters:
   - Fields highlighted in **green** indicate upgraded parameters.
   - Fields highlighted in **red** indicate downgraded parameters. If any parameter is downgraded, a red warning banner is displayed: *"The new license is more restrictive than the old one."*
5. Click **Submit** to deploy the new license, or **Abort** to cancel.
6. A notification *"New license submitted successfully"* confirms the deployment. Verify the updated fields in the License tab.

> **Note**
>
> The ELS Data Node and the ELS GUI apply the new license without a restart.
>
> On a deployment that runs the `Network Probe` with MSSP mode, restart the probes after the upload. Each probe reads the MSSP state from the license when its pipelines start and keeps that value until it is restarted. See *Applying a New License*.

4.4.6 Updating the License via CLI

To replace the license files directly on the filesystem (e.g. in environments without GUI access):

1. Back up the current license files:

```
cp /usr/share/logserver/license/es_*.license /tmp/license_backup/
cp /usr/share/logserver/license/es_*.info /tmp/license_backup/
```

2. Copy the new license files to the license directory (repeat on every ELS Data Node in the cluster):

```
cp es_*.license es_*.info /usr/share/logserver/license/
```

3. Set correct file ownership. Inherit the owner already set on the license directory instead of hardcoding it:

```
chown --reference=/usr/share/logserver/license/ /usr/share/logserver/license/
↪es_*.*
chmod 640 /usr/share/logserver/license/es_*.*
```

To set the owner explicitly instead, check the current value first with `stat -c '%U:%G' /usr/share/logserver/license/` and reuse it (for example `chown logserver:logserver /usr/share/logserver/license/es_*.*`).

> **Warning**
>
> Without correct file ownership, the ELS Data Node service cannot read the license files and will fail to start or reload.

4. Reload or restart the ELS Data Node service:

Option A, Hot reload (recommended for production, no downtime):

```
curl -sk -u "$USER:$PASSWORD" -X POST "https://localhost:9200/_logserver/
↪license/reload"
```

Note

`$USER:$PASSWORD` are the ELS Data Node credentials. On a host that runs the Data Node without the ELS GUI, for example an Energy SOAR appliance, read them from the Energy SOAR configuration:

```
grep "http.auth.basic" /etc/energysoar/application.conf.d/database.conf
```

On a cluster without TLS, use `http://localhost:9200` and drop the `-k` flag. Do not wait for the periodic automatic re-read to pick up the renewed license. The reload endpoint forces it immediately.

Option B, Full service restart:

```
systemctl restart logserver
```

5. Verify the updated license in **Config** → **License**.
6. On a deployment that runs the `Network Probe` with MSSP mode, restart the probes so they read the new license, as described in *Applying a New License*.

Warning

Expired License, Emergency License Upload

When the license expires, access to both the GUI and the REST API is blocked and normal login is not possible. To upload a new license without logging in, navigate directly to:

```
https://<ui-ip>/license_upload
```

This URL remains accessible regardless of license status and allows uploading new `.license` and `.info` files to restore system access.

This page is served by the ELS GUI. On deployments where the ELS GUI is not present on the Data Node host, for example an Energy SOAR appliance that embeds the ELS Data Node, `/license_upload` is unavailable; use the CLI method above instead.

After uploading the new license, verify that all services have reloaded it:

```
POST /_logserver/license/reload
```

The reload command must be executed from an SSH console, not from the GUI.

Alternatively, check the license status page in the GUI after regaining access. To avoid service interruption, it is recommended to renew the license before it expires.

4.4.7 License Service

Energy Logserver includes a dedicated **license-service** component that validates add-on licenses at runtime. It is required when the Network Probe or SIEM Plan add-ons are licensed.

Property	Value
Installation path	`/opt/license-service/`
Configuration file	`/opt/license-service/license-service.conf`
Service name	`license-service`

Start or restart the service:

```
systemctl start license-service
systemctl status license-service
```

The license-service validates the add-ons configured in `license-service.conf`. Each add-on has its own section, and both can be present at once:

- **`network_probe`**: validates the Network Probe license and manages probe registration and services
- **`siem`**: validates the SIEM Plan license and manages the related services

Configure the section for each add-on you have licensed.

Key configuration directives in `license-service.conf`:

Directive	Description
`network_probe`	Section enabling Network Probe license validation and probe service/file management
`siem`	Section enabling SIEM Plan license validation and service/file management
`logserver_connection.hosts`	ELS Data Node connection endpoints (default: `127.0.0.1:9200`)
`logserver_connection.username`	Authentication username for the ELS Data Node
`logserver_connection.password`	Authentication password for the ELS Data Node
`logserver_connection.https`	Enable HTTPS communication with the ELS Data Node
`http_server.https_enabled`	Enable HTTPS for the license-service API
`status_refreshing_interv`	Interval for refreshing managed services' status, set per section under `services_management` (default: `60`)
`host_ip`	Overrides the auto-detected IP address under which the probe registers (8.1.0 and later)
`host_name`	Overrides the auto-detected host name shown for the probe in **Network Probe** (8.1.0 and later)

Set `host_ip` and `host_name` at the top level of the file and restart `license-service`. The probe keeps its registration and only its address or name changes. After an upgrade from 8.0, the commented examples of both settings are in `license-service.conf.rpmnew`.

Note

The license-service is installed automatically as part of the Network Probe or SIEM Plan setup. It does not need to be installed separately.

4.4.8 License Renewal

Contact your vendor support to obtain a renewed license file. Apply it using the upload procedure described above.

4.4.9 License Compatibility Across Versions

Licenses are forward- and backward-compatible across Energy Logserver releases starting from version **7.4.0**. An existing license issued for an earlier 7.x version continues to work after upgrading to 8.x, no license replacement is required to complete the upgrade itself.

A new license is only needed when enabling **add-on features introduced in a later release** (for example, the `AI Assistant` key added with 7.9 / 8.0). In that case, perform the upgrade first, then upload the new license through the GUI as described in *Updating the License*. Uploading a newer license to an older version via the CLI is also supported, but the GUI upload form on pre-8.0 versions may reject licenses containing new keys, upgrade first, then upload.

4.5 User Interface

Comprehensive data exploration, visualization creation, and interactive dashboard management for security operations.

The User Interface module provides intuitive tools for security analysts and administrators to discover, visualize, and monitor security data. From powerful search capabilities and flexible data filtering to creating custom visualizations and building comprehensive security dashboards, this module serves as the primary interaction layer for security operations and threat analysis.

4.5.1 Overview

Energy Logserver's user interface transforms complex security data into actionable visual intelligence. Key features include:

- **Discover and Search**: Advanced search interface with Lucene query syntax, time-based filtering, and data export capabilities
- **Flexible Visualizations**: Create custom charts, graphs, and visual representations optimized for security use cases
- **Interactive Dashboards**: Build and manage multi-panel dashboards combining multiple data sources and visualizations
- **Pre-built Templates**: Leverage ready-to-use security dashboards and visualization templates
- **Real-time Updates**: Dynamic data refresh for monitoring live security events

4.5.2 In This Section

This chapter covers the complete user interface capabilities, from basic data discovery to advanced dashboard creation.

Discover

- *Index Pattern Configuration*
  - *First Time Setup*
  - *Index Selection*
  - *Time settings and refresh*

- *Fields*
- *Filtering and syntax building*
- *Saving and deleting queries*
- *Manual incident*
- *Quick Data Export Wizard*
- *AI Assistant*

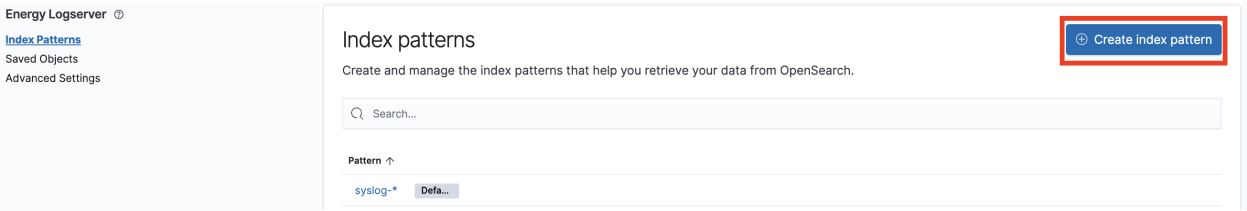
Index Pattern Configuration

Before you can explore data in Discover, you must configure an index pattern.

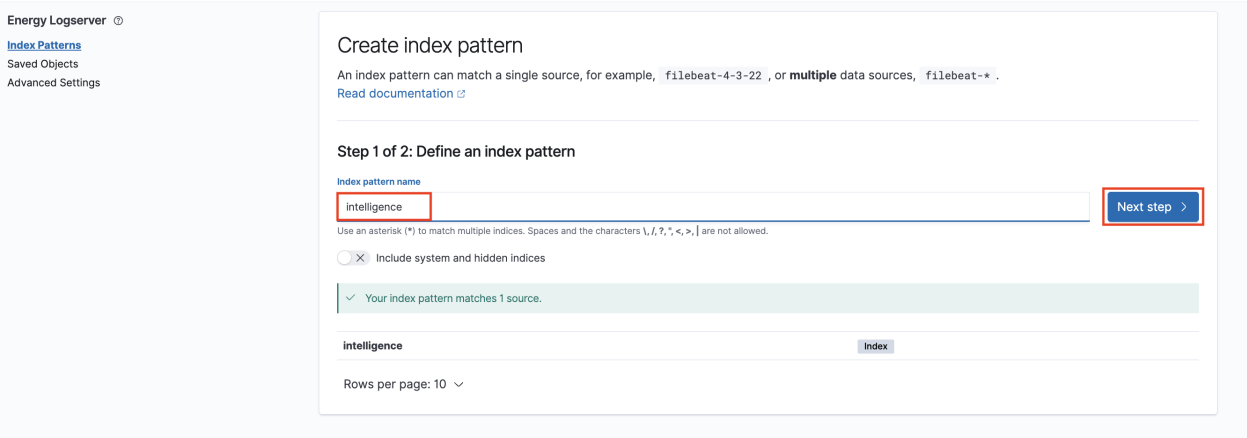
First Time Setup

If you log in to Energy Logserver for the first time, you must specify the Index to be searched. You can enter the name of your index, indicate a specific index from a given day, or use the asterisk (*) to match all indexes matching a specific index pattern.

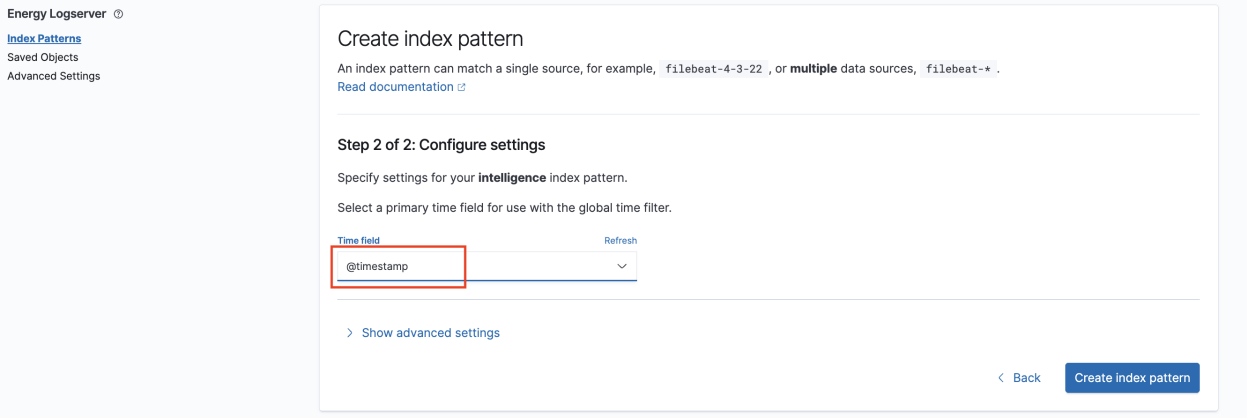
After logging in to the application click the button "Set up index pattern" to add a new index pattern:



In the "Index pattern" field enter the name of the index or index pattern (after confirming that the index or sets of indexes exist) and click the "Next step" button.



In the next step, from a drop-down menu select the "Time filter field name", after which individual events should be sorted. By default the *timestamp* is set, which is the time of occurrence of the event. Depending on your preferences, it may also be the time of indexing or another field available in the event.

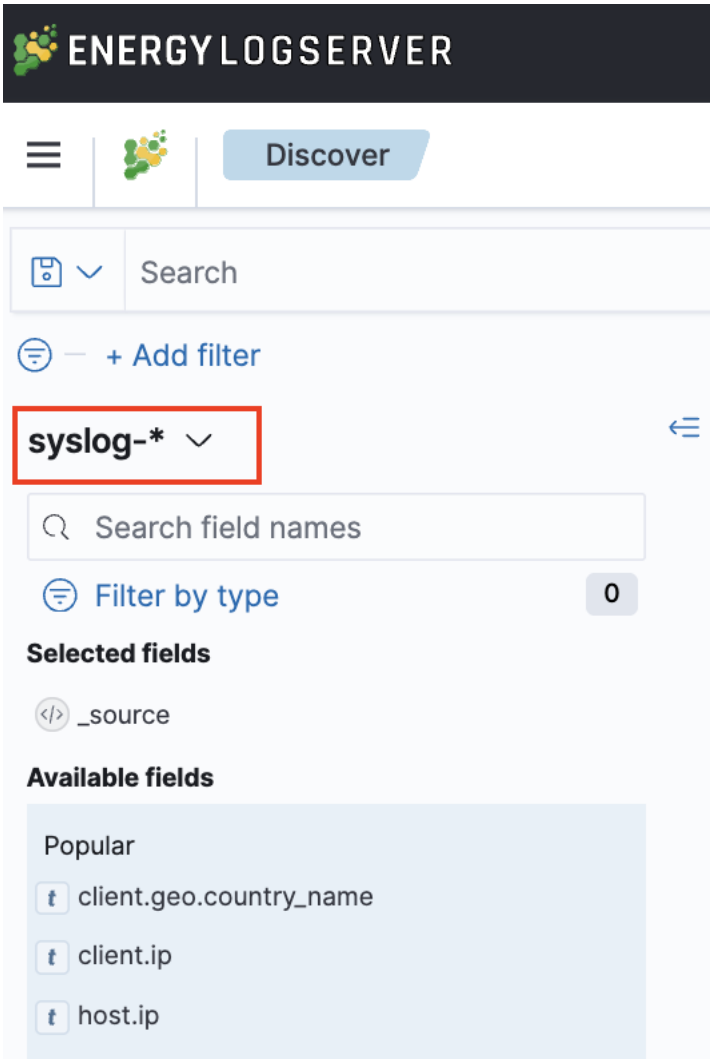


At any time, you can add more indexes or index patterns by going to the main tab selecting “Management” and next selecting “Index Patterns”.

Index Selection

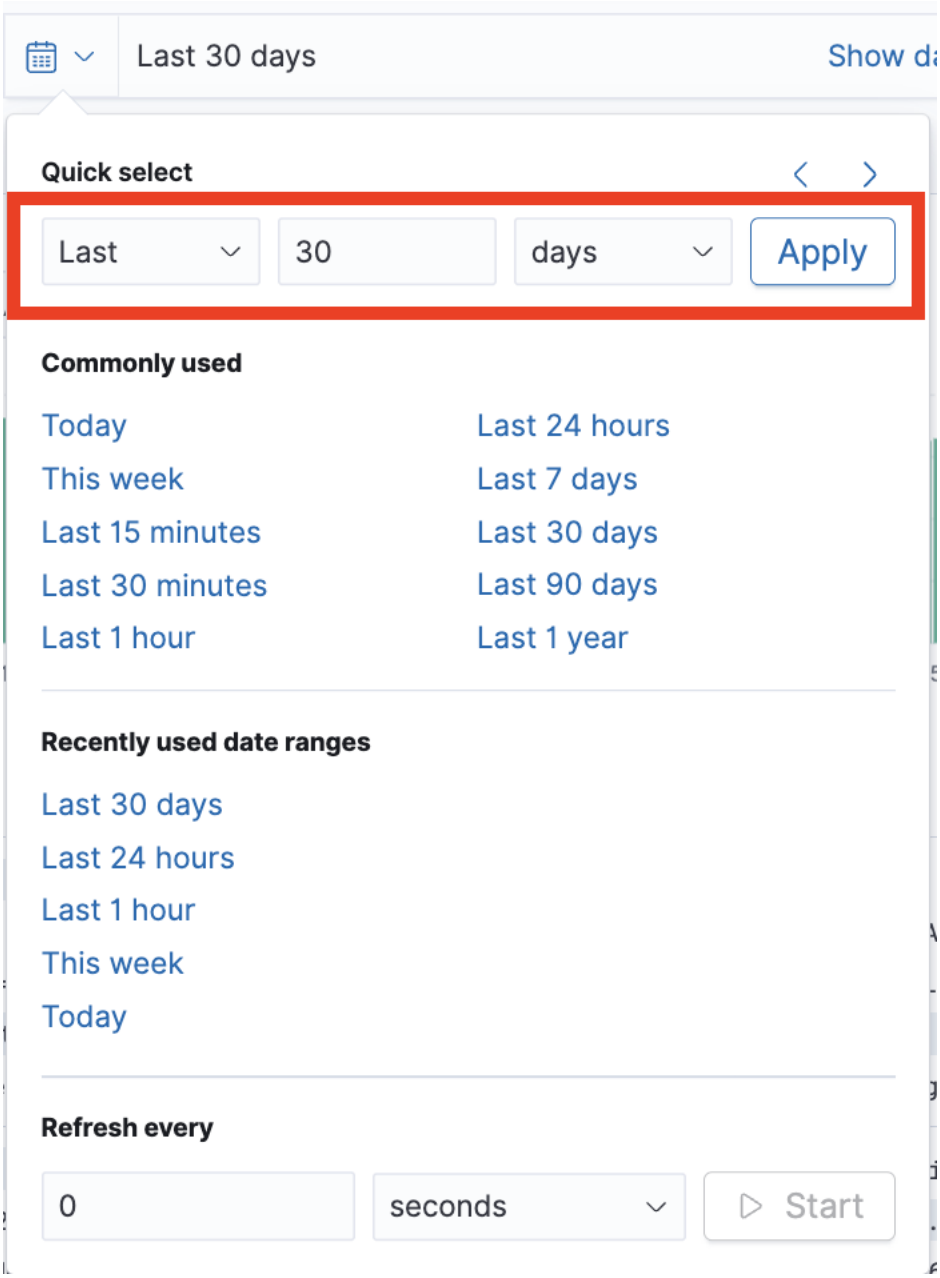
After logging into Energy Logserver, you will be taken to the “Discover” tab, where you can interactively explore your data. You have access to every document in every index that matches the selected index patterns.

If you want to change the selected index, click the drop-down menu with the name of the current index pattern in the left panel. Clicking on an entry from the expanded list will change the searched index.



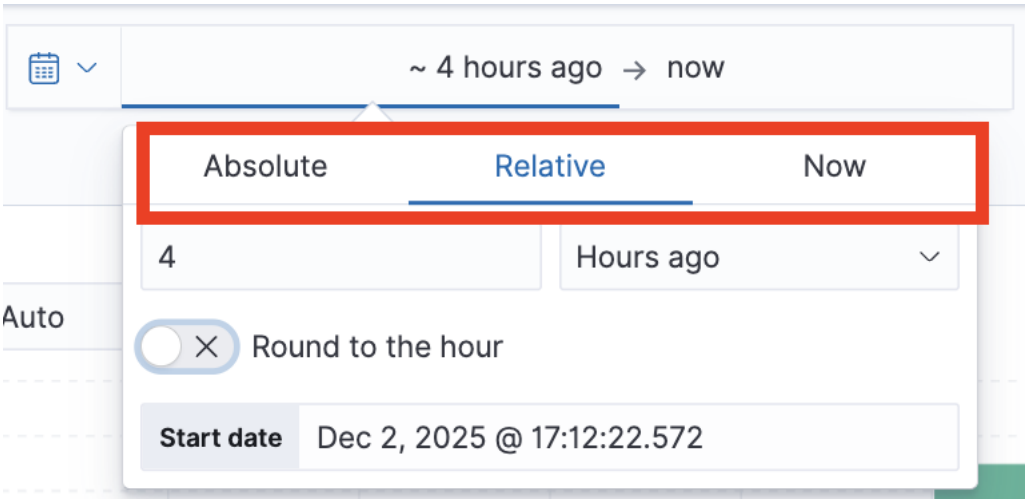
Time settings and refresh

In the upper right corner, there is a section that defines the range of time that Energy Logserver will search in terms of conditions contained in the search bar. The default value is the last 15 minutes.



After clicking this selection, you can adjust the scope of the search by selecting one of the three tabs in the drop-down window:

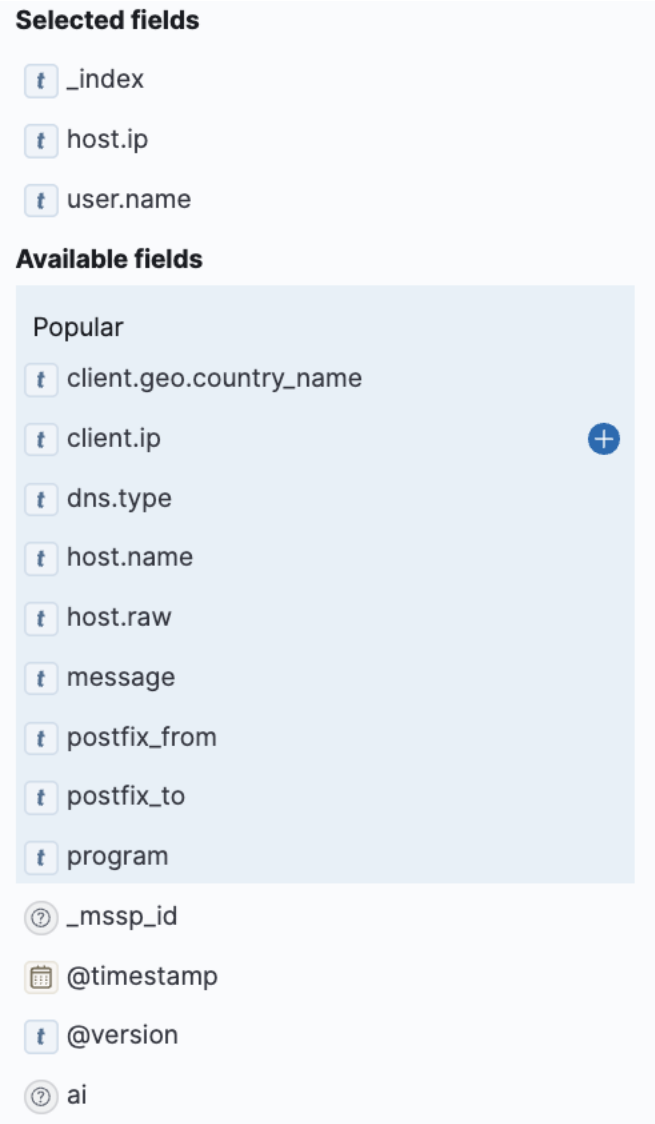
- **Absolute**: using two calendars you define the exact start and end time for which the search results are to be returned.
- **Relative**: in this window specify a relative time offset (e.g., 4 hours ago) from which Energy Logserver should search for data. You can also enable the "Round to the hour" option for approximate time ranges.
- **Now**: sets the time boundary to the current moment.



Fields

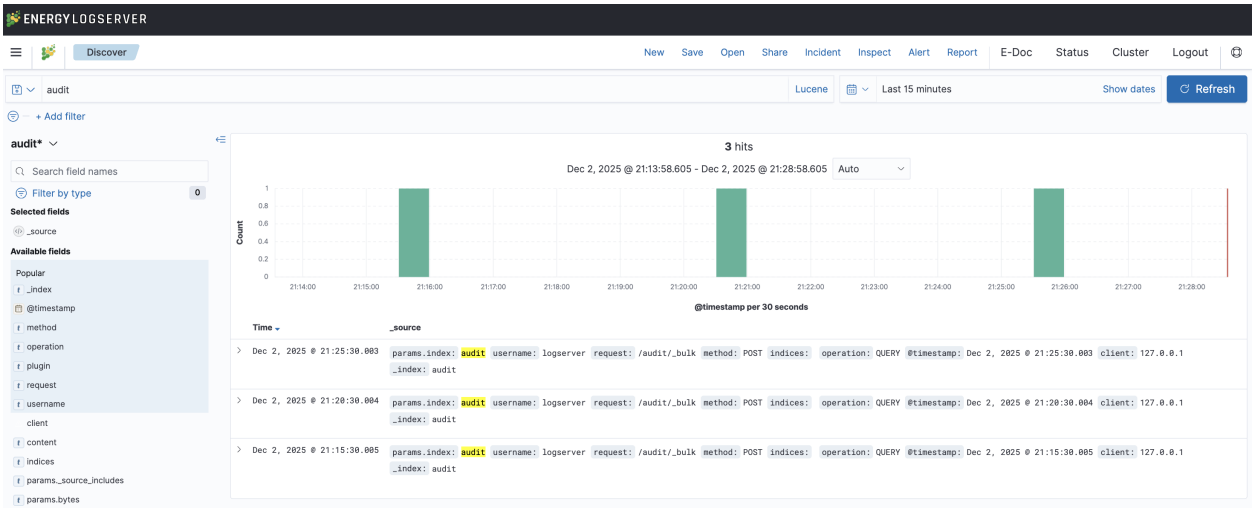
Energy Logserver recognizes fields in the body of searched events that can be used to create more precise queries. The extracted fields are visible in the left panel. They are divided into three types: timestamp, marked on the clock icon; text, marked with the letter "t" t params.level, and digital, marked with a hashtag # _score.

Point to a field and click the icon to transfer it to the "Selected Fields" column. In the place of events, a table with the selected columns is created. In the "Selected Fields" selection you can also delete specific fields from the table by clicking on the selected element.



Filtering and syntax building

Use the query bar to search for interesting events. For example, after entering the word “error”, all events that contain the word will be displayed, additionally highlighting them with a yellow background.



Syntax

Fields can be used similarly by defining conditions that interest you. The syntax of such queries is:

```
fields_name:<fields_value>
```

Example:

```
status:500
```

This query will display all events that contain the "status" fields with a value of 500.

Filters

The field value does not have to be a single, specific value. For digital fields you can specify a range in the following scheme:

```
fields_name:[<range_from> TO <range_to>]
```

Example:

```
status:[500 TO 599]
```

This query will return events with status fields that are in the range 500 to 599.

Operators

The search language used in Energy Logserver allows you to use logical operators "AND", "OR" and "NOT", which are key and necessary to build more complex queries.

- **AND** is used to combine expressions, e.g. `error AND "access denied"`. If an event contains only one expression or the word `error` and `denied` but not the word access, then it will not be displayed.
- **OR** is used to search for the events that contain one OR other expression, e.g. `status:500 OR denied`. This query will display events that contain the word "denied" or a status field value of 500. Energy Logserver uses this operator by default, so query `"status:500" "denied"` would return the same results.
- **NOT** is used to exclude the following expression e.g. "status:[500 TO 599] NOT status:505" will display all events that have a status field, and the value of the field is between 500 and 599 but will eliminate from the result events whose status field value is exactly 505.

- **The above methods** can be combined by building even more complex queries. Understanding how they work and combining them is the basis for effective searching and full use of Energy Logserver.

  Example of query built from connected logical operations:

```
status:[500 TO 599] AND ("access denied" OR error) NOT status:505
```

Returns in the results all events for which the value of status fields are in the range of 500 to 599, simultaneously contain the word "access denied" or "error", omitting those events for which the status field value is 505.

Wildcards

Wildcard searches can be run on individual terms, using ? to replace a single character, and * to replace zero or more characters:

```
qu?ck bro*
```

Be aware that wildcard queries can use an enormous amount of memory and perform very badly, just think how many terms need to be queried to match the query string "a* b* c*".

Regular expressions

Regular expression patterns can be embedded in the query string by wrapping them in forward-slashes ("/"):

```
name:/joh?n(ath[oa]n)/
```

Fuzziness

You can run fuzzy queries using the ~ operator:

```
quikc~ brwn~ fox~
```

For these queries, the query string is normalized. If present, only certain filters from the analyzer are applied.

The query uses the Damerau-Levenshtein distance to find all terms with a maximum of two changes, where a change is the insertion, deletion, or substitution of a single character or transposition of two adjacent characters.

The default edit distance is 2, but an edit distance of 1 should be sufficient to catch 80% of all human misspellings. It can be specified as:

```
quikc~1
```

Proximity searches

While a phrase query (e.g. "john smith") expects all of the terms in the same order, a proximity query allows the specified words to be further apart or in a different order. In the same way that fuzzy queries can specify a maximum edit distance for characters in a word, a proximity search allows us to specify a maximum edit distance of words in a phrase:

```
"fox quick"~5
```

The closer the text in a field is to the original order specified in the query string, the more relevant that document is considered to be. When compared to the above example query, the phrase "quick fox" would be considered more relevant than "quick brown fox".

Ranges

Ranges can be specified for date, numeric, or string fields. Inclusive ranges are specified with square brackets [min TO max] and exclusive ranges with curly brackets {min TO max}.

- All days in 2012:

  ```
  date:[2012-01-01 TO 2012-12-31]
  ```

- Numbers 1...5

  ```
  count:[1 TO 5]
  ```

- Tags between alpha and omega, excluding alpha and omega:

  ```
  tag:{alpha TO omega}
  ```

- Numbers from 10 upwards

  ```
  count:[10 TO *]
  ```

- Dates before 2012

  ```
  date:{* TO 2012-01-01}
  ```

Curly and square brackets can be combined:

- Numbers from 1 up to but not including 5

  ```
  count:[1 TO 5}
  ```

- Ranges with one side unbounded can use the following syntax:

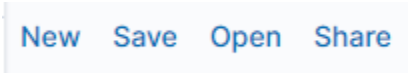
```
age:>10
age:>=10
age:<10
age:<=10
```

Saving and deleting queries

Saving queries enables you to reload and use them in the future.

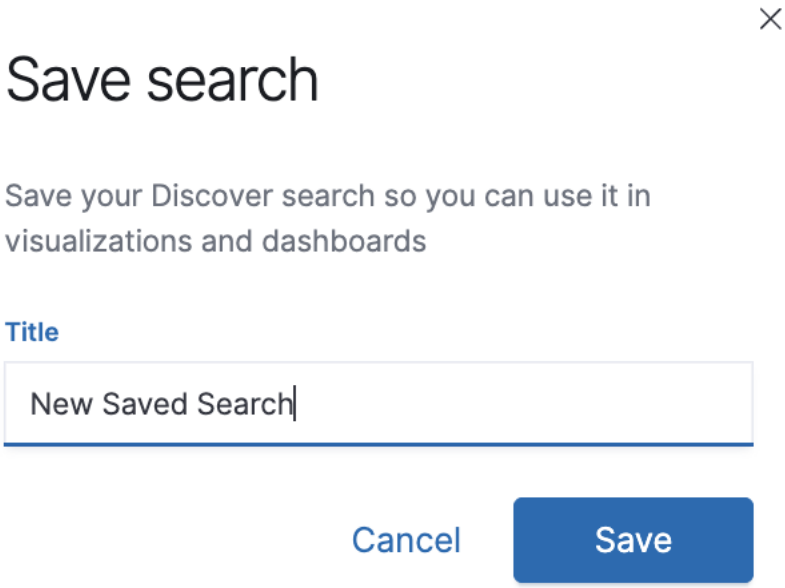
Save query

To save the query, click on the "Save" button under the query bar:



This will bring up a window in which you give the query a name and then click the button 

.



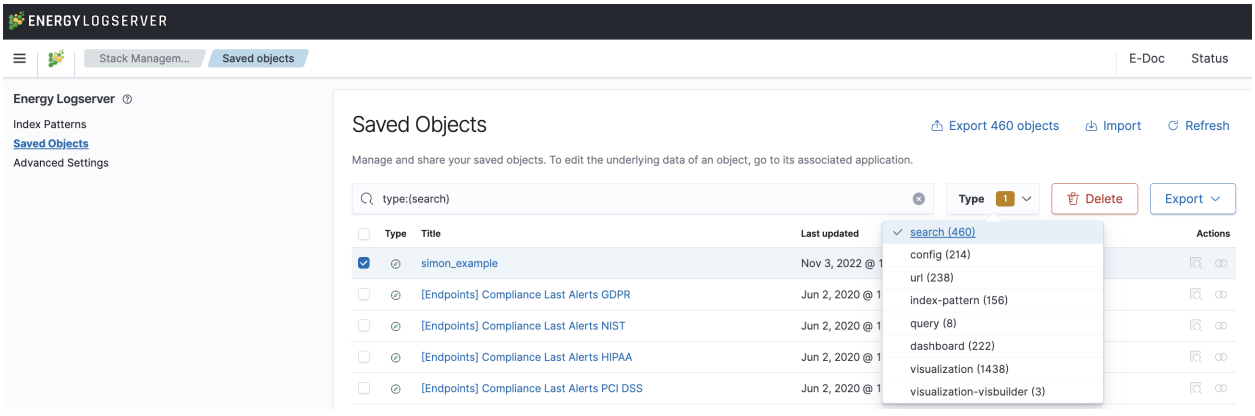
Saved queries can be managed by clicking the "Manage searches" button, which redirects to the Saved Objects section in Management where you can view, edit, export, and delete all saved searches.

Delete query

To delete a saved query, open it from the search list, and then click on the button 

.

If you want to delete many saved queries simultaneously, go to **Management** → **Saved Objects** and filter by type "Searches". Select the queries in the list (the checkbox to the left of the query name), and then click the "Delete" button.



From this level, you can also export saved queries in the same way. To do this, you need to click on 

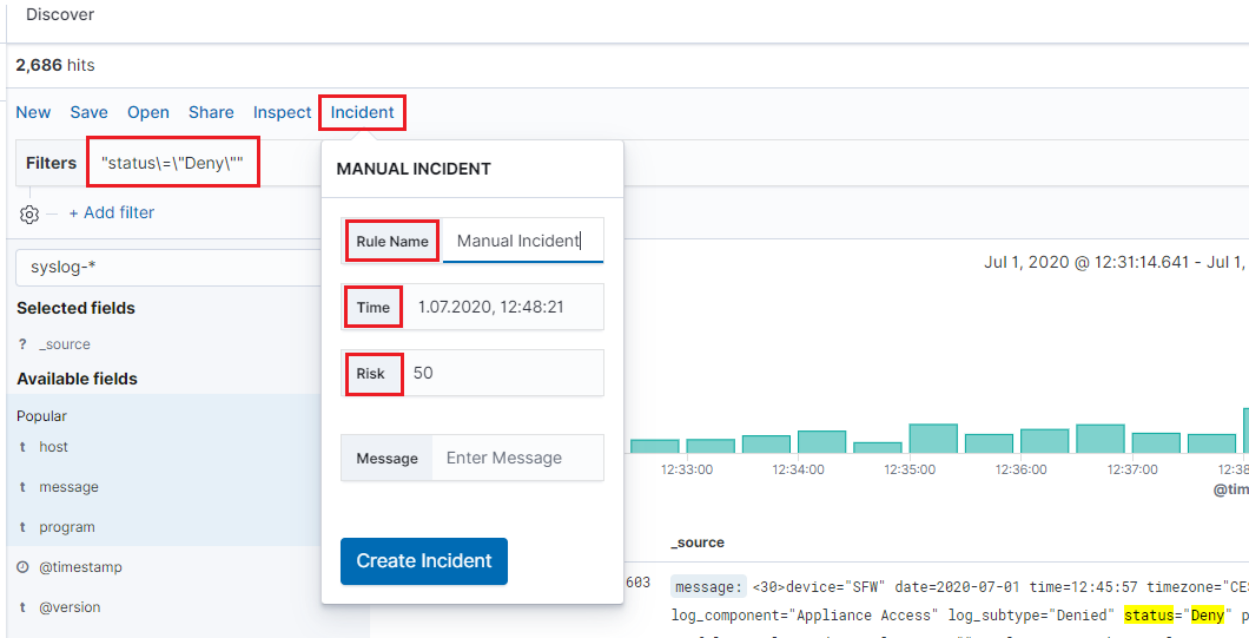
and choose the save location. The file will be saved in .json format. If you then want to import such a file to Energy Logserver, click on a button 

, at the top of the page and select the desired file.

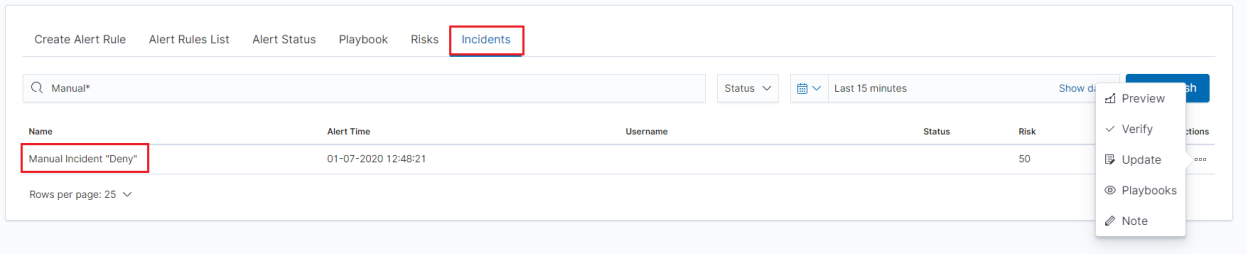
Manual incident

The `Discover` module allows you to manually create incidents that are saved in the `Incidents` tab of the `Alerts` module. Manual incidents are based on search results or filtering. For a manual incident, you can save the following parameters:

- Rule name
- Time
- Risk
- Message



After saving the manual incident, you can go to the Incident tab in the Alert module to perform the incident handling procedure.



Quick Data Export Wizard

The Quick Export wizard allows you to download the currently viewed data frame to your local machine. The data is processed into a `CSV` or `HTML` file.

To use the Quick Export Wizard:

1. Select the `Report` tab from the top bar and go to `Quick` tab.
2. Type the name of the task in the special field, which will also serve as the file name.
3. Select the file extension that you are interested in (`HTML` or `CSV`).

4. Press the `Export` button and wait for the file to be ready to download.
5. When the `Download` button appears, click on it to download the file to your local machine.

AI Assistant

The AI Assistant provides LLM-based analysis capabilities in two modes: Document Analysis for contextual log inspection and Knowledge Chat for general queries.

> **Note**
>
> Access to AI Assistant requires the `ai_assistant` capability assigned to the user role.

Document Analysis

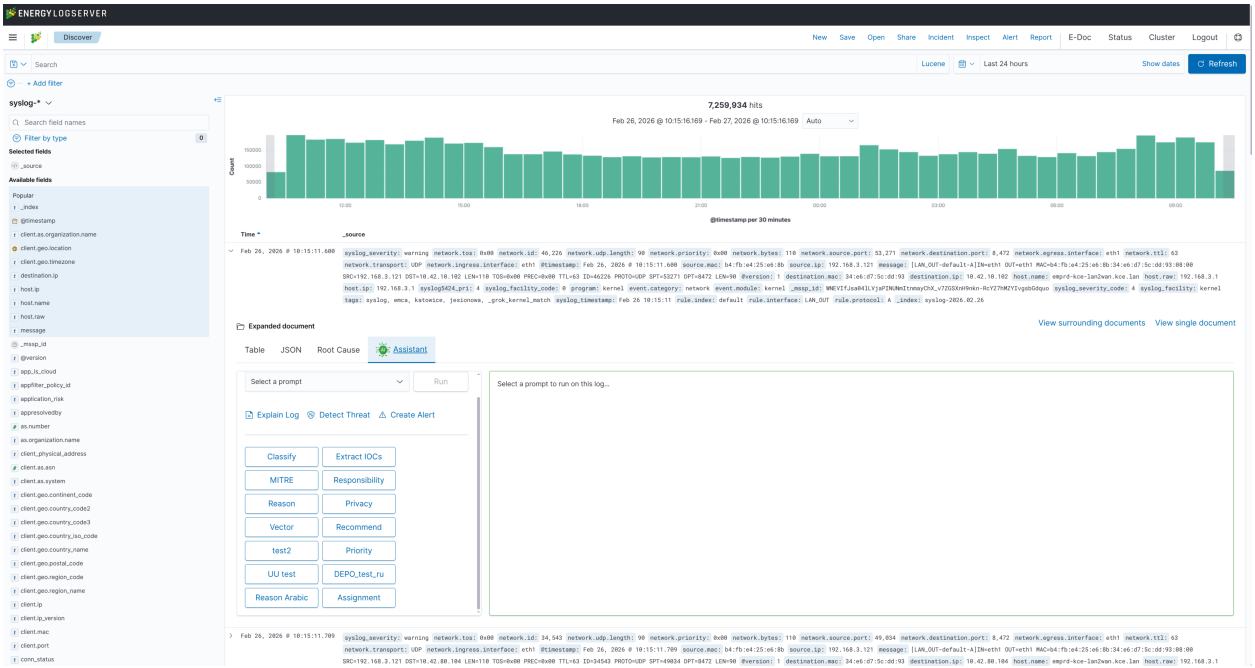
When viewing an expanded document in Discover, the **Assistant** tab in the document viewer toolbar provides contextual analysis of the selected log entry. The expanded document view contains three tabs:

- **Table**: standard field-value display
- **JSON**: raw document in JSON format
- **Assistant**: AI-powered log analysis

In the Assistant tab, select a provider from the drop-down menu and use the available prompt buttons to analyze the current log entry. Prompts are displayed as buttons (e.g., Classify, Extract IOCs, MITRE, Priority, Responsibility) or as select fields, depending on their configuration. The response from the LLM is displayed in the panel on the right.

Additional actions are available via the links above the prompt area:

- **Cluster Log**: group similar log entries
- **Select Preset**: apply a predefined prompt set
- **Create Alert**: create an alert rule based on the current event



Knowledge Chat

The Knowledge Chat is a floating chat panel available globally across the application. It is not tied to a specific document. Use it for general queries, troubleshooting guidance, or knowledge base lookups. Open it by clicking the AI Assistant floating button visible in the bottom corner of the interface.

> **Note**
>
> For AI Assistant configuration (providers, prompts, AI on Prem connection), see *AI Assistant*.

Visualizations

Visualize enables you to create visualizations of the data in your Energy Logserver indices. You can then build dashboards that display related visualizations. Visualizations are based on Energy Logserver queries. By using a series of Energy Logserver aggregations to extract and process your data, you can create charts that show you the trends, spikes, and dips.

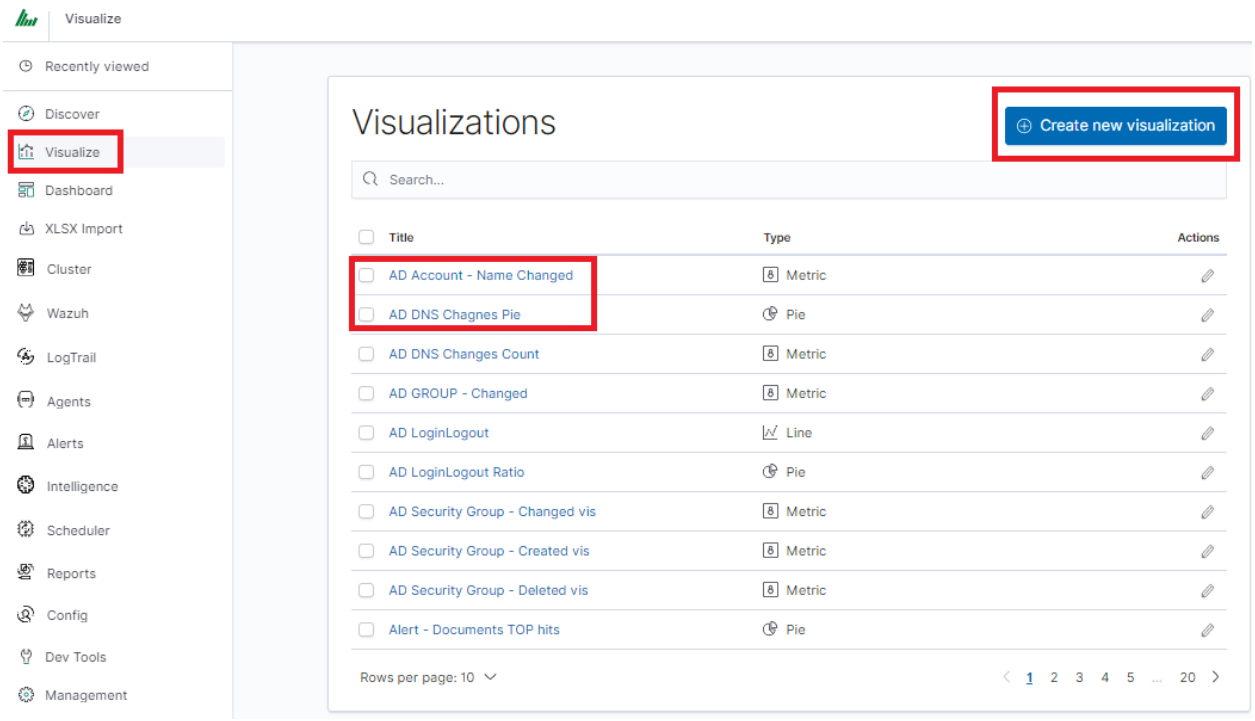
Creating visualization

Create

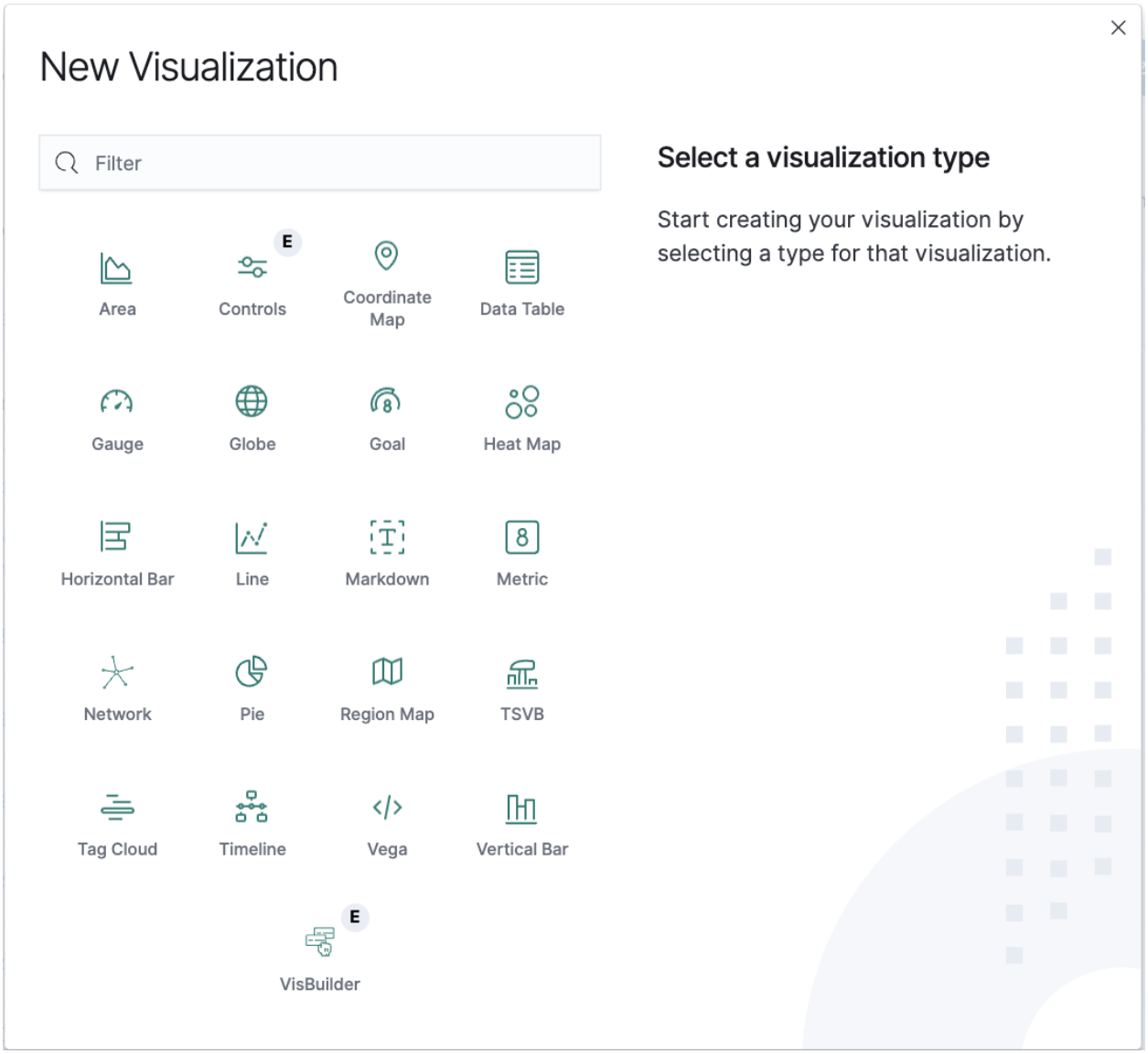
To create a visualization, go to the "Visualize" tab from the main menu. A new page will appear where you can create or load visualization.

Load

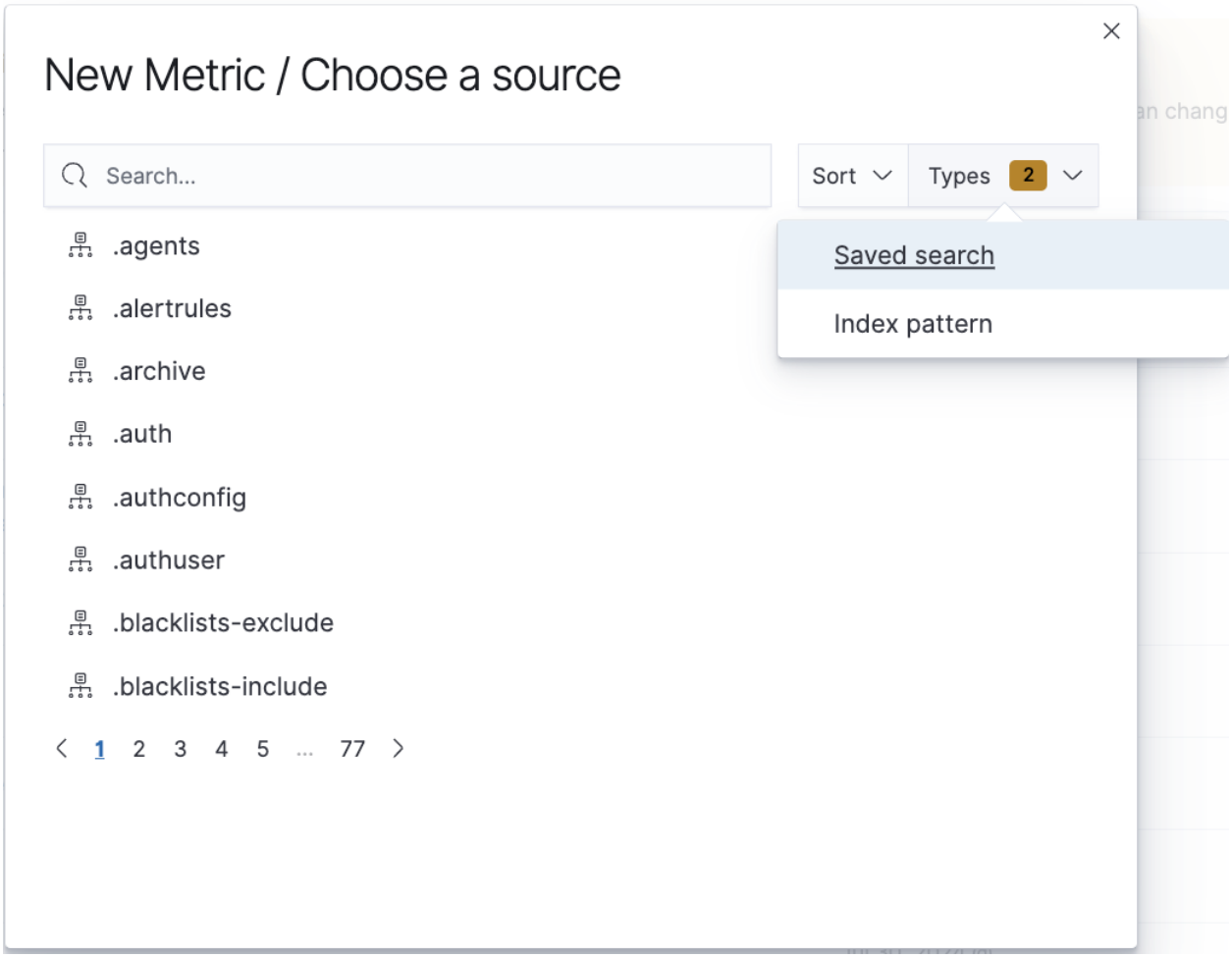
To load previously created and saved visualization, you must select it from the list.



To create a new visualization, you should choose the preferred method of data presentation.



Next, specify whether the created visualization will be based on a new or previously saved query. If on a new one, select the index whose visualization should concern. If visualization is created from a saved query, you just need to select the appropriate query from the list, or (if there are many saved searches) search for them by name.

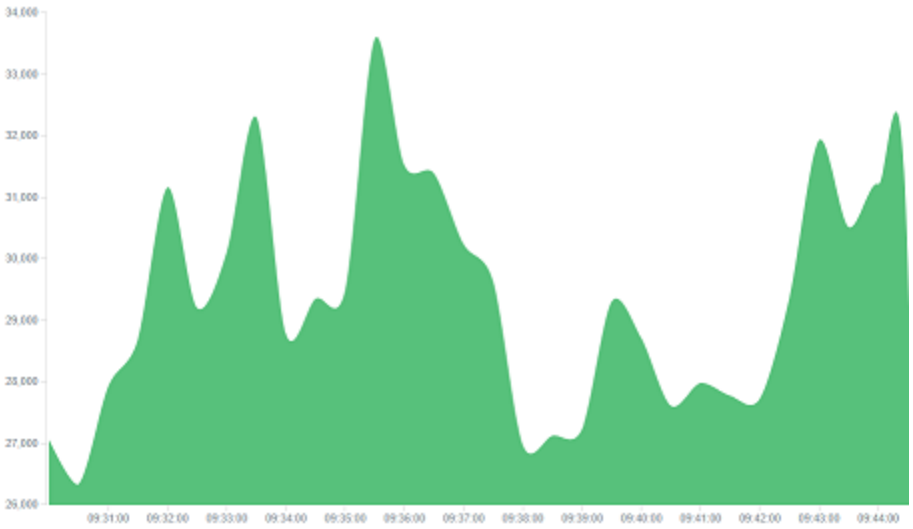


Visualization types

Before the data visualization will be created, first you have to choose the presentation method from an existing list. Currently, there are five groups of visualization types. Each of them serves different purposes. If you want to see only the current number of products sold, it is best to choose "Metric", which presents one value.



However, if we would like to see user activity trends on pages at different hours and days, a better choice will be the "Area chart", which displays a chart with time division.



The "Globe" visualization displays geolocation data on a 3D globe. It maps events to countries or regions using ISO country codes derived from GeoIP fields. Use this visualization to identify the geographic distribution of network traffic, login attempts, or other location-based events. This visualization is available only with the **SIEM Plan** license.

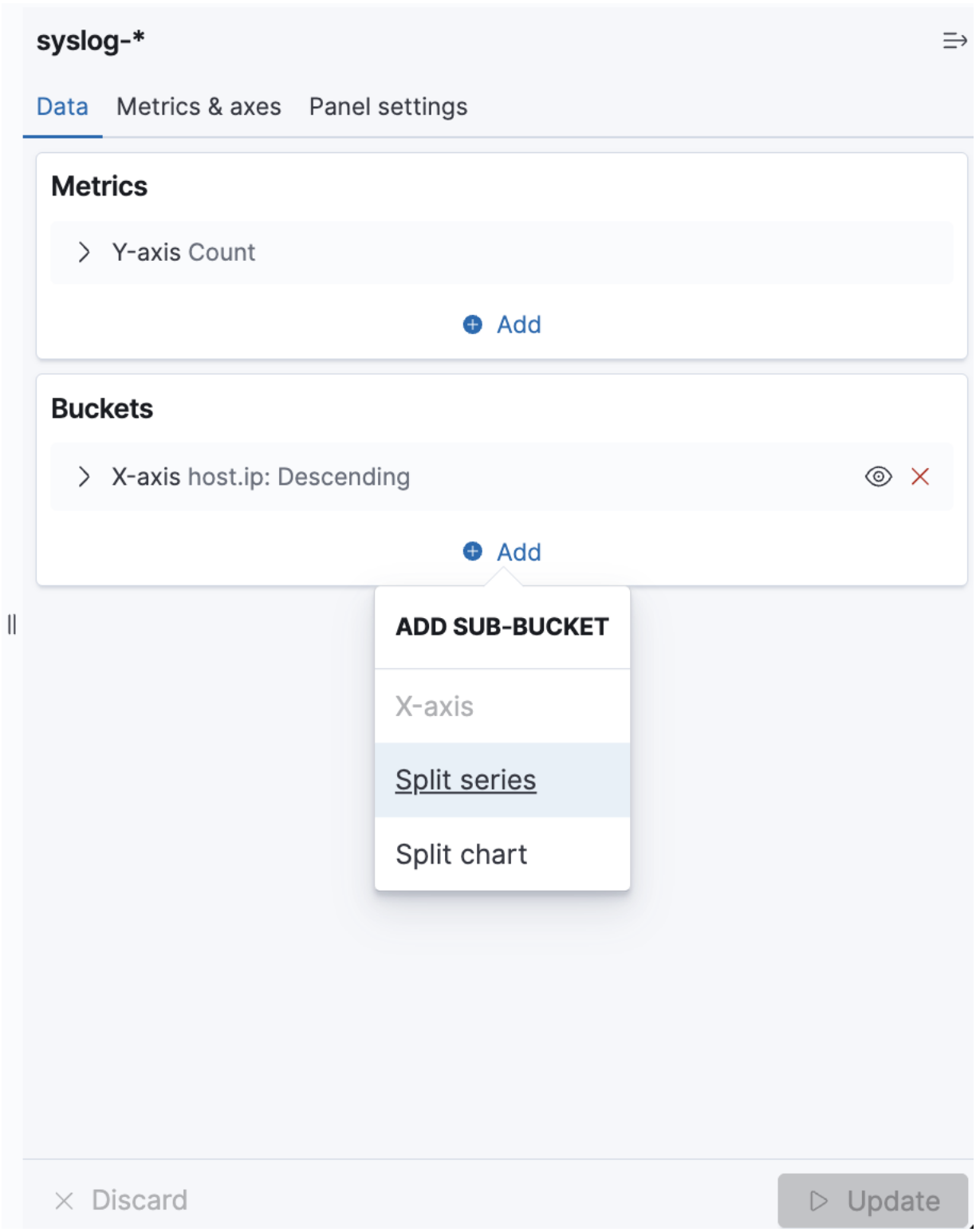
The "Network" visualization draws the data as a graph: two aggregations define the nodes, a third defines the edges between them, and metrics control node and edge size. Node shapes, colors, arrows, labels, and the physics of the layout are set in the visualization options. A node can also link through to the matching asset in *CMDB*. Use it for relationship views such as source to destination traffic, user to host activity, or process trees.

The "Markdown widget" view is used to place text e.g. information about the dashboard, explanations, and instructions on how to navigate. Markdown language was used to format the text (the most popular use is GitHub). More information and instructions can be found at this link: https://docs.github.com/en/get-started/writing-on-github

Edit visualization and saving

Editing

Editing a saved visualization enables you to directly modify the object definition. You can change the object title, add a description, and modify the JSON that defines the object properties. After selecting the index and the method of data presentation, you can enter the editing mode. This will open a new window with an empty visualization.



At the very top, there is a bar of queries that can be edited throughout the creation of the visualization. It works in the same way as in the "Discover" tab, which means searching the raw data, but instead of the data being displayed, the visualization will be edited. The following example will be based on the "Area chart". The visualization modification panel on the left is divided into three tabs: "Data", "Metric & Axes" and "Panel Settings".

In the "Data" tab, you can modify the elements responsible for which data and how should be presented. In this tab,

there are two sectors: “metrics”, in which we set what data should be displayed, and “buckets” in which we specify how they should be presented.

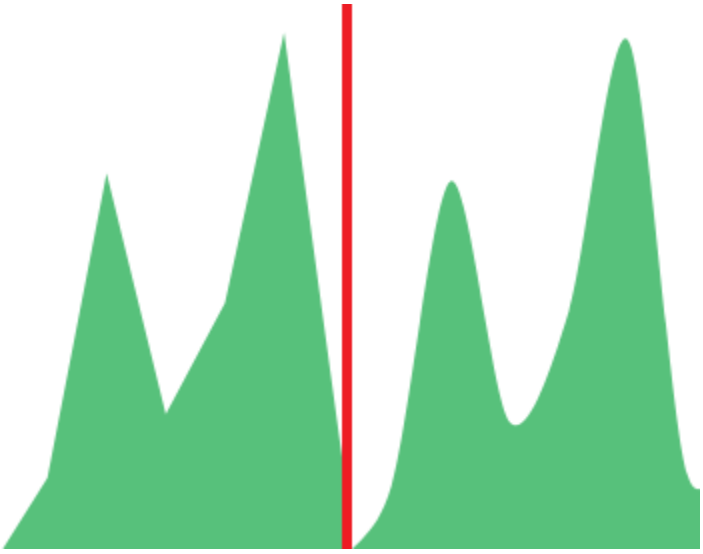
Select the Metrics & Axes tab to change the way each metric is shown on the chart. The data series are styled in the Metrics section, while the axes are styled in the X and Y axis sections.

In the “Panel Settings” tab, there are settings relating mainly to visual aesthetics. Each type of visualization has separate options.

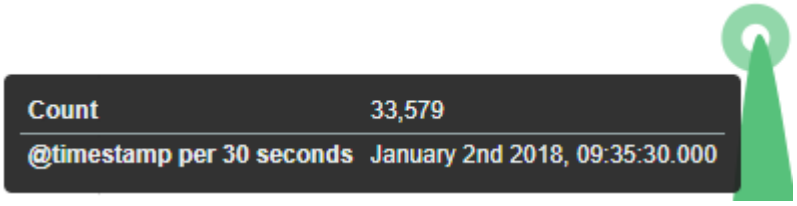
To create the first graph in the char modification panel, in the “Data” tab we add X-Axis in the “buckets” sections. In “Aggregation” choose “Histogram”, in “Field” should automatically be located “timestamp” and “interval”: “Auto” (if not, set it accordingly). Click the **Update** button in the bottom right of the configuration panel. Your first graph should now appear.

Some of the options for “Area Chart” are:

- **Smooth Lines** - is used to smooth the graph line.

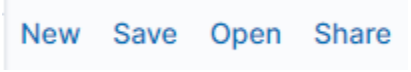


- **Current time marker** – places a vertical line on the graph that determines the current time.
- **Set Y-Axis Extents** – allows you to set minimum and maximum values for the Y axis, which increases the readability of the graphs. This is useful, if we know that the data will never be less than (the minimum value), or to indicate the goals of the company (maximum value).
- **Show Tooltip** – option for displaying the information window under the mouse cursor, after pointing to the point on the graph.



Saving

To save the visualization, click on the “Save” button under the query bar:



give it a name and click the button 

.

Load Saved Visualization

To load a previously saved visualization, go to **Management** → **Saved Objects** and filter by type "Visualizations". Select the desired visualization from the list to open it.

Dashboards

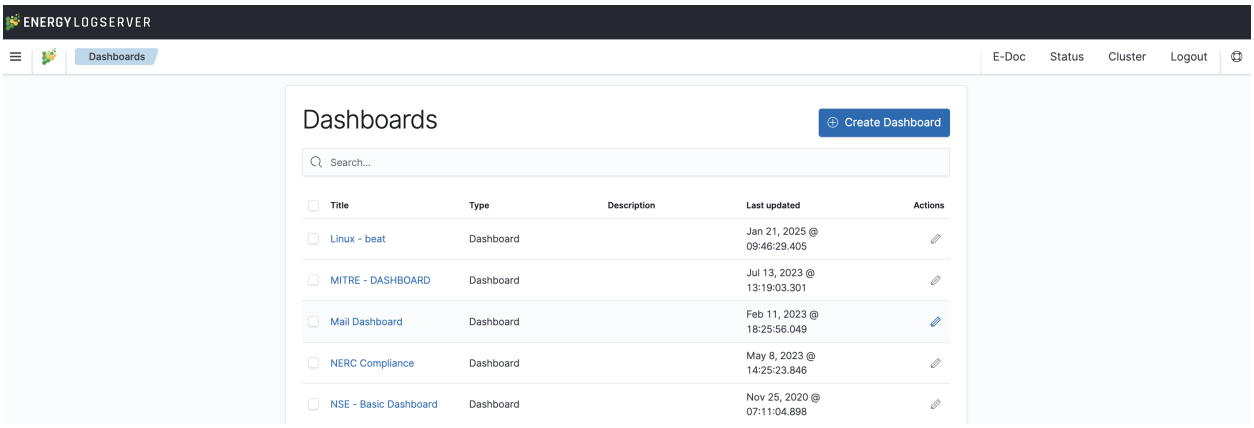
A dashboard is a collection of several visualizations or searches. Depending on how it is built and what visualization it contains, it can be designed for different teams e.g.:

- SOC - which is responsible for detecting failures or threats in the company
- Business - which thanks to the listings can determine the popularity of products and define the strategy of future sales and promotions
- Managers and directors - who may immediately have access to information about the performance units or branches

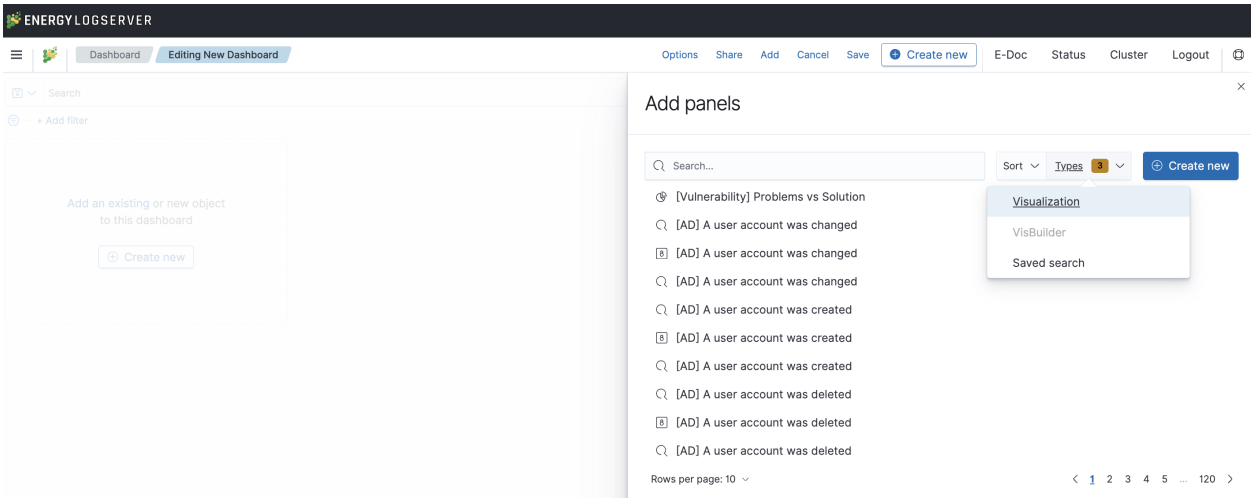
Creating Dashboards

Create

To create a dashboard from previously saved visualizations and queries, go to the "Dashboard" tab in the main menu. When you open it, a new page will appear.



Click the "Add" icon at the top of the page and select the "Visualization" or "Saved Search" tab.



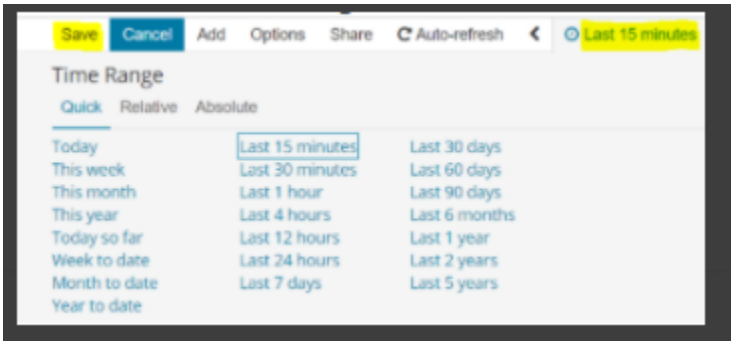
Selecting a saved query and/or visualization from the list will add them to the dashboard. If there are a large number of saved objects, use the bar to search for them by name.

Elements of the dashboard can be enlarged arbitrarily (by clicking on the right bottom corner of the object and dragging the border) and moved (by clicking on the title bar of the object and dragging it).

Saving

You may change the time period of your dashboard.

At the upper right-hand corner, you may choose the time range of your dashboard.



Click save and choose 'Store time with dashboard' if you are editing an existing dashboard. Otherwise, you may choose 'Save as a new dashboard' to create a new dashboard with the new time range.

To save a dashboard, click on the "Save" button at the top of the query bar and give it a name.

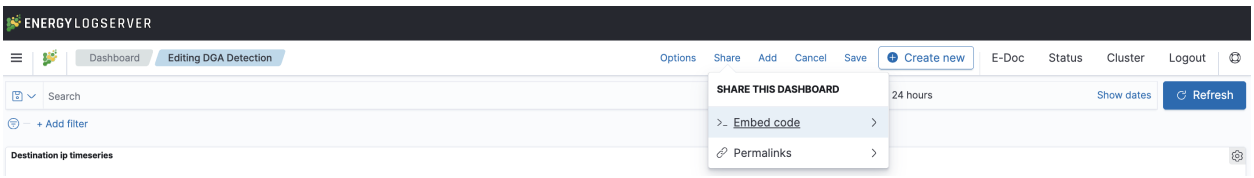
Load

To load a previously saved Dashboard, go to **Management** → **Saved Objects** and filter by type "Dashboard". Select the desired dashboard from the list to open it.

Sharing dashboards

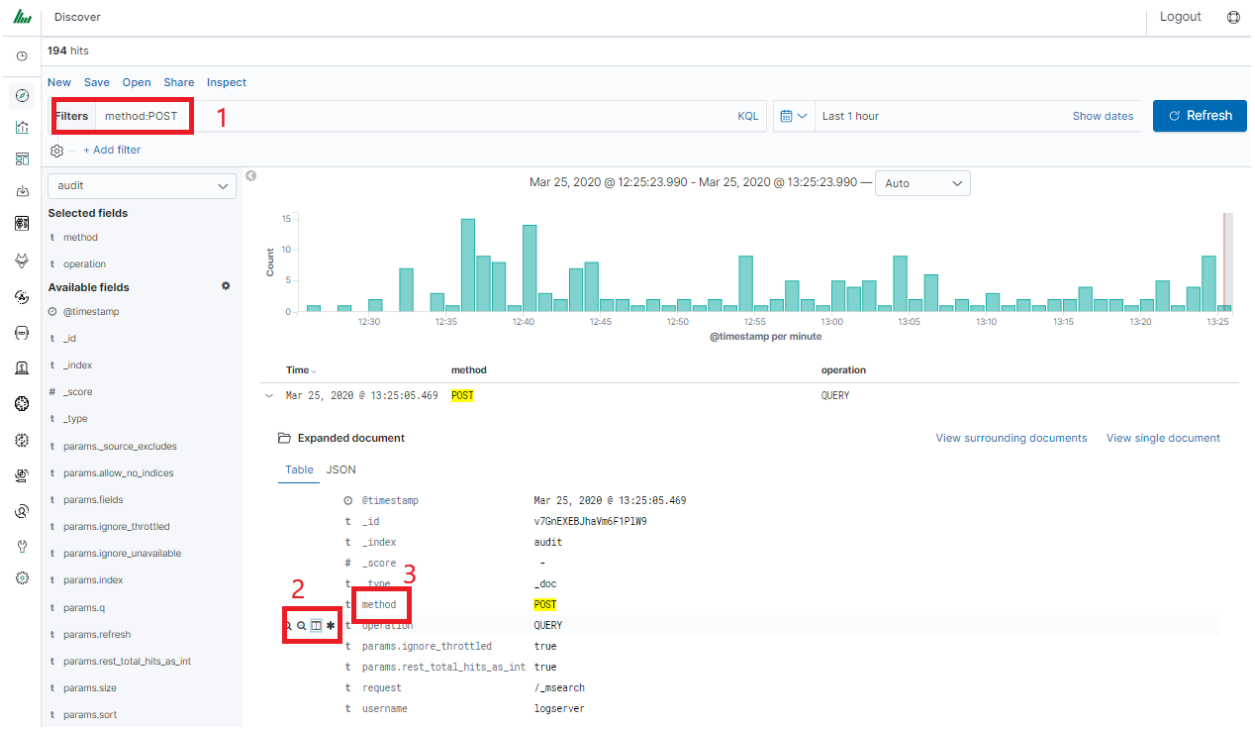
The dashboard can be shared with other Energy Logserver users as well as on any page - by placing a snippet of code. Provided that it can retrieve information from Energy Logserver.

To do this, create a new dashboard or open the saved dashboard and click on "Share" at the top of the page. A window will appear with the generated two URLs. The content of the first one "Embedded iframe" is used to provide the dashboard in the page code, and the second "Link" is a link that can be passed on to another user. There are two options for each, the first is to shorten the length of the link, and the second is to copy to the clipboard the content of the given bar.



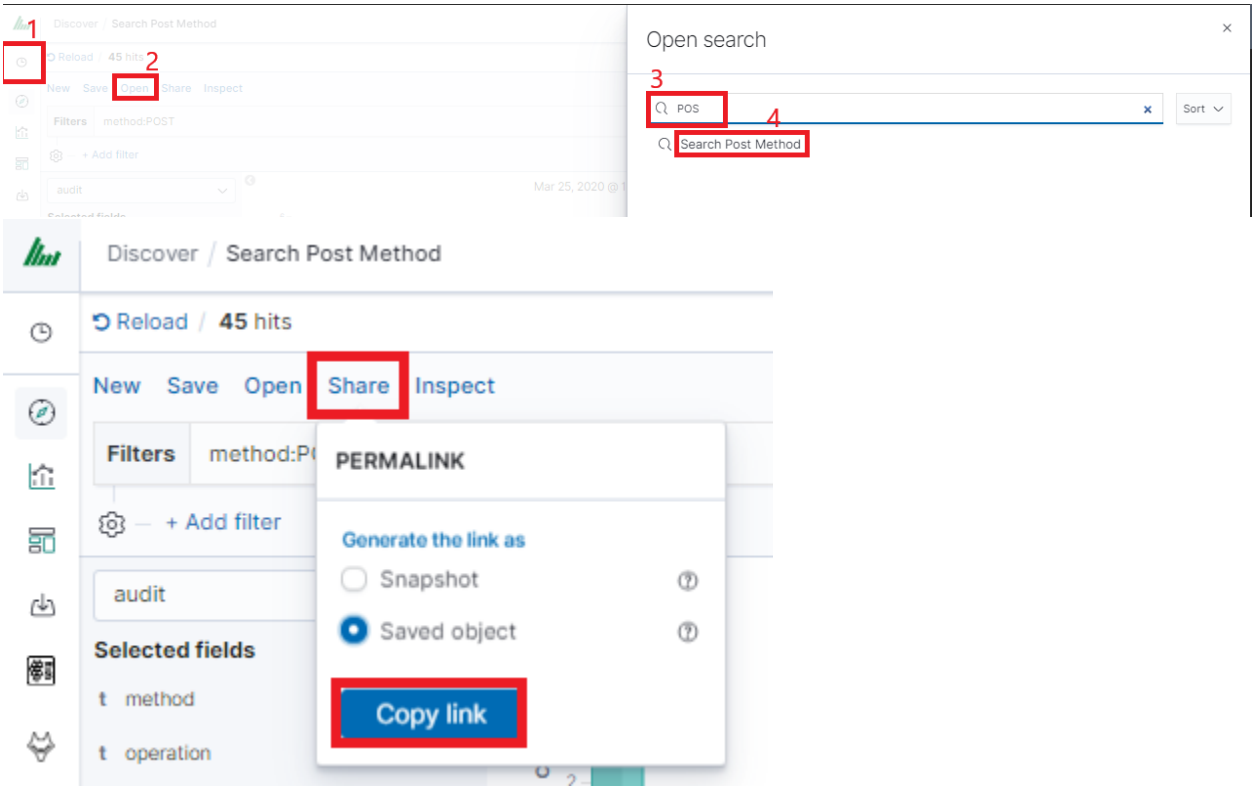
Dashboard drill down

In the discovery tab search for a message of your interest



Save your search

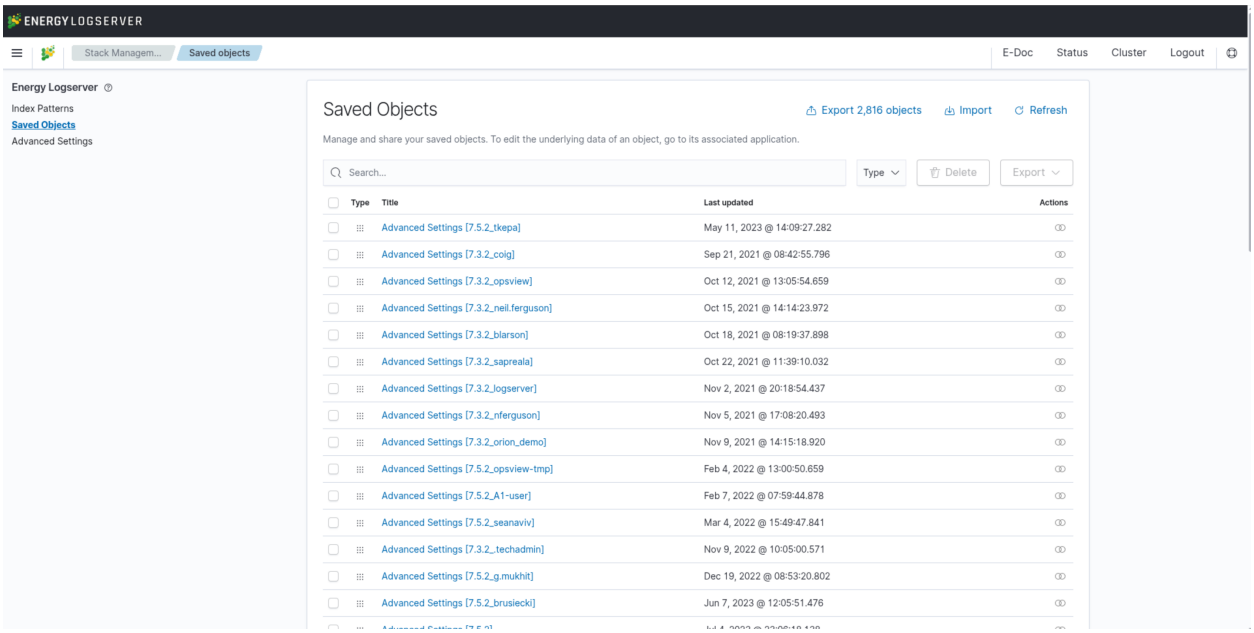
Check your “Shared link” and copy it



> **Warning**
>
> Do not copy `"?_g=()"` at the end.

Saved Objects

Saved Objects is a management interface for all objects stored in the system, including dashboards, visualizations, searches, index patterns, and queries. It is accessible via **Management** → **Saved Objects**.



Overview

The page displays all saved objects in a paginated table with 50 rows per page. A search bar and a **Type** filter allow narrowing the list to specific object types.

Available object types:

- **config**: application configuration objects
- **url**: shortened URLs
- **index-pattern**: index patterns used across the application
- **query**: saved queries
- **dashboard**: saved dashboards
- **visualization**: saved visualizations
- **visualization-visbuilder**: visualizations created with the Visual Builder
- **search**: saved searches (from Discover)

Table Columns

Column	Description
Type	Icon indicating the object type
Title	Object name, clickable: opens the object in its associated application
Last updated	Timestamp of the most recent modification
Actions	Per-row action buttons

Actions

Per-row actions

Each row provides a **Relationships** button (chain-link icon) that opens a view showing dependencies between the selected object and other saved objects, useful for identifying what a visualization depends on before deleting it.

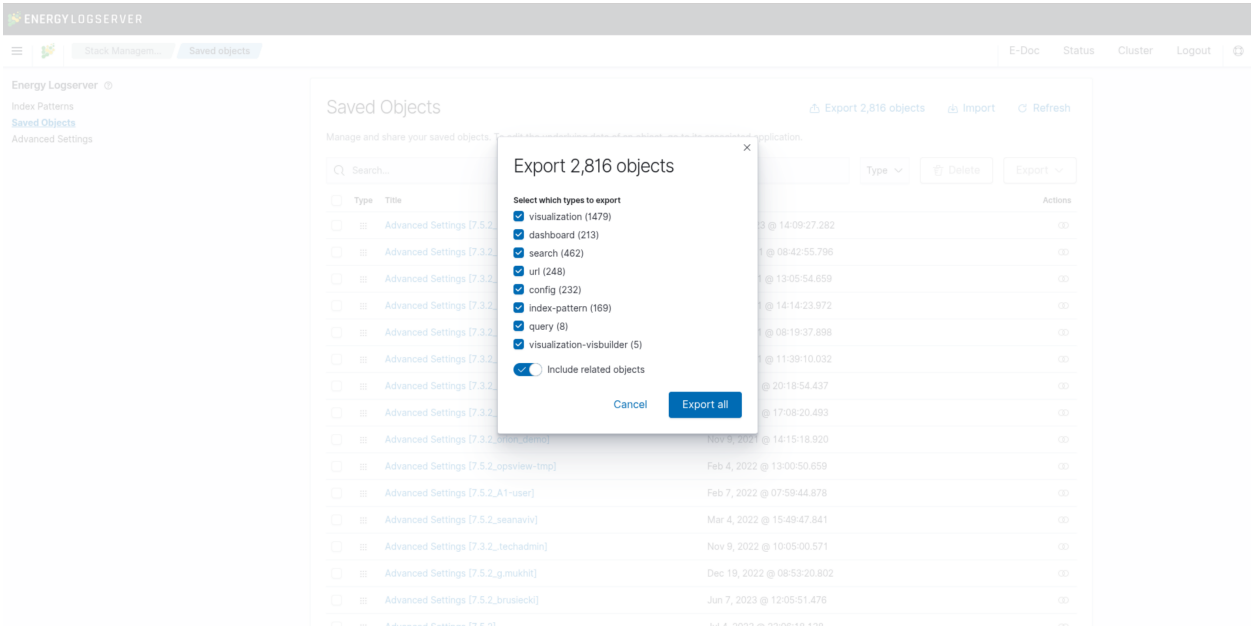
Bulk actions

Select one or more rows using the checkboxes to activate bulk actions:

- **Delete**: permanently deletes all selected objects
- **Export**: downloads selected objects as an `.ndjson` file. The export panel includes an **Include related objects** toggle (enabled by default)

Import and Export

Export all objects



Click the **Export** button (top right) to open the export dialog. Select which object types to include (all types are selected by default) and click **Export all** to download the full catalog as an `.ndjson` file.

Import

Click the **Import** button (top right) to open the import panel. Upload an `.ndjson` file using the drag-and-drop area or file picker. Select the import behavior:

- **Check for existing objects** (default), checks for conflicts with existing objects:
  - **Automatically overwrite conflicts** (default), replaces existing objects without prompting
  - **Request action on conflict**: prompts for each conflict individually
- **Create new objects with random IDs**: imports all objects as new, ignoring existing ones

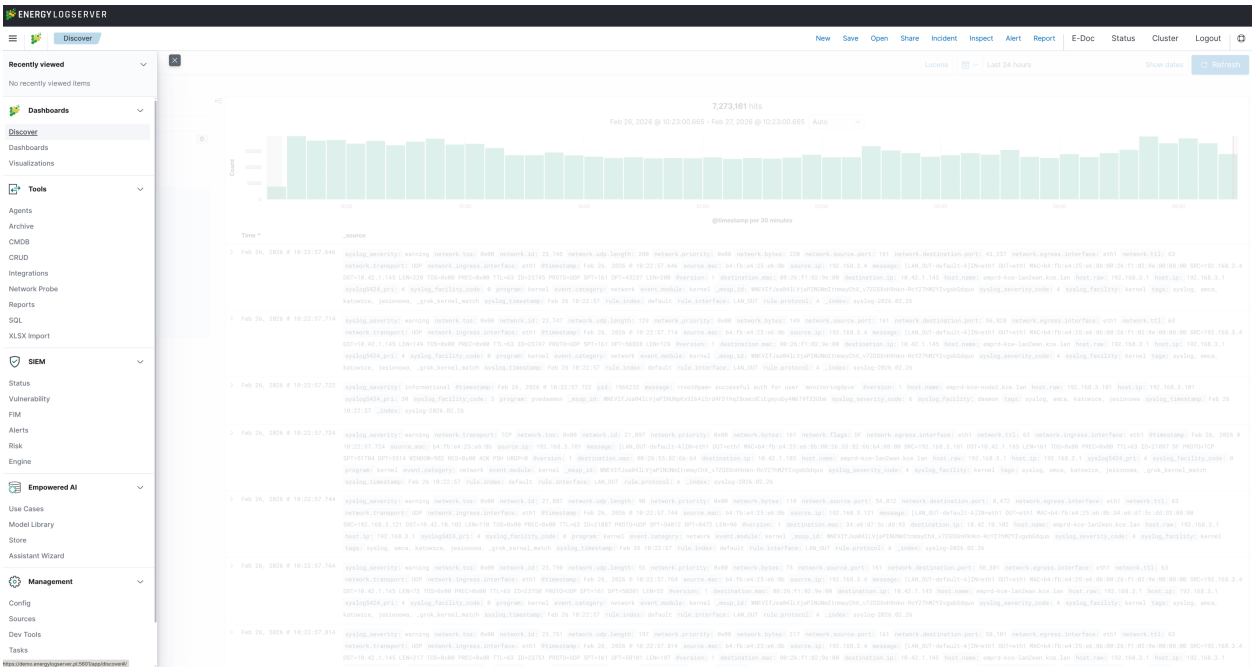
Click **Import** to complete the operation.

Refresh

The **Refresh** button (top right) reloads the list to reflect any recent changes.

Sidebar Navigation

The sidebar menu provides access to all modules and tools in Energy Logserver. Open it by clicking the hamburger icon in the upper left corner of the interface.



> **Note**
>
> The sidebar shows only the modules listed in the **Apps** field of the user's role, so what you see depends on the role you log in with and on which components are installed. The `admin` role sees every installed module. See *Role-Based Access Control* for how Apps are assigned. The screenshot above may differ from your environment.

Menu Structure

The sidebar is organized into the following categories:

Recently Viewed

Displays a list of recently accessed dashboards, visualizations, and saved searches for quick navigation.

Dashboards

Data exploration and visualization:

Item	Description
Discover	Interactive data exploration interface for searching and filtering log events
Dashboards	Create, open, and manage dashboards composed of visualizations
Visualizations	Build and edit data visualizations (charts, tables, maps, metrics)

Tools

Operational modules:

Item	Description
Agents	Agent inventory and configuration, see *Agents*
Archive	Long-term archiving and retrieval of index data, see *Archive*
Automation	n8n workflow engine integration, opened in a new browser tab. Shown only when `automation.enabled: true` is set in `/etc/logserver-gui/logserver-gui.yml`; the default is `false`.
CMDB	Asset inventory and infrastructure relations, see *CMDB*
CRUD	Document management interface for custom indices
Integrations	Plugin catalog for pipelines, dashboards, and alert rules, see *Integrations*
Network Probe	Probe, pipeline, and file management, see *Management Interface*
Reports	Scheduled and on-demand report generation and export
SQL	SQL and PPL query interface, see *SQL Query Language*
XLSX Import	Import data from XLSX spreadsheet files into indices

Note

Several of these modules open only for the `admin` role: Agents, Archive, CMDB, CRUD, Integrations, Network Probe, and XLSX Import. Another role can still have them in its **Apps** list and reach the page, but the page then reports *"This page is accessible for administrator users only."* Reports is gated differently, at the API level, where the `report` role is accepted alongside `admin`.

SIEM

Security Information and Event Management modules for threat detection and incident handling:

Item	Description
Status	SIEM operational status overview and health indicators
Vulnerability	Vulnerability detection and assessment results
FIM	File Integrity Monitoring, tracks changes to critical system files
Alerts	Alert rule management, incident handling, and notification configuration
Risk	Risk scoring and risk-based event prioritization
SIEM Engine	Endpoint Security engine overview, see *SIEM Interface Overview*

Note

The SIEM category appears only on deployments where the SIEM Plan package is installed. Within it, the entries follow the `alerts` app on the role, and incident handling additionally requires the `security` or `security-tenant` role. SIEM Engine has its own entry in the **Apps** list.

Empowered AI

AI and machine learning capabilities:

Item	Description
Use Cases	Manage configured AI detection rules, scheduling, and execution status
Model Library	Browse and manage trained AI models
Store	Download pre-built AI use cases from the online store
Assistant Wizard	Configure AI Assistant providers and prompts

Note

The Empowered AI entries follow the `intelligence` app on the role. The AI Assistant itself is license-gated separately: it is enabled only when the active license carries the `aiAssistant` flag, see *Licensing*.

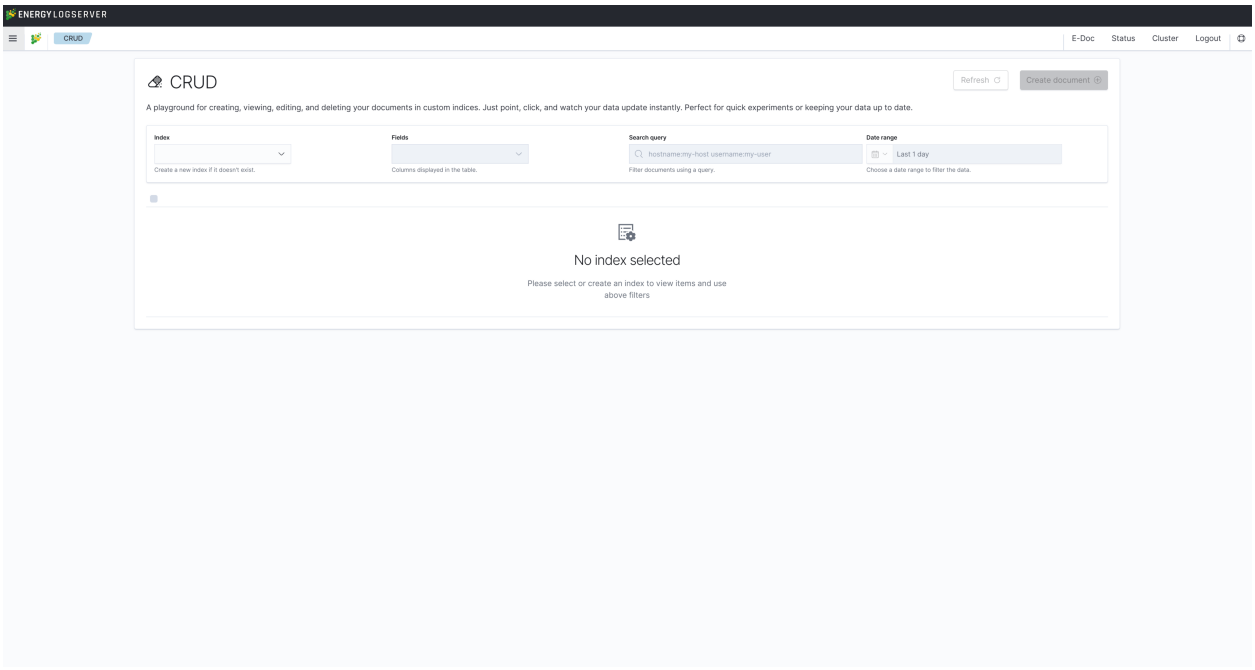
Management

System administration and configuration tools:

Item	Description
Config	Application configuration settings, license management, and user administration
Sources	Data source approval and management within the license limit, visible to administrators only, see *MSSP Configuration*
Tasks	Background task queue, view status and history of scheduled and on-demand tasks
Index Management	Index lifecycle management, index settings, and data operations. When deleting indices, a confirmation warning is displayed to prevent accidental data loss.
Management	Saved objects, advanced settings, and index pattern management
Dev Tools	Developer console for direct API interaction and Query Workbench

CRUD

The CRUD module provides an interface for creating, viewing, editing, and deleting documents in custom indices. It is accessible from the sidebar under **Tools** → **CRUD**.



Interface

The module header contains the following filter controls:

Control	Description
Index	Select an existing index or create a new one if it does not exist
Fields	Choose which columns are displayed in the results table
Search query	Filter documents using a query string (e.g., `hostname:my-host username:my-user`)
Date range	Restrict results to a specific time window

Click **Refresh** to reload the current view after changing filters.

Operations

Create document

Click the **Create document** button in the upper right corner to add a new document to the selected index. Fill in the field values in the form and submit. If the specified index does not exist, it is created automatically.

View

Select an index and optionally apply filters to browse existing documents. Results are displayed in a table with the columns selected in the **Fields** control. Click on a row to expand the full document.

Edit

Click the edit icon in the **Actions** column of a document row to modify its field values. The updated document is saved to the same index.

Delete

Click the delete icon in the **Actions** column to remove a document from the index.

Bulk Operations

The CRUD module supports bulk operations on multiple documents at once. Use the checkboxes in the leftmost column to select documents, then choose an action from the bulk action toolbar:

- **Bulk Update**: apply the same field value changes to all selected documents.
- **Bulk Delete**: remove all selected documents from the index.

Use the select-all checkbox in the table header to select all documents on the current page.

Access Control

> **Note**
>
> The CRUD module is available to the `admin` role only. A user with any other role who opens it sees *"This page is accessible for administrator users only."* instead of the document table. Whether the item appears in the sidebar at all is a separate matter, controlled by the **Apps** list on the role.

> **Note**
>
> The CRUD module operates on custom indices and is intended for managing reference data, lookup tables, or manually maintained records. It is not designed for modifying indices that receive data from automated pipelines.

Appearance

Energy Logserver uses a light theme by default. Each user account keeps its own theme setting, so switching to dark mode on one account does not change the interface for other users.

Switching Dark Mode

1. Open the sidebar and go to **Management > Advanced settings**.
2. In the **Appearance** section, turn **Dark mode** on or off.
3. Click **Save changes** and refresh the page.

Quick Switching from the Top Bar

To switch the theme without opening Advanced settings, turn on **Enable user control** in the same **Appearance** section, click **Save changes** and refresh the page. A brush icon appears in the upper right corner, next to the user avatar. Click it, choose **Screen mode** and click **Apply**:

Screen mode	Effect
Light mode	Light theme
Dark mode	Dark theme
Use browser settings	Takes the light or dark theme from the operating system at the moment you click **Apply**. A later change in the system theme takes effect only after you click **Apply** again.

Note

With **Enable user control** on, the theme selected in the top bar is stored in the browser, not on the server, so it applies only to the browser where you selected it. The **Dark mode** switch in Advanced settings is then greyed out with the message *This setting is overridden by user or browser preferences*. Turning **Enable user control** off restores the theme saved in Advanced settings and removes the browser setting.

4.6 Log Management

Comprehensive log collection, processing, querying, and lifecycle management.

4.6.1 Overview

Energy Logserver's log management capabilities provide comprehensive tools for querying, analyzing, and managing security data throughout its lifecycle. Key features include:

- **Flexible Query Languages**: SQL and PPL (Piped Processing Language) for powerful data analysis
- **Multiple Query Formats**: REST API with JSON, CSV, JDBC, and raw response formats
- **Advanced Search**: Full-text search with match, multi-match, and query string capabilities
- **Data Type System**: Complete type system with identifiers, functions, and operators
- **Long-term Archive**: Cost-effective storage with searchable archive and restoration capabilities
- **Index Lifecycle Management**: Automated policies for data retention, rollover, and deletion

4.6.2 In This Section

- **SQL Query Language**: Complete SQL syntax with practical SIEM examples
- **PPL Language**: Piped Processing Language reference and commands
- **Query API and Formats**: API endpoints and response format options (JSON, CSV, JDBC, Raw)
- **Data Types and Functions**: Data types, identifiers, and built-in functions reference
- **Full-Text Search**: Advanced search capabilities with match and query string syntax
- **Archive**: Long-term data storage, search, and restoration
- **Index Management**: Index lifecycle policies and optimization actions

Index Management

> **Note**
>
> Before using the Index Management module, it is necessary to set an appropriate password for the *logserver* user in the following file: `/usr/share/logserver-gui/curator/curator.yml`

The Index Management module allows you to manage indexes and perform activities such as:

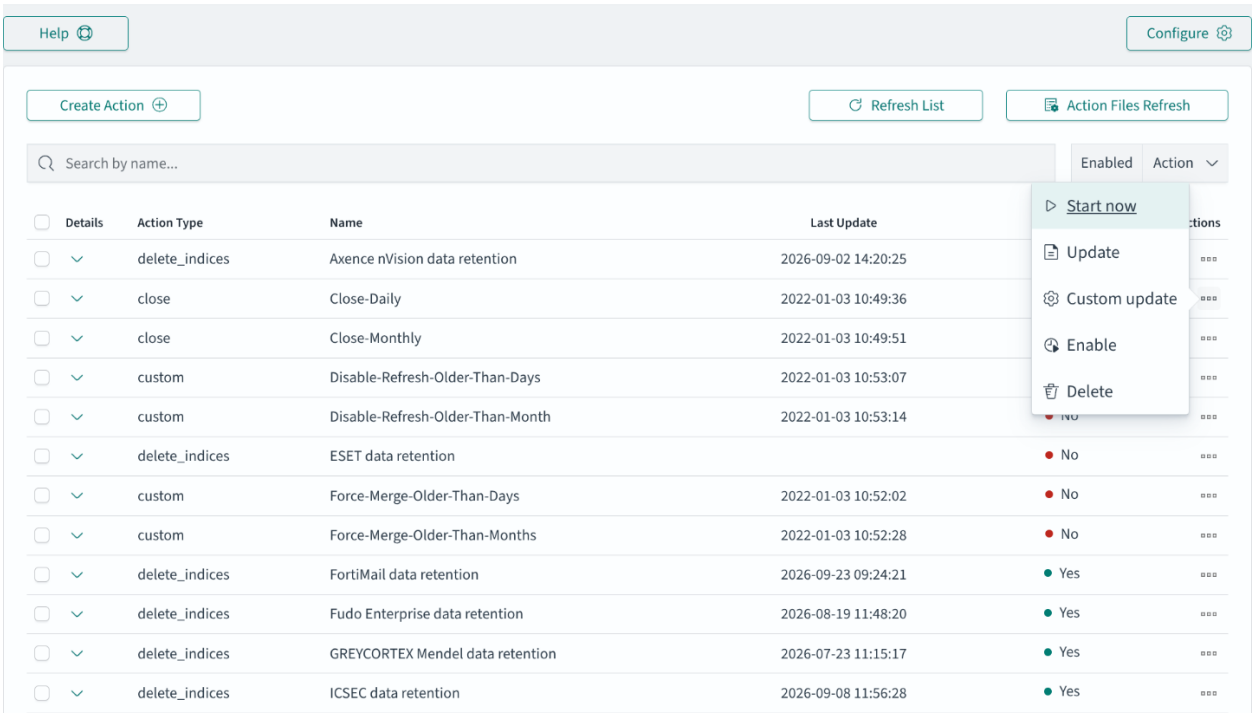
- Closing indexes,
- Delete indexes,
- Performing a merge operation for index,
- Shrink index shards,
- Index rollover.

The *Index Management* module is accessible through the main menu tab.

The main module window allows you to create new *Create Task* tasks, view and manage created tasks, that is:

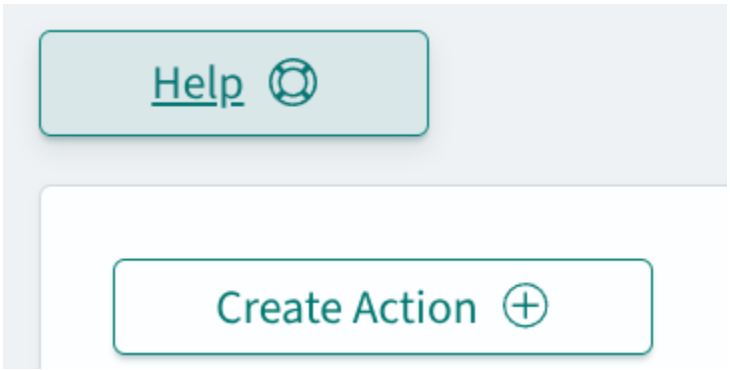
- Start now,
- Update,
- Custom update,
- Enable / Disable,
- Delete.

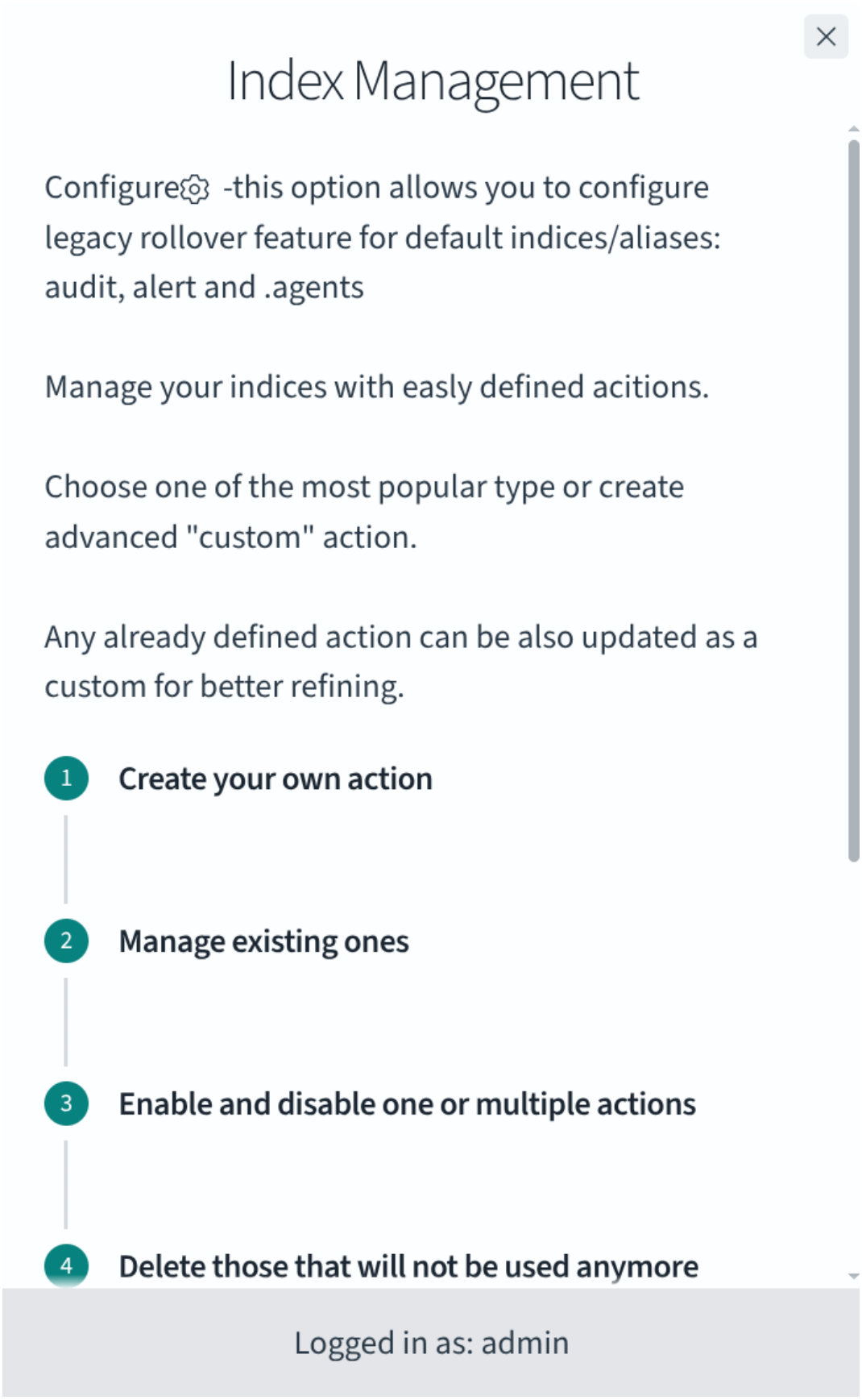
The *Scheduled* column shows whether a task runs on its schedule.



Note

Use the `Help` button to get a detailed description of the current action.




Index Management
Configure -this option allows you to configure legacy rollover feature for default indices/aliases: audit, alert and .agents
Manage your indices with easly defined acitions.
Choose one of the most popular type or create advanced "custom" action.
Any already defined action can be also updated as a custom for better refining.
1 Create your own action
2 Manage existing ones
3 Enable and disable one or multiple actions
4 Delete those that will not be used anymore
Logged in as: admin

- *Data tiering and retention strategy*
- *Close action*
- *Delete action*
- *Force Merge action*
- *Shrink action*
- *Rollover action*
- *Index rollover*
- *Custom action*
- *Preinstalled actions*

Data tiering and retention strategy

As data ages it is queried less, so it should move to cheaper storage and eventually leave the cluster, without shrinking the window that has to stay searchable. The Index Management actions below combine into one lifecycle, run as scheduled tasks:

1. **Hot.** Recent, most-queried data stays on fast SSD nodes that carry the indexing load.
2. **Move to cheaper storage.** After a set age, relocate older shards to high-capacity nodes with the *Index allocation* action (for example `require tag=cold`). Tag each node with a node attribute first (see `node.attr` in the node configuration), then route by that attribute. The data stays searchable on the slower tier; only the newest data needs sub-second speed.
3. **Optimize what no longer changes.** Reduce segment count on older read-only indices with *Force Merge*, and stop refreshes on indices that no longer receive writes with the preinstalled `Disable-Refresh-Older-Than` actions.
4. **Cap index size.** Use *Rollover* to start a new index once one reaches a size or age limit, which keeps shards in a healthy range.
5. **Retire.** Beyond the retention period, *Close* or *Delete* indices. For data that must be kept long term, send it to the *Archive* first; a zstd-compressed archive is a fraction of the live size and can still be searched and restored.

Size each tier from the retention split rather than keeping everything on fast disk. See *Capacity Planning Guidelines* for the storage math.

Close action

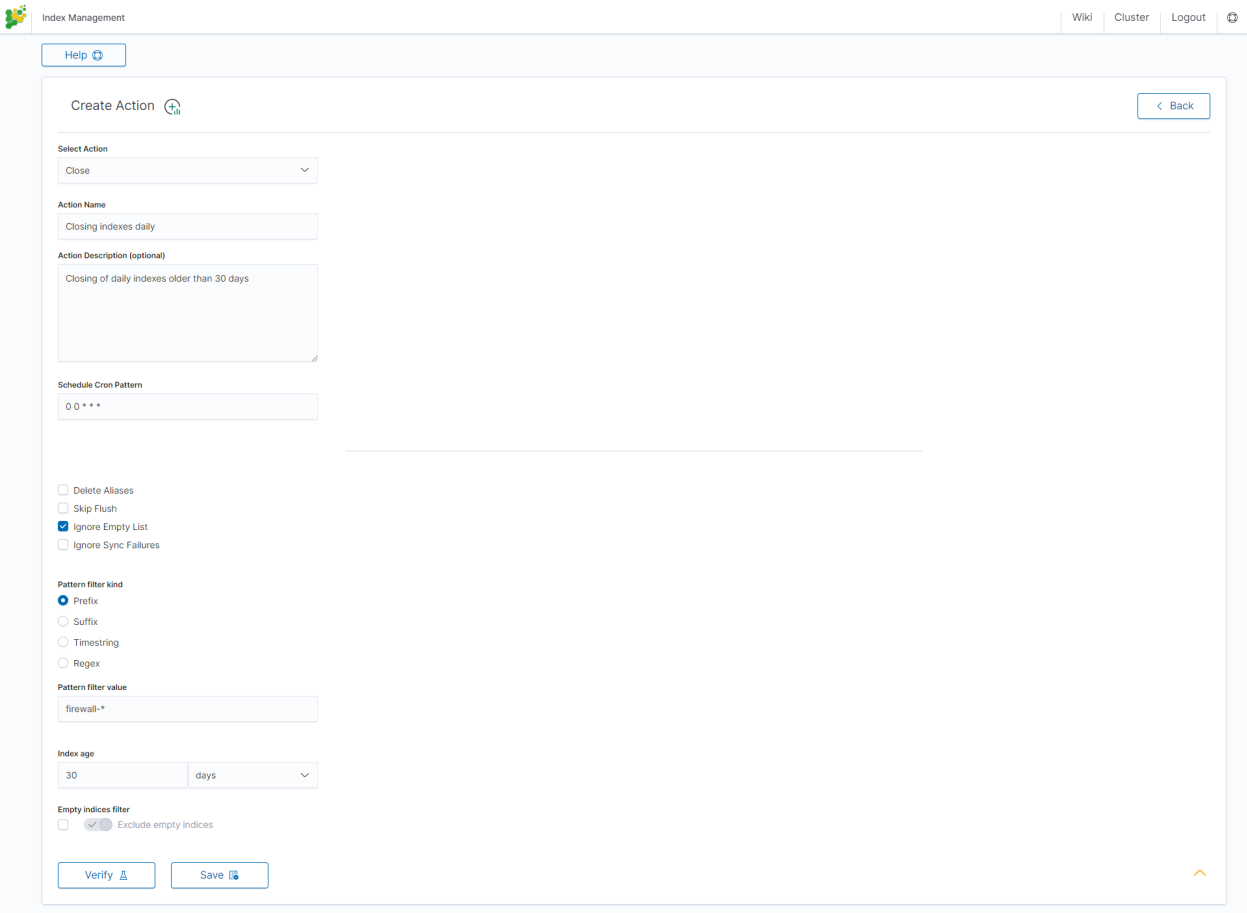
This action closes the selected indices and optionally deletes associated aliases beforehand.

Settings required:

- Action Name
- Schedule Cron Pattern - it sets when the task is to be executed, to decode cron format use the online tool: https://crontab.guru,
- Pattern filter kind - it sets the index filtertype for the task,
- Pattern filter value - it sets the value for the index filter,
- Index age - it sets the index age for the task.

Optional settings:

- Timeout override
- Ignore Empty List
- Continue if exception
- Closed indices filter
- Empty indices filter



Delete action

This action deletes the selected indices.

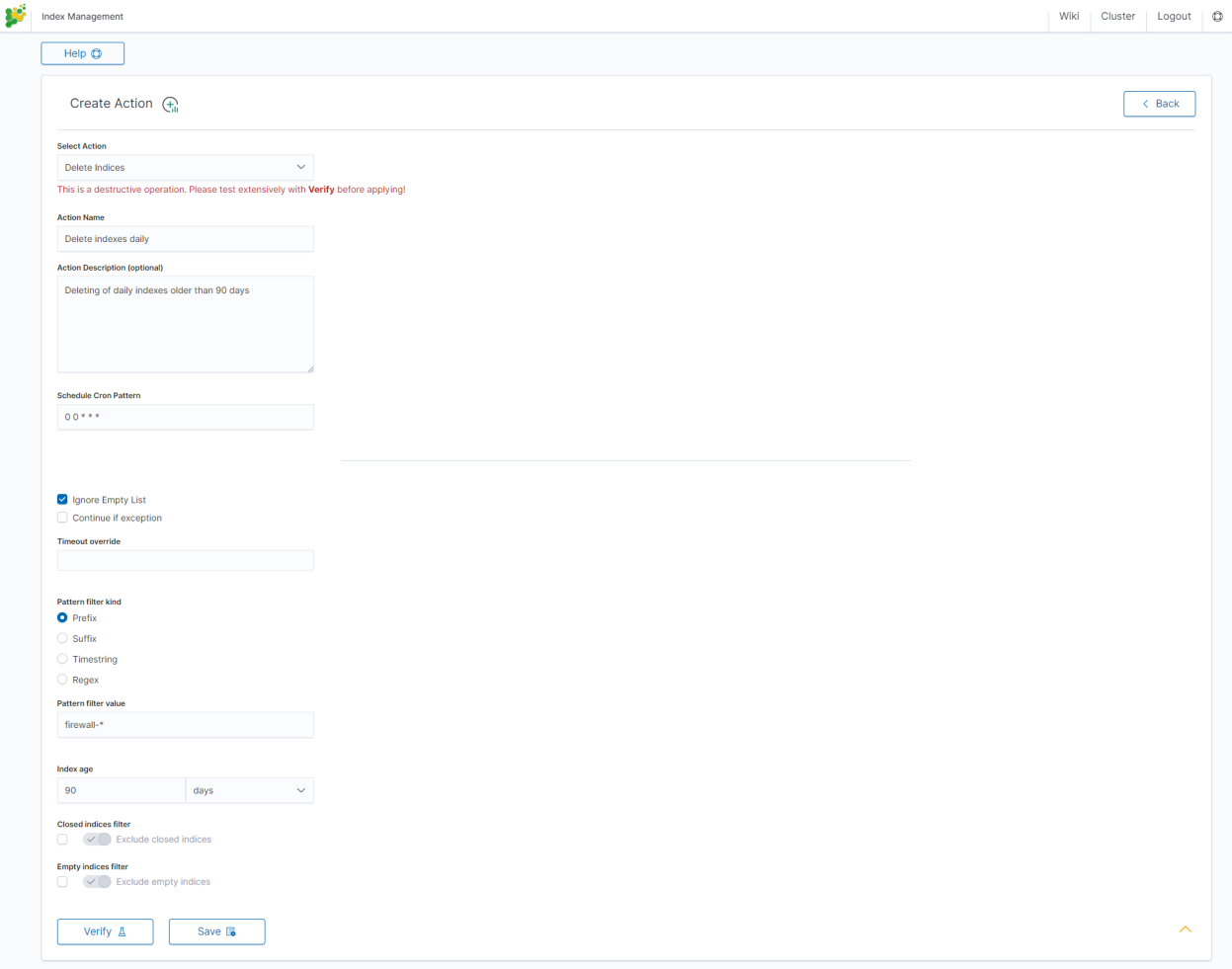
Settings required:

- Action Name
- Schedule Cron Pattern - it sets when the task is to be executed, to decode cron format use the online tool: https://crontab.guru,
- Pattern filter kind - it sets the index filtertype for the task,
- Pattern filter value - it sets the value for the index filter,
- Index age - it sets the index age for the task.

Optional settings:

- Delete Aliases

- Skip Flush
- Ignore Empty List
- Ignore Sync Failures



Force Merge action

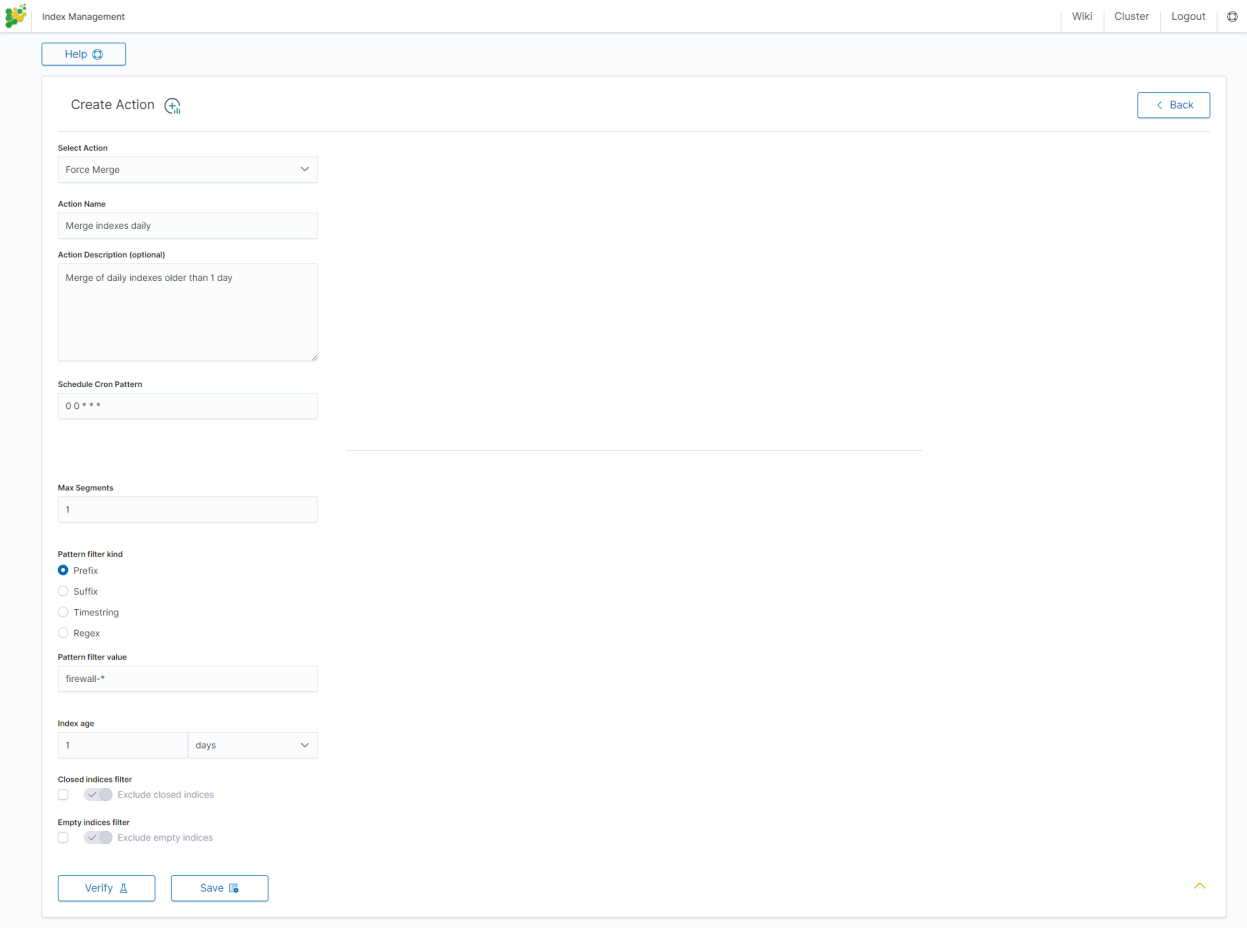
This action performs a Force Merge on the selected indices, merging them in the specific number of segments per shard.

Settings required:

- Action Name
- Schedule Cron Pattern - it sets when the task is to be executed, to decode cron format use the online tool: https://crontab.guru,
- Max Segments - it sets the number of segments for the shard,
- Pattern filter kind - it sets the index filtertype for the task,
- Pattern filter value - it sets the value for the index filter,
- Index age - it sets the index age for the task.

Optional settings:

- Ignore Empty List
- Ignore Sync Failures



Shrink action

Shrinking an index is a good way to reduce the total shard count in your cluster.

Several conditions need to be met in order for index shrinking to take place:

- The index must be marked as read-only
- A (primary or replica) copy of every shard in the index must be relocated to the same node
- The cluster must have health green
- The target index must not exist
- The number of primary shards in the target index must be a factor of the number of primary shards in the source index.
- The source index must have more primary shards than the target index.
- The index must not contain more than 2,147,483,519 documents in total across all shards that will be shrunk into a single shard on the target index as this is the maximum number of docs that can fit into a single shard.
- The node handling the shrink process must have sufficient free disk space to accommodate a second copy of the existing index.

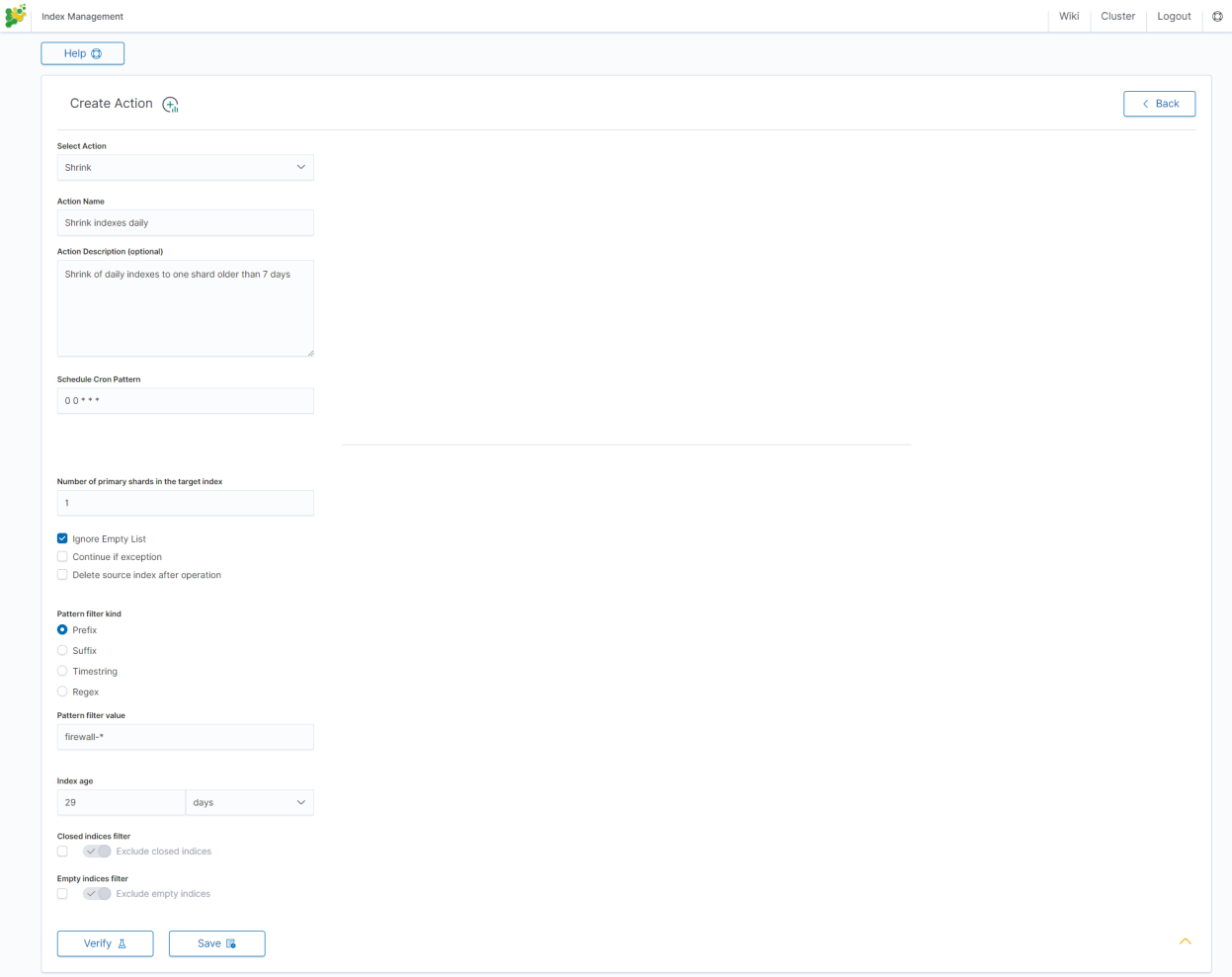
The task will try to meet these conditions. If it is unable to meet them all, it will not perform a shrink operation.

Settings required:

- Action Name
- Schedule Cron Pattern - it sets when the task is to be executed, to decode cron format use the online tool: https://crontab.guru,
- Number of primary shards in the target index - it sets the number of shards for the target index,
- Pattern filter kind - it sets the index filtertype for the task,
- Pattern filter value - it sets the value for the index filter,
- Index age - it sets the index age for the task.

Optional settings:

- Ignore Empty List
- Continue if exception
- Delete source index after operation
- Closed indices filter
- Empty indices filter



Rollover action

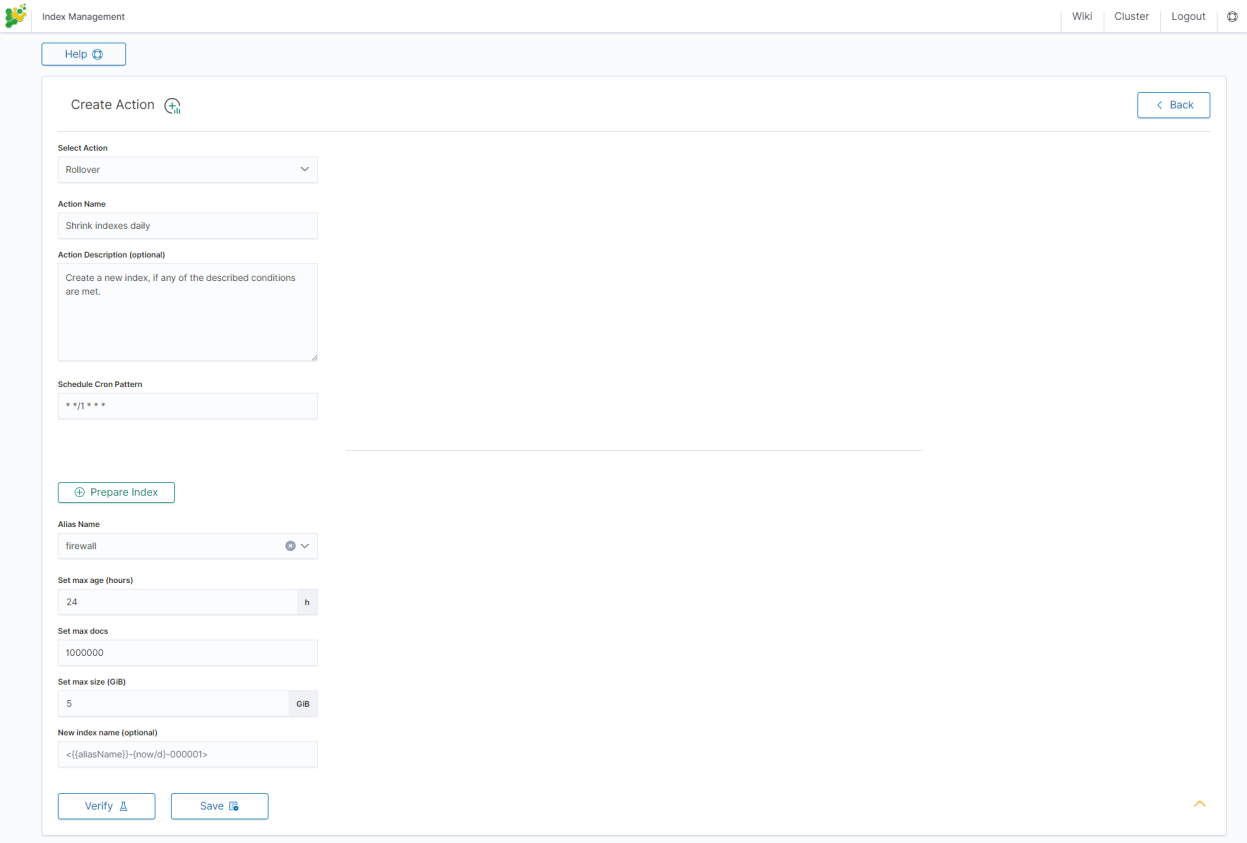
This action uses the Data Node Rollover API to create a new index if any of the described conditions are met.

Settings required:

- Action Name
- Schedule Cron Pattern - it sets when the task is to be executed, to decode cron format use the online tool: https://crontab.guru,
- Alias Name - it sets an alias for the index,
- Set max age (hours) - it sets an age for the index after then index will rollover,
- Set max docs - it sets a number of documents for the index after which the index will rollover,
- Set max size (GiB) - it sets index size in GB after which the index will rollover.

Optional settings:

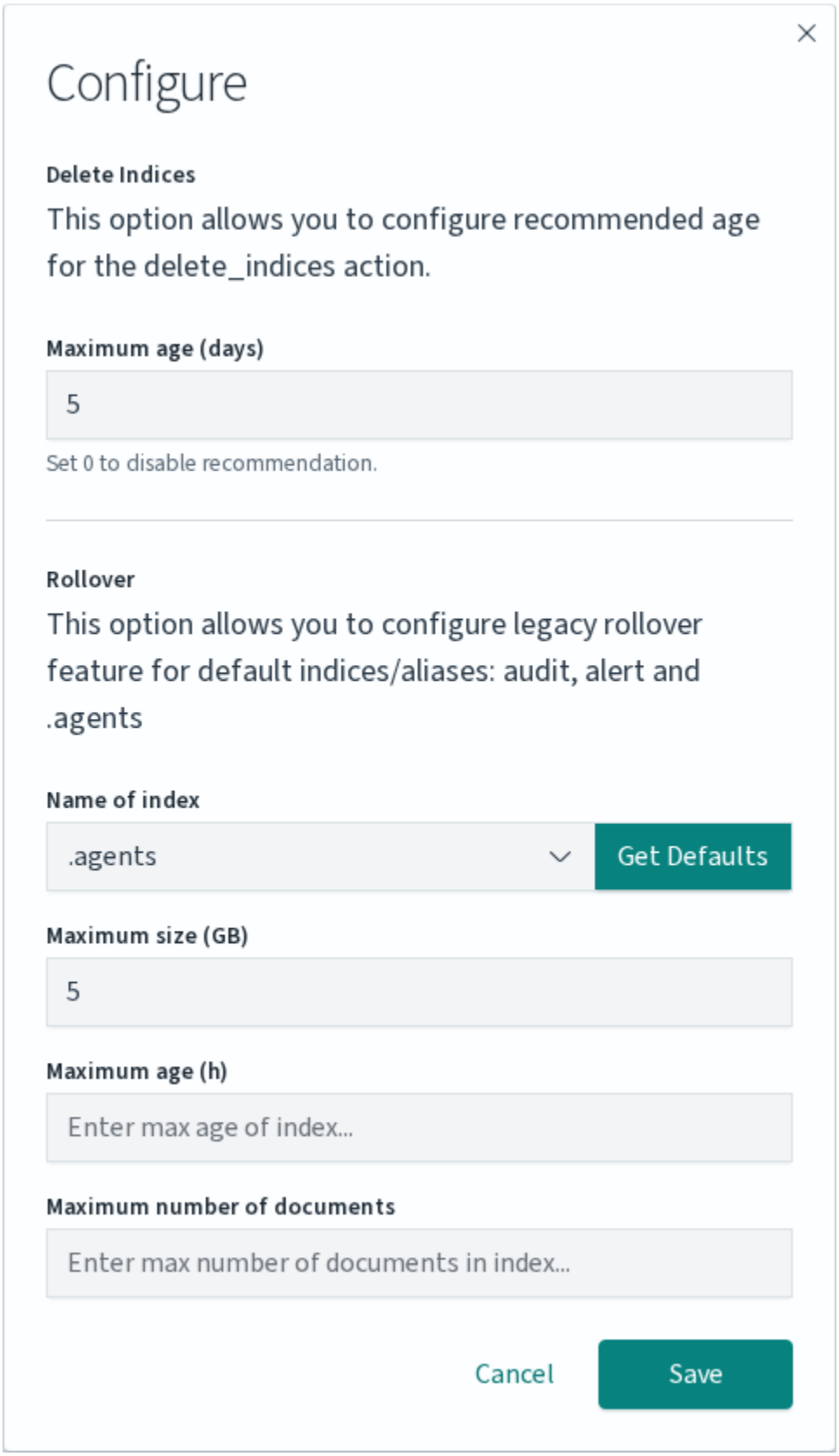
- New index name (optional)



Index rollover

Using the rollover function, you can configure retention policies for the *audit*, *.agents*, and *alert* indices.

To access the rollover configuration, click the **Configure** button (gear icon) within the **Index Management** module:



You can set the following retention parameters for the above indices:

- Maximum size (GB);
- Maximum age (h);
- Maximum number of documents.

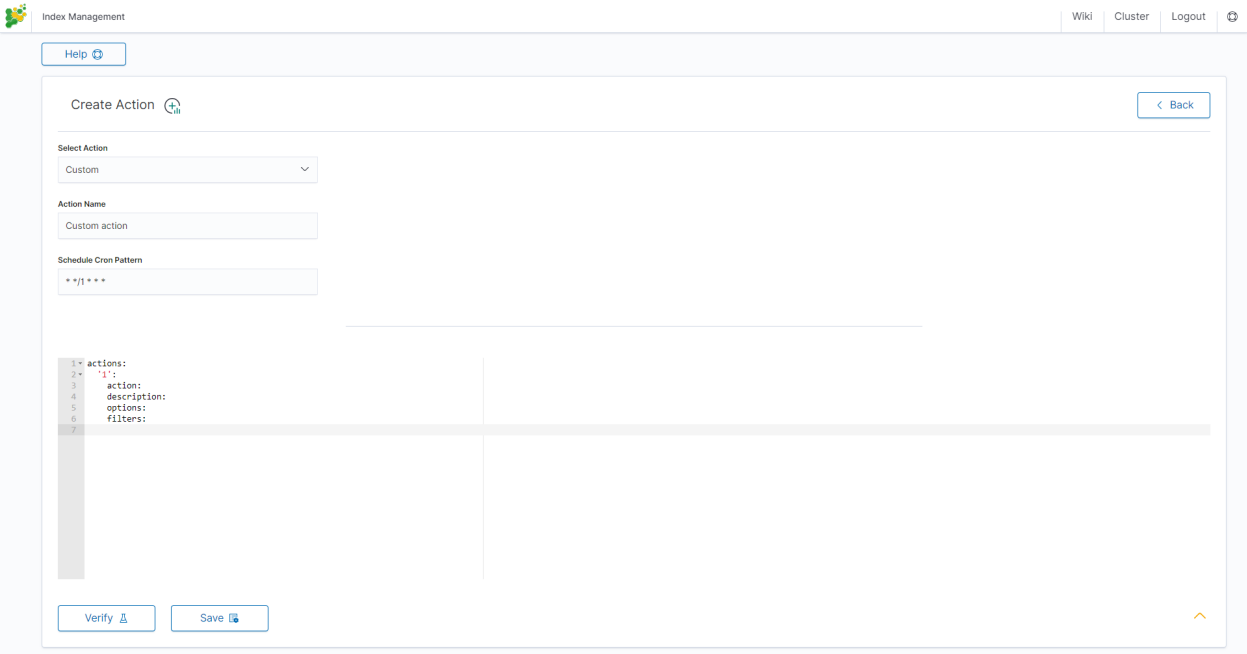
Delete Indices

The Configure modal also includes a **Delete Indices** section that defines the maximum retention age for system indices before automatic deletion.

Setting	Description
Maximum age (days)	Defines the retention period in days. Documents older than this threshold are automatically deleted from the `.agents`, `alert`, and `audit` indices.

Custom action

To create a Custom action, select *Custom* from *Select Action*, enter a name in the *Action Name* field, and set the schedule in the *Schedule Cron Pattern* field. In the edit field, enter the definition of a custom action:



Custom Action examples:

Open index

```
actions:
  1:
    action: open
    description: >-
      Open indices older than 30 days but younger than 60 days (based on index
      name), for syslog- prefixed indices.
    options:
      timeout_override:
```

(continues on next page)

(continued from previous page)

```
      continue_if_exception: False
      disable_action: True
    filters:
    - filtertype: pattern
      kind: prefix
      value: syslog-
      exclude:
    - filtertype: age
      source: name
      direction: older
      timestring: '%Y.%m.%d'
      unit: days
      unit_count: 30
      exclude:
    - filtertype: age
      source: name
      direction: younger
      timestring: '%Y.%m.%d'
      unit: days
      unit_count: 60
      exclude:
```

Replica reduce

```
actions:
  1:
    action: replicas
    description: >-
      Reduce the replica count to 0 for syslog- prefixed indices older than
      10 days (based on index creation_date)
    options:
      count: 0
      wait_for_completion: False
      timeout_override:
      continue_if_exception: False
      disable_action: True
    filters:
    - filtertype: pattern
      kind: prefix
      value: syslog-
      exclude:
    - filtertype: age
      source: creation_date
      direction: older
      unit: days
      unit_count: 10
      exclude:
```

Index allocation

```
actions:
  1:
    action: allocation
    description: >-
      Apply shard allocation routing to 'require' 'tag=cold' for hot/cold node
      setup for syslog- indices older than 3 days, based on index_creation
      date
    options:
      key: tag
      value: cold
      allocation_type: require
      disable_action: True
    filters:
    - filtertype: pattern
      kind: prefix
      value: syslog-
    - filtertype: age
      source: creation_date
      direction: older
      unit: days
      unit_count: 3
```

Cluster routing

```
actions:
  1:
    action: cluster_routing
    description: >-
      Disable shard routing for the entire cluster.
    options:
      routing_type: allocation
      value: none
      setting: enable
      wait_for_completion: True
      disable_action: True
  2:
    action: (any other action details go here)
    ...
  3:
    action: cluster_routing
    description: >-
      Re-enable shard routing for the entire cluster.
    options:
      routing_type: allocation
      value: all
      setting: enable
      wait_for_completion: True
      disable_action: True
```

Preinstalled actions

Since 8.1.0 the preinstalled actions are added with *Scheduled* set to *No*. They run only after you choose *Enable* in the task's *Actions* menu, so review the filters below first, because `Close-Daily` and `Close-Monthly` close indices.

Close-Daily

This action closes the selected indices older than 93 days and optionally deletes associated aliases beforehand. For example, if today is 21 December this action will close or optionally delete every index older than 19 September of the same year, action starts every day at 01:00 AM.

`Action type`: CLOSE
`Action name`: Close-Daily
`Action Description (optional)`: Close daily indices older than 93 days
`Schedule Cron Pattern`: 0 1 * * *
`Delete Aliases`: enabled
`Skip Flush`: disabled
`Ignore Empty List`: enabled
`Ignore Sync Failures`: enabled
`Pattern filter kind`: Timestring
`Pattern filter value`: %Y.%m.%d
`Index age`: 93 days
`Empty indices filter`: disable

Close-Monthly

This action closes the selected indices older than 93 days (3 months) and optionally deletes associated aliases beforehand. If today is 21 December, this action will close or optionally delete every index older than October the same year, the action starts every day at 01:00 AM.

`Action type`: CLOSE
`Action name`: Close-Monthly
`Action Description (optional)`: Close monthly indices older than 93 days
`Schedule Cron Pattern`: 0 1 * * *
`Delete Aliases`: enabled
`Skip Flush`: disabled
`Ignore Empty List`: enabled
`Ignore Sync Failures`: enabled
`Pattern filter kind`: Timestring
`Pattern filter value`: %Y.%m$
`Index age`: 93 days
`Empty indices filter`: disable

Disable-Refresh-Older-Than-Days

This action disables the daily refresh of indices older than 2 days. the action is performed daily at 01:00.

`Action type`: CUSTOM
`Action name`: Disable-Refresh-Older-Than-Days
`Schedule Cron Pattern`: 0 1 * * *

YAML:

```
actions:
  '1':
    action: index_settings
```

(continues on next page)

(continued from previous page)

```
    description: Disable refresh for older daily indices
    options:
      index_settings:
        index:
          refresh_interval: -1
      ignore_unavailable: False
      ignore_empty_list: true
      preserve_existing: False
    filters:
      - filtertype: pattern
        kind: timestring
        value: '%Y.%m.%d$'
      - filtertype: age
        source: creation_date
        direction: older
        unit: days
        unit_count: 2
```

Disable-Refresh-Older-Than-Month

This action disables the refresh of indices older than one month. The action is performed daily at 01:00.

`Action type`: CUSTOM
`Action name`: Disable-Refresh-Older-Than-Month
`Schedule Cron Pattern`: 0 1 * * *

YAML:

```
actions:
  '1':
    action: index_settings
    description: Disable refresh for older monthly indices
    options:
      index_settings:
        index:
          refresh_interval: -1
      ignore_unavailable: False
      ignore_empty_list: true
      preserve_existing: False
    filters:
      - filtertype: pattern
        kind: timestring
        value: '%Y.%m$'
      - filtertype: age
        source: creation_date
        direction: older
        unit: days
        unit_count: 32
```

Force-Merge-Older-Than-Days

This action forces the daily merge of indices older than two days. The action is performed daily at 01:00.

`Action type`: CUSTOM
`Action name`: Force-Merge-Older-Than-Days
`Schedule Cron Pattern`: 0 1 * * *

YAML:

```
actions:
  '1':
    action: forcemerge
    description: Force merge on older daily indices
    options:
      max_num_segments: 1
      ignore_empty_list: true
      continue_if_exception: false
      delay: 60
    filters:
      - filtertype: pattern
        kind: timestring
        value: '%Y.%m.%d$'
      - filtertype: age
        source: creation_date
        direction: older
        unit: days
        unit_count: 2
      - filtertype: forcemerged
        max_num_segments: 1
        exclude: True
```

Force-Merge-Older-Than-Months

This action forces the daily merge of indices older than one month. The action is performed daily at 01:00.

`Action type`: CUSTOM
`Action name`: Force-Merge-Older-Than-Months
`Schedule Cron Pattern`: 0 1 * * *

YAML:

```
actions:
  '1':
    action: forcemerge
    description: Force merge on older monthly indices
    options:
      max_num_segments: 1
      ignore_empty_list: true
      continue_if_exception: false
      delay: 60
    filters:
      - filtertype: pattern
        kind: timestring
        value: '%Y.%m$'
```

(continues on next page)

(continued from previous page)

```
      - filtertype: age
        source: creation_date
        direction: older
        unit: days
        unit_count: 32
      - filtertype: forcemerged
        max_num_segments: 1
        exclude: True
```

Note

Logtrail indices are created automatically by the internal audit logging pipeline. The following two preinstalled actions manage their size and retention.

Logtrail-default-delete

This action leaves only two last indices from each logtrail rollover index (allows for up to 10GB of data). The action is performed daily at 03:30.

Action type: CUSTOM
Action name: Logtrail-default-delete
Schedule Cron Pattern: 30 3 * * *

YAML:

```
actions:
  '1':
    action: delete_indices
    description: >-
      Leave only two last indices from each logtrail rollover index - allows␣
↪for up to
      10GB data.
    options:
      ignore_empty_list: true
      continue_if_exception: true
    filters:
      - filtertype: count
        count: 2
        pattern: '^logtrail-(.*?)-\d{4}.\d{2}.\d{2}-\d+$'
        reverse: true
```

Logtrail-default-rollover

This action rolls over default Logtrail indices. The action is performed every hour at minute 5.

Action type: CUSTOM
Action name: Logtrail-default-rollover
Schedule Cron Pattern: 5 * * * *

YAML:

```
actions:
  '1':
    action: rollover
    description: >-
      This action works on default logtrail indices. It is recommended to␣
↪enable
      it.
    options:
      name: logtrail-alert
      conditions:
        max_size: 5GB
      continue_if_exception: true
      allow_ilm_indices: true
  '2':
    action: rollover
    description: >-
      This action works on default logtrail indices. It is recommended to␣
↪enable
      it.
    options:
      name: logtrail-elasticsearch
      conditions:
        max_size: 5GB
      continue_if_exception: true
      allow_ilm_indices: true
  '3':
    action: rollover
    description: >-
      This action works on default logtrail indices. It is recommended to␣
↪enable
      it.
    options:
      name: logtrail-kibana
      conditions:
        max_size: 5GB
      continue_if_exception: true
      allow_ilm_indices: true
  '4':
    action: rollover
    description: >-
      This action works on default logtrail indices. It is recommended to␣
↪enable
      it.
    options:
      name: logtrail-logstash
      conditions:
        max_size: 5GB
      continue_if_exception: true
      allow_ilm_indices: true
```

SQL Query Language

Energy Logserver SQL lets you write queries in SQL rather than the Query domain-specific language (DSL). This section covers practical examples and complete SQL syntax reference.

- *SIEM Examples*
  - *Example 1: Check number of failed login attempts*
  - *Example 2: Gather host data from different sources in one place using* JOIN
  - *Example 3: See MAC addresses and their assigned IP addresses:*
  - *Example 4: Check total number of warnings from* syslog*:*
  - *Example 5: Check number of failed login attempts for every client:*
  - *Basic queries*
  - *Complex queries*
  - *Functions*
  - *JSON Support*
  - *Metadata queries*
  - *Aggregate functions*
  - *Delete*

SIEM Examples

Use SQL query to get security related data:

Example 1: Check number of failed login attempts

Query:

```
SELECT COUNT(sys.client.ip) AS failed_login_attempts FROM syslog-2024.02.23␣
↪sys
WHERE postfix_message = "SASL LOGIN authentication failed: UGFzc3dvcmQ6"
```

Result:

```
        failed_login_attempts
1       1329
```

Example 2: Gather host data from different sources in one place using *JOIN*

Query:

```
SELECT syslog.host.ip, syslog.host.name, server_addr, mac
FROM syslog-2024.02.23 syslog JOIN stream-2024.02.23 ON server_addr = syslog.
↪host.ip
```

Result:

```
       syslog.host.ip  syslog.host.name        server_addr     mac
       192.168.10.1       example-hostname-1   192.168.10.
↪1    bc:24:11:0g:f9:28
       192.168.10.1       example-hostname-1   192.168.10.
↪1    bc:24:11:0g:f9:28
       192.168.10.2       example-hostname-2   192.168.10.
↪2    bs:12:18:0g:f2:66
       192.168.10.1       example-hostname-1   192.168.10.
↪1    bc:24:11:0g:f9:28
       192.168.10.1       example-hostname-1   192.168.10.
↪1    bc:24:11:0g:f9:28
```

Example 3: See MAC addresses and their assigned IP addresses:

Query:

```
POST /_plugins/_sql
{
  "query" : "SELECT mac, client_addr FROM stream-2024.02.20 WHERE netflow.
↪type ='dhcp'"
}
```

Result:

```
{
  "schema": [
    {
      "name": "mac",
      "type": "keyword"
    },
    {
      "name": "client_addr",
      "type": "keyword"
    }
  ],
  "total": 3369,
  "datarows": [
    [
      "cs:96:d5:98:55:72",
      "10.7.7.2"
    ],
    [
      "bw:85:58:6a:b7:5a",
      "10.7.7.232"
    ],
    [
      "bc:34:11:0d:s9:88",
      "192.168.10.200"
    ]
    ]
    }
```

Example 4: Check total number of warnings from *syslog*:

Query:

```
SELECT COUNT(sys.syslog_severity_code) AS warnings_total FROM syslog-2024.02.
↪23 sys
WHERE syslog_severity = "warning"
```

Result:

```
warnings_total
429822
```

Example 5: Check number of failed login attempts for every client:

Query:

```
SELECT sys.client.ip, COUNT(*) AS failed_login_attempts FROM syslog-2024.02.
↪23 sys
WHERE  postfix_message = "SASL LOGIN authentication failed: CGZFzc3evxmQ6"
GROUP BY sys.client.ip
```

Result:

```
client.ip       failed_login_attempts
144.220.71.224  3
127.174.131.70  2
146.114.84.226  3
142.169.105.19  5
142.180.112.241 3
155.211.74.18   1
```

SQL in Energy Logserver bridges the gap between traditional relational database concepts and the flexibility of Energy Logserver's document-oriented data storage. This integration gives you the ability to use your SQL knowledge to query, analyze, and extract insights from your data.

SQL and Energy Logserver terminology

Here's how core SQL concepts map to Energy Logserver:

SQL	Energy Logserver
Table	Index
Row	Document
Column	Field

REST API

To use the SQL plugin with your own applications, send requests to the `_plugins/_sql` endpoint:

```
POST _plugins/_sql
{
  "query": "SELECT * FROM my-index LIMIT 50"
}
```

You can query multiple indexes by using a comma-separated list:

```
POST _plugins/_sql
{
  "query": "SELECT * FROM my-index1,myindex2,myindex3 LIMIT 50"
}
```

You can also specify an index pattern with a wildcard expression:

```
POST _plugins/_sql
{
  "query": "SELECT * FROM my-index* LIMIT 50"
}
```

To run the above query in the command line, use the curl command:

```
curl -XPOST https://localhost:9200/_plugins/_sql -u 'admin:admin' -k -H
↪'Content-Type: application/json' -d '{"query": "SELECT * FROM my-index*␣
↪LIMIT 50"}'
```

You can specify the *response format* as JDBC, standard Energy Logserver JSON, CSV, or raw. By default, queries return data in JDBC format. The following query sets the format to JSON:

```
POST _plugins/_sql?format=json
{
  "query": "SELECT * FROM my-index LIMIT 50"
}
```

See the rest of this guide for more information about request parameters, settings, supported operations, and tools.

Basic queries

Use the `SELECT` clause, along with `FROM`, `WHERE`, `GROUP BY`, `HAVING`, `ORDER BY`, and `LIMIT` to search and aggregate data.

Among these clauses, `SELECT` and `FROM` are required, as they specify which fields to retrieve and which indexes to retrieve them from. All other clauses are optional. Use them according to your needs.

Syntax

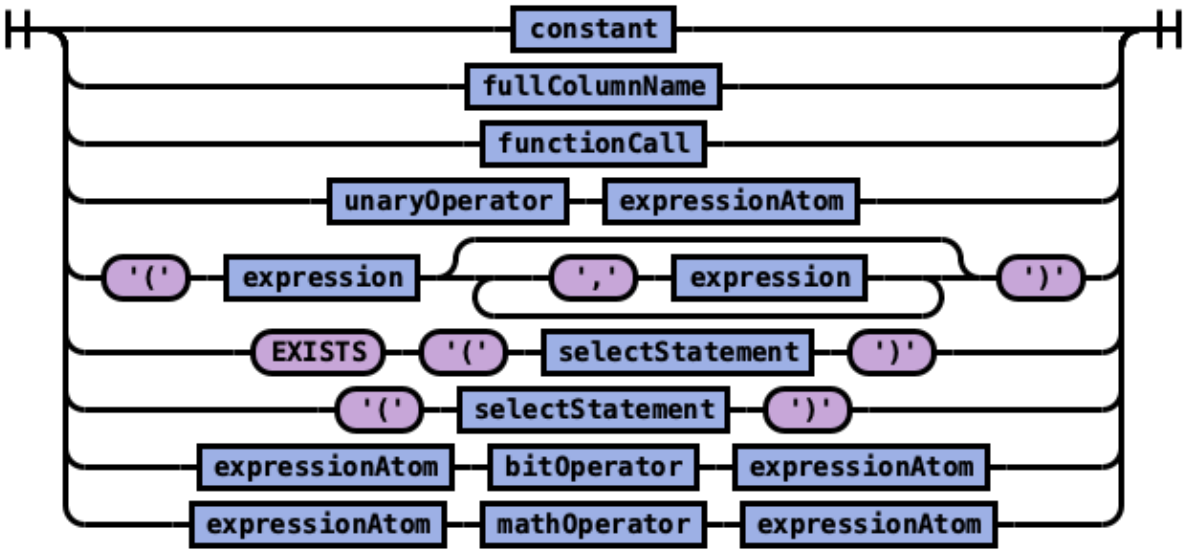
The complete syntax for searching and aggregating data is as follows:

```
SELECT [DISTINCT] (* | expression) [[AS] alias] [, ...]
FROM index_name
[WHERE predicates]
[GROUP BY expression [, ...]
 [HAVING predicates]]
[ORDER BY expression [IS [NOT] NULL] [ASC | DESC] [, ...]]
[LIMIT [offset, ] size]
```

Fundamentals

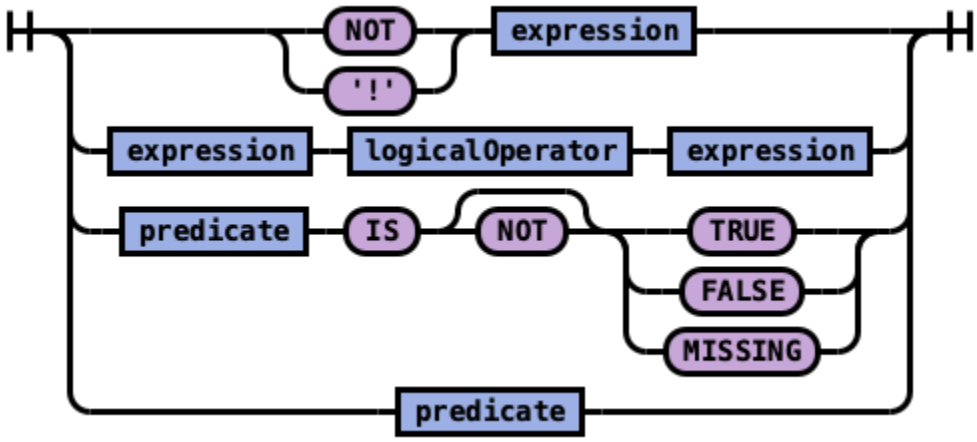
Apart from the predefined keywords of SQL, the most basic elements are literal and identifiers. A literal is a numeric, string, date or boolean constant. An identifier is an Energy Logserver index or field name. With arithmetic operators and SQL functions, use literals and identifiers to build complex expressions.

Rule `expressionAtom`:

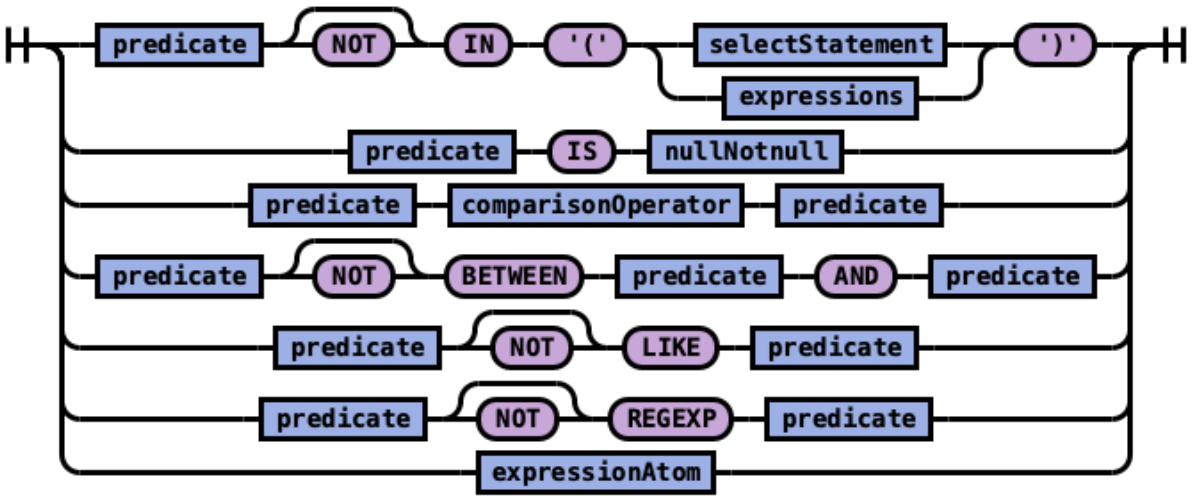


The expression in turn can be combined into a predicate with logical operator. Use a predicate in the `WHERE` and `HAVING` clause to filter out data by specific conditions.

Rule `expression`:



Rule `predicate`:



Execution Order

These SQL clauses execute in an order different from how they appear:

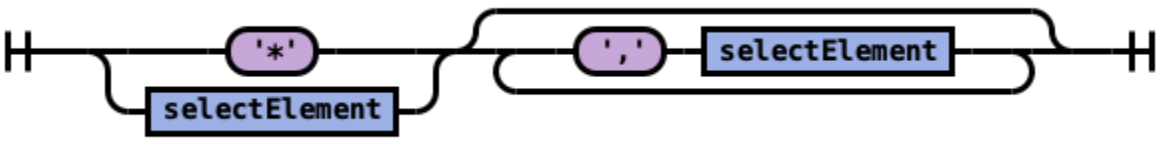
```
FROM index
 WHERE predicates
  GROUP BY expressions
   HAVING predicates
    SELECT expressions
     ORDER BY expressions
      LIMIT size
```

Select

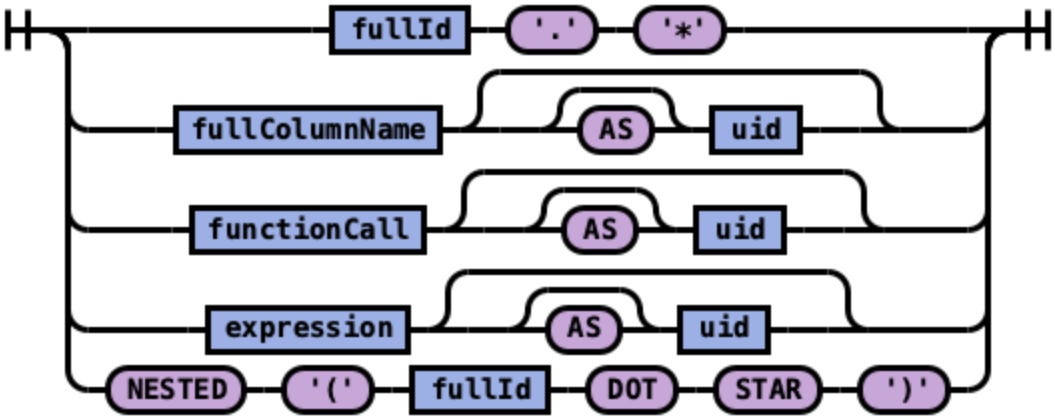
Specify the fields to be retrieved.

Syntax

Rule `selectElements`:



Rule `selectElement`:



Example 1: Use * to retrieve all fields in an index:

```
SELECT *
FROM accounts
```

account_numb	firstname	gender	city	balance	employer	state	email	address	lastname	age
1	Amber	M	Brogan	39225	Pyrami	IL	amberduke@pyrami.cor	880 Holmes Lane	Duke	32
16	Hattie	M	Dante	5686	Netagy	TN	hattiebond@netagy.c	671 Bristol Street	Bond	36
13	Nanette	F	Nogal	32838	Quility	VA	nanettebates@quility.con	789 Madison Street	Bates	28
18	Dale	M	Orick	4180		MD	daleadams@boink	467 Hutchinson Court	Adams	33

Example 2: Use field name(s) to retrieve only specific fields:

```
SELECT firstname, lastname
FROM accounts
```

firstname	lastname
Amber	Duke
Hattie	Bond
Nanette	Bates
Dale	Adams

Example 3: Use field aliases instead of field names. Field aliases are used to make field names more readable:

```
SELECT account_number AS num
FROM accounts
```

num
1
6
13
18

Example 4: Use the `DISTINCT` clause to get back only unique field values. You can specify one or more field names:

```
SELECT DISTINCT age
FROM accounts
```

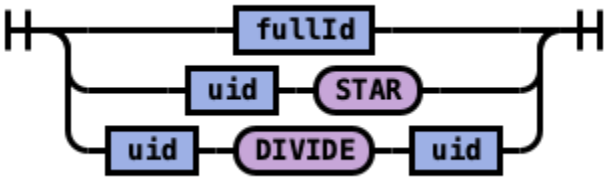
age
28
32
33
36

From

Specify the index that you want search. You can specify subqueries within the `FROM` clause.

Syntax

Rule `tableName`:



Example 1: Use index aliases to query across indexes. In this sample query, `acc` is an alias for the `accounts` index:

```
SELECT account_number, accounts.age
FROM accounts
```

or

```
SELECT account_number, acc.age
FROM accounts acc
```

account_number	age
1	32
6	36
13	28
18	33

Example 2: Use index patterns to query indexes that match a specific pattern:

```
SELECT account_number
FROM account*
```

account_number
1
6
13
18

Where

Specify a condition to filter the results.

Operators	Behavior
=	Equal to.
<>	Not equal to.
>	Greater than.
<	Less than.
>=	Greater than or equal to.
<=	Less than or equal to.
IN	Specify multiple `OR` operators.
BETWEEN	Similar to a range query.
LIKE	Use for full-text search. For more information about full-text queries.
IS NULL	Check if the field value is `NULL`.
IS NOT NULL	Check if the field value is `NOT NULL`.

Combine comparison operators (=, <>, >, >=, <, <=) with boolean operators `NOT`, `AND`, or `OR` to build more complex expressions.

Example 1: Use comparison operators for numbers, strings, or dates:

```
SELECT account_number
FROM accounts
WHERE account_number = 1
```

account_number
1

Example 2: Energy Logserver allows for flexible schema, so documents in an index may have different fields. Use `IS NULL` or `IS NOT NULL` to retrieve only missing fields or existing fields. Energy Logserver does not differentiate between missing fields and fields explicitly set to `NULL`:

```
SELECT account_number, employer
FROM accounts
WHERE employer IS NULL
```

account_number	employer
18	

Example 3: Deletes a document that satisfies the predicates in the `WHERE` clause:

```
DELETE FROM accounts
WHERE age > 30
```

Group By

Group documents with the same field value into buckets.

Example 1: Group by fields:

```
SELECT age
FROM accounts
GROUP BY age
```

id	age
0	28
1	32
2	33
3	36

Example 2: Group by field alias:

```
SELECT account_number AS num
FROM accounts
GROUP BY num
```

id	num
0	1
1	6
2	13
3	18

Example 4: Use scalar functions in the `GROUP BY` clause:

```
SELECT ABS(age) AS a
FROM accounts
GROUP BY ABS(age)
```

id	a
0	28.0
1	32.0
2	33.0
3	36.0

Having

Use the `HAVING` clause to aggregate inside each bucket based on aggregation functions (`COUNT`, `AVG`, `SUM`, `MIN`, and `MAX`). The `HAVING` clause filters results from the `GROUP BY` clause:

Example 1:

```
SELECT age, MAX(balance)
FROM accounts
GROUP BY age HAVING MIN(balance) > 10000
```

id	age	MAX (balance)
0	28	32838
1	32	39225

Order By

Use the `ORDER BY` clause to sort results into your desired order.

Example 1: Use `ORDER BY` to sort by ascending or descending order. Besides regular field names, using `ordinal`, `alias`, or `scalar` functions are supported:

```
SELECT account_number
FROM accounts
ORDER BY account_number DESC
```

account_number
18
13
6
1

Example 2: Specify if documents with missing fields are to be put at the beginning or at the end of the results. The default behavior of Energy Logserver is to return nulls or missing fields at the end. To push them before non-nulls, use the `IS NOT NULL` operator:

```
SELECT employer
FROM accounts
ORDER BY employer IS NOT NULL
```

employer
Netagy
Pyrami
Quility

Limit

Specify the maximum number of documents that you want to retrieve. Used to prevent fetching large amounts of data into memory.

Example 1: If you pass in a single argument, it's mapped to the `size` parameter in Energy Logserver and the `from` parameter is set to 0.

```
SELECT account_number
FROM accounts
ORDER BY account_number LIMIT 1
```

account_number
1

Example 2: If you pass in two arguments, the first is mapped to the `from` parameter and the second to the `size` parameter in Energy Logserver. You can use this for simple pagination for small indexes, as it's inefficient for large indexes. Use `ORDER BY` to ensure the same order between pages:

```
SELECT account_number
FROM accounts
ORDER BY account_number LIMIT 1, 1
```

account_number
6

Complex queries

Besides simple SFW (`SELECT-FROM-WHERE`) queries, the SQL plugin supports complex queries such as subquery, join, union, and minus. These queries operate on more than one Energy Logserver index. To examine how these queries execute behind the scenes, use the `explain` operation.

Joins

Energy Logserver SQL supports inner joins, cross joins, and left outer joins.

Constraints

Joins have a number of constraints:

1. You can only join two indexes.
2. You must use aliases for indexes (for example, `people p`).
3. Within an ON clause, you can only use AND conditions.
4. In a WHERE statement, don't combine trees that contain multiple indexes. For example, the following statement works:

   ```
   WHERE (a.type1 > 3 OR a.type1 < 0) AND (b.type2 > 4 OR b.type2 < -1)
   ```

   The following statement does not:

   ```
   WHERE (a.type1 > 3 OR b.type2 < 0) AND (a.type1 > 4 OR b.type2 < -1)
   ```

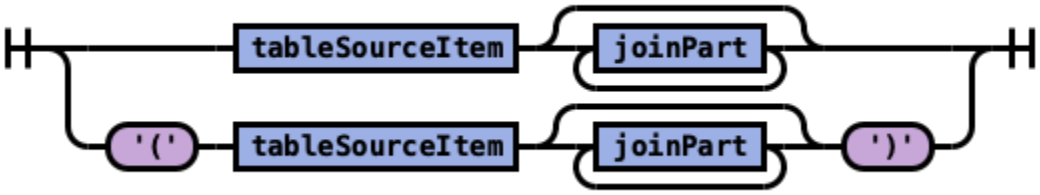
5. You can't use GROUP BY or ORDER BY for results.
6. LIMIT with OFFSET (e.g. `LIMIT 25 OFFSET 25`) is not supported.

Description

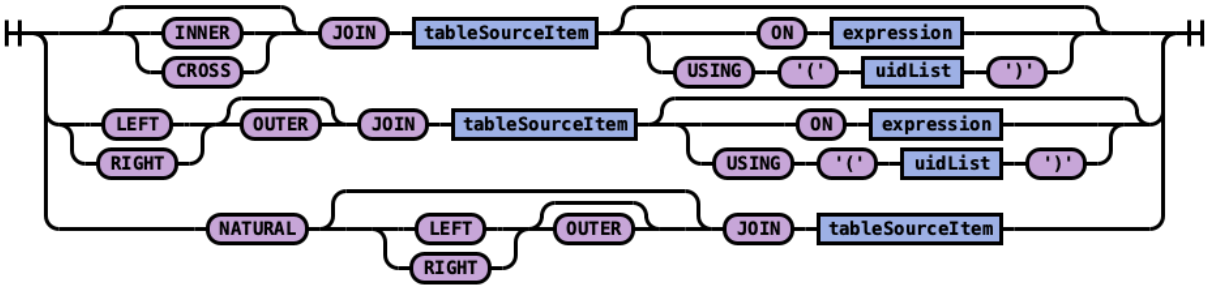
The `JOIN` clause combines columns from one or more indexes using values common to each.

Syntax

Rule `tableSource`:



Rule `joinPart`:



Example 1: Inner join

Inner join creates a new result set by combining columns of two indexes based on your join predicates. It iterates the two indexes and compares each document to find the ones that satisfy the join predicates. You can optionally precede the `JOIN` clause with an `INNER` keyword.

The join predicate(s) is specified by the ON clause.

SQL query:

```
SELECT
  a.account_number, a.firstname, a.lastname,
  e.id, e.name
FROM accounts a
JOIN employees_nested e
 ON a.account_number = e.id
```

Explain:

The `explain` output is complicated, because a `JOIN` clause is associated with two Energy Logserver DSL queries that execute in separate query planner frameworks. You can interpret it by examining the `Physical Plan` and `Logical Plan` objects.

```
{
  "Physical Plan" : {
    "Project [ columns=[a.account_number, a.firstname, a.lastname, e.name, e.
```

(continues on next page)

(continued from previous page)

```
↪id] ]" : {
     "Top [ count=200 ]" : {
       "BlockHashJoin[ conditions=( a.account_number = e.id ), type=JOIN,␣
↪blockSize=[FixedBlockSize with size=10000] ]" : {
         "Scroll [ employees_nested as e, pageSize=10000 ]" : {
           "request" : {
             "size" : 200,
             "from" : 0,
             "_source" : {
               "excludes" : [ ],
               "includes" : [
                 "id",
                 "name"
               ]
             }
           }
         },
         "Scroll [ accounts as a, pageSize=10000 ]" : {
           "request" : {
             "size" : 200,
             "from" : 0,
             "_source" : {
               "excludes" : [ ],
               "includes" : [
                 "account_number",
                 "firstname",
                 "lastname"
               ]
             }
           }
         },
         "useTermsFilterOptimization" : false
       }
     }
   }
 },
 "description" : "Hash Join algorithm builds hash table based on result of␣
↪first query, and then probes hash table to find matched rows for each row␣
↪returned by second query",
 "Logical Plan" : {
   "Project [ columns=[a.account_number, a.firstname, a.lastname, e.name, e.
↪id] ]" : {
     "Top [ count=200 ]" : {
       "Join [ conditions=( a.account_number = e.id ) type=JOIN ]" : {
         "Group" : [
           {
             "Project [ columns=[a.account_number, a.firstname, a.lastname] ]
↪" : {
               "TableScan" : {
                 "tableAlias" : "a",
                 "tableName" : "accounts"
               }
```

(continues on next page)

(continued from previous page)

```
                }
              },
              {
                "Project [ columns=[e.name, e.id] ]" : {
                  "TableScan" : {
                    "tableAlias" : "e",
                    "tableName" : "employees_nested"
                  }
                }
              }
            ]
          }
        }
      }
    }
}
```

Result set:

a.account_number	a.firstname	a.lastname	e.id	e.name
6	Hattie	Bond	6	Jane Smith

Example 2: Cross join

Cross join, also known as cartesian join, combines each document from the first index with each document from the second. The result set is the cartesian product of documents of both indexes. This operation is similar to the inner join without the `ON` clause that specifies the join condition.

> **Warning**
>
> It's risky to perform cross join on two indexes of large or even medium size. It might trigger a circuit breaker that terminates the query to avoid running out of memory.

SQL query:

```
SELECT
  a.account_number, a.firstname, a.lastname,
  e.id, e.name
FROM accounts a
JOIN employees_nested e
```

Result set:

a.account_number	a.firstname	a.lastname	e.id	e.name
1	Amber	Duke	3	Bob Smith
1	Amber	Duke	4	Susan Smith
1	Amber	Duke	6	Jane Smith
6	Hattie	Bond	3	Bob Smith
6	Hattie	Bond	4	Susan Smith
6	Hattie	Bond	6	Jane Smith
13	Nanette	Bates	3	Bob Smith
13	Nanette	Bates	4	Susan Smith
13	Nanette	Bates	6	Jane Smith
18	Dale	Adams	3	Bob Smith
18	Dale	Adams	4	Susan Smith
18	Dale	Adams	6	Jane Smith

Example 3: Left outer join

Use left outer join to retain rows from the first index if it does not satisfy the join predicate. The keyword `OUTER` is optional.

SQL query:

```
SELECT
  a.account_number, a.firstname, a.lastname,
  e.id, e.name
FROM accounts a
LEFT JOIN employees_nested e
 ON a.account_number = e.id
```

Result set:

a.account_number	a.firstname	a.lastname	e.id	e.name
1	Amber	Duke	null	null
6	Hattie	Bond	6	Jane Smith
13	Nanette	Bates	null	null
18	Dale	Adams	null	null

Subquery

A subquery is a complete `SELECT` statement used within another statement and enclosed in parenthesis. From the explain output, you can see that some subqueries are actually transformed to an equivalent join query to execute.

Example 1: Table subquery

SQL query:

```
SELECT a1.firstname, a1.lastname, a1.balance
FROM accounts a1
WHERE a1.account_number IN (
  SELECT a2.account_number
  FROM accounts a2
  WHERE a2.balance > 10000
)
```

Explain:

```
{
  "Physical Plan" : {
    "Project [ columns=[a1.balance, a1.firstname, a1.lastname] ]" : {
      "Top [ count=200 ]" : {
        "BlockHashJoin[ conditions=( a1.account_number = a2.account_number ),␣
↪type=JOIN, blockSize=[FixedBlockSize with size=10000] ]" : {
          "Scroll [ accounts as a2, pageSize=10000 ]" : {
            "request" : {
              "size" : 200,
              "query" : {
                "bool" : {
                  "filter" : [
                    {
                      "bool" : {
                        "adjust_pure_negative" : true,
                        "must" : [
                          {
                            "bool" : {
                              "adjust_pure_negative" : true,
                              "must" : [
                                {
                                  "bool" : {
                                    "adjust_pure_negative" : true,
                                    "must_not" : [
                                      {
                                        "bool" : {
                                          "adjust_pure_negative" : true,
                                          "must_not" : [
                                            {
                                              "exists" : {
                                                "field" : "account_number",
                                                "boost" : 1
                                              }
                                            }
                                          ],
```

(continues on next page)

(continued from previous page)

```
                                        "boost" : 1
                                      }
                                    }
                                  ],
                                  "boost" : 1
                                }
                              },
                              {
                                "range" : {
                                  "balance" : {
                                    "include_lower" : false,
                                    "include_upper" : true,
                                    "from" : 10000,
                                    "boost" : 1,
                                    "to" : null
                                  }
                                }
                              }
                            ],
                            "boost" : 1
                          }
                        }
                      ],
                      "boost" : 1
                    }
                  }
                ],
                "adjust_pure_negative" : true,
                "boost" : 1
              }
            },
            "from" : 0
          }
        },
        "Scroll [ accounts as a1, pageSize=10000 ]" : {
          "request" : {
            "size" : 200,
            "from" : 0,
            "_source" : {
              "excludes" : [ ],
              "includes" : [
                "firstname",
                "lastname",
                "balance",
                "account_number"
              ]
            }
          }
        },
        "useTermsFilterOptimization" : false
      }
    }
```

(continues on next page)

(continued from previous page)

```
    }
  },
  "description" : "Hash Join algorithm builds hash table based on result of␣
↪first query, and then probes hash table to find matched rows for each row␣
↪returned by second query",
  "Logical Plan" : {
    "Project [ columns=[a1.balance, a1.firstname, a1.lastname] ]" : {
      "Top [ count=200 ]" : {
        "Join [ conditions=( a1.account_number = a2.account_number )␣
↪type=JOIN ]" : {
          "Group" : [
            {
              "Project [ columns=[a1.balance, a1.firstname, a1.lastname, a1.
↪account_number] ]" : {
                "TableScan" : {
                  "tableAlias" : "a1",
                  "tableName" : "accounts"
                }
              }
            },
            {
              "Project [ columns=[a2.account_number] ]" : {
                "Filter [ conditions=[AND ( AND account_number ISN null, AND␣
↪balance GT 10000 ) ] ]" : {
                  "TableScan" : {
                    "tableAlias" : "a2",
                    "tableName" : "accounts"
                  }
                }
              }
            }
          ]
        }
      }
    }
  }
}
```

Result set:

a1.firstname	a1.lastname	a1.balance
Amber	Duke	39225
Nanette	Bates	32838

Example 2: From subquery

SQL query:

```
SELECT a.f, a.l, a.a
FROM (
  SELECT firstname AS f, lastname AS l, age AS a
```

(continues on next page)

(continued from previous page)

```
  FROM accounts
  WHERE age > 30
) AS a
```

Explain:

```
{
  "from" : 0,
  "size" : 200,
  "query" : {
    "bool" : {
      "filter" : [
        {
          "bool" : {
            "must" : [
              {
                "range" : {
                  "age" : {
                    "from" : 30,
                    "to" : null,
                    "include_lower" : false,
                    "include_upper" : true,
                    "boost" : 1.0
                  }
                }
              }
            ],
            "adjust_pure_negative" : true,
            "boost" : 1.0
          }
        }
      ],
      "adjust_pure_negative" : true,
      "boost" : 1.0
    }
  },
  "_source" : {
    "includes" : [
      "firstname",
      "lastname",
      "age"
    ],
    "excludes" : [ ]
  }
}
```

Result set:

f	l	a
Amber	Duke	32
Dale	Adams	33
Hattie	Bond	36

Functions

The SQL language supports all SQL plugin common functions, including relevance search, but also introduces a few function synonyms, which are available in SQL only. These synonyms are provided by the `V1` engine.

Match query

The `MATCHQUERY` and `MATCH_QUERY` functions are synonyms for the MATCH relevance function. They don't accept additional arguments but provide an alternate syntax.

Syntax

To use `matchquery` or `match_query`, pass in your search query and the field name that you want to search against:

```
match_query(field_expression, query_expression[, option=<option_value>]*)
matchquery(field_expression, query_expression[, option=<option_value>]*)
field_expression = match_query(query_expression[, option=<option_value>]*)
field_expression = matchquery(query_expression[, option=<option_value>]*)
```

You can specify the following options in any order:

- `analyzer`
- `boost`

Example

You can use `MATCHQUERY` to replace `MATCH`:

```
SELECT account_number, address
FROM accounts
WHERE MATCHQUERY(address, 'Holmes')
```

Alternatively, you can use `MATCH_QUERY` to replace `MATCH`:

```
SELECT account_number, address
FROM accounts
WHERE address = MATCH_QUERY('Holmes')
```

The results contain documents in which the address contains "Holmes":

account_number	address
1	880 Holmes Lane

Multi-match

There are three synonyms for `MULTI_MATCH`, each with a slightly different syntax. They accept a query string and a fields list with weights. They can also accept additional optional parameters.

Syntax

```
multimatch('query'=query_expression[, 'fields'=field_expression][, option=
↪<option_value>]*)
```

(continues on next page)

(continued from previous page)

```
multi_match('query'=query_expression[, 'fields'=field_expression][, option=
↪<option_value>]*)
multimatchquery('query'=query_expression[, 'fields'=field_expression][,␣
↪option=<option_value>]*)
```

The `fields` parameter is optional and can contain a single field or a comma-separated list (whitespace characters are not allowed). The weight for each field is optional and is specified after the field name. It should be delimited by the `caret` character – ^ – without whitespace.

Example

The following queries show the `fields` parameter of a multi-match query with a single field and a field list:

```
multi_match('fields' = "Tags^2,Title^3.4,Body,Comments^0.3", ...)
multi_match('fields' = "Title", ...)
```

You can specify the following options in any order:

- `analyzer`
- `boost`
- `slop`
- `type`
- `tie_breaker`
- `operator`

Query string

The `QUERY` function is a synonym for `QUERY_STRING`.

Syntax

```
query('query'=query_expression[, 'fields'=field_expression][, option=<option_
↪value>]*)
```

The `fields` parameter is optional and can contain a single field or a comma-separated list (whitespace characters are not allowed). The weight for each field is optional and is specified after the field name. It should be delimited by the `caret` character – ^ – without whitespace.

Example

The following queries show the `fields` parameter of a multi-match query with a single field and a field list:

```
query('fields' = "Tags^2,Title^3.4,Body,Comments^0.3", ...)
query('fields' = "Tags", ...)
```

You can specify the following options in any order:

- `analyzer`
- `boost`
- `slop`

- default_field

Example of using query_string in SQL and PPL queries:

The following is a sample REST API search request in Energy Logserver DSL.

```
GET accounts/_search
{
  "query": {
    "query_string": {
      "query": "Lane Street",
      "fields": [ "address" ],
    }
  }
}
```

The request above is equivalent to the following query function:

```
SELECT account_number, address
FROM accounts
WHERE query('address:Lane OR address:Street')
```

The results contain addresses that contain “Lane” or “Street”:

account_number	address
1	880 Holmes Lane
6	671 Bristol Street
13	789 Madison Street

Match phrase

The MATCHPHRASEQUERY function is a synonym for MATCH_PHRASE.

Syntax

```
matchphrasequery(query_expression, field_expression[, option=<option_value>]*)
```

You can specify the following options in any order:

- analyzer
- boost
- slop

Score query

To return a relevance score along with every matching document, use the SCORE, SCOREQUERY, or SCORE_QUERY functions.

Syntax

The `SCORE` function expects two arguments. The first argument is the *`MATCH_QUERY`* expression. The second argument is an optional floating-point number to boost the score (the default value is 1.0):

```
SCORE(match_query_expression, score)
SCOREQUERY(match_query_expression, score)
SCORE_QUERY(match_query_expression, score)
```

Example

The following example uses the `SCORE` function to boost the documents' scores:

```
SELECT account_number, address, _score
FROM accounts
WHERE SCORE(MATCH_QUERY(address, 'Lane'), 0.5) OR
  SCORE(MATCH_QUERY(address, 'Street'), 100)
ORDER BY _score
```

The results contain matches with corresponding scores:

account_number	address	score
1	880 Holmes Lane	0.5
6	671 Bristol Street	100
13	789 Madison Street	100

Wildcard query

To search documents by a given wildcard, use the `WILDCARDQUERY` or `WILDCARD_QUERY` functions.

Syntax

```
wildcardquery(field_expression, query_expression[, boost=<value>])
wildcard_query(field_expression, query_expression[, boost=<value>])
```

Example

The following example uses a wildcard query:

```
SELECT account_number, address
FROM accounts
WHERE wildcard_query(address, '*Holmes*');
```

The results contain documents that match the wildcard expression:

account_number	address
1	880 Holmes Lane

JSON Support

SQL plugin supports JSON by following PartiQL specification, a SQL-compatible query language that lets you query semi-structured and nested data for any data format. The SQL plugin only supports a subset of the PartiQL specification.

Querying nested collection

PartiQL extends SQL to allow you to query and unnest nested collections. In Energy Logserver, this is very useful to query a JSON index with nested objects or fields.

To follow along, use the `bulk` operation to index some sample data:

```
POST employees_nested/_bulk?refresh
{"index":{"_id":"1"}}
{"id":3,"name":"Bob Smith","title":null,"projects":[{"name":"SQL Spectrum␣
↪querying","started_year":1990},{"name":"SQL security","started_year":1999},{
↪"name":"Logserver security","started_year":2015}]}
{"index":{"_id":"2"}}
{"id":4,"name":"Susan Smith","title":"Dev Mgr","projects":[]}
{"index":{"_id":"3"}}
{"id":6,"name":"Jane Smith","title":"Software Eng 2","projects":[{"name":"SQL␣
↪security","started_year":1998},{"name":"Hello security","started_year":2015,
↪"address":[{"city":"Dallas","state":"TX"}]}]}
```

Example 1: Unnesting a nested collection

This example finds the nested document (`projects`) with a field value (`name`) that satisfies the predicate (contains `security`). Because each parent document can have more than one nested documents, the nested document that matches is flattened. In other words, the final result is the cartesian product between the parent and nested documents.

```
SELECT e.name AS employeeName,
       p.name AS projectName
FROM employees_nested AS e,
       e.projects AS p
WHERE p.name LIKE '%security%'
```

Explain:

```
{
  "from" : 0,
  "size" : 200,
  "query" : {
    "bool" : {
      "filter" : [
        {
          "bool" : {
            "must" : [
              {
                "nested" : {
                  "query" : {
                    "wildcard" : {
                      "projects.name" : {
                        "wildcard" : "*security*",
```

(continues on next page)

(continued from previous page)

```
                        "boost" : 1.0
                      }
                    }
                  },
                  "path" : "projects",
                  "ignore_unmapped" : false,
                  "score_mode" : "none",
                  "boost" : 1.0,
                  "inner_hits" : {
                    "ignore_unmapped" : false,
                    "from" : 0,
                    "size" : 3,
                    "version" : false,
                    "seq_no_primary_term" : false,
                    "explain" : false,
                    "track_scores" : false,
                    "_source" : {
                      "includes" : [
                        "projects.name"
                      ],
                      "excludes" : [ ]
                    }
                  }
                }
              }
            ],
            "adjust_pure_negative" : true,
            "boost" : 1.0
          }
        }
      ],
      "adjust_pure_negative" : true,
      "boost" : 1.0
    }
  },
  "_source" : {
    "includes" : [
      "name"
    ],
    "excludes" : [ ]
  }
}
```

Result set:

employeeName	projectName
Bob Smith	Energy Logserver Security
Bob Smith	SQL security
Jane Smith	Hello security
Jane Smith	SQL security

Example 2: Unnesting in existential subquery

To unnest a nested collection in a subquery to check if it satisfies a condition:

```
SELECT e.name AS employeeName
FROM employees_nested AS e
WHERE EXISTS (
    SELECT *
    FROM e.projects AS p
    WHERE p.name LIKE '%security%'
)
```

Explain:

```
{
  "from" : 0,
  "size" : 200,
  "query" : {
    "bool" : {
      "filter" : [
        {
          "bool" : {
            "must" : [
              {
                "nested" : {
                  "query" : {
                    "bool" : {
                      "must" : [
                        {
                          "bool" : {
                            "must" : [
                              {
                                "bool" : {
                                  "must_not" : [
                                    {
                                      "bool" : {
                                        "must_not" : [
                                          {
                                            "exists" : {
                                              "field" : "projects",
                                              "boost" : 1.0
                                            }
                                          }
                                        ],
                                        "adjust_pure_negative" : true,
                                        "boost" : 1.0
                                      }
                                    }
                                  ],
                                  "adjust_pure_negative" : true,
                                  "boost" : 1.0
                                }
                              },
                              {
```

(continues on next page)

(continued from previous page)

```
                                "wildcard" : {
                                  "projects.name" : {
                                    "wildcard" : "*security*",
                                    "boost" : 1.0
                                  }
                                }
                              }
                            ],
                            "adjust_pure_negative" : true,
                            "boost" : 1.0
                          }
                        }
                      ],
                      "adjust_pure_negative" : true,
                      "boost" : 1.0
                    }
                  },
                  "path" : "projects",
                  "ignore_unmapped" : false,
                  "score_mode" : "none",
                  "boost" : 1.0
                }
              }
            ],
            "adjust_pure_negative" : true,
            "boost" : 1.0
          }
        }
      ],
      "adjust_pure_negative" : true,
      "boost" : 1.0
    }
  },
  "_source" : {
    "includes" : [
      "name"
    ],
    "excludes" : [ ]
  }
}
```

Result set:

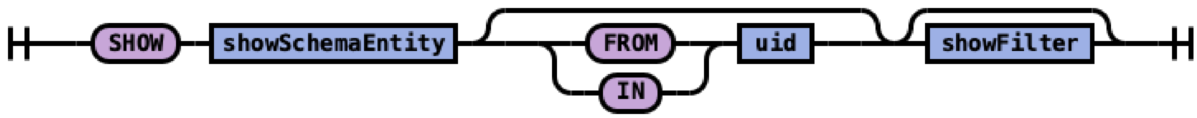
| employeeName | :— | :— Bob Smith | Jane Smith |

Metadata queries

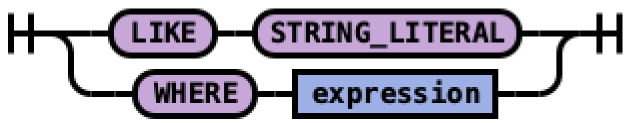
To see basic metadata about your indexes, use the `SHOW` and `DESCRIBE` commands.

Syntax

Rule `showStatement`:



Rule `showFilter`:



Example 1: See metadata for indexes

To see metadata for indexes that match a specific pattern, use the `SHOW` command. Use the wildcard `%` to match all indexes:

```
SHOW TABLES LIKE %
```

| TABLE_CAT | TABLE_SCHEM | TABLE_NAME | TABLE_TYPE | REMARKS | TYPE_CAT | TYPE_SCHEM | TYPE_NAME | SELF_REFERENCING_COL_NAME | REF_GENERATION :— | :— docker-cluster | null | accounts | BASE TABLE | null | null | null | null | null | null docker-cluster | null | employees_nested | BASE TABLE | null | null | null | null | null | null

Example 2: See metadata for a specific index

To see metadata for an index name with a prefix of `acc`:

```
SHOW TABLES LIKE acc%
```

| TABLE_CAT | TABLE_SCHEM | TABLE_NAME | TABLE_TYPE | REMARKS | TYPE_CAT | TYPE_SCHEM | TYPE_NAME | SELF_REFERENCING_COL_NAME | REF_GENERATION :— | :— docker-cluster | null | accounts | BASE TABLE | null | null | null | null | null | null

Example 3: See metadata for fields

To see metadata for field names that match a specific pattern, use the `DESCRIBE` command:

```
DESCRIBE TABLES LIKE accounts
```

| TABLE_CAT | TABLE_SCHEM | TABLE_NAME | COLUMN_NAME | DATA_TYPE | TYPE_NAME | COLUMN_SIZE | BUFFER_LENGTH | DECIMAL_DIGITS | NUM_PREC_RADIX | NULLABLE | REMARKS | COLUMN_DEF | SQL_DATA_TYPE | SQL_DATETIME_SUB | CHAR_OCTET_LENGTH | ORDINAL_POSITION | IS_NULLABLE | SCOPE_CATALOG | SCOPE_SCHEMA | SCOPE_TABLE | SOURCE_DATA_TYPE | IS_AUTOINCREMENT | IS_GENERATEDCOLUMN :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— | :— docker-cluster | null | accounts | account_number | null | long | null | null | null | 10 | 2 | null | null | null | null | null | 1 | | null | null | null | null | NO | docker-cluster | null | accounts | firstname | null | text | null | null | null | 10 | 2 | null | null | null | null | null | 2 | | null | null | null | null | NO | docker-cluster | null | accounts | address | null | text | null | null | null | 10 | 2 | null | null | null | null | null | 3 | | null | null

| null | null | NO | docker-cluster | null | accounts | balance | null | long | null | null | null | 10 | 2 | null | null | null | null | null | 4 | | null | null | null | null | NO | docker-cluster | null | accounts | gender | null | text | null | null | null | 10 | 2 | null | null | null | null | null | 5 | | null | null | null | null | NO | docker-cluster | null | accounts | city | null | text | null | null | null | 10 | 2 | null | null | null | null | null | 6 | | null | null | null | null | NO | docker-cluster | null | accounts | employer | null | text | null | null | null | 10 | 2 | null | null | null | null | null | 7 | | null | null | null | null | NO | docker-cluster | null | accounts | state | null | text | null | null | null | 10 | 2 | null | null | null | null | null | 8 | | null | null | null | null | NO | docker-cluster | null | accounts | age | null | long | null | null | null | 10 | 2 | null | null | null | null | null | 9 | | null | null | null | null | NO | docker-cluster | null | accounts | email | null | text | null | null | null | 10 | 2 | null | null | null | null | null | 10 | | null | null | null | null | NO | docker-cluster | null | accounts | lastname | null | text | null | null | null | 10 | 2 | null | null | null | null | null | 11 | | null | null | null | null | NO |

Aggregate functions

Aggregate functions operate on subsets defined by the `GROUP BY` clause. In the absence of a `GROUP BY` clause, aggregate functions operate on all elements of the result set. You can use aggregate functions in the `GROUP BY`, `SELECT`, and `HAVING` clauses.

Energy Logserver supports the following aggregate functions.

Function	Description
`AVG`	Returns the average of the results.
`COUNT`	Returns the number of results.
`SUM`	Returns the sum of the results.
`MIN`	Returns the minimum of the results.
`MAX`	Returns the maximum of the results.
`VAR_POP` or `VARIANCE`	Returns the population variance of the results after discarding nulls. Returns 0 when there is only one row of results.
`VAR_SAMP`	Returns the sample variance of the results after discarding nulls. Returns null when there is only one row of results.
`STD` or `STDDEV`	Returns the sample standard deviation of the results. Returns 0 when there is only one row of results.
`STDDEV_POP`	Returns the population standard deviation of the results. Returns 0 when there is only one row of results.
`STDDEV_SAMP`	Returns the sample standard deviation of the results. Returns null when there is only one row of results.

The examples below reference an `employees` table. You can try out the examples by indexing the following documents into Energy Logserver using the bulk index operation:

```
PUT employees/_bulk?refresh
{"index":{"_id":"1"}}
{"employee_id": 1, "department":1, "firstname":"Amber", "lastname":"Duke",
↪"sales":1356, "sale_date":"2020-01-23"}
{"index":{"_id":"2"}}
{"employee_id": 1, "department":1, "firstname":"Amber", "lastname":"Duke",
↪"sales":39224, "sale_date":"2021-01-06"}
{"index":{"_id":"6"}}
{"employee_id":6, "department":1, "firstname":"Hattie", "lastname":"Bond",
↪"sales":5686, "sale_date":"2021-06-07"}
{"index":{"_id":"7"}}
{"employee_id":6, "department":1, "firstname":"Hattie", "lastname":"Bond",
↪"sales":12432, "sale_date":"2022-05-18"}
{"index":{"_id":"13"}}
```

(continues on next page)

(continued from previous page)

```
{"employee_id":13,"department":2, "firstname":"Nanette", "lastname":"Bates",
↪"sales":32838, "sale_date":"2022-04-11"}
{"index":{"_id":"18"}}
{"employee_id":18,"department":2, "firstname":"Dale", "lastname":"Adams",
↪"sales":4180, "sale_date":"2022-11-05"}
```

GROUP BY (Aggregate functions)

The `GROUP BY` clause defines subsets of a result set. Aggregate functions operate on these subsets and return one result row for each subset.

You can use an identifier, ordinal, or expression in the `GROUP BY` clause.

Using an identifier in GROUP BY

You can specify the field name (column name) to aggregate on in the `GROUP BY` clause. For example, the following query returns the department numbers and the total sales for each department:

```
SELECT department, sum(sales)
FROM employees
GROUP BY department;
```

department	sum(sales)
1	58700<
2	37018

Using an ordinal in GROUP BY

You can specify the column number to aggregate on in the `GROUP BY` clause. The column number is determined by the column position in the `SELECT` clause. For example, the following query is equivalent to the query above. It returns the department numbers and the total sales for each department. It groups the results by the first column of the result set, which is `department`:

```
SELECT department, sum(sales)
FROM employees
GROUP BY 1;
```

department	sum(sales)
1	58700
2	37018

Using an expression in GROUP BY

You can use an expression in the `GROUP BY` clause. For example, the following query returns the average sales for each year:

```
SELECT year(sale_date), avg(sales)
FROM employees
GROUP BY year(sale_date);
```

year(start_date)	avg(sales)
2020	1356.0
2021	22455.0
2022	16484.0

SELECT (Aggregate functions)

You can use aggregate expressions in the `SELECT` clause either directly or as part of a larger expression. In addition, you can use expressions as arguments of aggregate functions.

Using aggregate expressions directly in SELECT

The following query returns the average sales for each department:

```
SELECT department, avg(sales)
FROM employees
GROUP BY department;
```

department	avg(sales)
1	14675.0
2	18509.0

Using aggregate expressions as part of larger expressions in SELECT

The following query calculates the average commission for the employees of each department as 5% of the average sales:

```
SELECT department, avg(sales) * 0.05 as avg_commission
FROM employees
GROUP BY department;
```

department	avg_commission
1	733.75
2	925.45

Using expressions as arguments to aggregate functions

The following query calculates the average commission amount for each department. First it calculates the commission amount for each `sales` value as 5% of the `sales`. Then it determines the average of all commission values:

```
SELECT department, avg(sales * 0.05) as avg_commission
FROM employees
GROUP BY department;
```

department	avg_commission
1	733.75
2	925.45

COUNT

The `COUNT` function accepts arguments, such as `*`, or literals, such as `1`. The following table describes how various forms of the `COUNT` function operate.

Function type	Description
COUNT(field)	Counts the number of rows where the value of the given field (or expression) is not null.
COUNT(*)	Counts the total number of rows in a table.
COUNT(1) [same as COUNT(*)]	Counts any non-null literal.

For example, the following query returns the count of sales for each year:

```
SELECT year(sale_date), count(sales)
FROM employees
GROUP BY year(sale_date);
```

year(sale_date)	count(sales)
2020	1
2021	2
2022	3

HAVING (Aggregate functions)

Both `WHERE` and `HAVING` are used to filter results. The `WHERE` filter is applied before the `GROUP BY` phase, so you cannot use aggregate functions in a `WHERE` clause. However, you can use the `WHERE` clause to limit the rows to which the aggregate is then applied.

The `HAVING` filter is applied after the `GROUP BY` phase, so you can use the `HAVING` clause to limit the groups that are included in the results.

HAVING with GROUP BY

You can use aggregate expressions or their aliases defined in a `SELECT` clause in a `HAVING` condition.

The following query uses an aggregate expression in the `HAVING` clause. It returns the number of sales for each employee who made more than one sale:

```
SELECT employee_id, count(sales)
FROM employees
GROUP BY employee_id
HAVING count(sales) > 1;
```

employee_id	count(sales)
1	2
6	2

The aggregations in a `HAVING` clause do not have to be the same as the aggregations in a `SELECT` list. The following query uses the `count` function in the `HAVING` clause but the `sum` function in the `SELECT` clause. It returns the total sales amount for each employee who made more than one sale:

```
SELECT employee_id, sum(sales)
FROM employees
GROUP BY employee_id
HAVING count(sales) > 1;
```

employee_id	sum (sales)
1	40580
6	18120

As an extension of the SQL standard, you are not restricted to using only identifiers in the `GROUP BY` clause. The following query uses an alias in the `GROUP BY` clause and is equivalent to the previous query:

```
SELECT employee_id as id, sum(sales)
FROM employees
GROUP BY id
HAVING count(sales) > 1;
```

id	sum (sales)
1	40580
6	18120

You can also use an alias for an aggregate expression in the `HAVING` clause. The following query returns the total sales for each department where sales exceed $40,000:

```
SELECT department, sum(sales) as total
FROM employees
GROUP BY department
HAVING total > 40000;
```

department	total
1	58700

If an identifier is ambiguous (for example, present both as a `SELECT` alias and as an index field), the preference is given to the alias. In the following query the identifier is replaced with the expression aliased in the `SELECT` clause:

```
SELECT department, sum(sales) as sales
FROM employees
GROUP BY department
HAVING sales > 40000;
```

department	sales
1	58700

HAVING without GROUP BY

You can use a `HAVING` clause without a `GROUP BY` clause. In this case, the whole set of data is to be considered one group. The following query will return `True` if there is more than one value in the `department` column:

```
SELECT 'True' as more_than_one_department FROM employees HAVING␣
↪min(department) < max(department);
```

more_than_one_department
True

If all employees in the employee table belonged to the same department, the query returns the `more_than_one_department` column with zero rows.

Delete

The `DELETE` statement deletes documents that satisfy the predicates in the `WHERE` clause. If you don't specify the `WHERE` clause, all documents are deleted.

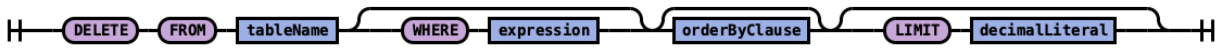
Setting

The `DELETE` statement is disabled by default. To enable the `DELETE` functionality in SQL, you need to update the configuration by sending the following request:

```
PUT _plugins/_query/settings
{
  "transient": {
    "plugins.sql.delete.enabled": "true"
  }
}
```

Syntax

Rule `singleDeleteStatement`:



Example

SQL query:

```
DELETE FROM accounts
WHERE age > 30
```

Explain:

```
{
  "size" : 1000,
  "query" : {
    "bool" : {
      "must" : [
```

(continues on next page)

(continued from previous page)

```
        {
          "range" : {
            "age" : {
              "from" : 30,
              "to" : null,
              "include_lower" : false,
              "include_upper" : true,
              "boost" : 1.0
            }
          }
        }
      ],
      "adjust_pure_negative" : true,
      "boost" : 1.0
    }
  },
  "_source" : false
}
```

Result set:

```
{
  "schema" : [
    {
      "name" : "deleted_rows",
      "type" : "long"
    }
  ],
  "total" : 1,
  "datarows" : [
    [
      3
    ]
  ],
  "size" : 1,
  "status" : 200
}
```

The `datarows` field shows the number of documents deleted.

PPL - Piped Processing Language

Piped Processing Language (PPL) is a query language that lets you use pipe (|) syntax to explore, discover, and query data stored in Energy Logserver.

To quickly get up and running with PPL, use **SQL** in Energy Logserver Dashboards.

The PPL syntax consists of commands delimited by the pipe character (|) where data flows from left to right through each pipeline.

```
search command | command 1 | command 2 ...
```

You can only use read-only commands like `search`, `where`, `fields`, `rename`, `dedup`, `stats`, `sort`, `eval`, `head`, `top`, and `rare`.

Quick start

To get started with PPL, choose **Dev Tools** in Energy Logserver Dashboards and use the `bulk` operation to index some sample data:

```
PUT accounts/_bulk?refresh
{"index":{"_id":"1"}}
{"account_number":1,"balance":39225,"firstname":"Amber","lastname":"Duke","age
 ↪":32,"gender":"M","address":"880 Holmes Lane","employer":"Pyrami","email":
 ↪"amberduke@pyrami.com","city":"Brogan","state":"IL"}
{"index":{"_id":"6"}}
{"account_number":6,"balance":5686,"firstname":"Hattie","lastname":"Bond","age
 ↪":36,"gender":"M","address":"671 Bristol Street","employer":"Netagy","email
 ↪":"hattiebond@netagy.com","city":"Dante","state":"TN"}
{"index":{"_id":"13"}}
{"account_number":13,"balance":32838,"firstname":"Nanette","lastname":"Bates",
 ↪"age":28,"gender":"F","address":"789 Madison Street","employer":"Quility",
 ↪"city":"Nogal","state":"VA"}
{"index":{"_id":"18"}}
{"account_number":18,"balance":4180,"firstname":"Dale","lastname":"Adams","age
 ↪":33,"gender":"M","address":"467 Hutchinson Court","email":"daleadams@boink.
 ↪com","city":"Orick","state":"MD"}
```

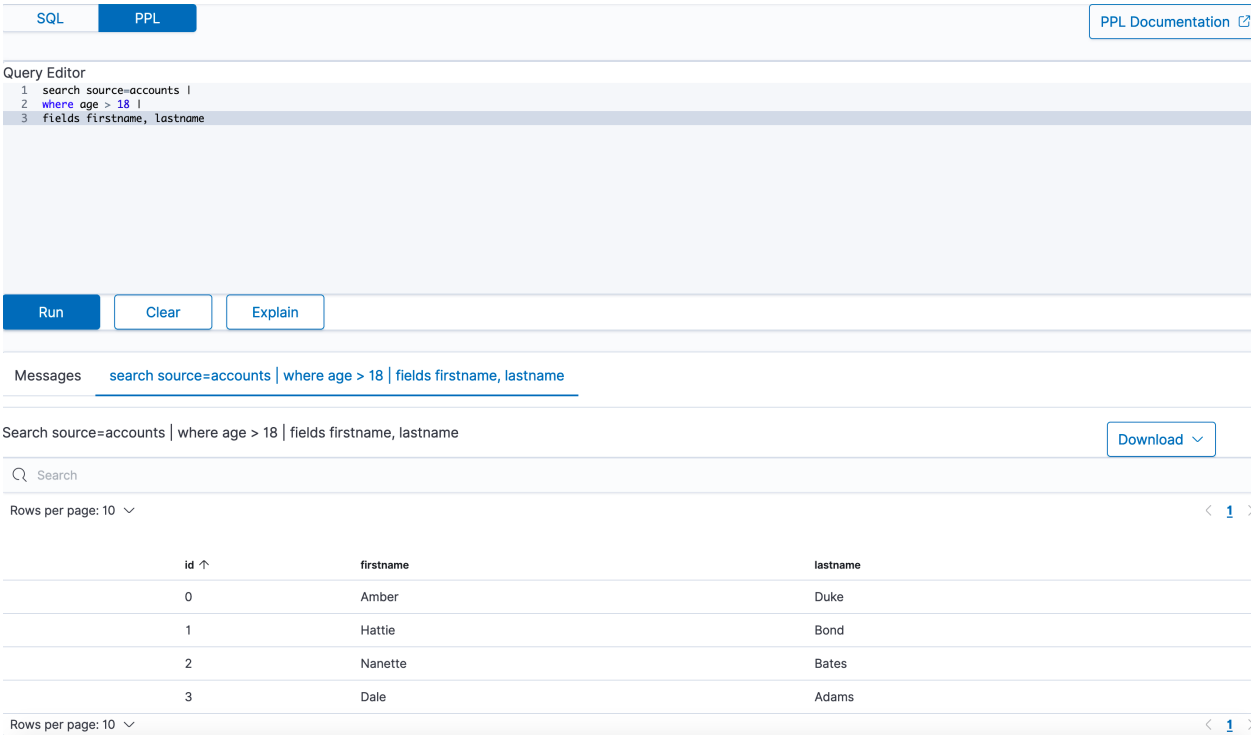
Go to **Dev Tools** → **Query Workbench**, then select the **PPL** language tab.

The following example returns `firstname` and `lastname` fields for documents in an `accounts` index with `age` greater than 18:

```
search source=accounts
| where age > 18
| fields firstname, lastname
```

Example response

firstname	lastname
Amber	Duke
Hattie	Bond
Nanette	Bates
Dale	Adams



PPL syntax

Every PPL query starts with the `search` command. It specifies the index to search and retrieve documents from. Subsequent commands can follow in any order.

Currently, `PPL` supports only one `search` command, which can be omitted to simplify the query.

Syntax

```
search source=<index> [boolean-expression]
source=<index> [boolean-expression]
```

Field	Description	Required
search	Specifies search keywords.	Yes
index	Specifies which index to query from.	No
bool-expression	Specifies an expression that evaluates to a Boolean value.	No

Examples

Example 1: Search through accounts index

In the following example, the `search` command refers to an `accounts` index as the source and uses `fields` and `where` commands for the conditions:

```
search source=accounts
| where age > 18
| fields firstname, lastname
```

In the following examples, angle brackets < > enclose required arguments and square brackets [ ] enclose optional arguments.

Example 2: Get all documents

To get all documents from the `accounts` index, specify it as the `source`:

```
search source=accounts;
```

account_numb	firstname	address	balance	gender	city	employer	state	age	email	lastname
1	Amber	880 Holmes Lane	39225	M	Brogan	Pyrami	IL	32	amberduke@pyrami.co	Duke
6	Hattie	671 Bristol Street	5686	M	Dante	Netagy	TN	36	hattiebond@netagy.c	Bond
13	Nanette	789 Madison Street	32838	F	Nogal	Quility	VA	28	null	Bates
18	Dale	467 Hutchinson Court	4180	M	Orick	null	MD	33	daleadams@boinl	Adams

Example 3: Get documents that match a condition

To get all documents from the `accounts` index that either have `account_number` equal to 1 or have `gender` as `F`, use the following query:

```
search source=accounts account_number=1 or gender=\"F\";
```

account_numb	firstname	address	balance	gender	city	employer	state	age	email	lastname
1	Amber	880 Holmes Lane	39225	M	Brogan	Pyrami	IL	32	amberduke@pyrami.cor	Duke
13	Nanette	789 Madison Street	32838	F	Nogal	Quility	VA	28	null	Bates

Commands

`PPL` supports all *`SQL` common* functions, including relevance search, but also introduces few more functions (called `commands`) which are available in `PPL` only.

dedup

The `dedup` (data deduplication) command removes duplicate documents defined by a field from the search result.

Syntax

```
dedup [int] <field-list> [keepempty=<bool>] [consecutive=<bool>]
```

Field	Description	Type	Re-quire	De-fault
int	Retain the specified number of duplicate events for each combination. The number must be greater than 0. If you do not specify a number, only the first occurring event is kept and all other duplicates are removed from the results.	string	No	1
keep-empt	If true, keep the document if any field in the field list has a null value or a field missing.	nested list of objects	No	False
con-sec-u-tive	If true, remove only consecutive events with duplicate combinations of values.	Boolean	No	False
field-list	Specify a comma-delimited field list. At least one field is required.	String or comma-separated list of strings	Yes	-

Example 1: Dedup by one field

To remove duplicate documents with the same gender:

```
search source=accounts | dedup gender | fields account_number, gender;
```

account_number	gender
1	M
13	F

Example 2: Keep two duplicate documents

To keep two duplicate documents with the same gender:

```
search source=accounts | dedup 2 gender | fields account_number, gender;
```

account_number	gender
1	M
6	M
13	F

Example 3: Keep or ignore an empty field by default

To keep two duplicate documents with a `null` field value:

```
search source=accounts | dedup email keepempty=true | fields account_number,␣
↪email;
```

account_number	email
1	amberduke@pyrami.com
6	hattiebond@netagy.com
13	null
18	daleadams@boink.com

To remove duplicate documents with the `null` field value:

```
search source=accounts | dedup email | fields account_number, email;
```

account_number	email
1	amberduke@pyrami.com
6	hattiebond@netagy.com
18	daleadams@boink.com

Example 4: Dedup of consecutive documents

To remove duplicates of consecutive documents:

```
search source=accounts | dedup gender consecutive=true | fields account_
↪number, gender;
```

account_number	gender
1	M
13	F
18	M

Limitations

The `dedup` command is not rewritten to Energy Logserver DSL, it is only executed on the coordination node.

eval

The `eval` command evaluates an expression and appends its result to the search result.

Syntax

```
eval <field>=<expression> ["," <field>=<expression> ]...
```

Field	Description	Required
field	If a field name does not exist, a new field is added. If the field name already exists, it's overwritten.	Yes
expression	Specify any supported expression.	Yes

Example 1: Create a new field

To create a new `doubleAge` field for each document. `doubleAge` is the result of `age` multiplied by 2:

```
search source=accounts | eval doubleAge = age * 2 | fields age, doubleAge;
```

age	doubleAge
32	64
36	72
28	56
33	66

Example 2: Overwrite the existing field

To overwrite the `age` field with `age` plus 1:

```
search source=accounts | eval age = age + 1 | fields age;
```

age
33
37
29
34

Example 3: Create a new field with a field defined with the `eval` command

To create a new field `ddAge`. `ddAge` is the result of `doubleAge` multiplied by 2, where `doubleAge` is defined in the `eval` command:

```
search source=accounts | eval doubleAge = age * 2, ddAge = doubleAge * 2 |␣
↪fields age, doubleAge, ddAge;
```

age	doubleAge	ddAge
32	64	128
36	72	144
28	56	112
33	66	132

Limitation

The `eval` command is not rewritten to Energy Logserver DSL, it is only executed on the coordination node.

fields

Use the `fields` command to keep or remove fields from a search result.

fields Syntax

```
fields [+|-] <field-list>
```

Field	Description	Required	Default
index	Plus (+) keeps only fields specified in the field list. Minus (-) removes all fields specified in the field list.	No	+
field list	Specify a comma-delimited list of fields.	Yes	No default

Example 1: Select specified fields from result

To get `account_number`, `firstname`, and `lastname` fields from a search result:

```
search source=accounts | fields account_number, firstname, lastname;
```

account_number	firstname	lastname
1	Amber	Duke
6	Hattie	Bond
13	Nanette	Bates
18	Dale	Adams

Example 2: Remove specified fields from a search result

To remove the `account_number` field from the search results:

```
search source=accounts | fields account_number, firstname, lastname | fields -
↪ account_number;
```

firstname	lastname
Amber	Duke
Hattie	Bond
Nanette	Bates
Dale	Adams

parse

Use the `parse` command to parse a text field using regular expression and append the result to the search result.

parse Syntax

```
parse <field> <regular-expression>
```

Field	Description	Required
field	A text field.	Yes
regular-expression	The regular expression used to extract new fields from the given test field. If a new field name exists, it will replace the original field.	Yes

The regular expression is used to match the whole text field of each document with Java regex engine. Each named capture group in the expression will become a new `STRING` field.

Example 1: Create new field

The example shows how to create new field `host` for each document. `host` will be the hostname after `@` in `email` field. Parsing a null field will return an empty string.

```
os> source=accounts | parse email '.+@(?<host>.+)' | fields email, host ;
fetched rows / total rows = 4/4
```

email	host
amberduke@pyrami.com	pyrami.com
hattiebond@netagy.com	netagy.com
null	null
daleadams@boink.com	boink.com

Example 2: Override the existing field

The example shows how to override the existing address field with street number removed.

```
os> source=accounts | parse address '\d+ (?<address>.+)' | fields address ;
fetched rows / total rows = 4/4
```

address
Holmes Lane
Bristol Street
Madison Street
Hutchinson Court

Example 3: Filter and sort be casted parsed field

The example shows how to sort street numbers that are higher than 500 in address field.

```
os> source=accounts | parse address '(?<streetNumber>\d+) (?<street>.+)' |
 ↪where cast(streetNumber as int) > 500 | sort num(streetNumber) | fields
 ↪streetNumber, street ;
fetched rows / total rows = 3/3
```

streetNumber	street
671	Bristol Street
789	Madison Street
880	Holmes Lane

Limitations (parse)

A few limitations exist when using the parse command:

- Fields defined by parse cannot be parsed again. For example, `source=accounts | parse address '\d+ (?<street>.+)' | parse street '\w+ (?<road>\w+)' ;` will fail to return any expressions.
- Fields defined by parse cannot be overridden with other commands. For example, when entering `source=accounts | parse address '\d+ (?<street>.+)' | eval street='1' | where street='1' ;` `where` will not match any documents since `street` cannot be overridden.

- The text field used by parse cannot be overridden. For example, when entering `source=accounts | parse address '\d+ (?<street>.+)' | eval address='1' ; street` will not be parse since address is overridden.
- Fields defined by parse cannot be filtered/sorted after using them in the `stats` command. For example, `source=accounts | parse email '.+@(?<host>.+)' | stats avg(age) by host | where host=pyrami.com ; where` will not parse the domain listed.

rename

Use the `rename` command to rename one or more fields in the search result.

rename Syntax

```
rename <source-field> AS <target-field>["," <source-field> AS <target-field>]...
```

Field	Description	Required
source-field	The name of the field that you want to rename.	Yes
target-field	The name you want to rename to.	Yes

Example 1: Rename one field

Rename the `account_number` field as `an`:

```
search source=accounts | rename account_number as an | fields an;
```

an
1
6
13
18

Example 2: Rename multiple fields

Rename the `account_number` field as `an` and `employer` as `emp`:

```
search source=accounts | rename account_number as an, employer as emp |␣
↪fields an, emp;
```

an	emp
1	Pyrami
6	Netagy
13	Quility
18	null

rename Limitations

The `rename` command is not rewritten to Energy Logserver DSL, it is only executed on the coordination node.

sort

Use the `sort` command to sort search results by a specified field.

sort Syntax

```
sort [count] <[+|-] sort-field>...
```

Field	Description	Required	Default
count	The maximum number results to return from the sorted result. If count=0, all results are returned.	No	1000
[+|-]	Use plus [+] to sort by ascending order and minus [-] to sort by descending order.	No	Ascending order
sort-field	Specify the field that you want to sort by.	Yes	-

Example 1: Sort by one field

To sort all documents by the `age` field in ascending order:

```
search source=accounts | sort age | fields account_number, age;
```

account_number	age
13	28
1	32
18	33
6	36

Example 2: Sort by one field and return all results

To sort all documents by the `age` field in ascending order and specify count as 0 to get back all results:

```
search source=accounts | sort 0 age | fields account_number, age;
```

account_number	age
13	28
1	32
18	33
6	36

Example 3: Sort by one field in descending order

To sort all documents by the `age` field in descending order:

```
search source=accounts | sort - age | fields account_number, age;
```

account_number	age
6	36
18	33
1	32
13	28

Example 4: Specify the number of sorted documents to return

To sort all documents by the `age` field in ascending order and specify count as 2 to get back two results:

```
search source=accounts | sort 2 age | fields account_number, age;
```

account_number	age
13	28
1	32

Example 5: Sort by multiple fields

To sort all documents by the `gender` field in ascending order and `age` field in descending order:

```
search source=accounts | sort + gender, - age | fields account_number, gender,
↪ age;
```

account_number	gender	age
13	F	28
6	M	36
18	M	33
1	M	32

stats

Use the `stats` command to aggregate from search results.

The following table lists the aggregation functions and also indicates how each one handles null or missing values:

Function	NULL	MISSING
COUNT	Not counted	Not counted
SUM	Ignore	Ignore
AVG	Ignore	Ignore
MAX	Ignore	Ignore
MIN	Ignore	Ignore

stats Syntax

```
stats <aggregation>... [by-clause]...
```

Field	Description	Required	Default
aggregation	Specify a statistical aggregation function. The argument of this function must be a field.	Yes	1000
by-clause	Specify one or more fields to group the results by. If not specified, the stats command returns only one row, which is the aggregation over the entire result set.	No	-

Example 1: Calculate the average value of a field

To calculate the average `age` of all documents:

```
search source=accounts | stats avg(age);
```

avg(age)
32.25

Example 2: Calculate the average value of a field by group

To calculate the average age grouped by gender:

```
search source=accounts | stats avg(age) by gender;
```

gender	avg(age)
F	28.0
M	33.666666666666664

Example 3: Calculate the average and sum of a field by group

To calculate the average and sum of age grouped by gender:

```
search source=accounts | stats avg(age), sum(age) by gender;
```

gender	avg(age)	sum(age)
F	28	28
M	33.666666666666664	101

Example 4: Calculate the maximum value of a field

To calculate the maximum age:

```
search source=accounts | stats max(age);
```

max(age)
36

Example 5: Calculate the maximum and minimum value of a field by group

To calculate the maximum and minimum age values grouped by gender:

```
search source=accounts | stats max(age), min(age) by gender;
```

gender	min(age)	max(age)
F	28	28
M	32	36

where

Use the `where` command with a bool expression to filter the search result. The `where` command only returns the result when the bool expression evaluates to true.

where Syntax

```
where <boolean-expression>
```

Field	Description	Required
bool-expression	An expression that evaluates to a boolean value.	No

Example: Filter result set with a condition

To get all documents from the `accounts` index where `account_number` is 1 or gender is `F`:

```
search source=accounts | where account_number=1 or gender=\"F\" | fields␣
↪account_number, gender;
```

account_number	gender
1	M
13	F

head

Use the `head` command to return the first N number of results in a specified search order.

head Syntax

```
head [N]
```

Field	Description	Required	Default
N	Specify the number of results to return.	No	10

Example 1: Get the first 10 results

To get the first 10 results:

```
search source=accounts | fields firstname, age | head;
```

firstname	age
Amber	32
Hattie	36
Nanette	28

Example 2: Get the first N results

To get the first two results:

```
search source=accounts | fields firstname, age | head 2;
```

firstname	age
Amber	32
Hattie	36

head Limitations

The `head` command is not rewritten to Energy Logserver DSL, it is only executed on the coordination node.

rare

Use the `rare` command to find the least common values of all fields in a field list. A maximum of 10 results are returned for each distinct set of values of the group-by fields.

rare Syntax

```
rare <field-list> [by-clause]
```

Field	Description	Required		
field-list	Specify a comma-delimited list of field names.	No	`by-clause`	Specify one or more fields to group the results by.	No

Example 1: Find the least common values in a field

To find the least common values of gender:

```
search source=accounts | rare gender;
```

gender
F
M

Example 2: Find the least common values grouped by gender

To find the least common age grouped by gender:

```
search source=accounts | rare age by gender;
```

gender	age
F	28
M	32
M	33

rare Limitations

The `rare` command is not rewritten to Energy Logserver DSL, it is only executed on the coordination node.

top

Use the `top` command to find the most common values of all fields in the field list.

top Syntax

```
top [N] <field-list> [by-clause]
```

Field	Description	Default
N	Specify the number of results to return.	10
field-list	Specify a comma-delimited list of field names.	-
by-clause	Specify one or more fields to group the results by.	-

Example 1: Find the most common values in a field

To find the most common genders:

```
search source=accounts | top gender;
```

gender
M
F

Example 2: Find the most common value in a field

To find the most common gender:

```
search source=accounts | top 1 gender;
```

gender
M

Example 3: Find the most common values grouped by gender

To find the most common age grouped by gender:

```
search source=accounts | top 1 age by gender;
```

gender	age
F	28
M	32

top Limitations

The `top` command is not rewritten to Energy Logserver DSL, it is only executed on the coordination node.

Query API and Formats

This section covers the SQL/PPL API endpoints and response format options.

- *SQL/PPL API*
  - *Query API*
  - *Explain API*
  - *Paginating results*
  - *Filtering results*
  - *Using parameters*
  - *Response formats*

SQL/PPL API

Use the SQL and PPL API to send queries to the SQL plugin. Use the `_sql` endpoint to send queries in SQL, and the `_ppl` endpoint to send queries in PPL. For both of these, you can also use the `_explain` endpoint to translate your query into Domain-specific language (DSL) or to troubleshoot errors.

Query API

Sends an SQL/PPL query to the SQL plugin. You can pass the format for the response as a query parameter.

Query parameters

Parameter	Data Type	Description
format	String	The format for the response. The _sql endpoint supports jdbc, csv, raw, and json formats. The _ppl endpoint supports jdbc, csv, and raw formats. Default is jdbc.
sanitize	Boolean	Specifies whether to escape special characters in the results. See *Response formats* for more information. Default is true.

Request fields

Field	Data Type	Description
query	String	The query to be executed. Required.
filter	JSON object	The filter for the results. Optional.
fetch_si	integer	The number of results to return in one response. Used for paginating results. Default is 1,000. Optional. Only supported for the `jdbc` response format.

Example request

```
POST /_plugins/_sql
{
  "query" : "SELECT * FROM accounts"
}
```

Example response

The response contains the schema and the results:

```
{
  "schema": [
    {
      "name": "account_number",
      "type": "long"
    },
    {
      "name": "firstname",
      "type": "text"
    },
    {
      "name": "address",
      "type": "text"
    },
    {
      "name": "balance",
      "type": "long"
    },
    {
      "name": "gender",
      "type": "text"
    },
    {
      "name": "city",
      "type": "text"
    },
    {
      "name": "employer",
      "type": "text"
    },
```

(continues on next page)

(continued from previous page)


```
    {
      "name": "state",
      "type": "text"
    },
    {
      "name": "age",
      "type": "long"
    },
    {
      "name": "email",
      "type": "text"
    },
    {
      "name": "lastname",
      "type": "text"
    }
  ],
  "datarows": [
    [
      1,
      "Amber",
      "880 Holmes Lane",
      39225,
      "M",
      "Brogan",
      "Pyrami",
      "IL",
      32,
      "amberduke@pyrami.com",
      "Duke"
    ],
    [
      6,
      "Hattie",
      "671 Bristol Street",
      5686,
      "M",
      "Dante",
      "Netagy",
      "TN",
      36,
      "hattiebond@netagy.com",
      "Bond"
    ],
    [
      13,
      "Nanette",
      "789 Madison Street",
      32838,
      "F",
      "Nogal",
      "Quility",
```


(continues on next page)

(continued from previous page)


```
      "VA",
      28,
      "nanettebates@quility.com",
      "Bates"
    ],
    [
      18,
      "Dale",
      "467 Hutchinson Court",
      4180,
      "M",
      "Orick",
      null,
      "MD",
      33,
      "daleadams@boink.com",
      "Adams"
    ]
  ],
  "total": 4,
  "size": 4,
  "status": 200
}
```


Response fields

Field	Data Type	Description
schema	Array	Specifies the field names and types for all fields.
data_rows	2D array	An array of results. Each result represents one matching row (document).
total	Integer	The total number of rows (documents) in the index.
size	Integer	The number of results to return in one response.
status	String	The HTTP response status Energy Logserver returns after running the query.

Explain API

The SQL plugin has an `explain` feature that shows how a query is executed against Energy Logserver, which is useful for debugging and development. A POST request to the `_plugins/_sql/_explain` or `_plugins/_ppl/_explain` endpoint returns Domain-specific language (DSL) in JSON format, explaining the query. You can execute the explain API operation either in command line using `curl` or in the Dashboards console, like in the example below.

Sample explain request for an SQL query

```
POST _plugins/_sql/_explain
{
  "query": "SELECT firstname, lastname FROM accounts WHERE age > 20"
}
```

Sample explain request for a PPL query

```
POST _plugins/_ppl/_explain
{
  "query" : "source=accounts | fields firstname, lastname"
}
```

Sample PPL query explain response

```
{
  "root": {
    "name": "ProjectOperator",
    "description": {
      "fields": "[firstname, lastname]"
    },
    "children": [
      {
        "name": "IndexScan",
        "description": {
          "request": """QueryRequest(indexName=accounts, sourceBuilder={"from
↪":0,"size":200,"timeout":"1m","_source":{"includes":["firstname","lastname
↪"],"excludes":[]}}, searchDone=false)"""
        },
        "children": []
      }
    ]
  }
}
```

For queries that require post-processing, the `explain` response includes a query plan in addition to the Energy Logserver DSL. For those queries that don't require post processing, you can see a complete DSL.

Paginating results

To get back a paginated response, use the `fetch_size` parameter. The value of `fetch_size` should be greater than 0. The default value is 1,000. A value of 0 will fall back to a non-paginated response.

> **Note**
>
> The `fetch_size` parameter is only supported for the `jdbc` response format.

Example

The following request contains an SQL query and specifies to return five results at a time:

```
POST _plugins/_sql/
{
  "fetch_size" : 5,
  "query" : "SELECT firstname, lastname FROM accounts WHERE age > 20 ORDER BY␣
↪state ASC"
}
```

The response contains all the fields that a query without `fetch_size` would contain, and a `cursor` field that is used to retrieve subsequent pages of results:


```
{
  "schema": [
    {
      "name": "firstname",
      "type": "text"
    },
    {
      "name": "lastname",
      "type": "text"
    }
  ],
  "cursor": "d:eyJhIjp7fSwicyI6IkRYRjFaWEo1UVc1a1JtVjBZMmdCQUFBQUFBQUFBQU1XZWp
↪kdFRFRkZUMlpTZEZkeFdsWnJkRlZoYnpaeVVRPT0iLCJjIjpbeyJuYW1lIjoiZmlyc3RuYW1lIiw
↪idHlwZSI6InRleHQifSx7Im5hbWUiOiJsYXN0bmFtZSIsInR5cGUiOiJ0ZXh0In1dLCJmIjo1LCJ
↪pIjoiYWNjb3VudHMiLCJsIjo5NTF9",
  "total": 956,
  "datarows": [
    [
      "Cherry",
      "Carey"
    ],
    [
      "Lindsey",
      "Hawkins"
    ],
    [
      "Sargent",
      "Powers"
    ],
    [
      "Campos",
      "Olsen"
    ],
    [
      "Savannah",
      "Kirby"
    ]
  ],
  "size": 5,
  "status": 200
}
```


To fetch subsequent pages, use the `cursor` from the previous response:


```
POST /_plugins/_sql
{
   "cursor": "d:eyJhIjp7fSwicyI6IkRYRjFaWEo1UVc1a1JtVjBZMmdCQUFBQUFBQUFBQU1XZW
↪pkdFRFRkZUMlpTZEZkeFdsWnJkRlZoYnpaeVVRPT0iLCJjIjpbeyJuYW1lIjoiZmlyc3RuYW1lIi
↪widHlwZSI6InRleHQifSx7Im5hbWUiOiJsYXN0bmFtZSIsInR5cGUiOiJ0ZXh0In1dLCJmIjo1LC
↪JpIjoiYWNjb3VudHMiLCJsIjo5NTF9"
}
```

The next response contains only the `datarows` of the results and a new `cursor`.

```
{
  "cursor": "d:eyJhIjp7fSwicyI6IkRYRjFaWEo1UVc1a1JtVjBZMmdCQUFBQUFBQUFBQU1XZWp
↪kdFRFRkZUMlpTZEZkeFdsWnJkRlZoYnpaeVVRPT0iLCJjIjpbeyJuYW1lIjoiZmlyc3RuYW1lIiw
↪idHlwZSI6InRleHQifSx7Im5hbWUiOiJsYXN0bmFtZSIsInR5cGUiOiJ0ZXh0In1dLCJmIjo1LCJ
↪pIjoiYWNjb3VudHMabcde12345",
  "datarows": [
    [
      "Abbey",
      "Karen"
    ],
    [
      "Chen",
      "Ken"
    ],
    [
      "Ani",
      "Jade"
    ],
    [
      "Peng",
      "Hu"
    ],
    [
      "John",
      "Doe"
    ]
  ]
}
```

The `datarows` can have more than the `fetch_size` number of records in case nested fields are flattened.

The last page of results has only `datarows` and no `cursor`. The `cursor` context is automatically cleared on the last page.

To explicitly clear the cursor context, use the `_plugins/_sql/close` endpoint operation:

```
POST /_plugins/_sql/close
{
   "cursor": "d:eyJhIjp7fSwicyI6IkRYRjFaWEo1UVc1a1JtVjBZMmdCQUFBQUFBQUFBQU1XZW
↪pkdFRFRkZUMlpTZEZkeFdsWnJkRlZoYnpaeVVRPT0iLCJjIjpbeyJuYW1lIjoiZmlyc3RuYW1lIi
↪widHlwZSI6InRleHQifSx7Im5hbWUiOiJsYXN0bmFtZSIsInR5cGUiOiJ0ZXh0In1dLCJmIjo1LC
↪JpIjoiYWNjb3VudHMiLCJsIjo5NTF9"
}'
```

The response is an acknowledgement from Energy Logserver:

```
{"succeeded":true}
```

Filtering results

You can use the `filter` parameter to add more conditions to the Energy Logserver DSL directly.

The following SQL query returns the names and account balances of all customers. The results are then filtered to contain only those customers with less than $10,000 balance.

```
POST /_plugins/_sql/
{
  "query" : "SELECT firstname, lastname, balance FROM accounts",
  "filter" : {
    "range" : {
      "balance" : {
        "lt" : 10000
      }
    }
  }
}
```

The response contains the matching results:

```
{
  "schema": [
    {
      "name": "firstname",
      "type": "text"
    },
    {
      "name": "lastname",
      "type": "text"
    },
    {
      "name": "balance",
      "type": "long"
    }
  ],
  "total": 2,
  "datarows": [
    [
      "Hattie",
      "Bond",
      5686
    ],
    [
      "Dale",
      "Adams",
      4180
    ]
  ],
  "size": 2,
  "status": 200
}
```

You can use the Explain API to see how this query is executed against Energy Logserver:

```
POST /_plugins/_sql/_explain
{
  "query" : "SELECT firstname, lastname, balance FROM accounts",
  "filter" : {
    "range" : {
```

(continues on next page)

(continued from previous page)

```
      "balance" : {
        "lt" : 10000
      }
    }
  }
}'
```

The response contains the Boolean query in Energy Logserver DSL that corresponds to the query above:

```
{
  "from": 0,
  "size": 200,
  "query": {
    "bool": {
      "filter": [{
        "bool": {
          "filter": [{
            "range": {
              "balance": {
                "from": null,
                "to": 10000,
                "include_lower": true,
                "include_upper": false,
                "boost": 1.0
              }
            }
          }],
          "adjust_pure_negative": true,
          "boost": 1.0
        }
      }],
      "adjust_pure_negative": true,
      "boost": 1.0
    }
  },
  "_source": {
    "includes": [
      "firstname",
      "lastname",
      "balance"
    ],
    "excludes": []
  }
}
```

Using parameters

You can use the `parameters` field to pass parameter values to a prepared SQL query.

The following explain operation uses an SQL query with an `age` parameter:

```
POST /_plugins/_sql/_explain
```

(continues on next page)

(continued from previous page)

```
{
  "query": "SELECT * FROM accounts WHERE age = ?",
  "parameters": [{
    "type": "integer",
    "value": 30
  }]
}
```

The response contains the Boolean query in Energy Logserver DSL that corresponds to the SQL query above:

```
{
  "from": 0,
  "size": 200,
  "query": {
    "bool": {
      "filter": [{
        "bool": {
          "must": [{
            "term": {
              "age": {
                "value": 30,
                "boost": 1.0
              }
            }
          }],
          "adjust_pure_negative": true,
          "boost": 1.0
        }
      }],
      "adjust_pure_negative": true,
      "boost": 1.0
    }
  }
}
```

Response formats

The SQL plugin provides the `jdbc`, `csv`, `raw`, and `json` response formats that are useful for different purposes. The `jdbc` format is widely used because it provides the schema information and adds more functionality, such as pagination. Besides the JDBC driver, various clients can benefit from a detailed and well-formatted response.

JDBC format

By default, the SQL plugin returns the response in the standard JDBC format. This format is provided for the JDBC driver and clients that need both the schema and the result set to be well formatted.

Example request

The following query does not specify the response format, so the format is set to `jdbc`:

```
POST _plugins/_sql
{
```

(continues on next page)

(continued from previous page)

```
  "query" : "SELECT firstname, lastname, age FROM accounts ORDER BY age LIMIT␣
↪2"
}
```

Example response

In the response, the `schema` contains the field names and types, and the `datarows` field contains the result set:

```
{
  "schema": [{
      "name": "firstname",
      "type": "text"
    },
    {
      "name": "lastname",
      "type": "text"
    },
    {
      "name": "age",
      "type": "long"
    }
  ],
  "total": 4,
  "datarows": [
    [
      "Nanette",
      "Bates",
      28
    ],
    [
      "Amber",
      "Duke",
      32
    ]
  ],
  "size": 2,
  "status": 200
}
```

If an error of any type occurs, Energy Logserver returns the error message.

The following query searches for a non-existent field `unknown`:

```
POST /_plugins/_sql
{
  "query" : "SELECT unknown FROM accounts"
}
```

The response contains the error message and the cause of the error:

```
{
  "error": {
    "reason": "Invalid SQL query",
```

(continues on next page)

(continued from previous page)

```
    "details": "Field [unknown] cannot be found or used here.",
    "type": "SemanticAnalysisException"
  },
  "status": 400
}
```

Energy Logserver DSL JSON format

If you set the format to `json`, the original Energy Logserver response is returned in JSON format. Because this is the native response from Energy Logserver, extra effort is needed to parse and interpret it.

Example request

The following query sets the response format to `json`:

```
POST _plugins/_sql?format=json
{
  "query" : "SELECT firstname, lastname, age FROM accounts ORDER BY age LIMIT␣
↪2"
}
```

Example response

The response is the original response from Energy Logserver:

```
{
  "_shards": {
    "total": 5,
    "failed": 0,
    "successful": 5,
    "skipped": 0
  },
  "hits": {
    "hits": [{
        "_index": "accounts",
        "_type": "account",
        "_source": {
          "firstname": "Nanette",
          "age": 28,
          "lastname": "Bates"
        },
        "_id": "13",
        "sort": [
          28
        ],
        "_score": null
      },
      {
        "_index": "accounts",
        "_type": "account",
        "_source": {
```

(continues on next page)

(continued from previous page)


```
          "firstname": "Amber",
          "age": 32,
          "lastname": "Duke"
        },
        "_id": "1",
        "sort": [
          32
        ],
        "_score": null
      }
    ],
    "total": {
      "value": 4,
      "relation": "eq"
    },
    "max_score": null
  },
  "took": 100,
  "timed_out": false
}
```


CSV format

You can also specify to return results in CSV format.

Example request

```
POST /_plugins/_sql?format=csv
{
  "query" : "SELECT firstname, lastname, age FROM accounts ORDER BY age"
}
```

Example response

```
firstname,lastname,age
Nanette,Bates,28
Amber,Duke,32
Dale,Adams,33
Hattie,Bond,36
```

Sanitizing results in CSV format

By default, Energy Logserver sanitizes header cells (field names) and data cells (field contents) according to the following rules:

- If a cell starts with +, −, = , or `@`, the sanitizer inserts a single quote (`'`) at the start of the cell.
- If a cell contains one or more commas (`,`), the sanitizer surrounds the cell with double quotes (`"`).

Example

The following query indexes a document with cells that either start with special characters or contain commas:

```
PUT /userdata/_doc/1?refresh=true
{
  "+firstname": "-Hattie",
  "=lastname": "@Bond",
  "address": "671 Bristol Street, Dente, TN"
}
```

You can use the query below to request results in CSV format:

```
POST /_plugins/_sql?format=csv
{
  "query" : "SELECT * FROM userdata"
}
```

In the response, cells that start with special characters are prefixed with `'`. The cell that has commas is surrounded with quotation marks:

```
'+firstname,'=lastname,address
'Hattie,'@Bond,"671 Bristol Street, Dente, TN"
```

To skip sanitizing, set the `sanitize` query parameter to false:

```
POST /_plugins/_sql?format=csv&sanitize=false
{
  "query" : "SELECT * FROM userdata"
}
```

The response contains the results in the original CSV format:

```
=lastname,address,+firstname
@Bond,"671 Bristol Street, Dente, TN",-Hattie
```

Raw format

You can use the raw format to pipe the results to other command line tools for post-processing.

Example request

```
POST /_plugins/_sql?format=raw
{
  "query" : "SELECT firstname, lastname, age FROM accounts ORDER BY age"
}
```

Example response

```
Nanette|Bates|28
Amber|Duke|32
Dale|Adams|33
Hattie|Bond|36
```

By default, Energy Logserver sanitizes results in `raw` format according to the following rule:

- If a data cell contains one or more pipe characters (`|`), the sanitizer surrounds the cell with double quotes.

Example

The following query indexes a document with pipe characters (`|`) in its fields:

```
PUT /userdata/_doc/1?refresh=true
{
  "+firstname": "|Hattie",
  "=lastname": "Bond|",
  "|address": "671 Bristol Street| Dente| TN"
}
```

You can use the query below to request results in `raw` format:

```
POST /_plugins/_sql?format=raw
{
  "query" : "SELECT * FROM userdata"
}
```

The query returns cells with the `|` character surrounded by quotation marks:

```
"|address"|=lastname|+firstname
"671 Bristol Street| Dente| TN"|"Bond|"|"|Hattie"
```

Data Types and Functions

Complete reference for data types, identifiers, and built-in functions.

- *Identifiers*
  - *Regular identifiers*
  - *Delimited identifiers*
  - *Case sensitivity*
  - *Data types*
  - *Functions*

Identifiers

An identifier is an ID to name your database objects, such as index names, field names, aliases, and so on. Energy Logserver supports two types of identifiers: regular identifiers and delimited identifiers.

Regular identifiers

A regular identifier is a string of characters that starts with an ASCII letter (lower or upper case). The next character can either be a letter, digit, or underscore (_). It can't be a reserved keyword. Whitespace and other special characters are also not allowed.

Energy Logserver supports the following regular identifiers:

1. Identifiers prefixed by a dot `.` sign. Use to hide an index. For example `.sample-dashboards`.
2. Identifiers prefixed by an `@` sign. Use for meta fields generated by Network Probe ingestion.
3. Identifiers with hyphen `-` in the middle. Use for index names with date information.
4. Identifiers with star `*` present. Use for wildcard match of index patterns.

For regular identifiers, you can use the name without any back tick or escape characters. In this example, `source`, `fields`, `account_number`, `firstname`, and `lastname` are all identifiers. Out of these, the `source` field is a reserved identifier.

```
SELECT account_number, firstname, lastname FROM accounts;
```

account_number	firstname	lastname
1	Amber	Duke
6	Hattie	Bond
13	Nanette	Bates
18	Dale	Adams

Delimited identifiers

A delimited identifier can contain special characters not allowed by a regular identifier. You must enclose delimited identifiers with back ticks (``). Back ticks differentiate the identifier from special characters.

If the index name includes a dot (`.`), for example, `log-2021.01.11`, use delimited identifiers with back ticks to escape it `` `log-2021.01.11` ``.

Typical examples of using delimited identifiers:

1. Identifiers with reserved keywords.
2. Identifiers with a `.` present. Similarly, `-` to include date information.
3. Identifiers with other special characters. For example, Unicode characters.

To quote an index name with back ticks:

```
source=`accounts` | fields `account_number`;
```

account_number
1
6
13
18

Case sensitivity

Identifiers are case sensitive. They must be exactly the same as what's stored in Energy Logserver.

For example, if you run `source=Accounts`, you'll get an index not found exception because the actual index name is in lower case.

Data types

The following table shows the data types supported by the SQL plugin and how each one maps to SQL and Energy Logserver data types:

Energy Logserver SQL Type	Energy Logserver Type	SQL Type
boolean	boolean	BOOLEAN
byte	byte	TINYINT
short	byte	SMALLINT
integer	integer	INTEGER
long	long	BIGINT
float	float	REAL
half_float	float	FLOAT
scaled_float	float	DOUBLE
double	double	DOUBLE
keyword	string	VARCHAR
text	text	VARCHAR
date	timestamp	TIMESTAMP
date_nanos	timestamp	TIMESTAMP
ip	ip	VARCHAR
date	timestamp	TIMESTAMP
binary	binary	VARBINARY
object	struct	STRUCT
nested	array	STRUCT

In addition to this list, the SQL plugin also supports the `datetime` type, though it doesn't have a corresponding mapping with Energy Logserver or SQL. To use a function without a corresponding mapping, you must explicitly convert the data type to one that does.

Date and time types

The date and time types represent a time period: `DATE`, `TIME`, `DATETIME`, `TIMESTAMP`, and `INTERVAL`. By default, the Energy Logserver DSL uses the `date` type as the only date-time related type that contains all information of an absolute time point.

To integrate with SQL, each type other than the `timestamp` type holds part of the time period information. Some functions might have restrictions for the input argument type.

Date

The `date` type represents the calendar date regardless of the time zone. A given date value is a 24-hour period, but this period varies in different timezones and might have flexible hours during daylight saving programs. The `date` type doesn't contain time information and it only supports a range of `1000-01-01` to `9999-12-31`.

Type	Syntax	Range
date	yyyy-MM-dd	0001-01-01 to 9999-12-31

Time

The `time` type represents the time of a clock regardless of its timezone. The `time` type doesn't contain date information.

Type	Syntax	Range
time	hh:mm:ss[.fraction]	00:00:00.0000000000 to 23:59:59.9999999999

Datetime

The `datetime` type is a combination of date and time. It doesn't contain timezone information. For an absolute time point that contains date, time, and timezone information, see *Timestamp*.

Type	Syntax	Range
datetime	yyyy-MM-dd hh:mm:ss[.fraction]	0001-01-01 00:00:00.0000000000 to 9999-12-31 23:59:59.9999999999

Timestamp

The `timestamp` type is an absolute instance independent of timezone or convention. For example, for a given point of time, if you change the timestamp to a different timezone, its value changes accordingly.

The `timestamp` type is stored differently from the other types. It's converted from its current timezone to UTC for storage and converted back to its set timezone from UTC when it's retrieved.

Type	Syntax	Range
timestamp	yyyy-MM-dd hh:mm:ss[.fraction]	0001-01-01 00:00:01.9999999999 UTC to 9999-12-31 23:59:59.9999999999

Interval

The `interval` type represents a temporal duration or a period.

Type	Syntax
interval	INTERVAL expr unit

The `expr` unit is any expression that eventually iterates to a quantity value. It represents a unit for interpreting the quantity, including `MICROSECOND`, `SECOND`, `MINUTE`, `HOUR`, `DAY`, `WEEK`, `MONTH`, `QUARTER`, and `YEAR`. The `INTERVAL` keyword and the unit specifier are not case sensitive.

The `interval` type has two classes of intervals: year-week intervals and day-time intervals.

- Year-week intervals store years, quarters, months, and weeks.
- Day-time intervals store days, hours, minutes, seconds, and microseconds.

Convert between date and time types

Apart from the `interval` type, all date and time types can be converted to each other. The conversion might alter the value or cause some information loss. For example, when extracting the `time` value from a `datetime` value, or converting a `date` value to a `datetime` value, and so on.

The SQL plugin supports the following conversion rules for each of the types:

Convert from date

- Because the `date` value doesn't have any time information, conversion to the `time` type isn't useful and always returns a zero time value of `00:00:00`.
- Converting from `date` to `datetime` has a data fill-up due to the lack of time information. It attaches the time `00:00:00` to the original date by default and forms a `datetime` instance. For example, conversion of `2020-08-17` to a `datetime` type is `2020-08-17 00:00:00`.
- Converting to `timestamp` type alternates both the `time` value and the `timezone` information. It attaches the zero time value `00:00:00` and the session timezone (UTC by default) to the date. For example, conversion of `2020-08-17` to a `datetime` type with a session timezone UTC is `2020-08-17 00:00:00 UTC`.

Convert from time

- You cannot convert the `time` type to any other date and time types because it doesn't contain any date information.

Convert from datetime

- Converting `datetime` to `date` extracts the date value from the `datetime` value. For example, conversion of `2020-08-17 14:09:00` to a `date` type is `2020-08-08`.
- Converting `datetime` to `time` extracts the time value from the `datetime` value. For example, conversion of `2020-08-17 14:09:00` to a `time` type is `14:09:00`.
- Because the `datetime` type doesn't contain timezone information, converting to `timestamp` type fills up the timezone value with the session timezone. For example, conversion of `2020-08-17 14:09:00` (UTC) to a `timestamp` type is `2020-08-17 14:09:00 UTC`.

Convert from timestamp

- Converting from a `timestamp` type to a `date` type extracts the date value and converting to a `time` type extracts the time value. Converting from a `timestamp` type to `datetime` type extracts only the `datetime` value and leaves out the timezone value. For example, conversion of `2020-08-17 14:09:00` UTC to a `date` type is `2020-08-17`, to a `time` type is `14:09:00`, and to a `datetime` type is `2020-08-17 14:09:00`.

Functions

You must enable fielddata in the document mapping for most string functions to work properly.

The specification shows the return type of the function with a generic type `T` as the argument. For example, `abs(number T) -> T` means that the function `abs` accepts a numerical argument of type `T`, which could be any subtype of the `number` type, and it returns the actual type of `T` as the return type.

The SQL plugin supports the following common functions shared across the SQL and PPL languages.

Mathematical

Function	Specification	Example
abs	abs(number T) -> T	SELECT abs(0.5)
add	add(number T, number T) -> T	SELECT add(1, 5)
cbrt	cbrt(number T) -> double	SELECT cbrt(8)
ceil	ceil(number T) -> T	SELECT ceil(0.5)
conv	conv(string T, integer, integer) -> string	SELECT conv('2C', 16, 10), conv(1111, 2, 10)
crc32	crc32(string) -> string	SELECT crc32('MySQL')
divide	divide(number T, number T) -> T	SELECT divide(1, 0.5)
e	e() -> double	SELECT e()
exp	exp(number T) -> double	SELECT exp(0.5)
expm1	expm1(number T) -> double	SELECT expm1(0.5)
floor	floor(number T) -> long	SELECT floor(0.5)
ln	ln(number T) -> double	SELECT ln(10)
log	log(number T) -> double OR log(number T, number T) -> double	SELECT log(10), SELECT log(2, 16)
log2	log2(number T) -> double	SELECT log2(10)
log10	log10(number T) -> double	SELECT log10(10)
mod	mod(number T, number T) -> T	SELECT mod(2, 3)
modulus	modulus(number T, number T) -> T	SELECT modulus(2, 3)
multiply	multiply(number T, number T) -> T	SELECT multiply(2, 3)
pi	pi() -> double	SELECT pi()
pow	pow(number T, number T) -> double	SELECT pow(2, 3)
power	power(number T, number T) -> double	SELECT power(2, 3)
rand	rand() -> float OR rand(number T) -> float	SELECT rand(), SELECT rand(0.5)
rint	rint(number T) -> double	SELECT rint(1.5)
round	round(number T) -> T OR round(number T, integer) -> T	SELECT round(1.5), SELECT round(1.175, 2)
sign	sign(number T) -> integer	SELECT sign(1.5)
signum	signum(number T) -> integer	SELECT signum(0.5)
sqrt	sqrt(number T) -> double	SELECT sqrt(0.5)
strcmp	strcmp(string T, string T) -> integer	SELECT strcmp('hello', 'hello world')
subtract	subtract(number T, number T) -> T	SELECT subtract(3, 2)
truncate	truncate(number T, number T) -> T	SELECT truncate(56.78, 1)
+	number T + number T -> T	SELECT 1 + 5
-	number T - number T -> T	SELECT 3 - 2
*	number T * number T -> T	SELECT 2 * 3
/	number T / number T -> T	SELECT 1 / 0.5
%	number T % number T -> T	SELECT 2 % 3

Trigonometric

Function	Specification	Example
acos	acos(number T) -> double	SELECT acos(0.5)
asin	asin(number T) -> double	SELECT asin(0.5)
atan	atan(number T) -> double	SELECT atan(0.5)
atan2	atan2(number T, number T) -> double	SELECT atan2(1, 0.5)
cos	cos(number T) -> double	SELECT cos(0.5)
cosh	cosh(number T) -> double	SELECT cosh(0.5)
cot	cot(number T) -> double	SELECT cot(0.5)
degrees	degrees(number T) -> double	SELECT degrees(0.5)
radians	radians(number T) -> double	SELECT radians(0.5)
sin	sin(number T) -> double	SELECT sin(0.5)
sinh	sinh(number T) -> double	SELECT sinh(0.5)
tan	tan(number T) -> double	SELECT tan(0.5)

Date and time

Functions marked with * are only available in SQL.

Function	Specification	Example
adddate	adddate(date, INTERVAL expr unit) -> date	SELECT adddate(date('2020-08-26'), INTERVAL 1 hour)
addtime	addtime(date, date) -> date	SELECT addtime(date('2008-12-12'), date('2008-12-12'))
convert_tz	convert_tz(date, string, string) -> date	SELECT convert_tz('2008-12-25 05:30:00', '+00:00', 'America/Los_Angeles')
curtime	curtime() -> time	SELECT curtime()
curdate	curdate() -> date	SELECT curdate()
current_date	current_date() -> date	SELECT current_date()
current_time	current_time() -> time	SELECT current_time()
current_timestamp	current_timestamp() -> date	SELECT current_timestamp()
date	date(date) -> date	SELECT date('2000-01-02')
datediff	datediff(date, date) -> integer	SELECT datediff(date('2000-01-02'), date('2000-01-01'))
datetime	datetime(string) -> datetime	SELECT datetime('2008-12-25 00:00:00')
date_add	date_add(date, INTERVAL integer UNIT)	SELECT date_add('2020-08-26', INTERVAL 1 HOUR)
date_format	date_format(date, string) -> string OR date_format(date, string, string) -> string	SELECT date_format(date('2020-08-26'), 'Y')
date_sub	date_sub(date, INTERVAL expr unit) -> date	SELECT date_sub(date('2008-01-02'), INTERVAL 31 day)
dayofmonth	dayofmonth(date) -> integer	SELECT dayofmonth(date('2001-05-07'))
day	day(date) -> integer	SELECT day(date('2020-08-25'))
dayname	dayname(date) -> string	SELECT dayname(date('2020-08-26'))

continues on next page

Table 2 – continued from previous page

Function	Specification	Example
dayofmonth	dayofmonth(date) -> integer	SELECT dayofmonth(date('2020-08-26'))
dayofweek	dayofweek(date) -> integer	SELECT dayofweek(date('2020-08-26'))
dayofyear	dayofyear(date) -> integer	SELECT dayofyear(date('2020-08-26'))
dayofweek	dayofweek(date) -> integer	SELECT dayofweek(date('2020-08-26'))
day_of_month*	day_of_month(date) -> integer	SELECT day_of_month(date('2020-08-26'))
day_of_week*	day_of_week(date) -> integer	SELECT day_of_week(date('2020-08-26'))
day_of_year*	day_of_year(date) -> integer	SELECT day_of_year(date('2020-08-26'))
extract*	extract(part FROM date) -> integer	SELECT extract(MONTH FROM datetime('2020-08-26 10:11:12'))
from_days	from_days(N) -> integer	SELECT from_days(733687)
from_unixtime	from_unixtime(N) -> date	SELECT from_unixtime(1220249547)
get_format	get_format(PART, string) -> string	SELECT get_format(DATE, 'USA')
hour	hour(time) -> integer	SELECT hour(time '01:02:03')
hour_of_day*	hour_of_day(time) -> integer	SELECT hour_of_day(time '01:02:03')
last_day*	last_day(date) -> integer	SELECT last_day(date('2020-08-26'))
localtime	localtime() -> date	SELECT localtime()
localtimestamp	localtimestamp() -> date	SELECT localtimestamp()
makedate	makedate(double, double) -> date	SELECT makedate(1945, 5.9)
maketime	maketime(integer, integer, integer) -> date	SELECT maketime(1, 2, 3)
microsecond	microsecond(expr) -> integer	SELECT microsecond(time '01:02:03.123456')
minute	minute(expr) -> integer	SELECT minute(time '01:02:03')
minute_of_day*	minute_of_day(expr) -> integer	SELECT minute_of_day(time '01:02:03')
minute_of_hour*	minute_of_hour(expr) -> integer	SELECT minute_of_hour(time '01:02:03')
month	month(date) -> integer	SELECT month(date('2020-08-26'))
month_of_year*	month_of_year(date) -> integer	SELECT month_of_year(date('2020-08-26'))
monthname	monthname(date) -> string	SELECT monthname(date('2020-08-26'))
now	now() -> date	SELECT now()
period_add	period_add(integer, integer)	SELECT period_add(200801, 2)
period_diff	period_diff(integer, integer)	SELECT period_diff(200802, 200703)
quarter	quarter(date) -> integer	SELECT quarter(date('2020-08-26'))

continues on next page

Table 2 – continued from previous page

Function	Specification	Example
second	second(time) -> integer	SELECT second(time '01:02:03')
second_of_minute*	second_of_minute(time) -> integer	SELECT second_of_minute(time '01:02:03')
sec_to_time*	sec_to_time(integer) -> date	SELECT sec_to_time(10000)
subdate	subdate(date, INTERVAL expr unit) -> date, datetime	SELECT subdate(date('2008-01-02'), INTERVAL 31 day)
subtime	subtime(date, date) -> date	SELECT subtime(date('2008-12-12'), date('2008-11-15'))
str_to_date*	str_to_date(string, format) -> date	SELECT str_to_date("01,5,2013", "%d,%m,%Y")
time	time(expr) -> time	SELECT time('13:49:00')
timediff	timediff(time, time) -> time	SELECT timediff(time('23:59:59'), time('13:00:00'))
timestamp	timestamp(date) -> date	SELECT timestamp('2001-05-07 00:00:00')
timestampadd	timestampadd(interval, integer, (date)) -> date	SELECT timestampadd(DAY, 17, datetime('2000-01-01 00:00:00'))
timestampdiff	timestampdiff(interval, date, date) -> integer	SELECT timestampdiff(YEAR, '1997-01-01 00:00:00', '2001-03-06 00:00:00')
time_format	time_format(date, string) -> string	SELECT time_format('1998-01-31 13:14:15.012345', '%f %H %h %I %i %p %r %S %s %T')
time_to_sec	time_to_sec(time) -> long	SELECT time_to_sec(time '22:23:00')
to_days	to_days(date) -> long	SELECT to_days(date '2008-10-07')
to_seconds	to_seconds(date) -> integer	SELECT to_seconds(date('2008-10-07'))
unix_timestamp	unix_timestamp(date) -> double	SELECT unix_timestamp(timestamp('1996-11-15 17:05:42'))
utc_date	utc_date() -> date	SELECT utc_date()
utc_time	utc_time() -> date	SELECT utc_time()
utc_timestamp	utc_timestamp() -> date	SELECT utc_timestamp()
week	week(date[mode]) -> integer	SELECT week(date('2008-02-20'))
weekofyear	weekofyear(date[mode]) -> integer	SELECT weekofyear(date('2008-02-20'))
week_of_year*	week_of_year(date[mode]) -> integer	SELECT week_of_year(date('2008-02-20'))
year	year(date) -> integer	SELECT year(date('2001-07-05'))
yearweek*	yearweek(date[mode]) -> integer	SELECT yearweek(date('2008-02-20'))

String

Function	Specification	Example
ascii	ascii(string) -> integer	SELECT ascii('h')
concat	concat(string, string) -> string	SELECT concat('hello', 'world')
concat_ws	concat_ws(separator, string, string...) -> string	SELECT concat_ws(" ", "Hello", "World!")
left	left(string, integer) -> string	SELECT left('hello', 2)
length	length(string) -> integer	SELECT length('hello')
locate	locate(string, string, integer) -> integer OR locate(string, string) -> integer	SELECT locate('o', 'hello'), locate('l', 'hello world', 5)
replace	replace(string, string, string) -> string	SELECT replace('hello', 'l', 'x')
right	right(string, integer) -> string	SELECT right('hello', 2)
rtrim	rtrim(string) -> string	SELECT rtrim('hello')
substring	substring(string, integer, integer) -> string	SELECT substring('hello', 2, 4)
trim	trim(string) -> string	SELECT trim('hello')
upper	upper(string) -> string	SELECT upper('hello world')

Aggregate

Function	Specification	Example
avg	avg(number T) -> T	SELECT avg(column) FROM my-index
count	count(number T) -> T	SELECT count(date) FROM my-index
min	min(number T) -> T	SELECT min(column) FROM my-index
show	show(string) -> string	SHOW TABLES LIKE my-index

Advanced

Function	Specification	Example
if	if(boolean, os_type, os_type) -> os_type	SELECT if(false, 0, 1),if(true, 0, 1)
ifnull	ifnull(os_type, os_type) -> os_type	SELECT ifnull(0, 1), ifnull(null, 1)
isnull	isnull(os_type) -> integer	SELECT isnull(null), isnull(1)

Relevance-based search (full-text search)

These functions are only available in the `WHERE` clause. For their descriptions and usage examples in SQL and PPL, see *Full-text search*.

Full-Text Search

Full-text search operators in Energy Logserver match documents against a query expression applied to indexed text fields. Each operator can be invoked through three query interfaces: the native query DSL (REST API), the SQL plugin, and the PPL plugin. The examples in this chapter show all three forms side by side for every operator.

- *Match*
- *Syntax*
- *Multi-match*
- *Query string*
- *Match phrase*
- *Simple query string*
- *Match phrase prefix*
- *Match boolean prefix*

Match

Use the `MATCH` function to search documents that match a `string`, `number`, `date`, or `boolean` value for a given field.

Syntax

```
match(field_expression, query_expression[, option=<option_value>]*)
```

You can specify the following options in any order:

- `analyzer`
- `auto_generate_synonyms_phrase`
- `fuzziness`
- `max_expansions`
- `prefix_length`
- `fuzzy_transpositions`
- `fuzzy_rewrite`
- `lenient`
- `operator`
- `minimum_should_match`
- `zero_terms_query`
- `boost`

Refer to the `match` query for parameter descriptions and supported values.

Example 1: Search the `message` field for the text "this is a test":

```
GET my_index/_search
{
  "query": {
    "match": {
      "message": "this is a test"
```

(continues on next page)

(continued from previous page)

```
    }
  }
}
```

SQL query:

```
SELECT message FROM my_index WHERE match(message, "this is a test")
```

PPL query:

```
SOURCE=my_index | WHERE match(message, "this is a test") | FIELDS message
```

Example 2: Search the `message` field with the `operator` parameter:

```
GET my_index/_search
{
  "query": {
    "match": {
      "message": {
        "query": "this is a test",
        "operator": "and"
      }
    }
  }
}
```

SQL query:

```
SELECT message FROM my_index WHERE match(message, "this is a test", operator=
↪'and')
```

PPL query:

```
SOURCE=my_index | WHERE match(message, "this is a test", operator='and') |␣
↪FIELDS message
```

Example 3: Search the `message` field with the `operator` and `zero_terms_query` parameters:

```
GET my_index/_search
{
  "query": {
    "match": {
      "message": {
        "query": "to be or not to be",
        "operator": "and",
        "zero_terms_query": "all"
      }
    }
  }
}
```

SQL query:

```
SELECT message FROM my_index WHERE match(message, "this is a test", operator=
↪'and', zero_terms_query='all')
```

PPL query:

```
SOURCE=my_index | WHERE match(message, "this is a test", operator='and', zero_
↪terms_query='all') | FIELDS message
```

Multi-match

To search for text in multiple fields, use `MULTI_MATCH` function. This function maps to the `multi_match` query used in ELS Data Node, to return the documents that match a provided text, number, date or boolean value with a given field or fields.

Multi-match Syntax

The `MULTI_MATCH` function lets you *boost* certain fields using **^** character. Boosts are multipliers that weigh matches in one field more heavily than matches in other fields. The syntax allows to specify the fields in double quotes, single quotes, surrounded by backticks, or unquoted. Use star `"*"` to search all fields. Star symbol should be quoted.

```
multi_match([field_expression+], query_expression[, option=<option_value>]*)
```

The weight is optional and is specified after the field name. It could be delimited by the `caret` character – `^` or by whitespace. Please, refer to examples below:

```
multi_match(["Tags" ^ 2, 'Title' 3.4, `Body`, Comments ^ 0.3], ...)
multi_match(["*"], ...)
```

You can specify the following options for `MULTI_MATCH` in any order:

- `analyzer`
- `auto_generate_synonyms_phrase`
- `cutoff_frequency`
- `fuzziness`
- `fuzzy_transpositions`
- `lenient`
- `max_expansions`
- `minimum_should_match`
- `operator`
- `prefix_length`
- `tie_breaker`
- `type`
- `slop`
- `zero_terms_query`
- `boost`

Please, refer to `multi_match` query *documentation* for parameter description and supported values.

For example, REST API search for `Dale` in either the `firstname` or `lastname` fields:

```
GET accounts/_search
{
  "query": {
    "multi_match": {
      "query": "Lane Street",
      "fields": [ "address" ],
    }
  }
}
```

could be called from *SQL* using `multi_match` function

```
SELECT firstname, lastname
FROM accounts
WHERE multi_match(['*name'], 'Dale')
```

or `multi_match` *PPL* function

```
SOURCE=accounts | WHERE multi_match(['*name'], 'Dale') | fields firstname,␣
↪lastname
```

firstname	lastname
Dale	Adams

Query string

To split text based on operators, use the `QUERY_STRING` function. The `QUERY_STRING` function supports logical connectives, wildcard, regex, and proximity search. This function maps to the `query_string` query used in ELS Data Node, to return the documents that match a provided text, number, date or boolean value with a given field or fields.

Query string Syntax

The `QUERY_STRING` function has syntax similar to `MATCH_QUERY` and lets you *boost* certain fields using **^** character. Boosts are multipliers that weigh matches in one field more heavily than matches in other fields. The syntax allows to specify the fields in double quotes, single quotes, surrounded by backticks, or unquoted. Use star `"*"` to search all fields. Star symbol should be quoted.

```
query_string([field_expression+], query_expression[, option=<option_value>]*)
```

The weight is optional and is specified after the field name. It could be delimited by the `caret` character – `^` or by whitespace. Please, refer to examples below:

```
query_string(["Tags" ^ 2, 'Title' 3.4, `Body`, Comments ^ 0.3], ...)
query_string(["*"], ...)
```

You can specify the following options for `QUERY_STRING` in any order:

- `analyzer`
- `allow_leading_wildcard`

- analyze_wildcard
- auto_generate_synonyms_phrase_query
- boost
- default_operator
- enable_position_increments
- fuzziness
- fuzzy_rewrite
- escape
- fuzzy_max_expansions
- fuzzy_prefix_length
- fuzzy_transpositions
- lenient
- max_determinized_states
- minimum_should_match
- quote_analyzer
- phrase_slop
- quote_field_suffix
- rewrite
- type
- tie_breaker
- time_zone

Example of using `query_string` in SQL and PPL queries:

The REST API search request

```
GET accounts/_search
{
  "query": {
    "query_string": {
      "query": "Lane Street",
      "fields": [ "address" ],
    }
  }
}
```

could be called from *SQL*

```
SELECT account_number, address
FROM accounts
WHERE query_string(['address'], 'Lane Street', default_operator='OR')
```

or from *PPL*

```
SOURCE=accounts | WHERE query_string(['address'], 'Lane Street', default_
↪operator='OR') | fields account_number, address
```

account_number	address
1	880 Holmes Lane
6	671 Bristol Street
13	789 Madison Street

Match phrase

To search for exact phrases, use `MATCHPHRASE` or `MATCH_PHRASE` functions.

Match phrase Syntax

```
matchphrasequery(field_expression, query_expression)
matchphrase(field_expression, query_expression[, option=<option_value>]*)
match_phrase(field_expression, query_expression[, option=<option_value>]*)
```

The `MATCHPHRASE`/`MATCH_PHRASE` functions let you specify the following options in any order:

- `analyzer`
- `slop`
- `zero_terms_query`
- `boost`

Example of using `match_phrase` in SQL and PPL queries:

The REST API search request

```
GET accounts/_search
{
  "query": {
    "match_phrase": {
      "address": {
        "query": "880 Holmes Lane"
      }
    }
  }
}
```

could be called from *SQL*

```
SELECT account_number, address
FROM accounts
WHERE match_phrase(address, '880 Holmes Lane')
```

or *PPL*

```
SOURCE=accounts | WHERE match_phrase(address, '880 Holmes Lane') | FIELDS␣
↪account_number, address
```

account_number	address
1	880 Holmes Lane

Simple query string

The `simple_query_string` function maps to the `simple_query_string` query in Energy Logserver. It returns the documents that match a provided text, number, date or boolean value with a given field or fields. The `^` lets you *boost* certain fields. Boosts are multipliers that weigh matches in one field more heavily than matches in other fields.

Simple query string Syntax

The syntax allows to specify the fields in double quotes, single quotes, surrounded by backticks, or unquoted. Use star `"*"` to search all fields. Star symbol should be quoted.

```
simple_query_string([field_expression+], query_expression[, option=<option_
↪value>]*)
```

The weight is optional and is specified after the field name. It could be delimited by the `caret` character – `^` or by whitespace. Please, refer to examples below:

```
simple_query_string(["Tags" ^ 2, 'Title' 3.4, `Body`, Comments ^ 0.3], ...)
simple_query_string(["*"], ...)
```

You can specify the following options for `SIMPLE_QUERY_STRING` in any order:

- `analyze_wildcard`
- `analyzer`
- `auto_generate_synonyms_phrase_query`
- `boost`
- `default_operator`
- `flags`
- `fuzzy_max_expansions`
- `fuzzy_prefix_length`
- `fuzzy_transpositions`
- `lenient`
- `minimum_should_match`
- `quote_field_suffix`

Example of using `simple_query_string` in SQL and PPL queries:

The REST API search request

```
GET accounts/_search
{
  "query": {
    "simple_query_string": {
```

(continues on next page)

(continued from previous page)

```
      "query": "Lane Street",
      "fields": [ "address" ],
    }
  }
}
```

could be called from *SQL*

```
SELECT account_number, address
FROM accounts
WHERE simple_query_string(['address'], 'Lane Street', default_operator='OR')
```

or from *PPL*

```
SOURCE=accounts | WHERE simple_query_string(['address'], 'Lane Street',␣
↪default_operator='OR') | fields account_number, address
```

account_number	address
1	880 Holmes Lane
6	671 Bristol Street
13	789 Madison Street

Match phrase prefix

To search for phrases by given prefix, use `MATCH_PHRASE_PREFIX` function to make a prefix query out of the last term in the query string.

Match phrase prefix Syntax

```
match_phrase_prefix(field_expression, query_expression[, option=<option_value>
↪]*)
```

The `MATCH_PHRASE_PREFIX` function lets you specify the following options in any order:

- `analyzer`
- `slop`
- `max_expansions`
- `zero_terms_query`
- `boost`

Example of using `match_phrase_prefix` in SQL and PPL queries:

The REST API search request

```
GET accounts/_search
{
  "query": {
    "match_phrase_prefix": {
      "author": {
```

(continues on next page)

(continued from previous page)

```
        "query": "Alexander Mil"
      }
    }
  }
}
```

could be called from *SQL*

```
SELECT author, title
FROM books
WHERE match_phrase_prefix(author, 'Alexander Mil')
```

or *PPL*

```
source=books | where match_phrase_prefix(author, 'Alexander Mil') | fields␣
↪author, title
```

author	title
Alan Alexander Milne	The House at Pooh Corner
Alan Alexander Milne	Winnie-the-Pooh

Match boolean prefix

Use the `match_bool_prefix` function to search documents that match text only for a given field prefix.

Match boolean prefix Syntax

```
match_bool_prefix(field_expression, query_expression[, option=<option_value>
↪]*)
```

The `MATCH_BOOL_PREFIX` function lets you specify the following options in any order:

- `minimum_should_match`
- `fuzziness`
- `prefix_length`
- `max_expansions`
- `fuzzy_transpositions`
- `fuzzy_rewrite`
- `boost`
- `analyzer`
- `operator`

Example of using `match_bool_prefix` in SQL and PPL queries:

The REST API search request

```
GET accounts/_search
{
  "query": {
    "match_bool_prefix": {
      "address": {
        "query": "Bristol Stre"
      }
    }
  }
}
```

could be called from *SQL*

```
SELECT firstname, address
FROM accounts
WHERE match_bool_prefix(address, 'Bristol Stre')
```

or *PPL*

```
source=accounts | where match_bool_prefix(address, 'Bristol Stre') | fields␣
↪firstname, address
```

firstname	address
Hattie	671 Bristol Street
Nanette	789 Madison Street

Archive

The Archive module allows you to create compressed data files (zstd algorithm) from Energy Logserver indices. The archive checks the age of each document in the index and if it is older than defined in the job, it is copied to the archive file.

- *Configuration*
- *Archive Task*
- *Archive Search*
- *Archive Restore*
- *Search/Restore task with archives without metadata*
- *Archive Clean*
- *Archive Directory Structure*
- *Archives Checksum Verification*
- *Identifying progress of archivisation/restoration process*
- *Command Line tools*

Configuration

Enabling module

To configure the module edit the UI main configuration file and set path to the archive directory - location where the archive files will be stored:

```
vim /etc/logserver-gui/logserver-gui.yml
```

The key ships uncommented with a default location, so change it only if the archives belong somewhere else, for example on a mounted volume:

```
archive.archivefolderpath: "/usr/share/logserver-gui/data/archive/archives/"
```

The remaining archive settings (`archive.compressionOptions`, `archive.requestTimeout`, `archive.scrollDuration`, `archive.integrityCheck`) ship commented out and fall back to their defaults.

Archives will be saved inside above directory in the subdirectories that describes year and month of its creation. For example:

```
/usr/share/logserver-gui/data/archive/archives
├── 2022
│   └── 08
│       ├── enc3_2022-08-15.json.zstd
│       └── skimmer-2022.08_2022-08-06.json.zstd
└── 2023
    ├── 05
    │   ├── enc1_2023-05-25.json.zstd
    │   ├── enc2_2023-05-25.json.zstd
    │   └── skimmer-2023.05_2023-05-25.json.zstd
    └── 07
        └── skimmer-2023.07_2023-07-30.json.zstd
```

Archive Task

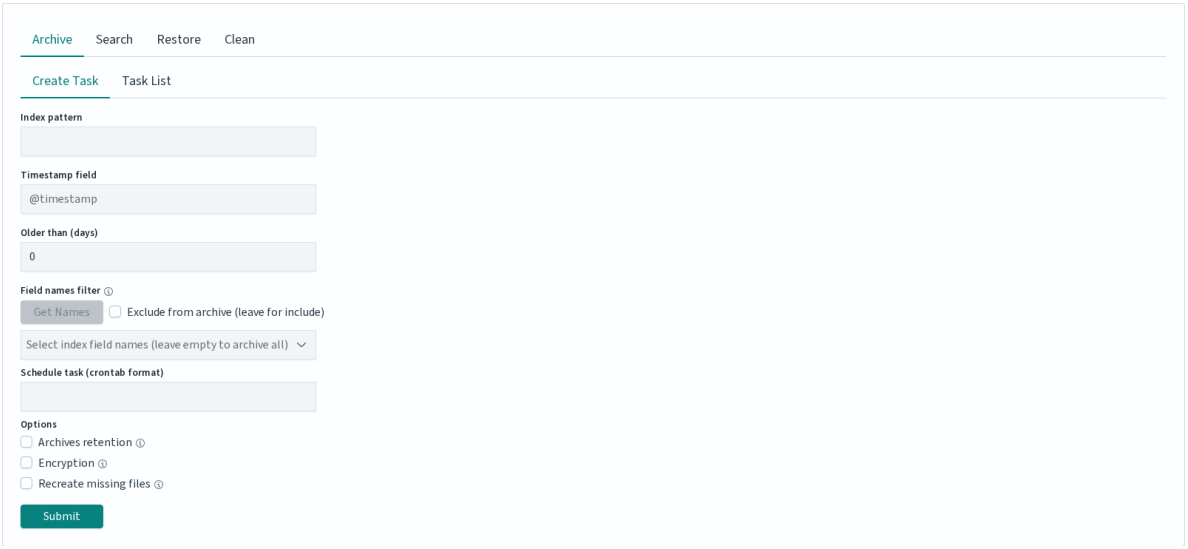
Create Archive task

1. From the main navigation go to the "Archive" module.
2. On the "Archive" tab select "Create Task" and define the following parameters:
   - `Index pattern` - for the indices that will be archived, for example, `syslog*`
   - `Timestamp field` - time field of the indices (default **@timestamp**)
   - `Older than (days)` - number of days after which documents will be archived
   - `Field names filter` - filter fields that should be archived
   - `Schedule task` (crontab format) - the work schedule of the ordered task.

   Under **Options**:

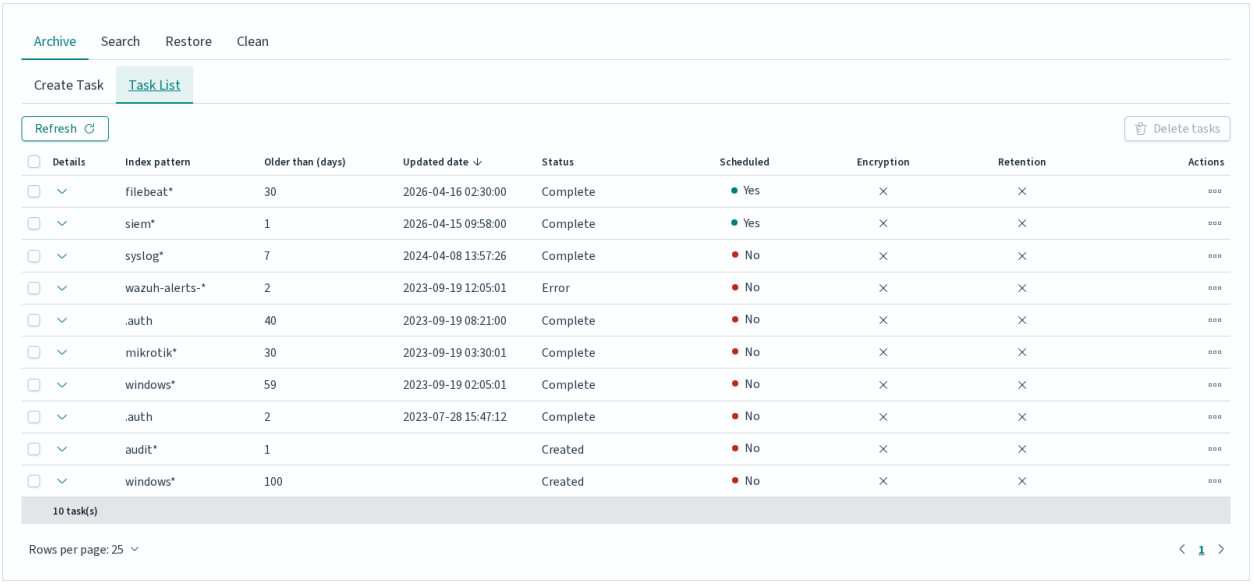
   - `Archives retention` - keep only archives from a chosen time window; older files are removed automatically.
   - `Encryption` - after enabling encryption, prompt with two password fields will be shown.

- `Recreate missing files` - re-run the job on dates whose archive file has been deleted, so that the archive set can be rebuilt.



Task List

In the `Task List`, you can follow the current status of ordered tasks. You can modify the task scheduler or delete a single or many tasks at once.



If the archiving task finds an existing archive file that matches the data being archived, it will check the number of documents in the archive and the number of documents in the index. If there is a difference in the number of documents then new documents will be added to the archive file.

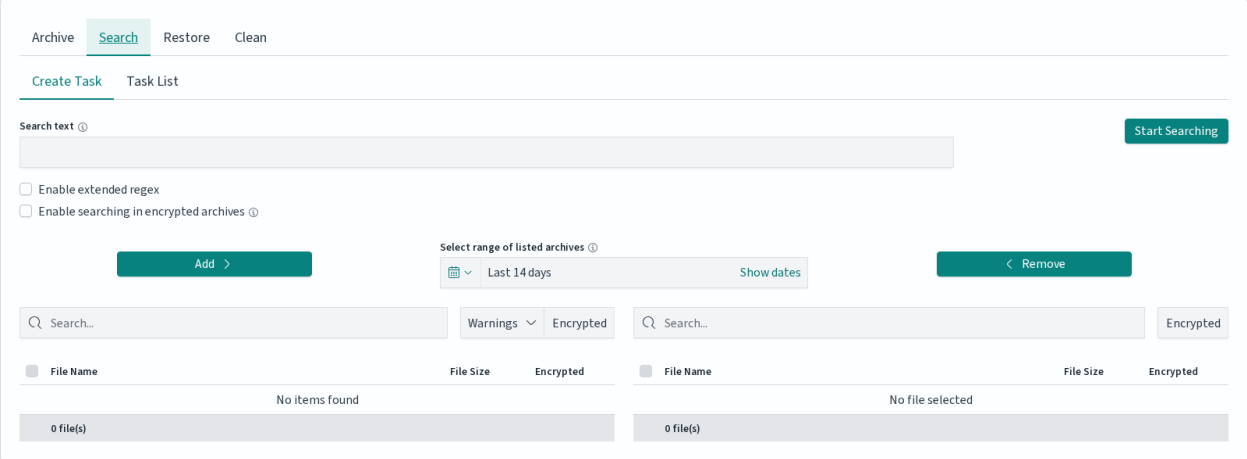
To show more details of the task, click on the details cell of the desired row.

Archive Search

The Archive Search module can search archive files for the specific content and return results in the `Task List`

Create Search task

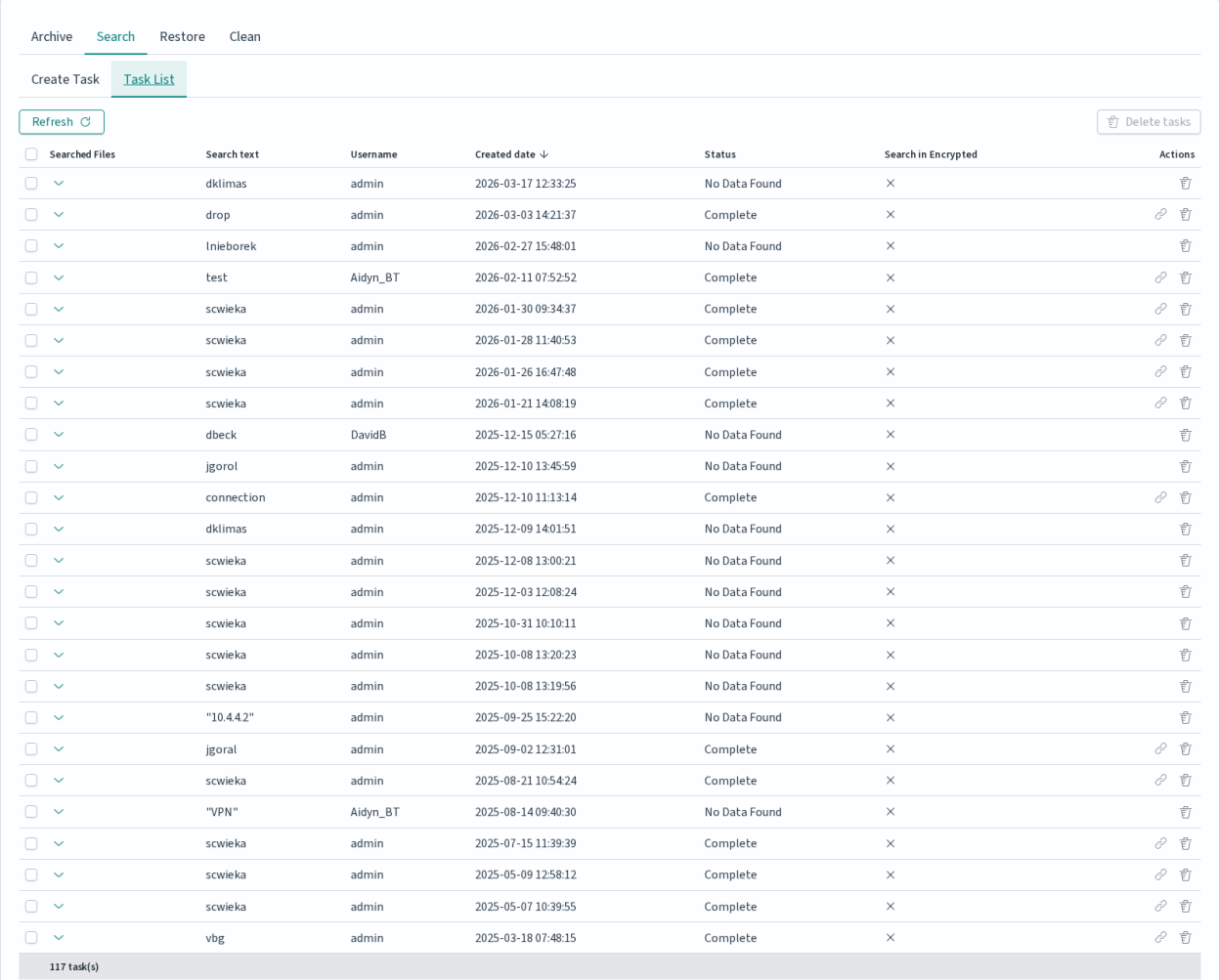
1. From the main navigation go to the `Archive` module.
2. On the `Search` tab select `Create Task` and define the following parameters:
    - `Search text` - field for entering the text to be searched.
    - `Enable extended regex` - treat the search text as an extended regular expression instead of a plain string.
    - `Enable searching in encrypted archives` - enable option to search in encrypted archives.
    - `Select range of listed archives` - files that match the selected range are displayed in the left-hand list (default **last 14 days**). Use the `Warnings` and `Encrypted` filters above the list to narrow it down.
    - Move the archives to search through from the left list to the right list with `Add`; use `Remove` to take them back.



The table footer shows the total number of found files for the specified date range.

Search Task List

The searching process can take a long time. On the `Task List`, you can follow the status of the searching process. Also, you can view results and delete tasks.



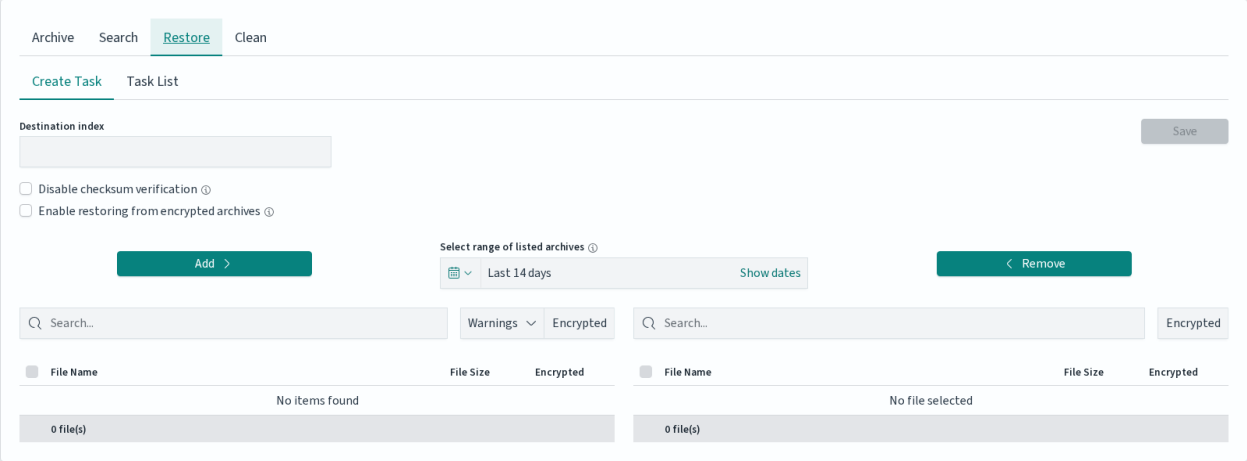
Archive Restore

The Archive Restore module moves data from the archive to the Energy Logserver index and makes it available online.

Create Restore task

1. From the main navigation go to the `Archive` module.
2. On the `Restore` tab select `Create Task` and define the following parameters:

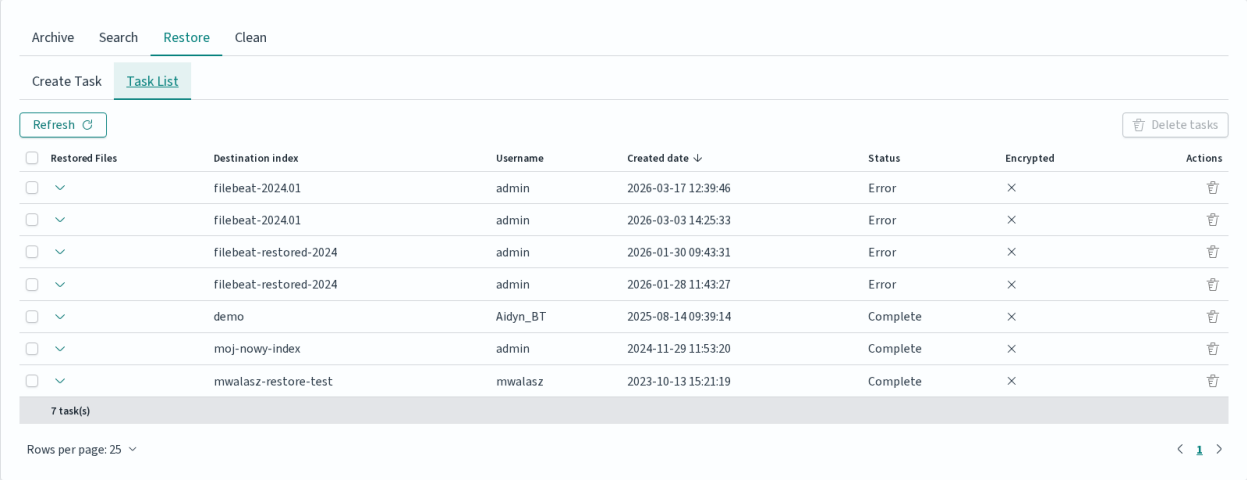
- `Destination index` - If a destination index does not exist it will be created. If exists data will be appended.
- `Disable checksum verification` - skip the checksum check that otherwise guards against corrupted archives. Use only when a known-good archive fails verification for reasons unrelated to its content.
- `Enable restoring from encrypted archives` - enable option to restore data from encrypted archives.
- `Select range of listed archives` - files that match the selected range are displayed in the left-hand list (default **last 14 days**). Use the `Warnings` and `Encrypted` filters above the list to narrow it down.
- Move the archives to restore from the left list to the right list with `Add`; use `Remove` to take them back.



The table footer shows the total number of found files for the specified date range.

Restore Task List

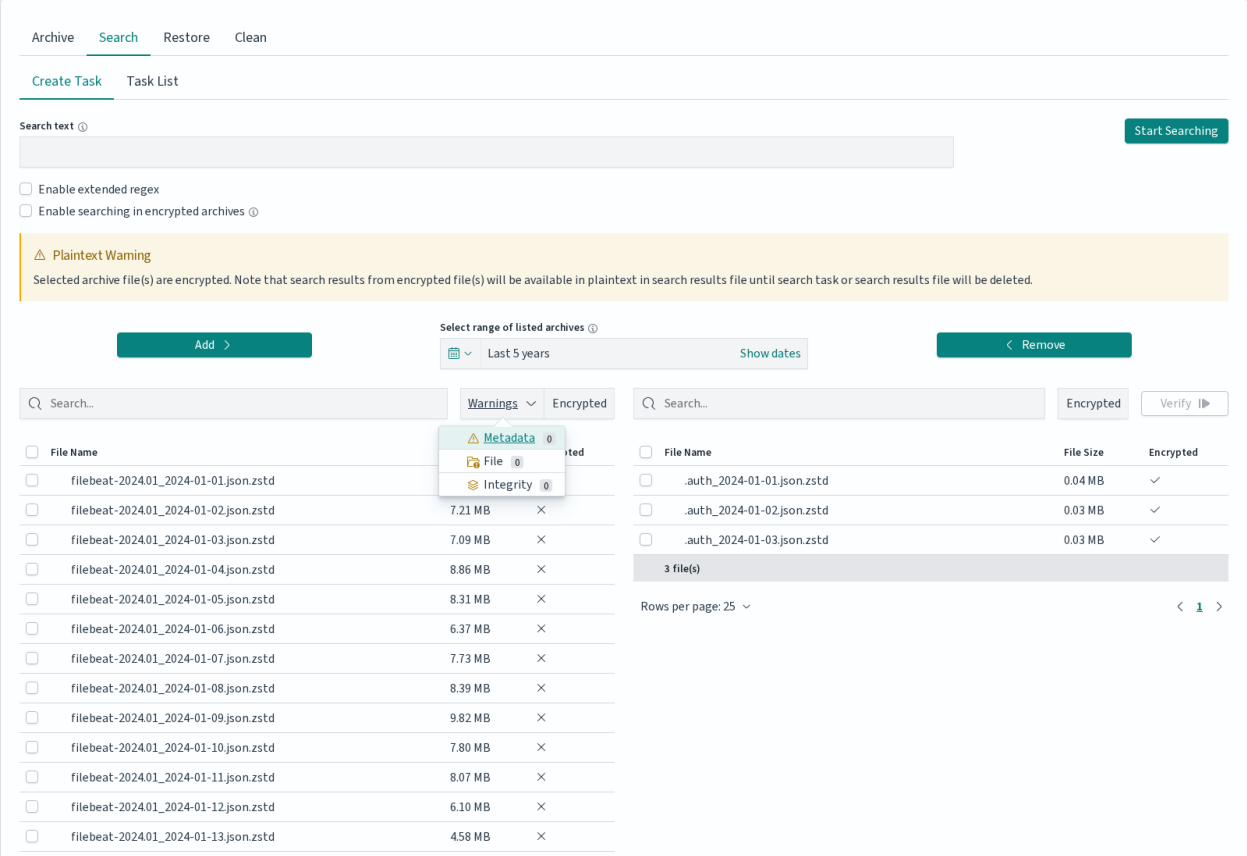
The process will index data back into the Data Node. Depending on archive size, the process can take a long time. On the `Task List` you can follow the status of the recovery process. Also you can view result and delete tasks.



Search/Restore task with archives without metadata

When creating Search or Restore tasks, the left-hand list of archives can be filtered by the `Warnings` dropdown above it. The dropdown offers three categories, each with a counter of matching files:

- `Metadata` - the archive is listed but its metadata record could not be found.
- `File` - metadata exists but the archive file itself is missing from disk.
- `Integrity` - metadata and file are present, but the stored checksum does not match the archive contents.



When particular archive's metadata could not be found following icon will be displayed:



That archive can be used for task creation, but there are some issues to keep in mind:

- encryption status of the archive without metadata cannot be established (can be either encrypted or not)
- when task has enabled encryption handling (e.g. `Enable restoring from encrypted archives` or `Enable searching in encrypted archives`), archives will be decrypted with provided password. If archive was not decrypted, an error is expected
- when archive is potentially encrypted and password is not provided, an error is expected.

On the other hand, when metadata is present, but archive itself could not be located, following icon will be displayed:



That archive cannot be used for task creation and so cannot be selected.

Archive Clean

The `Clean` tab allows you to delete data from the archive.

To do so, go to the `Clean` section in the `Archive` tab.

Then, check the boxes next to the files you want to delete and press the `Delete` button in the top right corner.

Archive Directory Structure

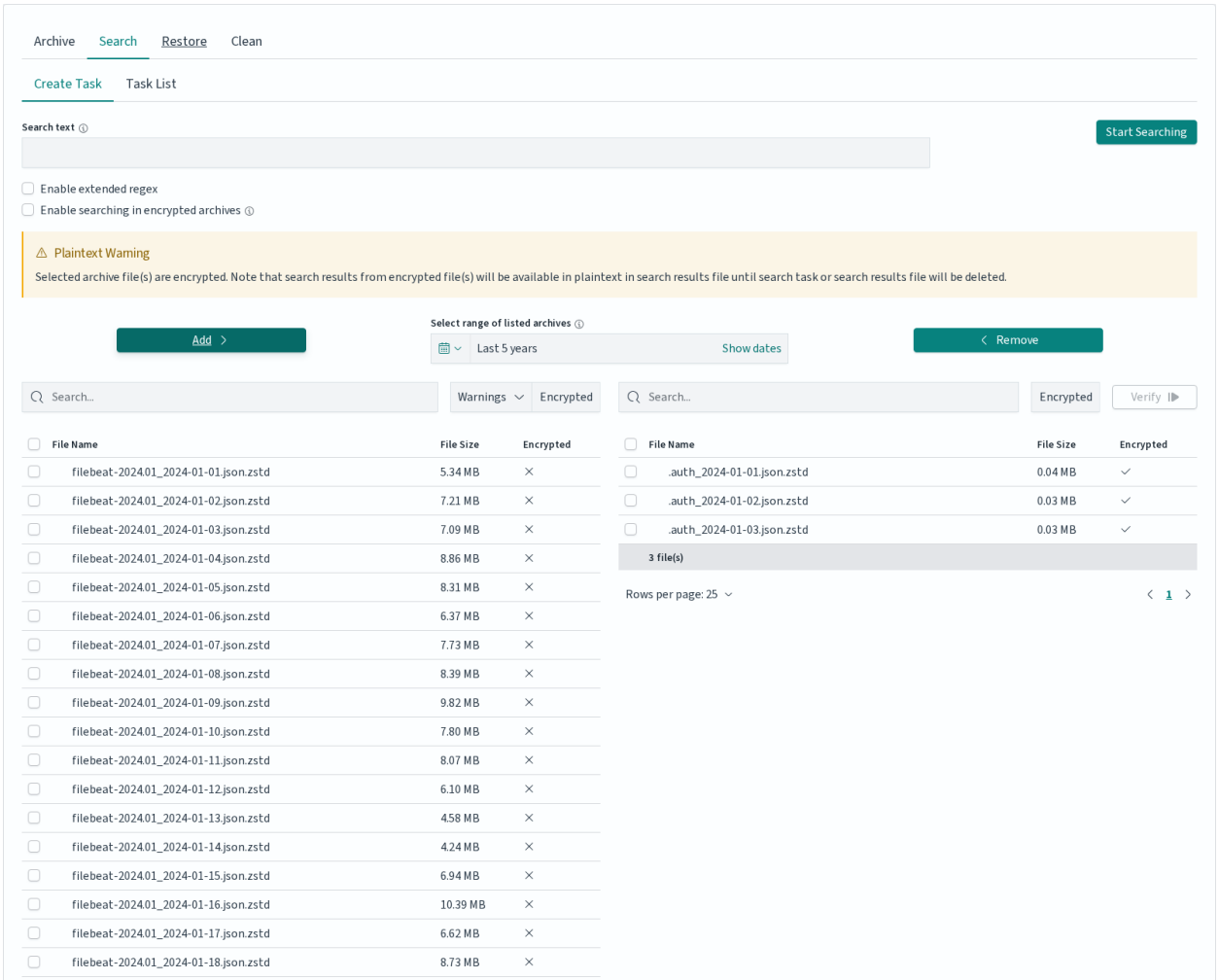
New archives will be created in the configured archive.archivefolderpath in a systematic order. They can be found under a path based on the date the archive was created: */archivefolderpath/*year/$month. This method of storing archives ensures better readability and significantly simplifies viewing large numbers of files. The final directory is determined by the last segment of the archive name, which contains the archive creation date. For example, assuming that the root archivefolderpath is set to `/download`, archive `sample-archive_2023-11-01.json.zstd` will be saved to the `/download/2023/11/` automatically created directory. Archives once saved to the root directory will be displayed normally in the GUI and will be accessible in the same way as those saved in a sorted manner.

Archives Checksum Verification

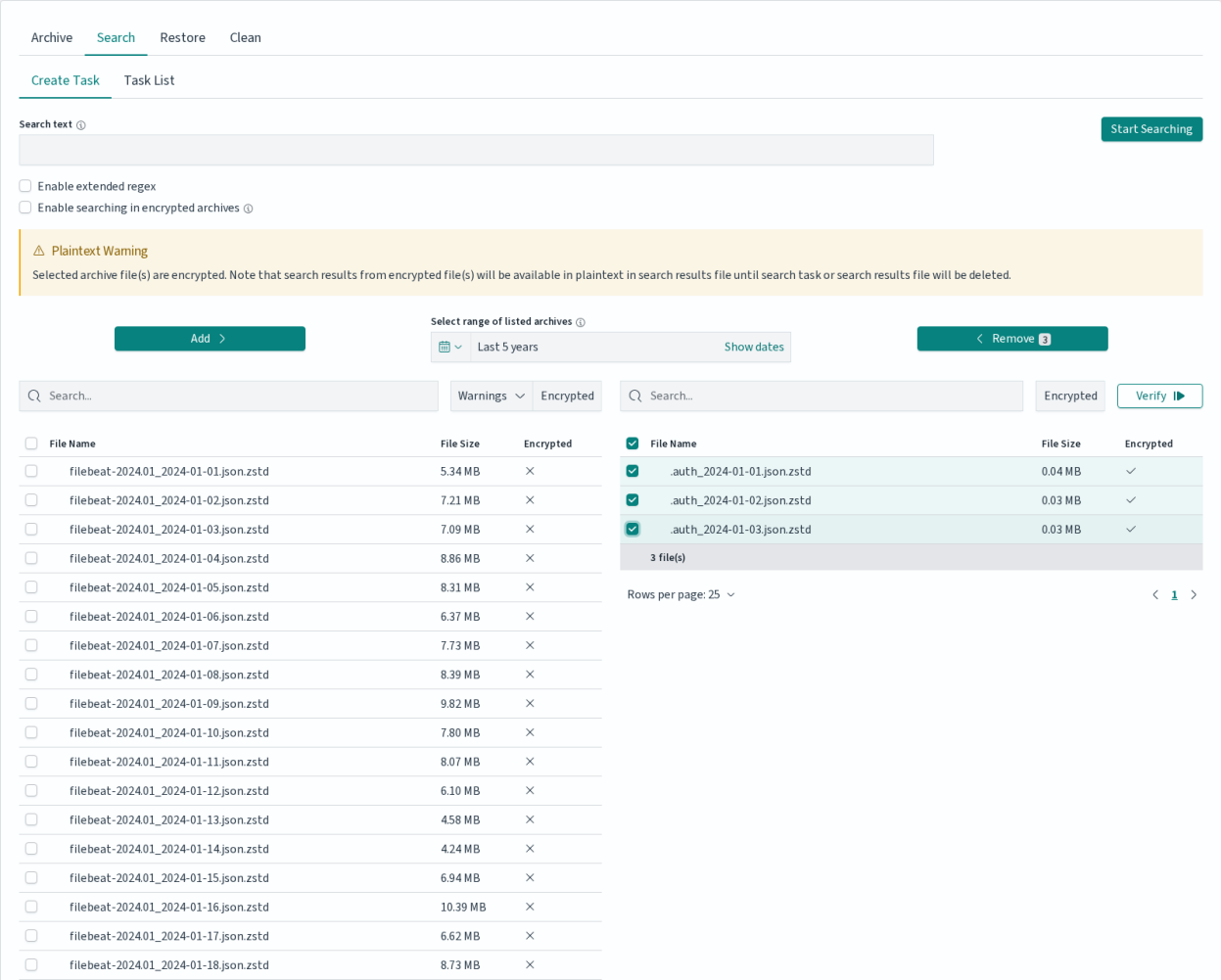
Archives checksum verification feature has been integrated into the `Create Task` section, enhancing the functionality of both the `Search` and `Restore` tabs. This feature adds an extra layer of confidence in the accuracy and reliability of the stored files.

Starting Verification

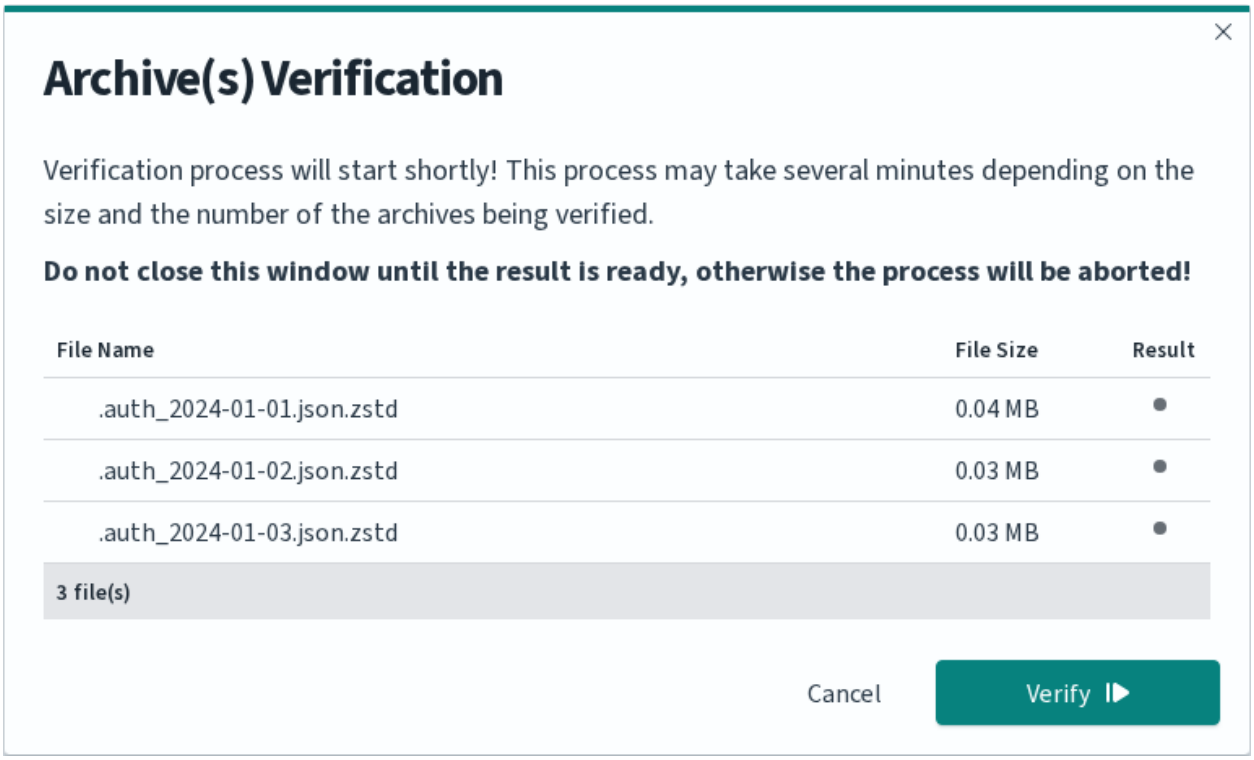
To start the verification process navigate to one of the mentioned tabs. Select archives that will be checked and move them to the right-side table. Below picture presents the button that will be activated, as soon as any complete archives (without any warning) will be selected in the right table. Archives without metadata cannot be utilized due to the lack of necessary details.



In order to optimize resource utilization and enhance the overall efficiency, a double selection mechanism has been implemented for the file checksum verification, acknowledging the resource-intensive nature of this feature. This ensures, that only desired files will be processed.

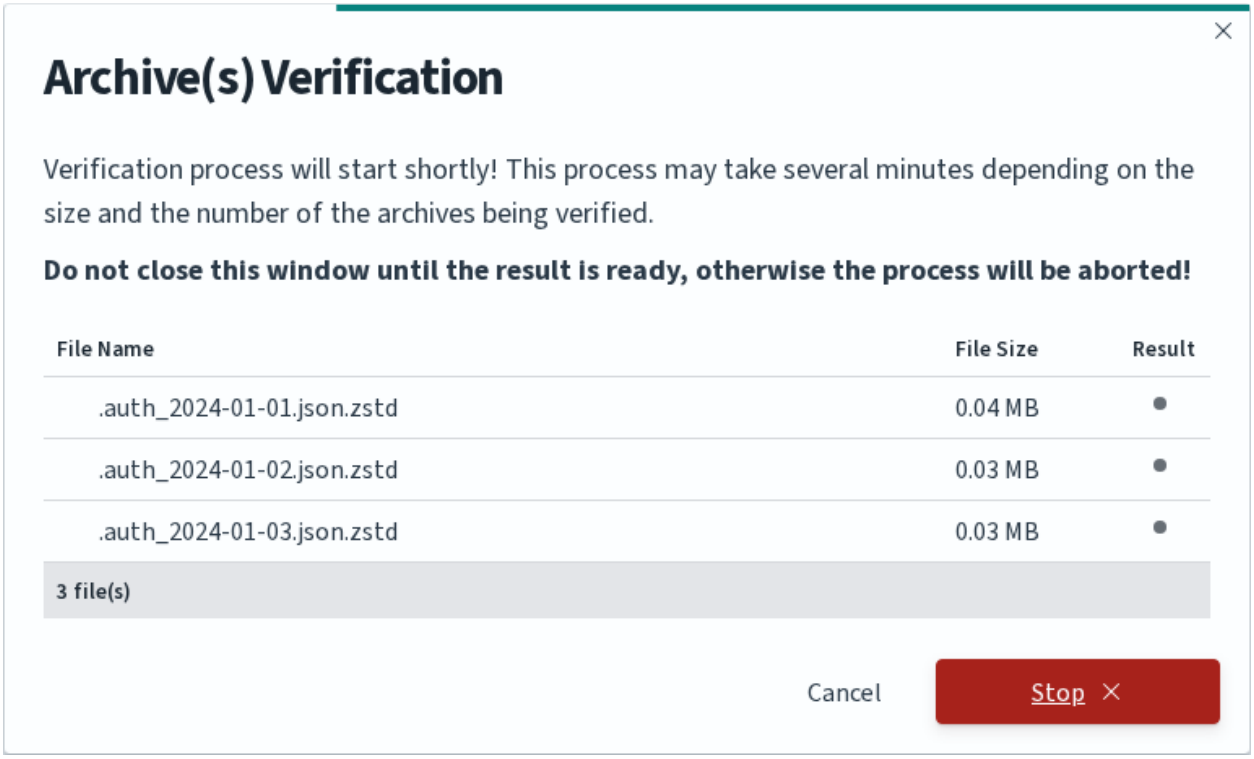


Above picture presents correctly selected archives and a situation when the verification can be started by clicking the `Verify` button. The following modal is displayed.



The modal lists the selected files along with their size and a placeholder for the per-file result. Hovering over a result dot reveals additional information - the current status of the verification or, if a file failed, the reason why.

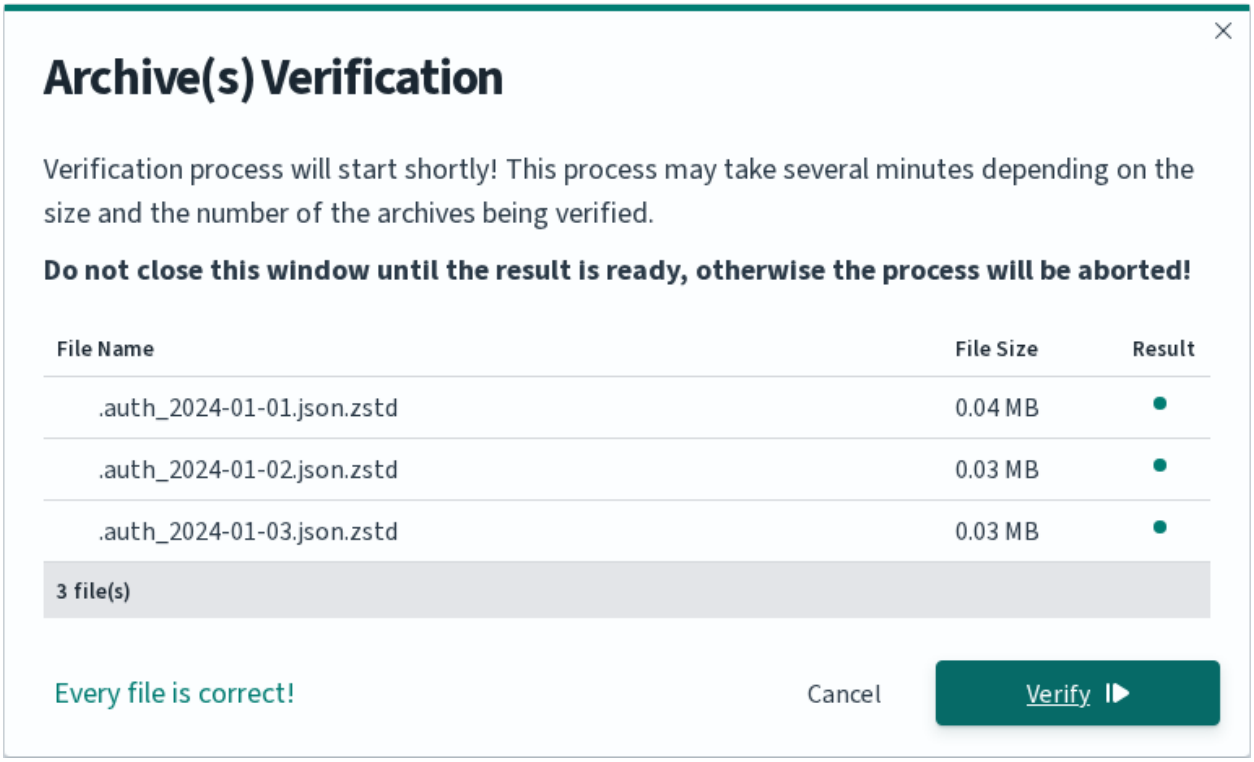
To start verification, click the `Verify` button in the bottom right corner. While the process runs, the button changes to `Stop` and the modal must stay open - closing or reloading the page aborts the verification.



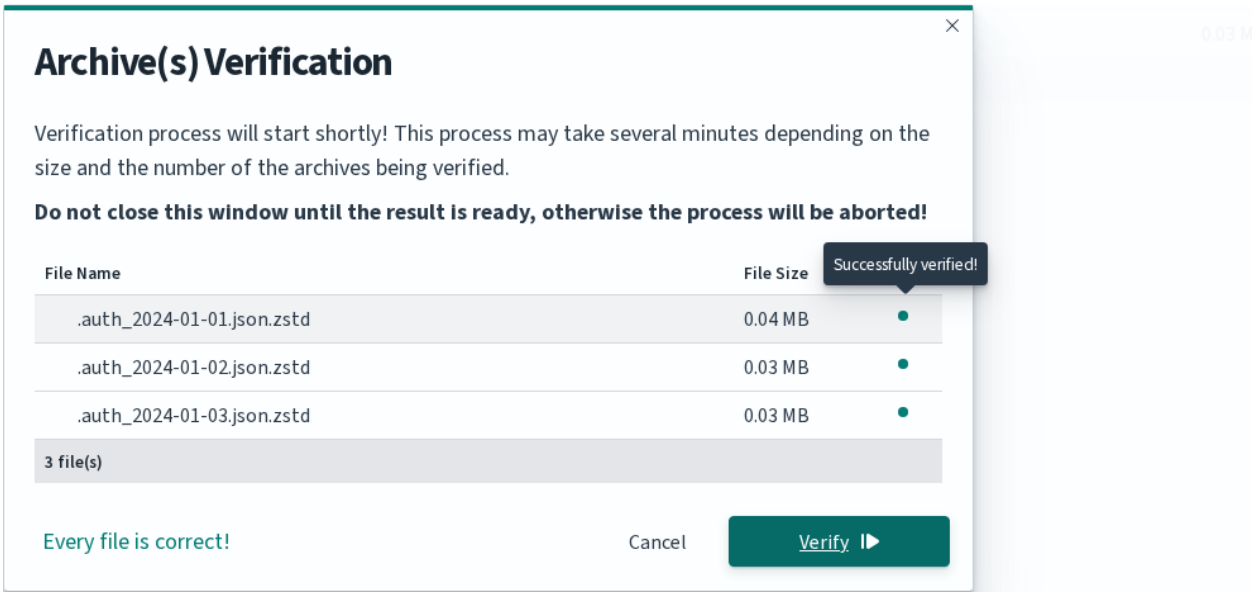
Verification Result

When verification completes, the summary appears in the bottom left corner of the modal and each row gets a coloured dot in the `Result` column. The summary message reflects the overall outcome:

- green - every archive passed the checksum check (message `Every file is correct!`).
- yellow - some archives passed, others did not.
- red - every selected archive failed verification.



Hovering over any result dot shows a tooltip with the status of that file. For failed rows the tooltip contains the error message that explains why verification did not pass.



Identifying progress of archivisation/restoration process

The `/usr/share/logserver-gui/data/archive/tasks` directory contains metadata files, that indicates the current status of the task. These files contain information about all indices, that:

- are about to be processed (**"Waiting"** status)
- are processing (**"Running"** status)
- were processed (**"Complete"** status)

If everything went according to the plan and the process has successfully finished, that metadata file will be removed. However, when some index cannot be processed or something unexpected happened, there will be **"Error"** status, with detailed message in the "error" field and metadata will remain in the system.

The above described situation is reflected in the GUI by the **Status** column in the Task List tables.

Moreover, in the metadata files can be found current process id (`pid`), total documents count and encryption details.

Uncompleted Tasks removal

1. List archive folder and find filename generated by uncompleted task.

```
ls -la /archivefolderpath/
-rw-r--r--. 1 user group          13 Mar 21 10:07 prd-srv-win-ad-2022.12.
21_2022-12-21.json.zstd
```

2. Find document in `.archive` index using filename from previous step

```
curl -s -k -X GET -ulogserver:...  http://127.0.0.1:9200/.archive/_search?
size=10000 |jq '.'| grep -B4 "prd-srv-win-ad-2022.12.21"
```

3. Write down it's ID

```
"_id": "Q8teA4cBj_ghAWXFcMJA",
      "_score": 1.0,
      "_source": {
```

(continues on next page)

(continued from previous page)

```
        "date": "2023-03-21T08:52:13.502Z",
        "filename": "prd-srv-win-ad-2022.12.21_2022-12-21.json.zstd",
```

4. Remove document using saved ID

```
curl -s -k -X DELETE -ulogserver:... http://127.0.0.1:9200/.archive/_doc/
↪Q8teA4cBj_ghAWXFcMJA
```

Command Line tools

Archive files can be handled by the following commands `zstd`, `zstdcat`, `zstdgrep`, `zstdless`, `zstdmt`.

zstd

The command for decompress `*.zstd` the Archive files, for example:

```
zstd -d winlogbeat-2020.10_2020-10-23.json.zstd -o
 winlogbeat-2020.10_2020-10-23.json
```

zstdcat

The command for concatenate `*.zstd` Archive files and print content on the standard output, for example:

```
zstdcat winlogbeat-2020.10_2020-10-23.json.zstd
```

zstdgrep

The command for print lines matching a pattern from `*.zstd` Archive files, for example:

```
zstdgrep "optima" winlogbeat-2020.10_2020-10-23.json.zstd
```

Above example is searching documents contain the "optima" phrase in winlogbeat-2020.10_2020-10-23.json.zstd archive file.

zstdless

The command for viewing Archive `*.zstd` files, for example:

```
zstdless winlogbeat-2020.10_2020-10-23.json.zstd
```

zstdmt

The command for compress and decompress Archive `*.zstd` file using multiple CPU core (default is 1), for example:

```
zstdmt -d winlogbeat-2020.10_2020-10-23.json.zstd -o winlogbeat-2020.10_2020-
↪10-23.json
```

4.7 Network Probe

The Network Probe is the ingestion tier. Every source, whether it runs an agent or not, delivers its data to a probe pipeline that parses and enriches it before it reaches the Data Node. The probe also provides deep packet inspection, flow analysis, and network behavior monitoring in its own right.

Start with *Connecting Data Sources* to choose a collection method for a new source.

4.7.1 Built-in Monitoring Tools

IDS and Full Packet Capture

Network Probe provides a high performance Network IDS, IPS and Network Security Monitoring engine. It captures anomalies and logs network traffic alarms based on defined rules, with the ability to save PCAP of all detected packets for detailed forensic analysis.

NDR (Network Detection and Response)

Network Probe acts as a passive network traffic analyzer supporting investigations of suspicious or malicious activity. It provides extensive logs describing network activity including comprehensive connection records, application-layer transcripts (HTTP sessions, DNS requests, SSL certificates, SMTP content), and support for external SIEM integration for querying and analysis.

NetFlow Collection

Network Probe includes a multi-flow accounting feature supporting NetFlow v5/v9, IPFIX and sFlow packets on multiple interfaces (IPv4 and IPv6). Beyond collecting flows, it can classify, aggregate, replicate to 3rd party collectors, and export forwarding-plane data for comprehensive traffic visibility.

Connecting Data Sources

Every source reaches Energy Logserver in one of two ways. Either you install software on the source and it ships data out, or the Network Probe receives or fetches the data without anything being installed on the source at all. The first is agent-based collection, the second is agentless.

This page helps you pick between them and points to the procedure for each method. It does not repeat those procedures.

- *How data reaches the system*
- *Choosing between agent and agentless*
- *Agent-based collection*
- *Agentless collection*
  - *Syslog*
  - *Windows Event Collector*
  - *WinRM polling*
  - *Flow export*
  - *SPAN through Kafka*
  - *Database polling*
  - *API and script polling*
  - *Files and shares*
- *Ports and firewall*
- *Verify the source is delivering*

- *Related pages*

How data reaches the system

Whichever method you choose, the data lands in a Network Probe pipeline, which receives it, parses it into fields, and writes it to the Data Node. The three pipeline stages and the files that hold them are described in *How data flows through a probe*.

What the method decides is only the first stage, the input. An agent connects to a listener the probe already runs. An agentless source either sends to a listener or is polled by the probe on a schedule.

Choosing between agent and agentless

Method	Runs on the source	Direction	Choose it when	Trade-off
Beats agent	Beat service	Source connects to probe	You control the host and need file, event log, metric, or packet data	One service to deploy and update per host
Beats managed by MasterAgent	Beat service plus MasterAgent	Source connects to probe	You run many hosts and want to push configuration and restart Beats from the GUI	Adds a second service and a certificate to manage
Endpoint Security agent	Endpoint Security agent	Source connects to manager	You need file integrity monitoring and security configuration assessment	Heavier than a Beat, separate enrollment
Syslog	Nothing	Source sends to probe	The device can only emit syslog, which covers most network and security appliances	No delivery guarantee over UDP, parsing depends on the vendor format
Windows Event Collector	Nothing	Source pushes to collector	You cannot install software on Windows hosts	Requires certificates, group policy, and a WS-Man subscription
WinRM polling	Nothing	Probe polls the source	You need Windows data from a small number of hosts without agents or subscriptions	Polling interval limits freshness, credentials stored on the probe
Flow export	Nothing	Source sends to probe	You need traffic visibility across a network rather than logs from a host	Flow records describe connections, not payloads
SPAN through Kafka	Nothing	Probe reads the mirror	You need full traffic analysis from a mirrored port	Requires switch configuration and the bundled Kafka broker
Database polling	Nothing	Probe polls the source	The application logs to a table rather than a file	Adds query load on the database, needs a read account
API or script polling	Nothing	Probe polls the source	The vendor exposes an API, as cloud and vulnerability scanners do	Credentials live on the probe, rate limits apply
File or share	Nothing	Probe reads the path	The source already writes to a share the probe can mount	Delivery depends on the share staying available

Agent-based collection

Path	Where the procedure lives
MasterAgent and Beats	*Agents*
Endpoint Security agent	*SIEM Agents Configuration* and *Endpoint Security Capabilities*
Collector shipped with an integration	*Advanced Installation*

Beats ship with the installer at version 7.17.29 for both Windows and Linux. Winlogbeat is Windows only. The installer copies the Linux packages and the Windows archives into the agent repository that the GUI serves, so you can download them from the Console rather than from the installation media.

Only a small number of integrations bundle a collector. For every other integration the wizard has no agent step, because the source sends data on its own or the probe fetches it.

Agentless collection

Syslog

The probe listens on TCP and UDP 5514, and the installer forwards the standard port 514 to it. This is the path for firewalls, routers, switches, and any appliance that emits syslog. See *Input "network"* for the listener and the firewall rules, and *Custom Integrations* for writing a parser when the vendor is not in the catalog.

Windows Event Collector

Windows hosts forward their event logs to a collector that subscribes to them over WS-Man, with nothing installed on the source. The collector listens on TCP 5986 and forwards the events to a probe pipeline. Certificates and a group policy on the source side are required. See *Event Collector*.

WinRM polling

The probe runs a script that queries Windows hosts over WinRM on a schedule and turns the results into events. This suits a handful of hosts where neither an agent nor a subscription is practical. The integration is installed from the catalog, see *Integrations*.

Flow export

Network devices export flow records to the probe: NetFlow on UDP 2055, IPFIX on 4739, and sFlow on UDP 6343. To account for traffic that the probe host itself sees, rather than traffic reported by a device, see *pmacct Local Traffic Monitoring*.

SPAN through Kafka

Traffic mirrored from a switch port is analyzed by the bundled sensors and carried to the pipeline through Kafka. Kafka and Zookeeper are installed but masked, and must be unmasked before use, see *Kafka and Zookeeper*. The sensor side is covered in *Suricata IDS* and *Zeek*.

Database polling

The probe queries a table on a schedule and turns each row into an event. MySQL, MSSQL, Oracle, and PostgreSQL are covered in *Input database*.

API and script polling

The probe runs a command or calls an API on a schedule. This is how cloud services and vulnerability scanners are collected. For an integration that is configured without the GUI, see *Manual Configuration*.

Files and shares

The probe tails a local path, or a remote share it has mounted. See *Input File* and *Getting data from share folder*.

Ports and firewall

The ports the installer opens are listed in *Network Communication Requirements*.

Integrations bind listener ports of their own, and the installer does not open those. After enabling an integration whose source sends data to the probe, find the port it listens on and open it:

```
grep -r "port =>" /etc/logserver-probe/conf.d/<integration>/
firewall-cmd --zone=public --add-port=<port>/tcp --permanent
firewall-cmd --zone=public --add-port=<port>/udp --permanent
firewall-cmd --reload
```

> **Note**
>
> Two integrations cannot share a listener port. If a pipeline fails to start with an address already in use error, check whether another enabled integration binds the same port and change one of them.

Verify the source is delivering

1. Confirm the probe is running: `systemctl status logserver-probe`.
2. Confirm the pipeline is registered in `/etc/logserver-probe/pipelines.yml`.
3. Confirm the listener is bound, for a source that sends data: `ss -lnp | grep <port>`.
4. Open **Discover** in the Console and select the index the pipeline writes to.
5. If no documents arrive, work through *Custom Integrations Troubleshooting*.

Related pages

- *Technical Configuration*, the full input, filter, and output reference
- *Custom Integrations*, building a pipeline for a source that is not in the catalog
- *Management Interface*, managing pipelines from the Console
- *Agents*
- *Event Collector*
- *Integrations*

Installation

- *Sizing and Placement*
  - *Count probes by zone, not by volume*
  - *Separate the flow rate from the log rate*
  - *Hardware*
- *Standalone Sensor Deployment*
- *Network Probe Service Installation*
  - *Service Management*
  - *Bundled Services*
  - *Configuration Files*
- *TLS Configuration*
  - *Default Configuration (7.6.1 and later)*
  - *Manual Configuration (7.6.0 and before)*
  - *MasterAgent TLS*
- *Post-installation Verification*

Sizing and Placement

Before installing sensors, decide how many you need and where they sit. Two rules answer both questions.

Count probes by zone, not by volume

A single probe handles a high flow rate on its own, so raw throughput rarely decides the count. What decides it is network segmentation: a probe sees only the traffic on the segment it sits in. Deploy one probe per isolated zone (for example Core, DMZ, and each cloud or branch segment) and let each probe monitor its own segment. Placing a probe inside the zone it watches also keeps captured traffic from crossing segment boundaries, which matters for audited environments. Only the pull from the cluster crosses the firewall.

Separate the flow rate from the log rate

Two figures often get quoted together and then double-counted:

- **Flows per second and NetFlow bandwidth** are the probe's workload.
- **Events per second (EPS)** is the cluster's log-ingest workload, handled by the data nodes, not the probe.

Size the probe from the flow rate and the cluster from the EPS. They are different resources; do not add them together.

Hardware

Size the probe to the production figures for the **ELS Network Node** in *System Requirements*; that table is the single source for component hardware. Local disk on the probe holds buffered and cached data, so size it against how long the probe must retain flows or captures. Full packet capture (PCAP) raises the disk and interface requirements substantially, so size those cases against the expected capture rate and retention.

Standalone Sensor Deployment

The Network Probe can be installed on a dedicated host as a standalone sensor, without the Data Node, ELS Console, or SIEM components. Run the installer in interactive mode with the `-p|--network-probe` flag to preselect Network Probe, then answer **No** to the remaining component prompts:

```
cd install
./install.sh -i -p
```

> **Note**
>
> Do not use `-n|--noninteractive` for a standalone sensor, because `-n` answers "Yes" to every component prompt and installs the full stack. See *Non-interactive Installation Mode* for details.

See *Installation* for installer prerequisites and options.

Network Probe Service Installation

Energy Logserver Network Probe is installed as part of the main installation package. The service is managed via systemd.

Service Management

> **Note**
>
> The Network Probe service requires `license-service` to be running. If `license-service` is not active, the Network Probe will fail to start.

Start Network Probe service:

```
systemctl start logserver-probe
```

Enable Network Probe to start on boot:

```
systemctl enable logserver-probe
```

Check service status:

```
systemctl status logserver-probe
```

Bundled Services

The Network Probe installation includes the following components, each running as a separate systemd service:

Component	Service	Description
Network Probe	`logserver-probe`	Pipeline engine for log ingestion and processing
Kafka	`kafka.service`	Message queue for data transport between probe and Data Node
Zookeeper	`zookeeper.service`	Coordination service required by Kafka
Zeek	`zeek`	Network traffic analyzer (NDR)
Suricata	`suricata.service`	Network IDS/IPS engine
pmacct (NetFlow)	`nfacctd@instance1`	NetFlow/IPFIX collector
pmacct (sFlow)	`sfacctd@instance1`	sFlow collector

Note

Only `logserver-probe` runs after installation. The installer leaves `kafka`, `zookeeper`, `zeek`, and `suricata` disabled and masked, so each one has to be unmasked before its first start. Zeek and Suricata are installed only when you answer **Yes** to the prompt for optional Network Probe components; `-n|--noninteractive` answers Yes for you.

Enable only the components relevant to your use case. Services can be managed from the GUI via the *Management Interface*.

Warning

Zeek and Suricata capture from a network interface. Bind them to the capture interface before starting:

- **Zeek:** `/opt/zeek/etc/node.cfg`
- **Suricata:** `/etc/suricata/suricata.yaml`

The pmacct collectors do not capture from an interface. `nfacctd` and `sfacctd` bind a UDP listen socket for flows that network devices export to the probe (2055 for NetFlow/IPFIX, 6343 for sFlow), configured in `/etc/pmacct/nfacctd-instance1.conf` and `/etc/pmacct/sfacctd-instance1.conf`. To account for a host's own traffic by capturing from its interface, run the `pmacctd` generator instead, described in *pmacct Local Traffic Monitoring*.

For the full Suricata setup, from unmasking the service to seeing alerts in the GUI, see *Suricata IDS*. For Zeek, from unmasking the service to reading events in Discover, see *Zeek*.

Setting Up the NetFlow and sFlow Collectors

The installer ships the pmacct collector templates and setup scripts in `/opt/network-probe/pmacct/`, but it creates no `nfacctd` or `sfacctd` instance. The `nfacctd@instance1` and `sfacctd@instance1` services from the table above exist only after you run the scripts. The installer already opens UDP 2055 and 6343 in the firewall.

1. Unmask and start Kafka and Zookeeper, because the collectors publish flows to the Kafka `netflow` topic:

```
systemctl unmask zookeeper kafka
systemctl enable --now zookeeper kafka
```

2. Create the instance. Run the script from its own directory, because it reads the templates from the current directory:

```
cd /opt/network-probe/pmacct
./create-nfacctd-instance.sh instance1
```

The script writes `/etc/pmacct/nfacctd-instance1.conf` and `/etc/pmacct/librdkafka.conf`, installs the `nfacctd@instance1` unit, and enables and starts it. For sFlow, run `./create-sfacctd-instance.sh instance1` instead.

3. Add `peer_src_ip` to the `aggregate` line in `/etc/pmacct/nfacctd-instance1.conf` and run `systemctl restart nfacctd@instance1`. Without it, `netflow.exporter` contains `%{peer_ip_src}` (8.1.0 template). For an instance that already sends flows, see *Netflow: missing exporter field*.
4. In the GUI, open **Network Probe > Pipelines Management**, click the edit icon in the `network-stream` row, select the probe, and click **Submit**. The `network-stream` pipeline reads the `netflow` topic and writes flows to `stream-netflow-pmacct-*`.
5. Confirm the collector is listening:

```
ss -ulnp | grep 2055
```

Every device that exports NetFlow or sFlow to the probe is collected from this point on, so check which devices already send flows to the probe address before you start the instance.

Configuration Files

Network Probe configuration files are located in:

- Main configuration: `/etc/logserver-probe/logserver-probe.yml`
- Pipeline configurations: `/etc/logserver-probe/pipelines.yml` and `/etc/logserver-probe/pipelines.d/`
- Input/Filter/Output configs: `/etc/logserver-probe/conf.d/`
- JVM options: `/etc/logserver-probe/jvm.options`

TLS Configuration

Default Configuration (7.6.1 and later)

Since version 7.6.1, there is a default configuration for enabled algorithms at `/etc/logserver-probe/java.properties`.

Make sure there is the following definition in `/etc/logserver-probe/jvm.options`:

```
-Djava.security.properties=/etc/logserver-probe/java.properties
```

You can edit this file and after restarting Network Probe, changes will be applied.

Manual Configuration (7.6.0 and before)

For versions 7.6.0 and earlier, follow these steps:

1. Create file `/etc/logserver-probe/java.properties` with content:

```
security.useSystemPropertiesFile=false
jdk.tls.ephemeralDHKeySize=2048
jdk.certpath.disabledAlgorithms=MD2, SHA1, MD5, DSA, RSA keySize < 2048
```

(continues on next page)

(continued from previous page)

```
  jdk.tls.disabledAlgorithms=DH keySize < 2048, TLSv1.1, TLSv1, SSLv3, SSLv2,
↪ TLS_RSA_WITH_AES_256_CBC_SHA256, TLS_RSA_WITH_AES_256_CBC_SHA, TLS_RSA_
↪WITH_AES_128_CBC_SHA256, TLS_RSA_WITH_AES_128_CBC_SHA, TLS_RSA_WITH_AES_256_
↪GCM_SHA384, TLS_RSA_WITH_AES_128_GCM_SHA256, DHE_DSS, RSA_EXPORT, DHE_DSS_
↪EXPORT, DHE_RSA_EXPORT, DH_DSS_EXPORT, DH_RSA_EXPORT, DH_anon, ECDH_anon,␣
↪DH_RSA, DH_DSS, ECDH, 3DES_EDE_CBC, DES_CBC, RC4_40, RC4_128, DES40_CBC,␣
↪RC2, HmacMD5, TLS_DHE_RSA_WITH_AES_128_CBC_SHA, TLS_DHE_RSA_WITH_AES_256_
↪CBC_SHA
  jdk.tls.legacyAlgorithms=
```

2. Add to `/etc/logserver-probe/jvm.options`:

```
  -Djava.security.properties=/etc/logserver-probe/java.properties
```

3. After restarting Network Probe service, changes will be applied.

MasterAgent TLS

Since version 7.6.1, MasterAgent has predefined strong ciphers already set. Updating is recommended.

For MasterAgent configuration, see *Agents*.

Post-installation Verification

Verify Network Probe is running:

```
systemctl status logserver-probe
```

Check Network Probe logs:

```
journalctl -u logserver-probe -f
```

Verify Network Probe is listening on configured ports:

```
ss -tulnp | grep java
```

Test pipeline configuration:

```
/usr/share/logserver-probe/bin/logstash --config.test_and_exit
```

Management Interface

The **Network Probe** plugin has two tabs, `Pipelines Management` and `Network Probes`, described below. The banner at the top shows how many probes are in use out of the number allowed by the license.

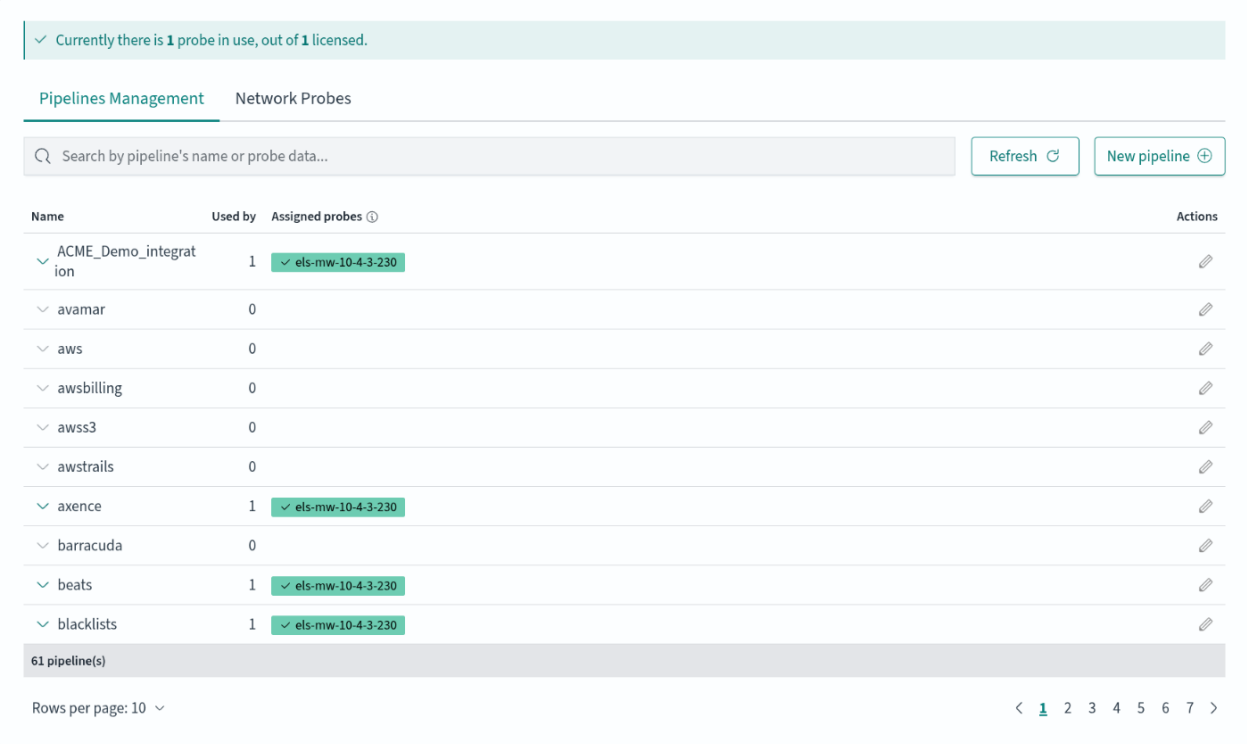
> **Note**
>
> The Management Interface communicates with each probe via the license-service API using TLS. The SSL certificates are located at `/opt/license-service/ssl/` and are configured automatically during license-service installation.

- *Pipelines Management*
  - *New Pipeline Creator*
  - *Pipeline Details*
  - *Manage pipeline*
- *Network Probes*
  - *Services Section*
  - *Pipelines Section*
  - *Files Section*

Pipelines Management

This tab lets you manage pipelines on all probes at once. Each row describes one pipeline available on the probes:

- `Name`: name of the pipeline
- `Used by`: number of probes that run the pipeline
- `Assigned probes`: probes on which the pipeline is enabled

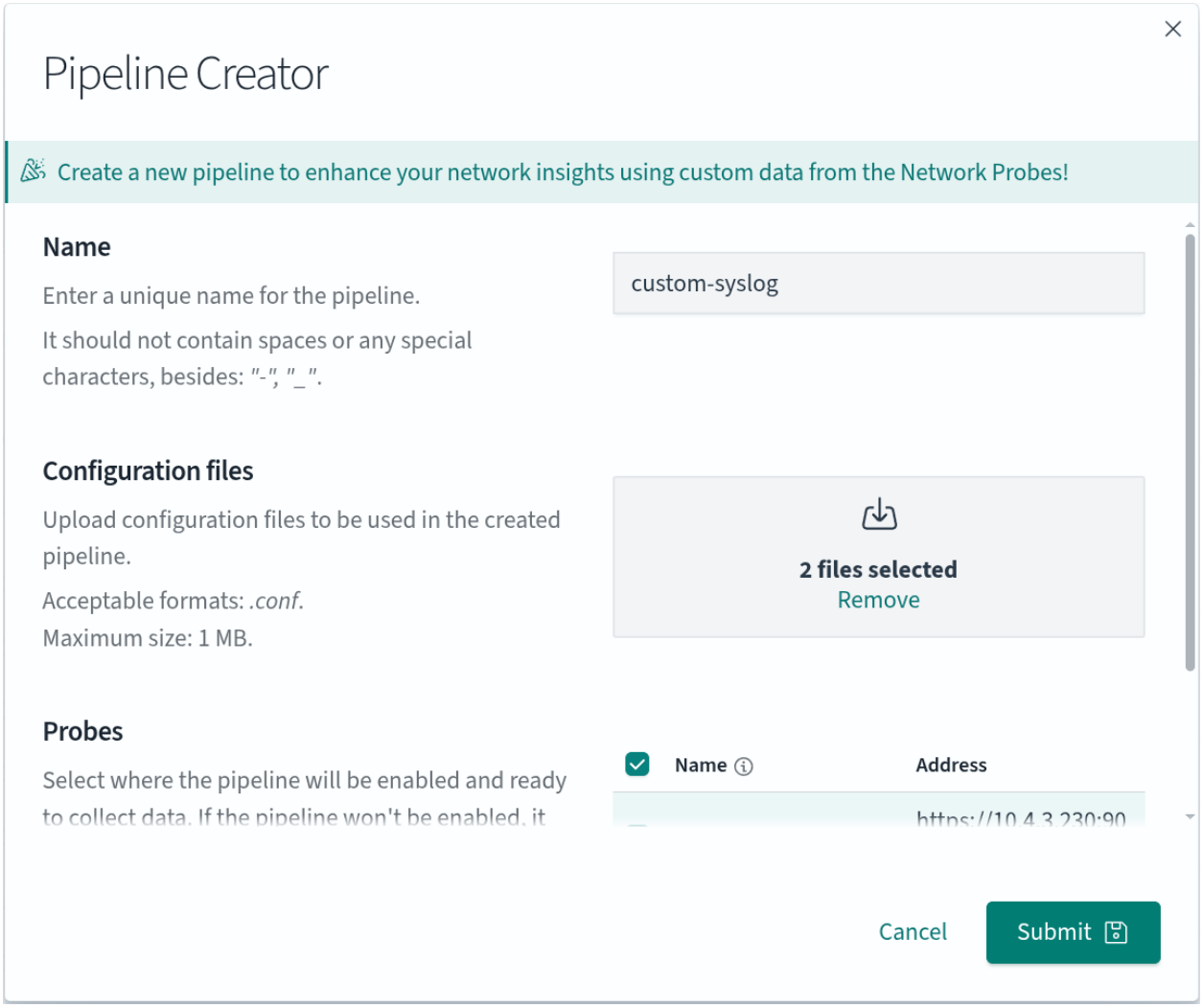


You can search the list by pipeline name or by probe data, for example the probe hostname. Use `Refresh` to reload the list and `New pipeline` to open the Pipeline Creator.

New Pipeline Creator

The **Pipeline Creator** lets you create a pipeline from the GUI.

1. In the `Pipelines Management` tab, click `New pipeline`.
2. Enter a unique name for the pipeline. The name cannot contain spaces or special characters other than – and _.
3. Select or drag and drop the configuration files. Each file must have the `.conf` extension and be no larger than 1 MB. You can add more than one file.
4. Select the probes on which the pipeline will be enabled. Only active probes can be selected. The pipeline is installed on every Network Probe, but it collects data only on the selected ones.
5. Click `Submit`.



The creator validates the configuration files before it saves anything. If a file cannot be parsed, a notification `Unable to create pipeline "<name>"` appears and nothing is written to the probes. Click `See the full error` to read the parser message with the file path, line, and column.

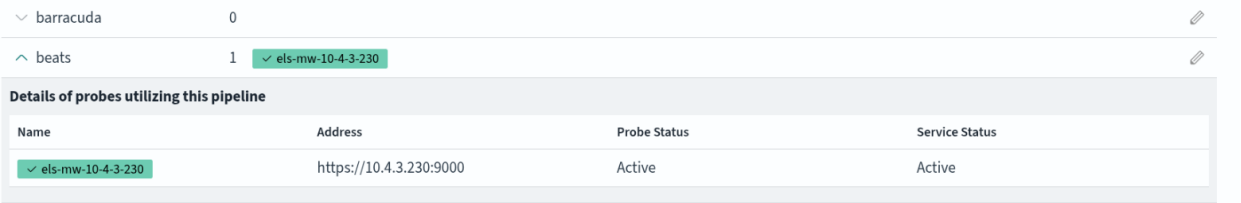
On a probe where the pipeline is not enabled, the creator saves the configuration files in `/etc/logserver-probe/conf.d/<name>/` and the definition file as `/etc/logserver-probe/pipelines.d/<name>.yml.off`.

Pipeline Details

A green label with a check icon means that the probe is working and the pipeline is enabled on it. A gray label with a cross means that the pipeline should be working, but the probe does not respond.

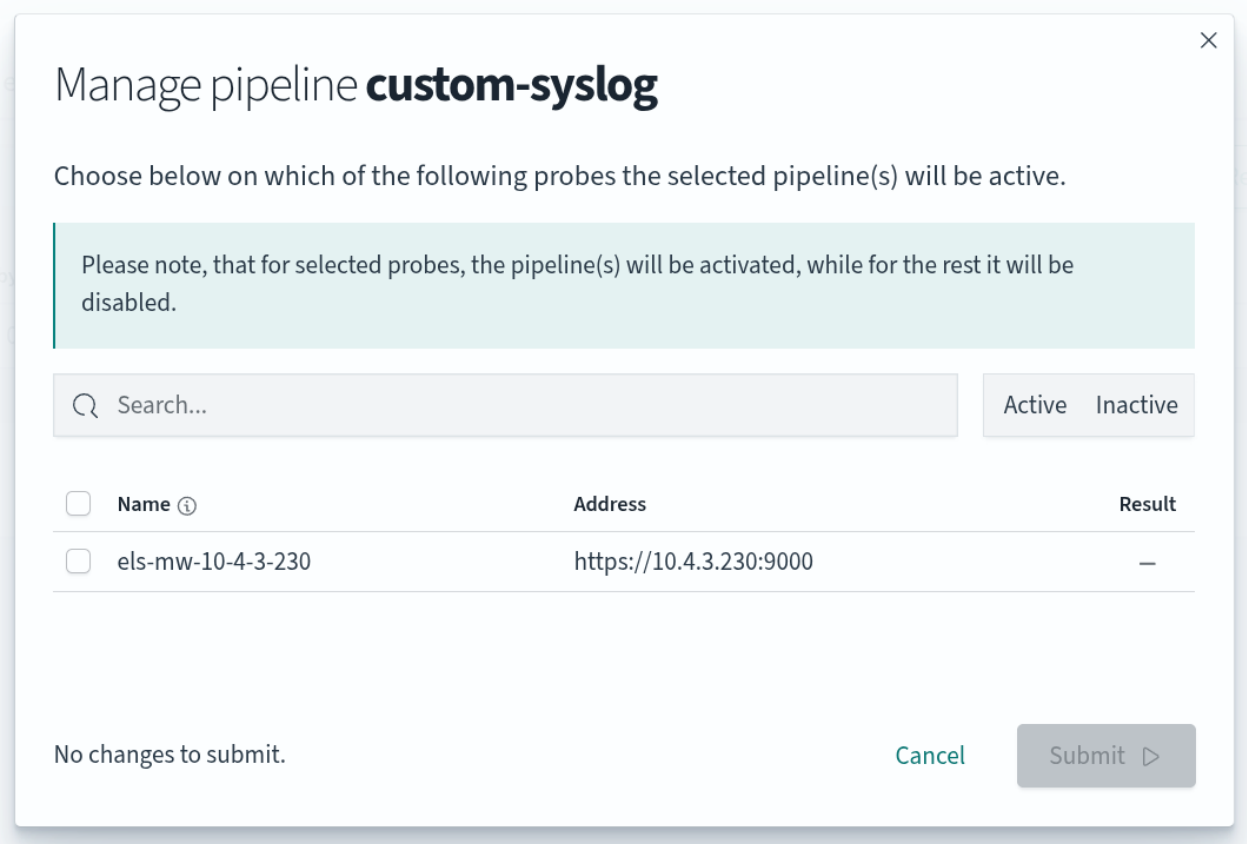
Click a probe label in the `Assigned probes` column to open the pipeline statuses on that probe.

Click the arrow on the left to expand the details of a pipeline. The details list the name, address, probe status, and service status of each probe that uses the pipeline.



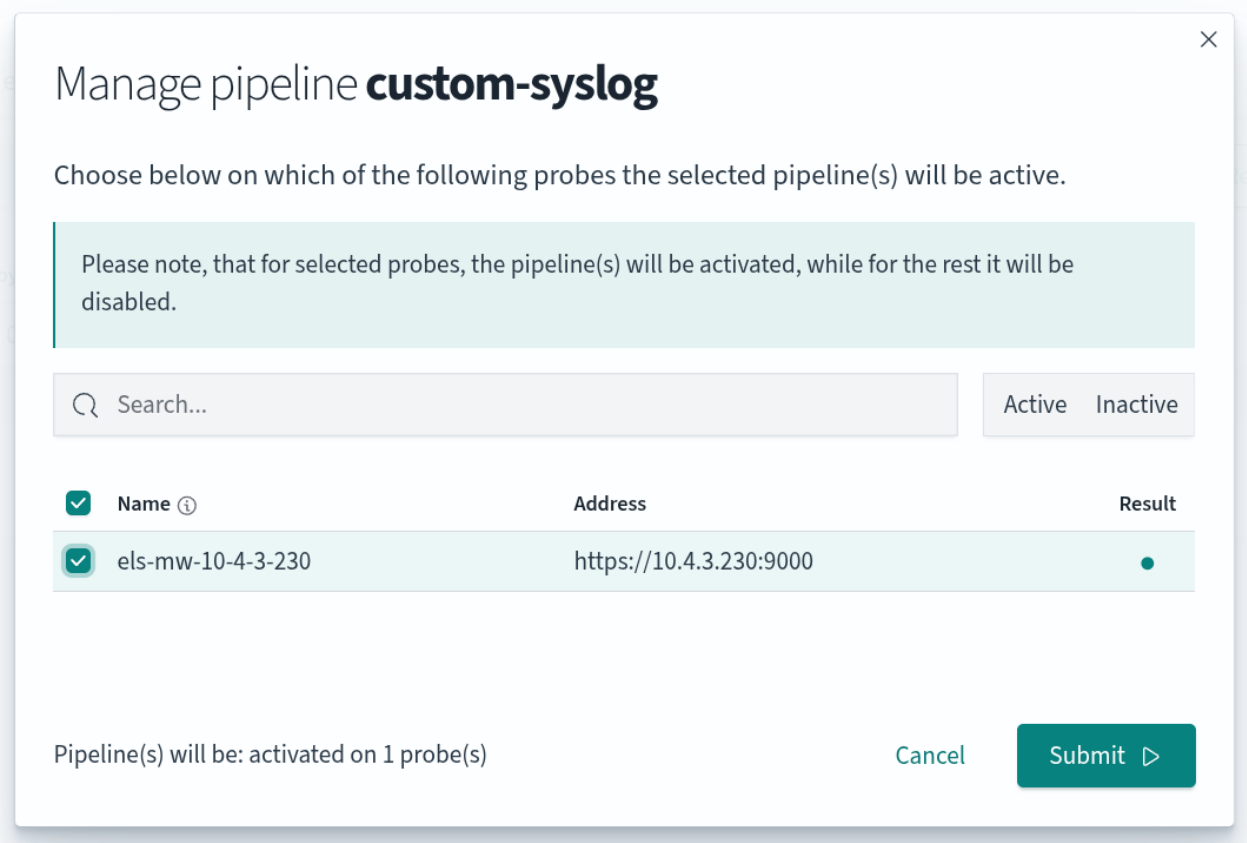
Manage pipeline

Click the pencil icon on the right to open the **Manage pipeline** window.

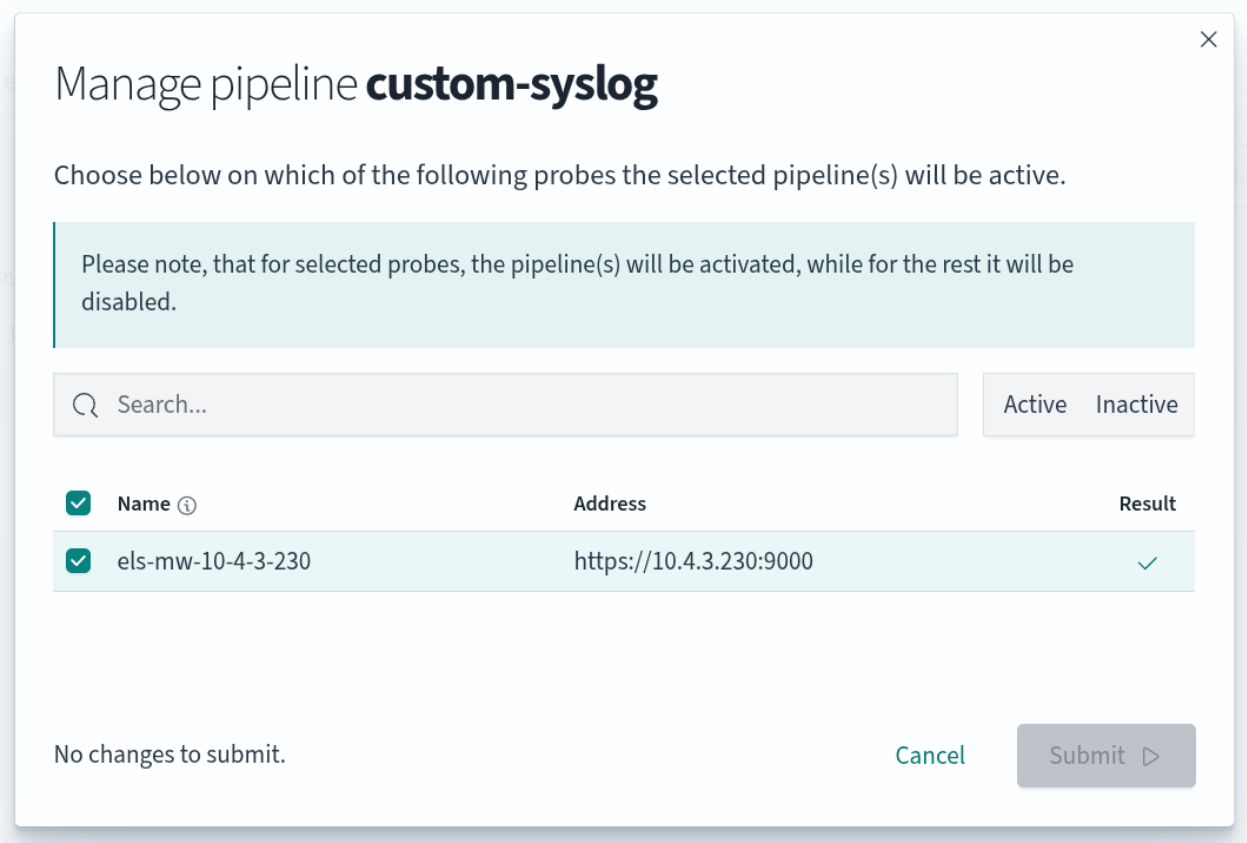


The list shows all probes. A selected row means that the pipeline is enabled on that probe. After you submit, the pipeline is enabled on the selected probes and disabled on the rest. A probe that is not running cannot be selected. Use the search field and the `Active`/`Inactive` filters to find a probe in a long list.

When you change the selection, the line in the lower-left corner summarizes the operations, for example `Pipeline(s) will be: activated on 1 probe(s).` Without changes it reads `No changes to submit.`



Click `Submit`. The `Result` column shows the progress for each probe and then a green icon when the operation succeeded or a red icon when it failed. Hover over a red icon to read the error message.

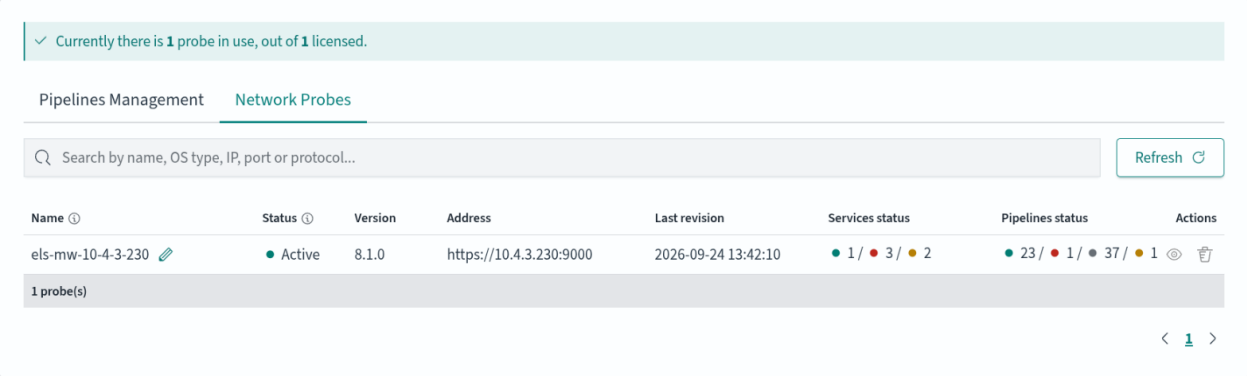


Click `Close` and refresh the list to see the change.

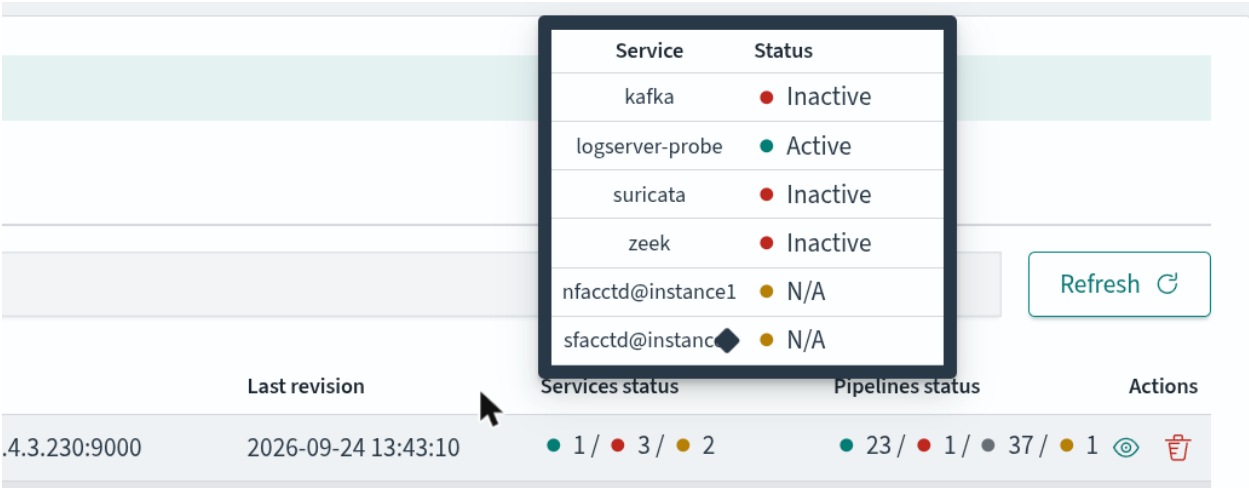
Network Probes

This tab lists all registered probes with the following columns:

- `Name`: probe hostname or a custom name. Click the pencil icon to set a custom name.
- `Status`: probe status during the last 60 seconds
- `Version`: Network Probe version installed on the probe. A warning icon means that it does not match the Energy Logserver version.
- `Address`: license-service address of the probe
- `Last revision`: last time the probe reported its status. A warning icon means that the probe clock is ahead, so its status may be wrong.
- `Services status` and `Pipelines status`: number of services and pipelines in each state
- `Actions`: `Show details` (eye icon) and `Delete` (trash icon)



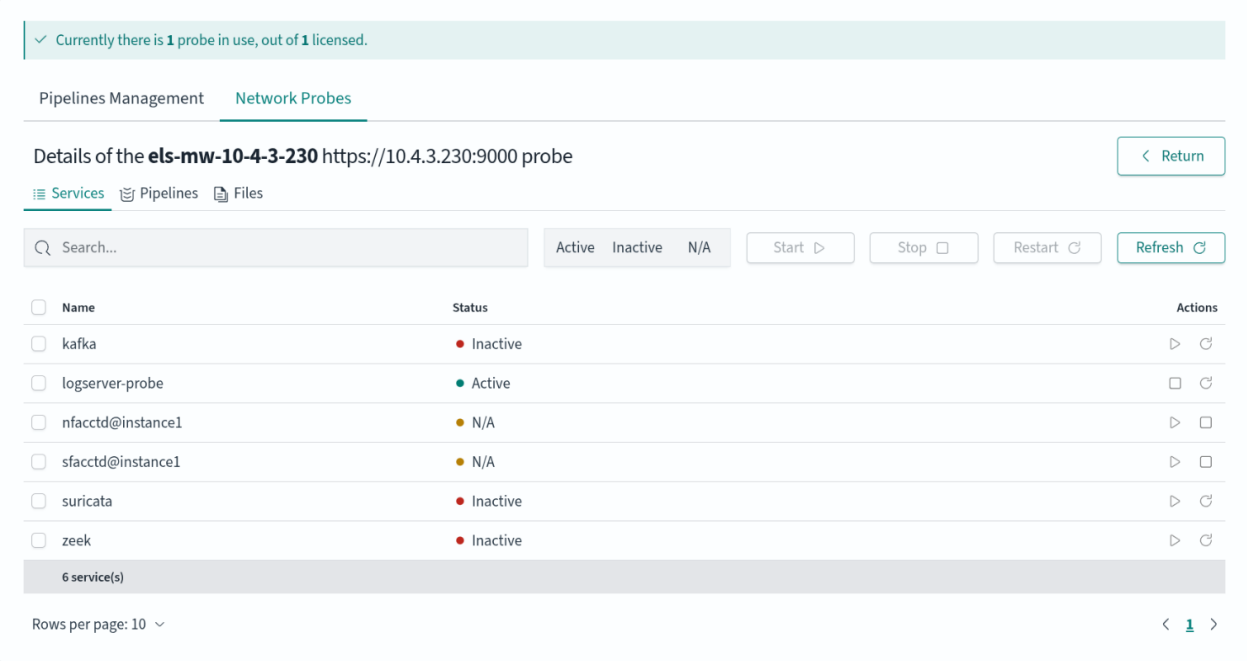
You can search the list by name, OS type, IP, port, or protocol. Hover over the `Services status` or `Pipelines status` column to see a table with the status of each service or pipeline.



Click the eye icon to open the probe details with the `Services`, `Pipelines`, and `Files` sections. `Delete` removes the probe from the GUI. If the registration of a probe failed, delete it so that the probe can register again.

Services Section

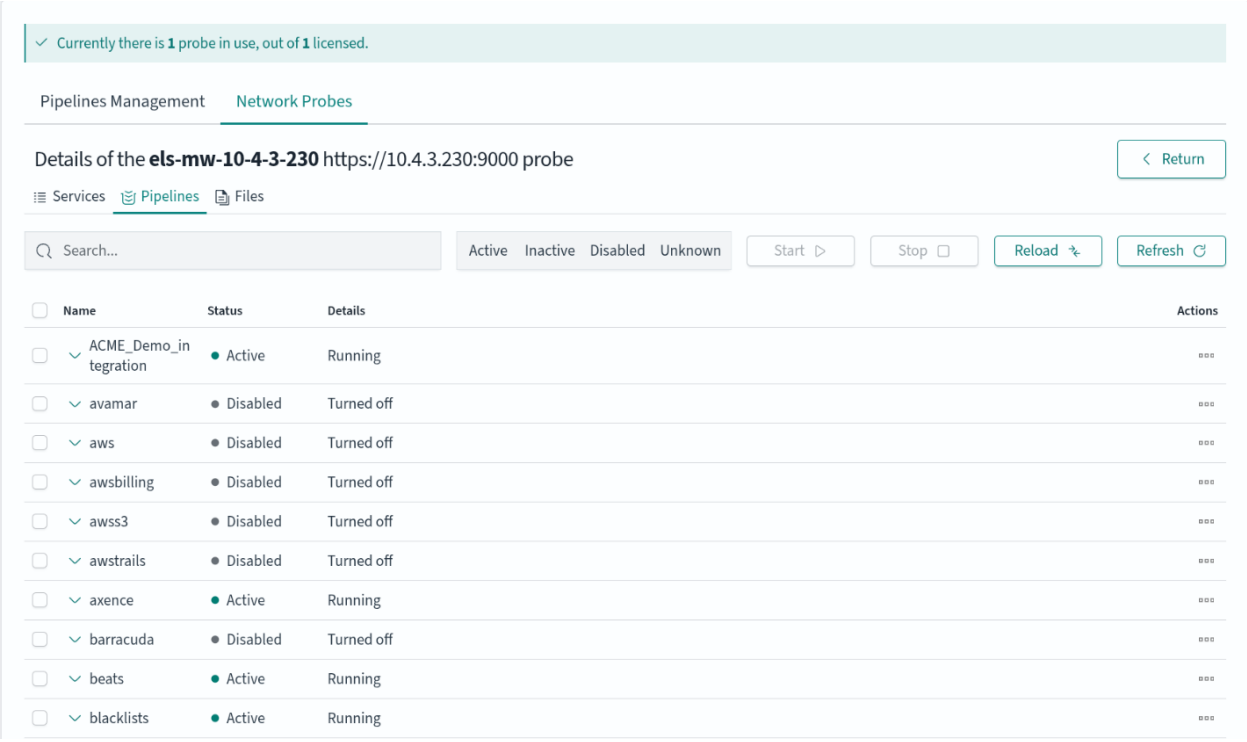
This section lists the services managed on the probe: `logserver-probe`, `kafka`, `suricata`, `zeek`, `nfacctd@instance1`, and `sfacctd@instance1`. You can see their status and start, stop, or restart them. A service that is not installed on the probe shows the status `N/A`.



Use the buttons in the `Actions` column to manage one service, or select several rows and use the `Start`, `Stop`, or `Restart` buttons above the list. The `Active`, `Inactive`, and `N/A` filters limit the list to services in that state.

Pipelines Section

This section lets you manage the pipelines on the probe. You can check their status and statistics, enable or disable them, delete them, and reload the pipeline configuration.

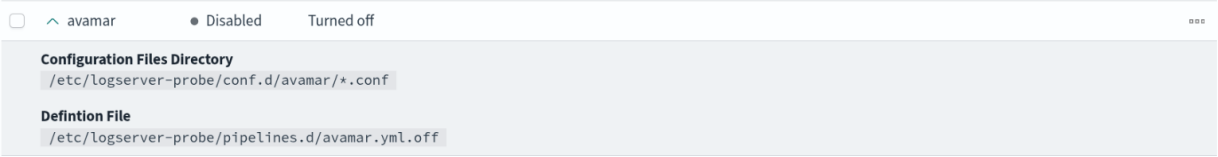


Pipeline configuration

A pipeline consists of:

- `definition file`: pipeline name, path to the configuration files, and optional pipeline settings. One file can define several pipelines. The default path is `/etc/logserver-probe/pipelines.d/<pipeline_name>.yml`.
- `configuration files`: input, filter, and output plugin configuration. The default path is `/etc/logserver-probe/conf.d/<pipeline_name>/*.conf`, and the definition file can point to another one.

Click the arrow on the left to see both paths in the pipeline details.



Pipeline status

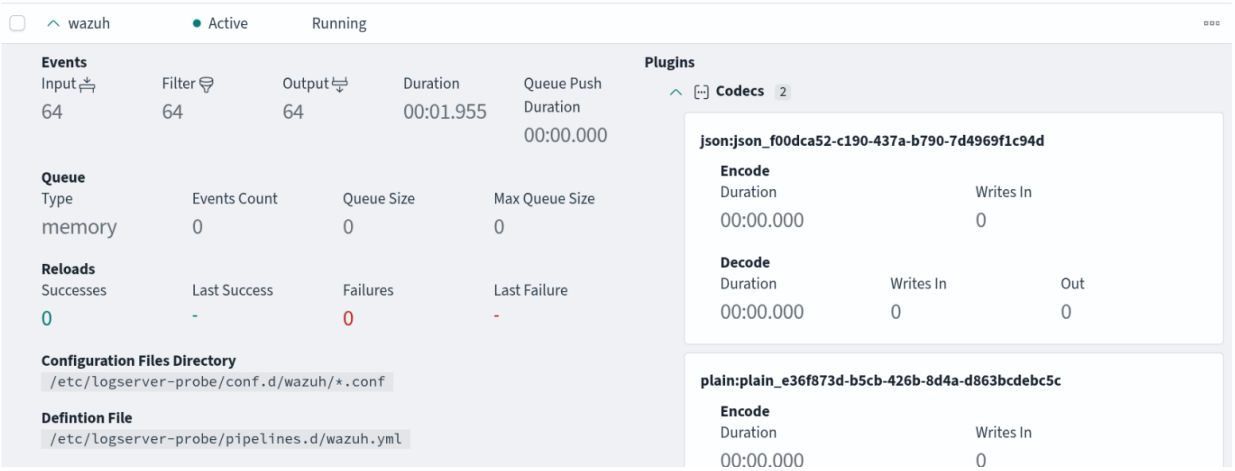
A pipeline can have one of the following statuses:

- **Active**: the pipeline is enabled and running, and its runtime statistics are available.
- **Active** *with warning*: the pipeline is enabled and running, but a reload error has occurred.
- **Inactive**: the pipeline is enabled, but it does not run and its statistics cannot be found.
- **Disabled**: the pipeline is disabled and does not run.
- **Unknown**: the pipeline has failed, for example because of a wrong configuration, corrupted files, or a stopped service.

The `Details` column describes the current status and can point to the cause of an error.

Click the arrow on the left to view more details. Every pipeline shows its configuration details. An *active* pipeline also shows runtime statistics:

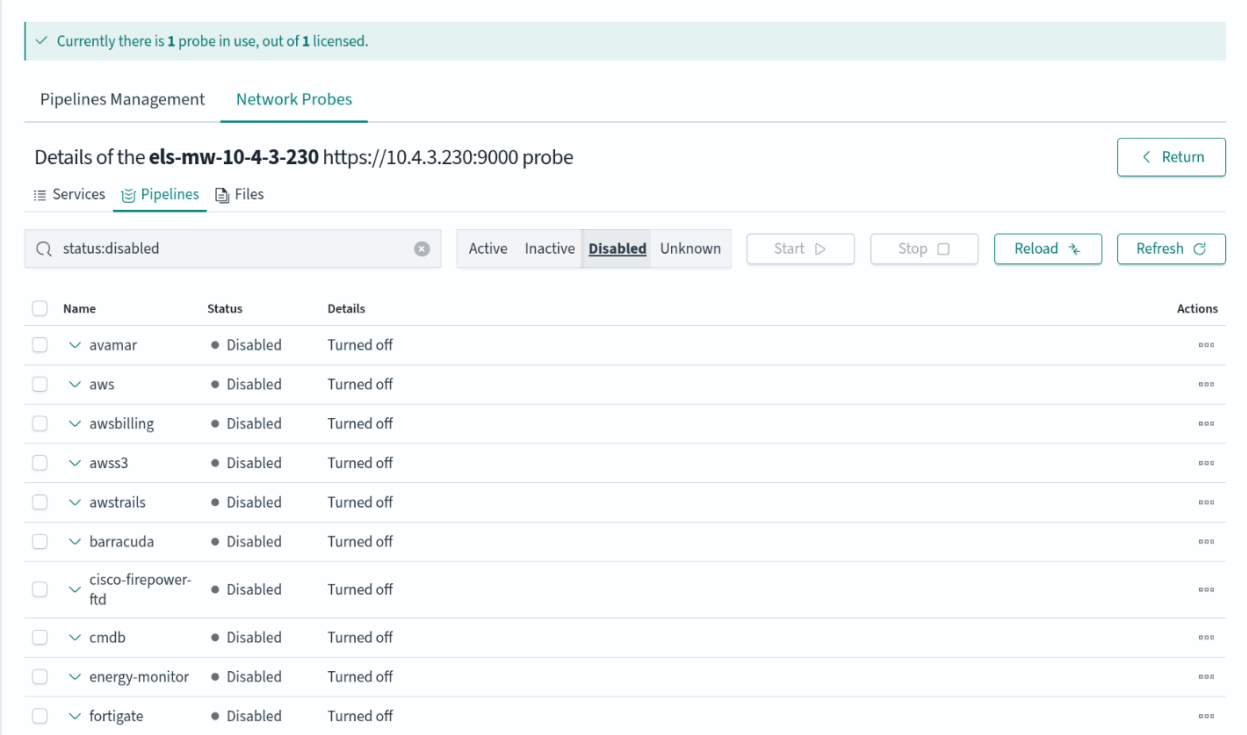
- input, filter, and output events
- queue details
- plugins in use (input, filter, output, and codecs)



If a reload error has occurred, the details show it as well.

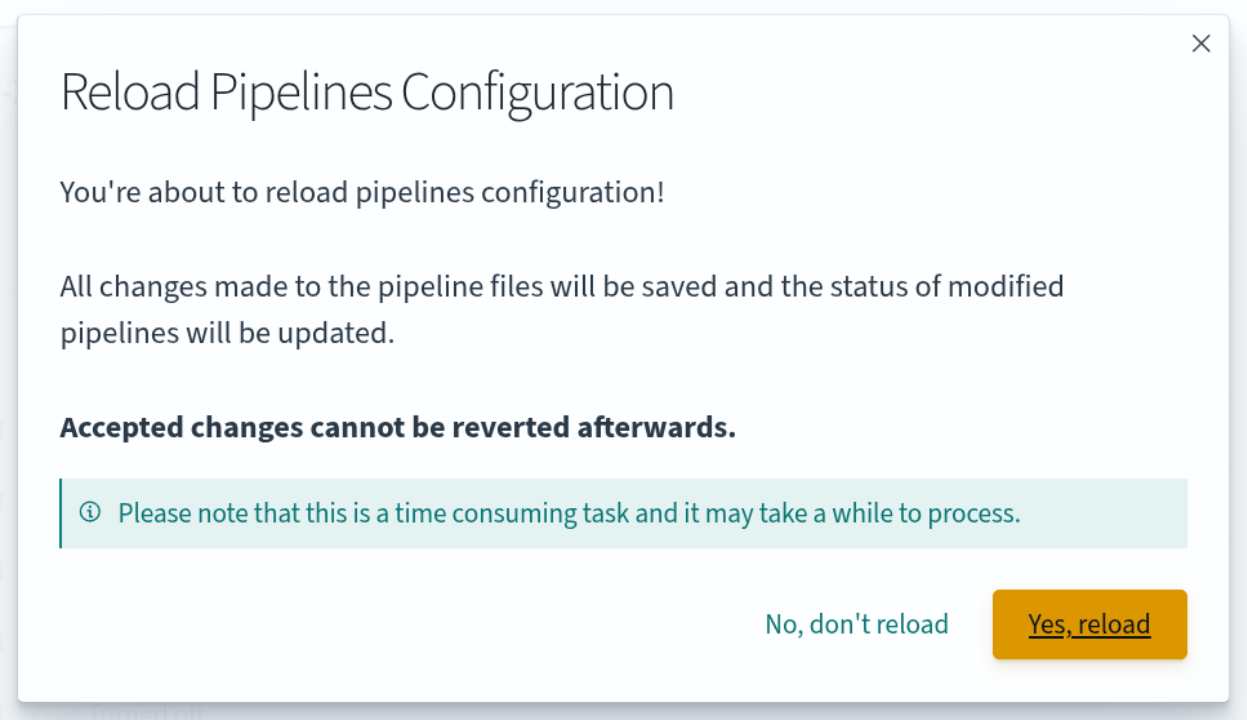
Filtering by status

Use the `Active`, `Inactive`, `Disabled`, and `Unknown` buttons next to the search field to show only pipelines with that status. The example below shows only disabled pipelines:



Reload pipelines configuration

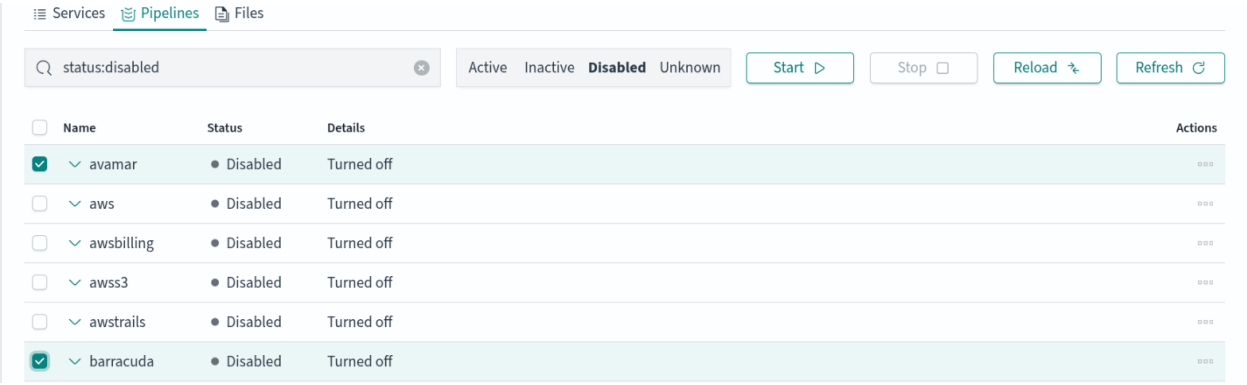
Click `Reload` to apply changes in the Network Probe configuration files without restarting the `logserver-probe` service. Confirm the operation in the window that opens.



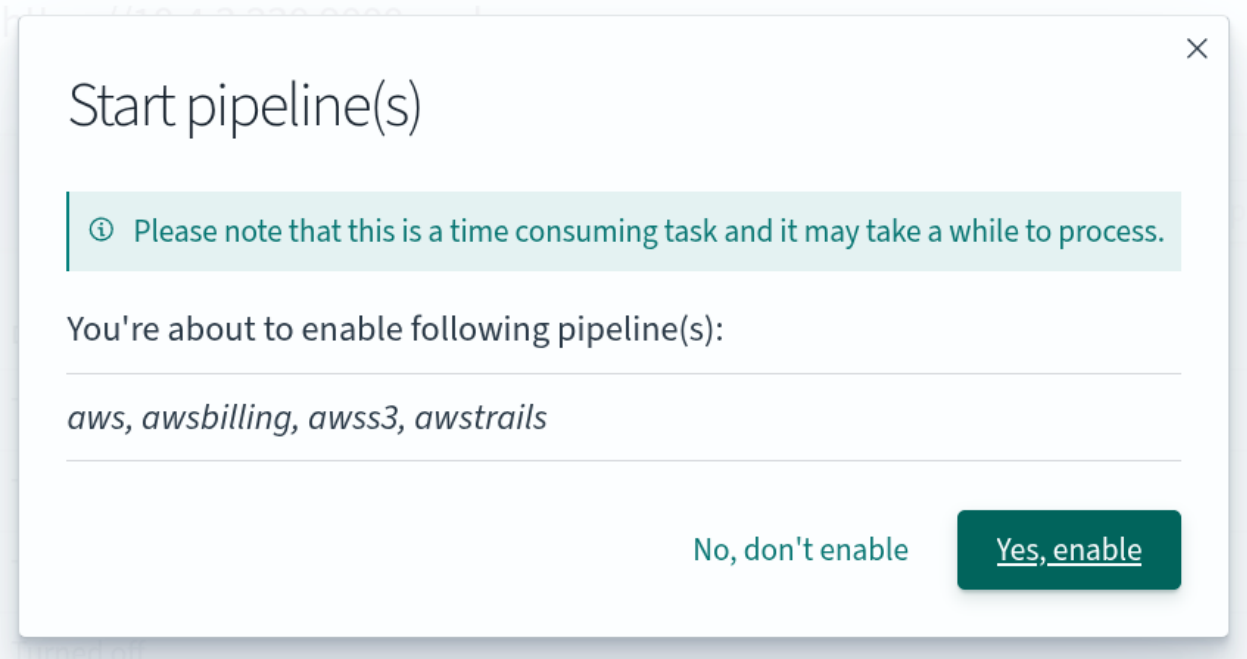
Enabling or disabling a pipeline reloads the configuration automatically.

Enable/disable pipeline

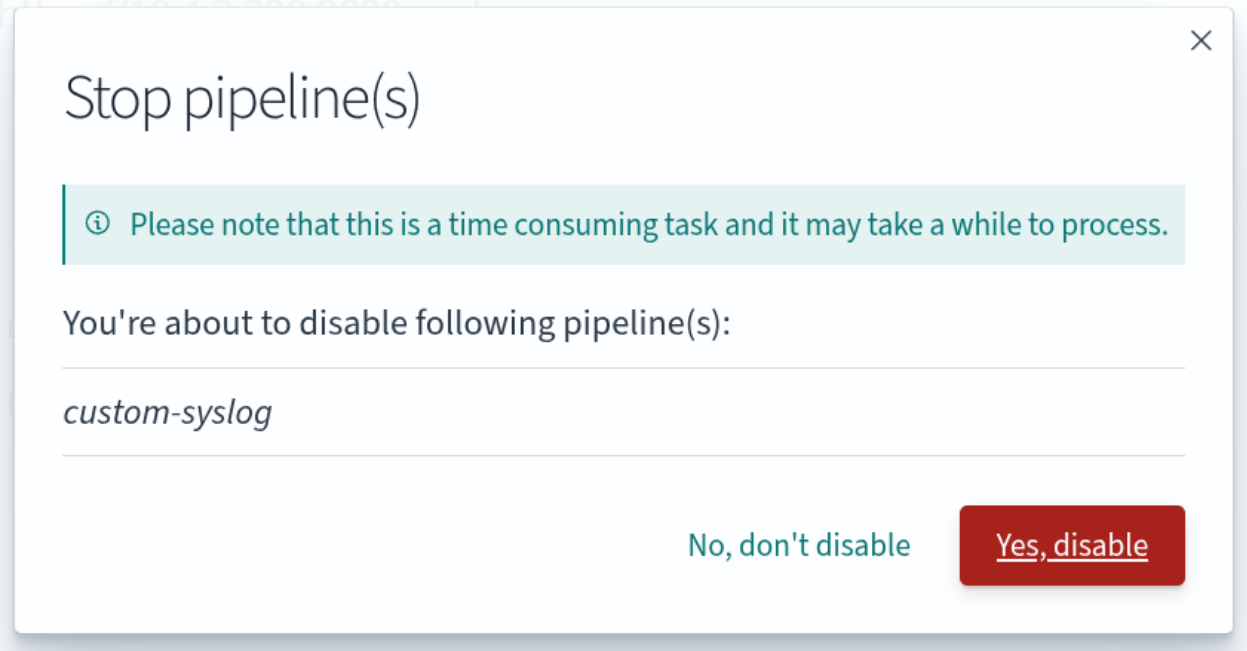
Open the `Actions` menu of a pipeline and choose `Start` or `Stop`. To change several pipelines at once, select their rows and use the `Start` or `Stop` button above the list. All selected pipelines must have the same state.



If one definition file defines several pipelines, the action applies to all of them, and the confirmation window lists them. Below is the message for one of the 4 pipelines defined in `aws.yml`:

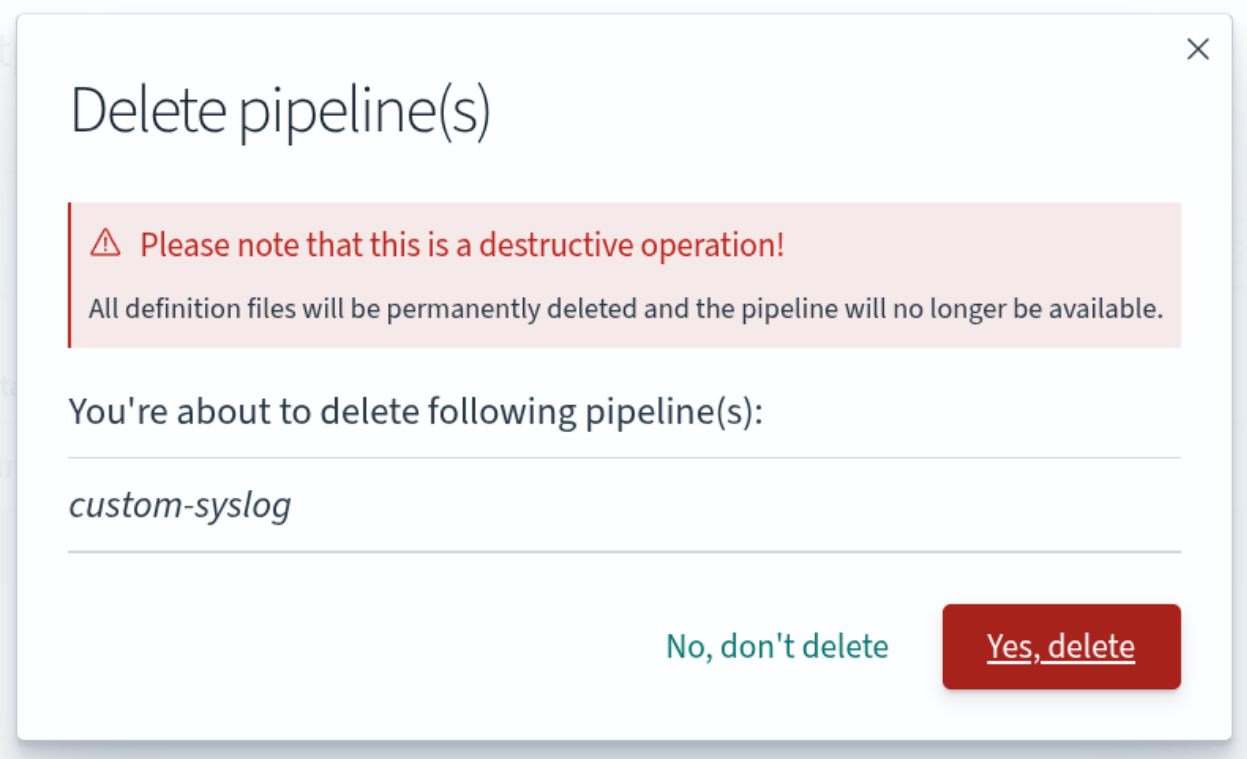


For a pipeline with its own definition file, the message looks like this:



Delete pipeline

Choose `Delete` in the `Actions` menu to remove a pipeline and all its files from the probe. The operation deletes the configuration files, the definition file, and the pipeline entry in `pipelines.yml`, and stops the pipeline. It cannot be undone.

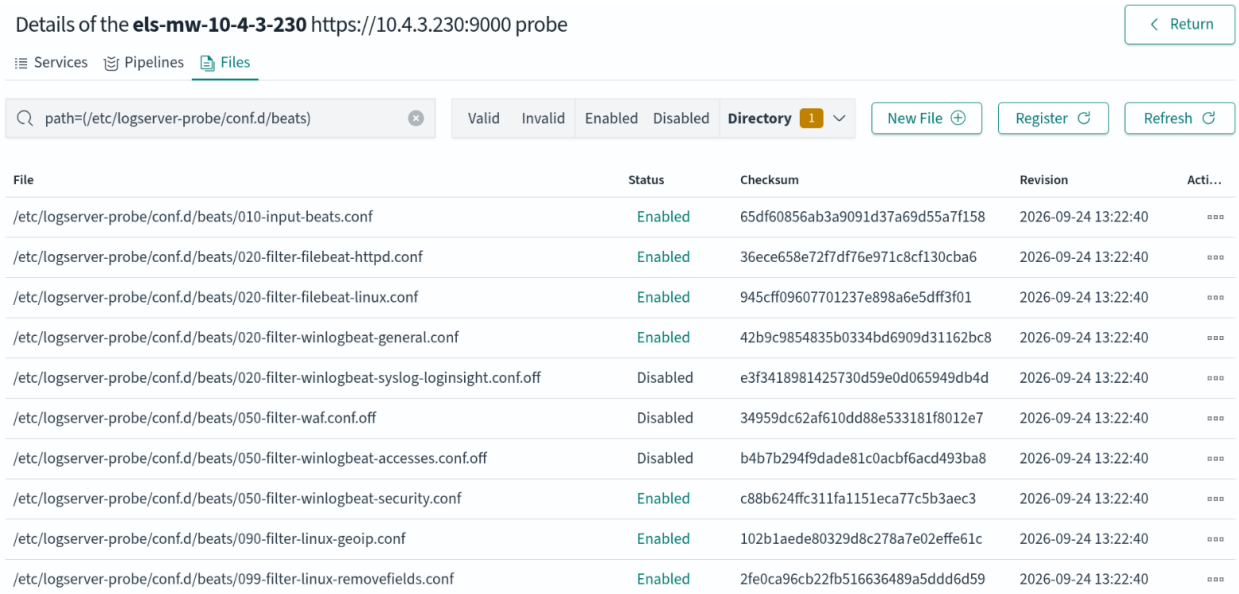


The empty directory `/etc/logserver-probe/conf.d/<pipeline_name>/` stays on the probe. The `logserver-probe` service keeps reporting the deleted pipeline, so the list can still show it with the status `Unknown`, also after `Reload`.

The `Files` action opens the `Files` section with the files of the pipeline.

Files Section

This section lists the files managed by the probe. You can create, edit, enable, disable, and delete them, and check their validation results.



The table shows:

- `File`: full path of the file
- `Status`: whether the pipeline uses the file. Configuration files in `/etc/logserver-probe/conf.d/` are either `Enabled` (`.conf`) or `Disabled` (`.conf.off`). Click the status to change it.
- `Checksum`: checksum of the file content
- `Revision`: date of the latest registration of the file

A yellow warning icon next to a file marks a parsing error or an outdated revision, described below.

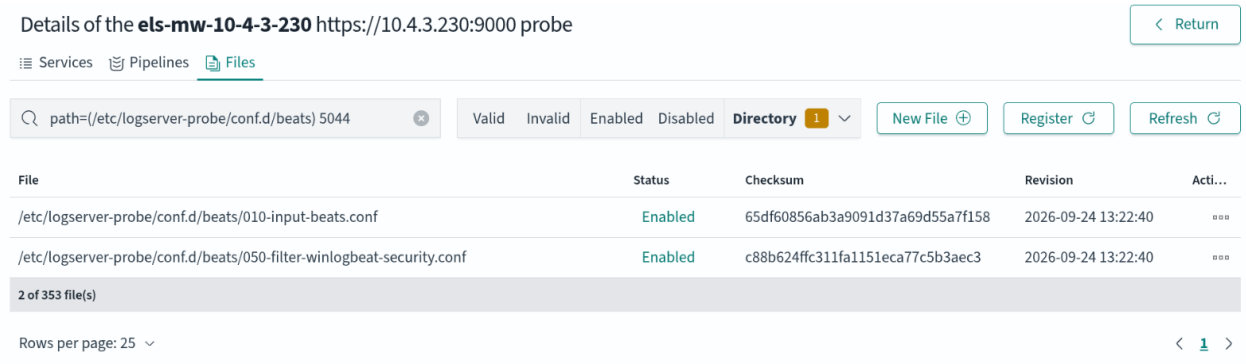
The probe manages the directories and file extensions set in the `files_management` section of `/opt/license-service/license-service.conf` on the probe. By default these are `/etc/logserver-probe/`, `/etc/logserver-probe/conf.d/`, `/etc/logserver-probe/pipelines.d/`, `/opt/zeek/etc/`, and `/etc/suricata/`, with the extensions `yml`, `yaml`, `conf`, `cfg`, `config`, and `off`.

Filtering and searching

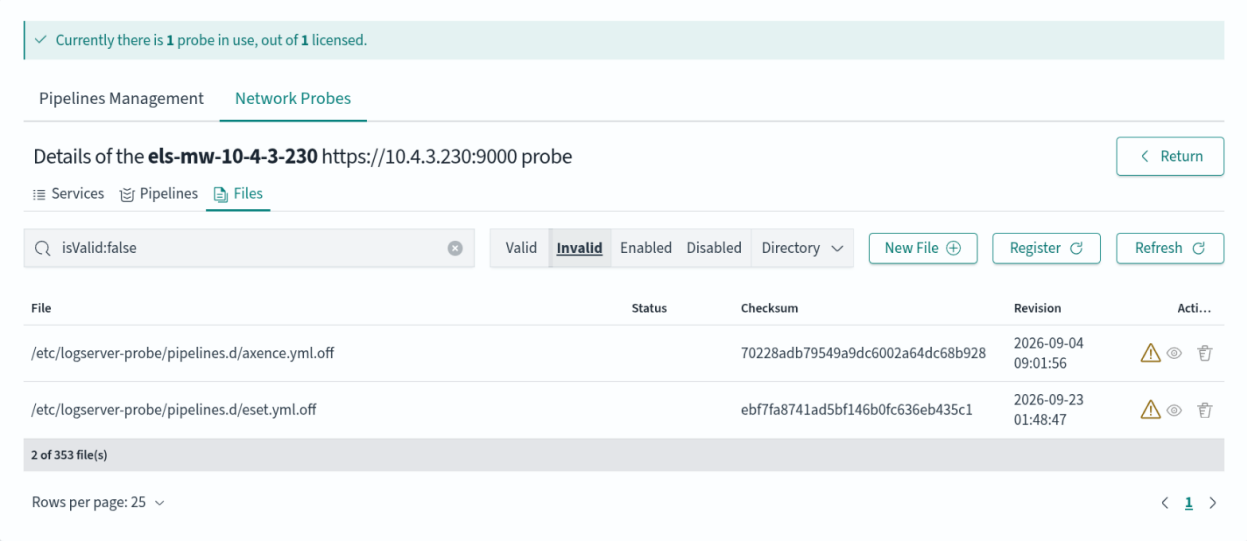
You can filter files by:

- `Valid`/`Invalid`: parsing result and whether the revision is current
- `Enabled`/`Disabled`: whether the pipeline uses the file
- `Directory`: the directory of the file

You can also search by a fragment of the path, the checksum, or the file content. The example below uses the directory filter and finds the files in `conf.d/beats` that contain `5044`:

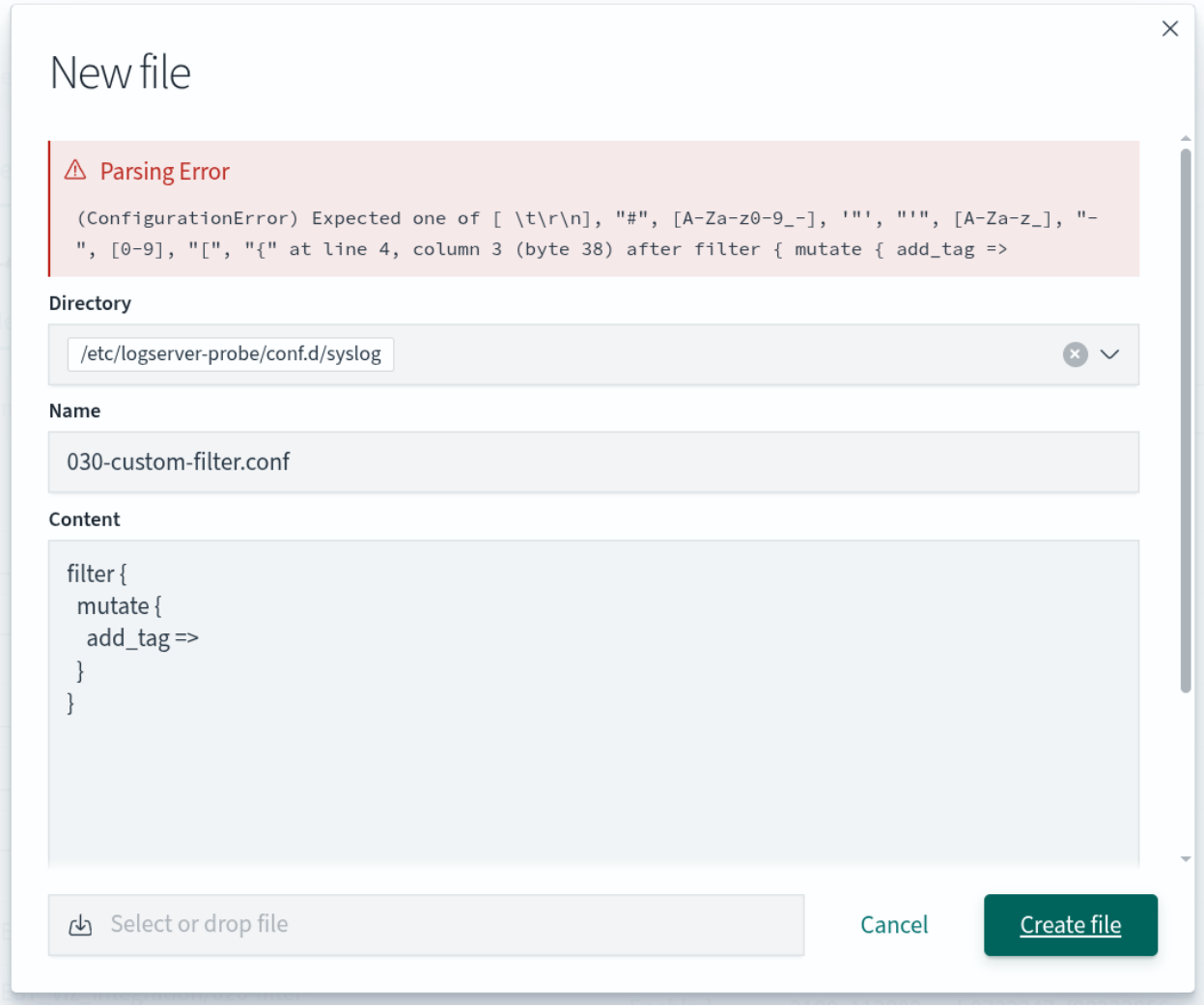


The next example uses the `Invalid` filter. The table shows only files with a parsing error or an outdated revision.



Create file

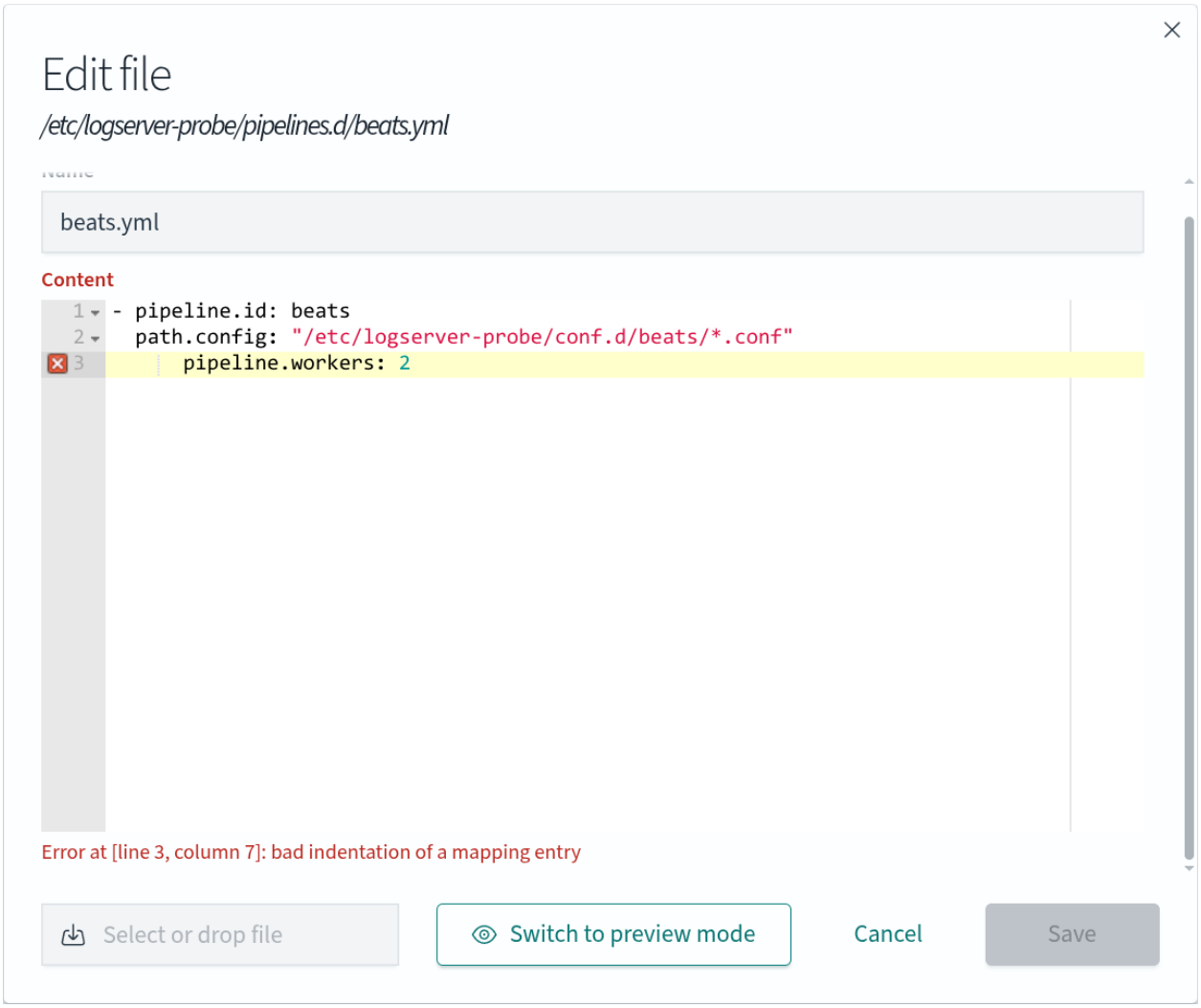
Click `New File`. Select the directory, enter the file name with its extension, and type the content, or upload an existing file with the field in the lower-left corner. You can create files only in the managed directories and only with an allowed extension. A `.conf` file is validated when you save it, and a parsing error blocks the save.



Update file

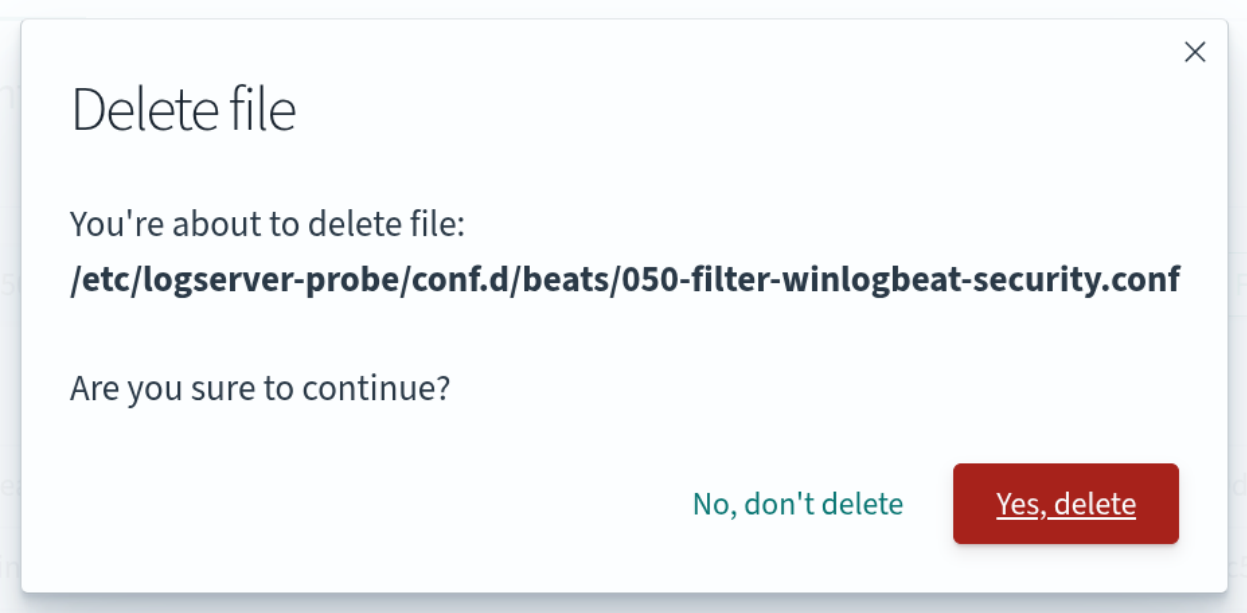
Choose `Show or edit` in the `Actions` menu, or click the eye icon for files without a status. The file opens in preview mode. Click `Switch to edit mode` to change the name or the content, or upload a new version of the file.

When you edit a `.yml` or `.yaml` file, the editor checks the syntax and marks the errors. `Save` stays disabled until you correct them.



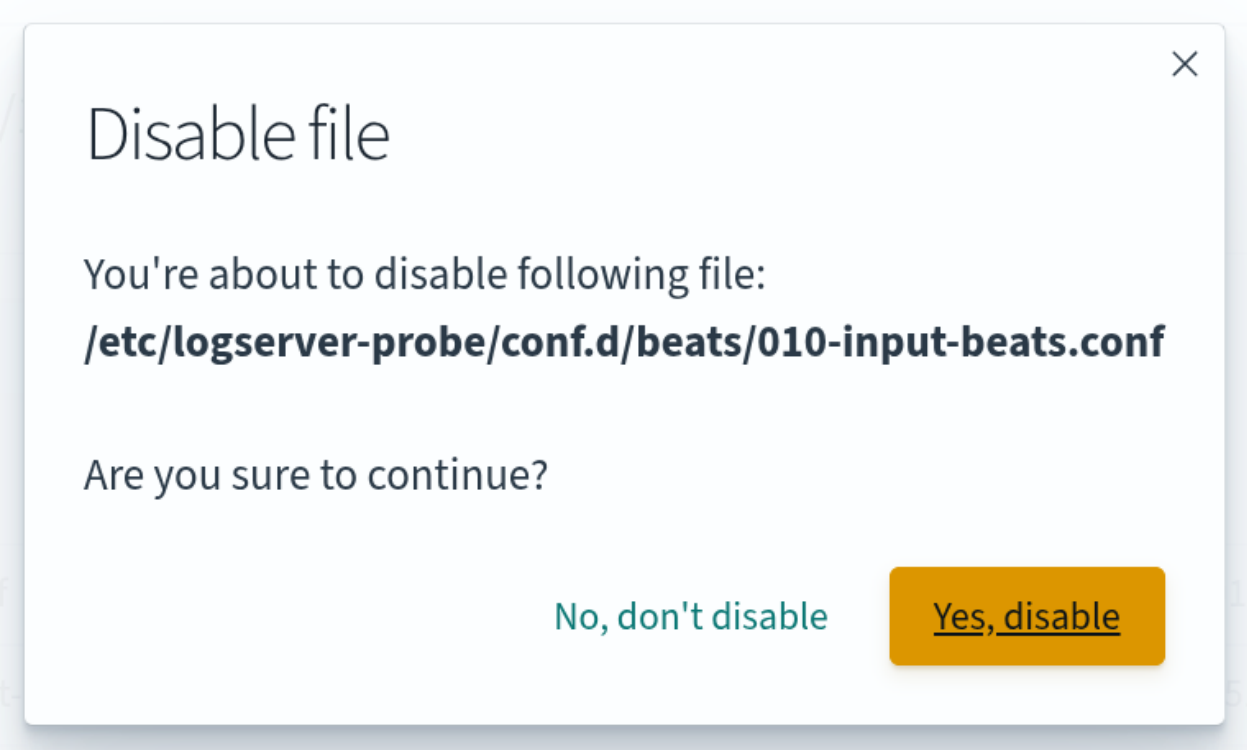
Delete file

Choose `Delete` in the `Actions` menu, or click the trash icon, and confirm.



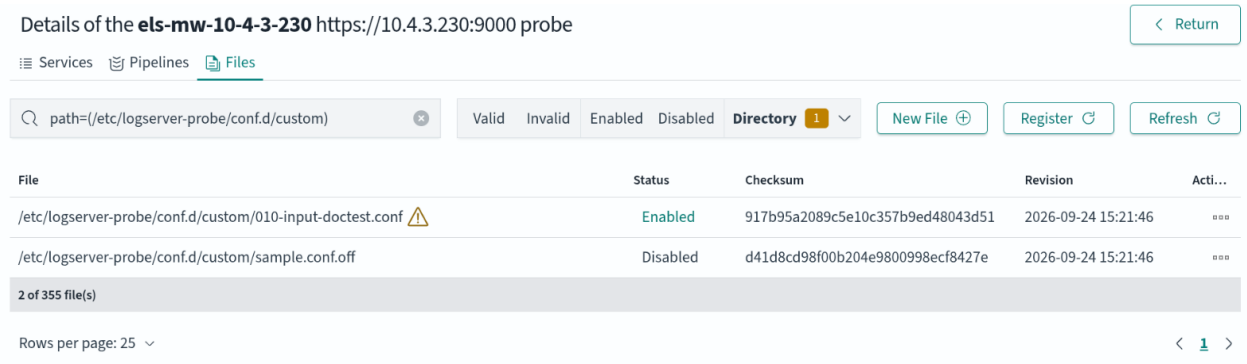
Enable/disable file

A configuration file can be enabled or disabled. The pipeline uses only enabled files, so you can disable a file, for example for testing. Choose `Enable` or `Disable` in the `Actions` menu, or click the status, and confirm.

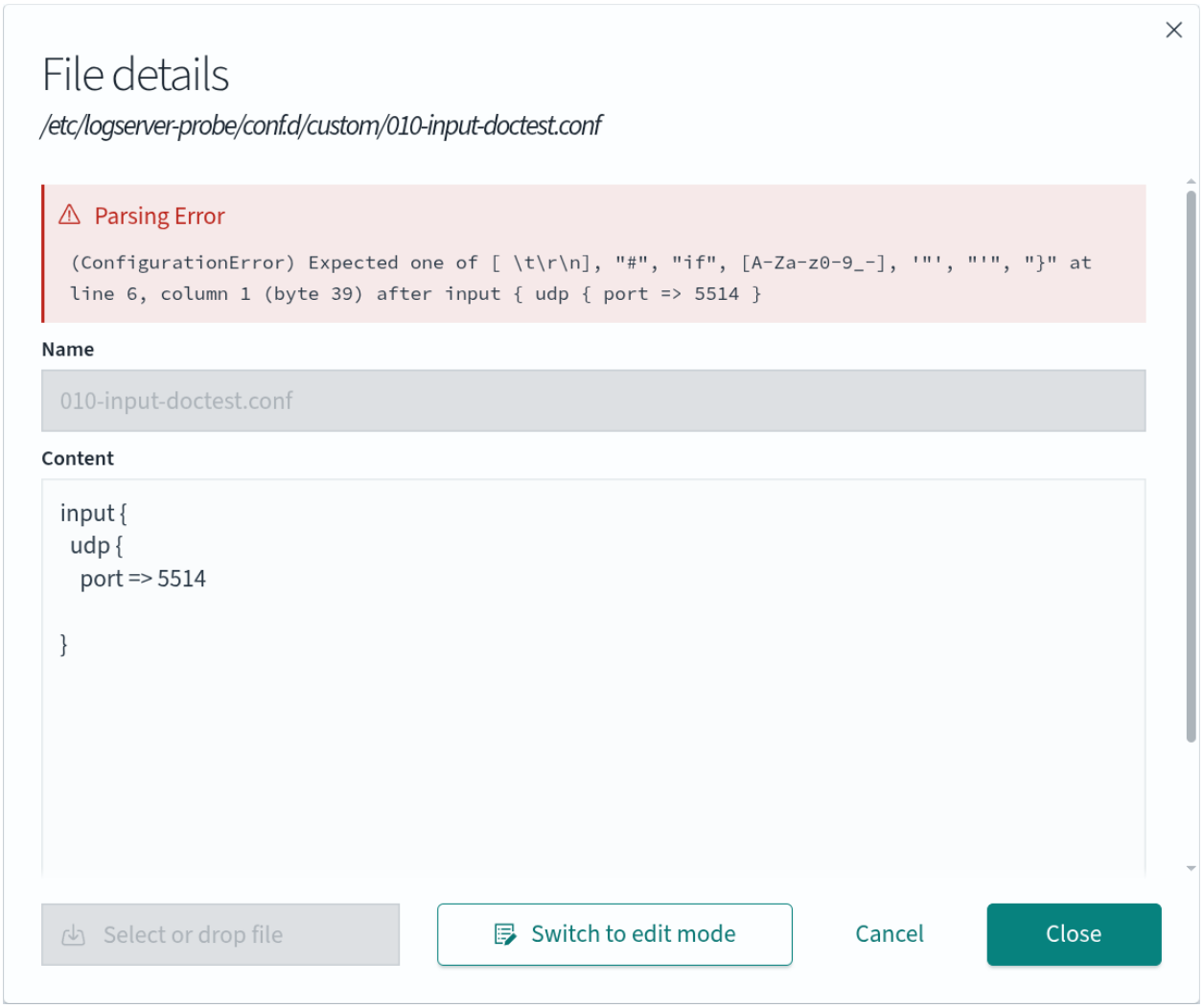


Files parsing

The probe parses `.conf` and `.conf.off` files in `/etc/logserver-probe/conf.d/` to check their syntax. A file that cannot be parsed has a warning icon next to its path.

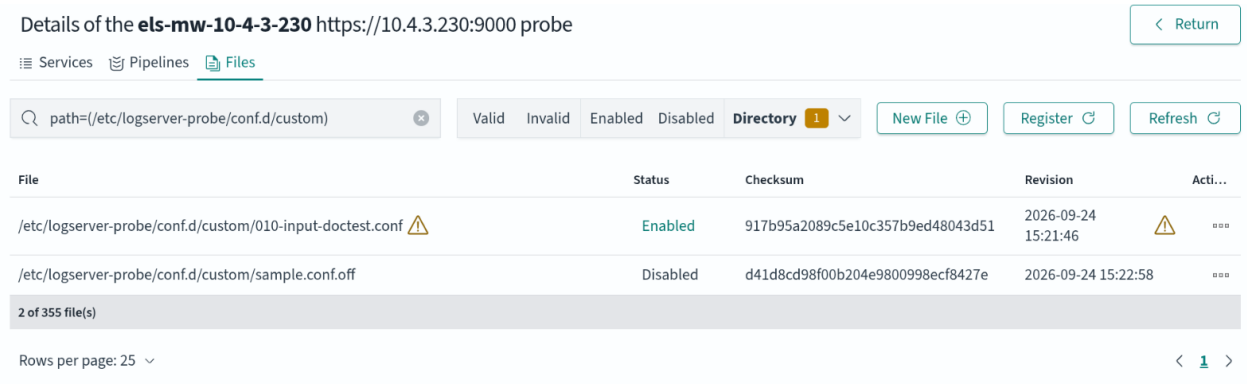


Click the icon to open the file with the parser message and correct the error.



Files' revision consistency

The probe registers its files every 12 hours (`registration_interval_hours` in `license-service.conf`). A file that was not found in the latest registration, for example because someone deleted it directly on the probe, shows a warning icon in the `Revision` column with the message `Outdated revision`.



Click the icon to write the stored copy of the file back to the probe. If you deleted the file on purpose, delete it in the GUI as well.

Files re-registration

Click `Register` to register all files on the probe again. The registration also parses the pipeline configuration files. Click `Refresh` to see the result.

Technical Configuration

The Energy Logserver uses Network Probe service to dynamically unify data from disparate sources and normalize the data into a destination of your choice. A Network Probe pipeline has two required elements, *input* and *output*, and one optional element *filter*. The input plugins consume data from a source, the filter plugins modify the data as you specify, and the output plugins write the data to a destination. The location for Network Probe pipeline configuration files is: `/etc/logserver-probe/conf.d/`. The following example configurations illustrate the available input, filter, and output plugins. These files are not installed by default, create them manually or use the *Management Interface*.

Example pipeline configurations:

- `01-input-beats.conf`
- `01-input-syslog.conf`
- `01-input-snmp.conf`
- `01-input-http.conf`
- `01-input-file.conf`
- `01-input-database.conf`
- `020-filter-beats-syslog.conf`
- `020-filter-network.conf`
- `099-filter-geoip.conf`
- `100-output-logserver.conf`
- `naemon_beat.example`
- `perflogs.example`

Note

`/etc/logserver-probe/pipelines.yml` is generated, not shipped. The package installs the upstream file with no pipelines enabled, and the installer then rebuilds it by concatenating every `/etc/logserver-probe/pipelines.d/*.yml` definition. Installing an integration registers its definition there, which is why the file grows as you add sources. Check what is actually active with `cat /etc/logserver-probe/pipelines.yml`.

Every integration places its definition in `pipelines.d`, either as `<name>.yml` if it is meant to run from the start, or as `<name>.yml.off` if it waits to be installed. Only the `.yml` files are concatenated, so a freshly installed probe runs a short list:

Pipeline	Purpose	Present in
`beats`	Data from Beats agents	Every installation
`logtrail`	Log viewer backend	Every installation
`syslog`	Syslog collector	Every installation
`syslog_elastic`, `syslog_wazuh`	Syslog routed to the SIEM indices	Installations with the SIEM add-on
`blacklists`	IOC and blacklist matching	Installations with the SIEM add-on
`wazuh`	Endpoint Security alerts	Installations with the SIEM add-on

Everything else, including the Suricata, Zeek, NetFlow and vendor pipelines, arrives disabled and starts running once you install its integration. Pipelines can also be created through the Management Interface or by adding definition files to `/etc/logserver-probe/pipelines.d/` by hand.

Several procedures in this documentation have you edit `pipelines.yml` directly. That works, but the installer rebuilds the file from `pipelines.d` when it runs, so a definition placed in `pipelines.d` is the one that survives a reinstallation.

- *Input "beats"*
- *Getting data from share folder*
- *Input "network"*
- *Input SNMP*
- *Input HTTP / HTTPS*
- *Input Relp*
- *Kafka and Zookeeper*
- *Input Kafka*
- *Input Google Pub/Sub*
- *Input File*
- *Input database*
- *Input CEF*
- *Input OPSEC*

- *Input SDEE*
- *Input XML*
- *Input WMI*
- *Filter "beats syslog"*
- *Filter "network"*
- *Filter "geoip"*
- *Avoiding duplicate documents*
- *Data enrichment*
- *Output to Data Node*
- *naemon beat*
- *perflog*
- *LDAP data enrichment*
- *Multiline codec*
- *Known issues and troubleshooting*

Input "beats"

This plugin waits for receiving data from remote beats services. It uses tcp /5044 port for communication:

```
input {
        beats {
                port => 5044
        }
}
```

Getting data from share folder

Using beats, you can read data from FTP, SFTP, SMB share. Connection to remote resources should be done as follows:

Input - FTP server

- Installation

```
yum install curlftpfs
```

- Create mount ftp directory

```
mkdir /mnt/my_ftp
```

- Use `curlftpfs` to mount your remote ftp site. Suppose my access credentials are as follows:

```
curlftpfs ftp-user:ftp-pass@my-ftp-location.local /mnt/my_ftp/
```

Input - SFTP server

- Install the required packages

```
yum install sshfs
```

- Add user

```
sudo adduser yourusername fuse
```

- Create local folder

```
mkdir ~/Desktop/sftp
```

- Mount remote folder to local:

```
sshfs HOSTuser@remote.host.or.ip:/host/dir/to/mount ~/Desktop/sftp
```

Input - SMB/CIFS server

- Create local folder

```
mkdir ~/Desktop/smb
```

- Mount remote folder to local:

```
mount -t smbfs //remote.host.or.ip/freigabe /mnt -o username=testuser
```

or

```
mount -t cifs //remote.host.or.ip/freigabe /mnt -o username=testuser
```

Input "network"

This plugin reads events over a TCP or UDP socket assigns the appropriate tags:

```
input {
       tcp {
                port => 5514
                type => "network"
                tags => [ "LAN", "TCP" ]
       }
       udp {
                port => 5514
                type => "network"
                tags => [ "LAN", "UDP" ]
       }
}
```

To redirect the default syslog port (514/TCP/UDP) to the dedicated collector port, follow these steps:

```
firewall-cmd --zone=public --add-masquerade --permanent
firewall-cmd --zone=public --add-forward-port=port=514:proto=udp:toport=5514 -
↪-permanent
```

(continues on next page)

(continued from previous page)

```
firewall-cmd --zone=public --add-forward-port=port=514:proto=tcp:toport=5514 -
↪-permanent
firewall-cmd --reload
```

These are the same rules the installer applies when firewalld is present. Masquerade must stay enabled on the zone, the forward-port rule depends on it.

Input SNMP

The SNMP input polls network devices using Simple Network Management Protocol (SNMP) to gather information related to the current state of the devices operation:

```
input {
  snmp {
    get => ["1.3.6.1.2.1.1.1.0"]
    hosts => [{host => "udp:127.0.0.1/161" community => "public" version =>
↪"2c"  retries => 2  timeout => 1000}]
  }
}
```

Input HTTP / HTTPS

Using this input you can receive single or multiline events over http(s). Applications can send an HTTP request to the endpoint started by this input and Network Probe will convert it into an event for subsequent processing. Sample definition:

```
input {
  http {
    host => "0.0.0.0"
    port => "8080"
  }
}
```

Events are by default sent in plain text. You can enable encryption by setting ssl to true and configuring the ssl_certificate and ssl_key options:

```
input {
  http {
    host => "0.0.0.0"
    port => "8080"
    ssl => "true"
    ssl_certificate => "path_to_certificate_file"
    ssl_key => "path_to_key_file"
  }
}
```

Input Relp

Note

The `relp` input plugin is not shipped with the Network Probe. Install it before using this configuration:

```
sudo -u logserver-probe /usr/share/logserver-probe/bin/logstash-plugin␣
↪install logstash-input-relp
```

Description

Read RELP events over a TCP socket.

This protocol implements application-level acknowledgments to help protect against message loss.

Message acks only function as far as messages being put into the queue for filters; anything lost after that point will not be retransmitted.

Relp input configuration options

This plugin supports the following configuration options plus the Common Options described later.

Nr.	Setting	Input type	Required
1	host	string	No
2	port	number	Yes
3	ssl_cacert	a valid filesystem path	No
4	ssl_cert	a valid filesystem path	No
5	ssl_enable	boolean	No
6	ssl_key	a valid filesystem path	No
7	ssl_key_passphrase	password	No
8	ssl_verify	string	boolean

`host` - The address to listen on.

`port` - The port to listen on.

`ssl_cacert` - The SSL CA certificate, chainfile or CA path. The system CA path is automatically included.

`ssl_cert` - SSL certificate path

`ssl_enable` - Enable SSL (must be set for other ssl_ options to take effect).

`ssl_key` - SSL key path

`ssl_key_passphrase` - SSL key passphrase

`ssl_verify` - Verify the identity of the other end of the SSL connection against the CA. For input, sets the field sslsubject to that of the client certificate.

Common Options The following configuration options are supported by all input plugins:

Nr.	Setting	Input type	Required
1	add_field	hash	No
2	codec	codec	No
3	enable_metric	boolean	No
4	id	string	No
5	tags	array	No
6	type	string	No

`add_field` - Add a field to an event

`codec` - The codec used for input data. Input codecs are a convenient method for decoding your data before it enters the input, without needing a separate filter in your Network Probe pipeline.

`enable_metric` - Disable or enable metric logging for this specific plugin instance by default we record all the metrics we can, but you can disable metrics collection for a specific plugin.

`id` - Add a unique ID to the plugin configuration. If no ID is specified, Network Probe will generate one. It is strongly recommended to set this ID in your configuration. This is particularly useful when you have two or more plugins of the same type, for example, if you have 2 relp inputs. Adding a named ID in this case will help in monitoring Network Probe when using the monitoring APIs.

```
input {
  relp {
    id => "my_plugin_id"
  }
}
```

`tags` - add any number of arbitrary tags to your event.

`type` - Add a type field to all events handled by this input.

Types are used mainly for filter activation.

The type is stored as part of the event itself, so you can also use the type to search for it in GUI.

If you try to set a type on an event that already has one (for example when you send an event from a shipper to an indexer) then a new input will not override the existing type. A type set at the shipper stays with that event for its life even when sent to another Network Probe server.

Kafka and Zookeeper

Kafka enables load distribution between data-receiving nodes and supports encrypted communication. It is bundled with the Network Probe package but **disabled (masked) by default** after installation.

Enabling Kafka and Zookeeper

Kafka and Zookeeper services are masked after installation. To enable them:

```
systemctl unmask zookeeper kafka
systemctl enable zookeeper kafka
systemctl start zookeeper kafka
```

Verify the services are running:

```
systemctl status zookeeper kafka
```

Service Dependencies

The `kafka.service` unit has the following dependency chain:

Directive	Services
`Requires`	`zookeeper.service`, `license-service.service`
`After`	`zookeeper.service`, `license-service.service`
`Before`	`logserver-probe.service`

Important

Ensure `license-service` is running before starting Kafka. If `license-service` is not active, the Kafka service will fail to start.

Default Configuration

Configuration files are located in `/opt/kafka/config/`.

server.properties

Key defaults in `/opt/kafka/config/server.properties`:

Parameter	Default	Description
`broker.id`	`0`	Unique broker identifier (must be unique per node in multi-broker setups)
`log.dirs`	`/tmp/kafka-logs`	Directory for Kafka log segments
`zookeeper.connect`	`localhost:2181`	Zookeeper connection string
`log.retention.hours`	`168`	Log retention period (7 days)
`num.partitions`	`1`	Default partitions per topic
`num.network.threads`	`3`	Threads for network requests
`num.io.threads`	`8`	Threads for I/O operations

Default listener port: **9092** (PLAINTEXT).

zookeeper.properties

Key defaults in `/opt/kafka/config/zookeeper.properties`:

Parameter	Default	Description
`dataDir`	`/tmp/zookeeper`	Snapshot storage directory
`clientPort`	`2181`	Client connection port
`maxClientCnxns`	`0`	Max client connections (0 = unlimited)
`admin.enableServer`	`false`	ZooKeeper admin server

Firewall Rules

Open the required ports for Kafka and Zookeeper:

```
firewall-cmd --permanent --add-port=2181/tcp
firewall-cmd --permanent --add-port=2888/tcp
firewall-cmd --permanent --add-port=3888/tcp
firewall-cmd --permanent --add-port=9092/tcp
firewall-cmd --reload
```

Port	Service
2181	Zookeeper client connections
2888	Zookeeper peer communication (multi-node)
3888	Zookeeper leader election (multi-node)
9092	Kafka broker (PLAINTEXT)
9093	Kafka broker (SSL, when TLS is enabled)

Testing Kafka

After enabling, verify Kafka is working by creating and consuming a test topic:

1. Create a test topic:

```
/opt/kafka/bin/kafka-topics.sh --bootstrap-server localhost:9092 --create␣
↪--partitions 1 --replication-factor 1 --topic test
```

2. List existing topics:

```
/opt/kafka/bin/kafka-topics.sh --bootstrap-server localhost:9092 --list
```

3. Produce test messages:

```
/opt/kafka/bin/kafka-console-producer.sh --topic test --bootstrap-server␣
↪localhost:9092
```

   Type a few messages and press `Ctrl+C` to exit.

4. Consume test messages:

```
/opt/kafka/bin/kafka-console-consumer.sh --topic test --from-beginning --
↪bootstrap-server localhost:9092
```

Log Retention

Kafka durably persists all published records using a configurable retention period. The default retention is 168 hours (7 days). After this period, records are discarded to free up space.

To change the retention period, edit `/opt/kafka/config/server.properties`:

```
log.retention.hours=168
```

Restart the Kafka service after changes:

```
systemctl restart kafka
```

TLS Encryption

Kafka supports TLS encryption for communication between brokers and clients. When TLS is enabled, the broker listens on port **9093** (SSL) in addition to or instead of port 9092 (PLAINTEXT).

Note

Energy Logserver 8.x ships with OpenJDK 21. JKS is deprecated since OpenJDK 9 (JEP 229) and `keytool` defaults to PKCS12. The procedure below uses `.jks` for backward compatibility. For new deployments, use the PKCS12 variant shown after step 6.

1. Generate Server Keystore

Create a server keystore with a certificate pair. Replace the placeholders with your values:

```
keytool -keystore server.keystore.jks -alias {alias_name} -validity {validity_
↪days} -genkey -keyalg RSA -ext SAN=DNS:{FQDN},IP:{server_IP}
```

2. Create a Certificate Authority (CA)

```
openssl req -new -x509 -keyout rootCA.key -out rootCA.crt -days 365
```

3. Import CA into Truststores

```
keytool -keystore server.truststore.jks -alias CARoot -import -file rootCA.crt
keytool -keystore client.truststore.jks -alias CARoot -import -file rootCA.crt
```

4. Create a Certificate Signing Request (CSR)

```
keytool -keystore server.keystore.jks -alias {alias_name} -certreq -file cert-
↪file -ext SAN=DNS:{FQDN},IP:{server_IP}
```

5. Sign the Certificate

```
openssl x509 -req -extfile <(printf "subjectAltName = DNS:{FQDN},IP:{server_
↪IP}") -CA rootCA.crt -CAkey rootCA.key -in cert-file -out cert-signed -days␣
↪3650 -CAcreateserial -passin pass:{password}
```

6. Import Signed Certificate into Server Keystore

```
keytool -keystore server.keystore.jks -alias CARoot -import -file rootCA.crt
keytool -keystore server.keystore.jks -alias {alias_name} -import -file cert-
↪signed
```

PKCS12 variant (recommended for OpenJDK 21)

Replace the `keytool` commands in steps 1-6 with `-storetype PKCS12` and the `.p12` file extension. Example for step 1:

```
keytool -keystore server.keystore.p12 -storetype PKCS12 -alias {alias_name} -
↪validity {validity_days} -genkey -keyalg RSA -ext SAN=DNS:{FQDN},IP:{server_
↪IP}
```

When using PKCS12 keystores, declare the type explicitly in the Kafka broker `server.properties`:

```
ssl.keystore.type=PKCS12
ssl.truststore.type=PKCS12
```

In the pipeline output and input shown below, set `ssl_truststore_type => "PKCS12"` and point `ssl_truststore_location` at the `.p12` file.

7. (Optional) Import Trusted Certificates

If you have externally trusted certificates, create a client keystore and import them:

```
keytool -keystore client.keystore.jks -alias {alias_name} -validity {validity_
↪days} -keyalg RSA -genkey
```

Combine the certificate and key into PKCS12 format:

```
openssl pkcs12 -export -in {your_cert} -inkey {your_key} -out {pair_name}.p12␣
↪-name {friendly_name} -CAfile ca.crt -caname root
```

Import the CA and PKCS12 certificate into the client keystore:

```
keytool -keystore client.keystore.jks -alias CARoot -import -file {ca_cert_
↪file}
keytool -importkeystore -deststorepass {keystore_password} -destkeystore␣
↪client.keystore.jks -srckeystore {pair_name}.p12 -srcstoretype PKCS12
```

Configuring Kafka Broker for TLS

Edit `/opt/kafka/config/server.properties`:

```
listeners=PLAINTEXT://localhost:9092,SSL://{FQDN}:9093
ssl.keystore.location={path_to}/server.keystore.jks
ssl.keystore.password={keystore_password}
ssl.key.password={key_password}
ssl.truststore.location={path_to}/server.truststore.jks
ssl.truststore.password={truststore_password}
ssl.enabled.protocols=TLSv1.2
ssl.client.auth=required
security.inter.broker.protocol=SSL
```

Restart Kafka:

```
systemctl restart kafka
```

Note

When TLS is enabled, also open port 9093 in the firewall:

```
firewall-cmd --permanent --add-port=9093/tcp
firewall-cmd --reload
```

Configuring Network Probe as a Kafka TLS Client

Output: sending data to Kafka over TLS:

```
output {
  kafka {
    bootstrap_servers => "{FQDN}:9093"
    security_protocol => "SSL"
    ssl_truststore_type => "JKS"
    ssl_truststore_location => "{path_to}/client.truststore.jks"
    ssl_truststore_password => "{truststore_password}"
    client_id => "host.name"
    topic_id => "Topic-1"
    codec => json
  }
}
```

Input: consuming data from Kafka over TLS:

```
input {
  kafka {
    bootstrap_servers => "{FQDN}:9093"
    security_protocol => "SSL"
    ssl_truststore_type => "JKS"
    ssl_truststore_location => "{path_to}/client.truststore.jks"
    ssl_truststore_password => "{truststore_password}"
    consumer_threads => 4
    topics => ["Topic-1"]
    codec => json
    tags => ["kafka"]
  }
}
```

Input Kafka

This input will read events from a Kafka topic.

Sample definition:

```
input {
  kafka {
    bootstrap_servers => "10.0.0.1:9092"
    consumer_threads => 3

    topics => ["example"]
    codec => json
    client_id => "hostname"
    group_id => "logserver"
    max_partition_fetch_bytes => "30000000"
    max_poll_records => "1000"

    fetch_max_bytes => "72428800"
    fetch_min_bytes => "1000000"
```

(continues on next page)

(continued from previous page)

```
    fetch_max_wait_ms => "800"

    check_crcs => false

  }
}
```

`bootstrap_servers` - A list of URLs of Kafka instances to use for establishing the initial connection to the cluster. This list should be in the form of host1:port1,host2:port2 These urls are just used for the initial connection to discover the full cluster membership (which may change dynamically) so this list need not contain the full set of servers (you may want more than one, though, in case a server is down).

`consumer_threads` - Ideally you should have as many threads as the number of partitions for a perfect balance, more threads than partitions means that some threads will be idle

`topics` - A list of topics to subscribe to, defaults to ["logserver"].

`codec` - The codec used for input data. Input codecs are a convenient method for decoding your data before it enters the input, without needing a separate filter in your Network Probe pipeline.

`client_id` - The id string to pass to the server when making requests. The purpose of this is to be able to track the source of requests beyond just ip/port by allowing a logical application name to be included.

`group_id` - The identifier of the group this consumer belongs to. Consumer group is a single logical subscriber that happens to be made up of multiple processors. Messages in a topic will be distributed to all Network Probe instances with the same group_id.

`max_partition_fetch_bytes` - The maximum amount of data per-partition the server will return. The maximum total memory used for a request will be #partitions * max.partition.fetch.bytes. This size must be at least as large as the maximum message size the server allows or else it is possible for the producer to send messages larger than the consumer can fetch. If that happens, the consumer can get stuck trying to fetch a large message on a certain partition.

`max_poll_records` - The maximum number of records returned in a single call to poll().

`fetch_max_bytes` - The maximum amount of data the server should return for a fetch request. This is not an absolute maximum, if the first message in the first non-empty partition of the fetch is larger than this value, the message will still be returned to ensure that the consumer can make progress.

`fetch_min_bytes` - The minimum amount of data the server should return for a fetch request. If insufficient data is available the request will wait for that much data to accumulate before answering the request.

`fetch_max_wait_ms` - The maximum amount of time the server will block before answering the fetch request if there isn't sufficient data to immediately satisfy fetch_min_bytes. This should be less than or equal to the timeout used in poll_timeout_ms.

`check_crcs` - Automatically check the CRC32 of the records consumed. This ensures no on-the-wire or on-disk corruption to the messages occurred. This check adds some overhead, so it may be disabled in cases seeking extreme performance.

Input Google Pub/Sub

This input reads messages from a Google Cloud Pub/Sub subscription. It is available from version 8.1.0 and is not installed by default. The plugin is in the installation package, in `install/logstash/logstash/plugins/`. Copy it to a directory the `logserver-probe` user can read, install it, and restart the service:

```
sudo -u logserver-probe /usr/share/logserver-probe/bin/logstash-plugin␣
↪install file:///tmp/logstash-input-google_pubsub-1.4.3.zip
systemctl restart logserver-probe
```

Sample definition:

```
input {
  google_pubsub {
    project_id => "my-project"
    topic => "logs"
    subscription => "logserver-sub"
    json_key_file => "/etc/logserver-probe/pubsub-key.json"
  }
}
```

`project_id`, `topic`, `subscription` - Required. The Google Cloud project, the topic, and the subscription to read from.

`json_key_file` - Service account key file. It must exist and be readable by the `logserver-probe` user, otherwise the pipeline does not start. Instead of a file you can pass the key base64-encoded in `json_key_file_content`; use only one of the two. On Google Compute Engine both can be omitted.

`create_subscription` - Tries to create the subscription; default `false`. It needs extra permissions for the service account.

`include_metadata` - Puts the full Pub/Sub message in `[@metadata][pubsub_message]`; default `false`.

Input File

This plugin stream events from files, normally by tailing them in a manner similar to tail -0F but optionally reading them from the beginning. Sample definition:

```
file {
    path => "/tmp/access_log"
    start_position => "beginning"
}
```

Input database

This plugin can read data in any database with a JDBC interface into Network Probe. You can periodically schedule ingestion using a cron syntax (see schedule setting) or run the query one time to load data into Network Probe. Each row in the resultset becomes a single event. Columns in the resultset are converted into fields in the event.

Input MySQL

Download jdbc driver: https://dev.mysql.com/downloads/connector/j/

Sample definition:

```
input {
  jdbc {
    jdbc_driver_library => "mysql-connector-java-5.1.36-bin.jar"
    jdbc_driver_class => "com.mysql.jdbc.Driver"
    jdbc_connection_string => "jdbc:mysql://localhost:3306/mydb"
    jdbc_user => "mysql"
    jdbc_password => "mysql"
    parameters => { "favorite_artist" => "Beethoven" }
    schedule => "* * * * *"
    statement => "SELECT * from songs where artist = :favorite_artist"
```

(continues on next page)

(continued from previous page)

```
  }
}
```

Input MSSQL

Download jdbc driver: https://learn.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server

Sample definition:

```
input {
  jdbc {
    jdbc_driver_library => "./mssql-jdbc-6.2.2.jre8.jar"
    jdbc_driver_class => "com.microsoft.sqlserver.jdbc.SQLServerDriver"
    jdbc_connection_string => "jdbc:sqlserver://VB201001000;
↪databaseName=Database;"
    jdbc_user => "mssql"
    jdbc_password => "mssql"
    jdbc_default_timezone => "UTC"
    statement_filepath => "/usr/share/logserver-probe/plugin/query"
    schedule => "*/5 * * * *"
    sql_log_level => "warn"
    record_last_run => "false"
    clean_run => "true"
  }
}
```

Input Oracle

Download jdbc driver: https://www.oracle.com/database/technologies/appdev/jdbc-downloads.html

Sample definition:

```
input {
  jdbc {
    jdbc_driver_library => "./ojdbc8.jar"
    jdbc_driver_class => "oracle.jdbc.driver.OracleDriver"
    jdbc_connection_string => "jdbc:oracle:thin:@hostname:PORT/SERVICE"
    jdbc_user => "oracle"
    jdbc_password => "oracle"
    parameters => { "favorite_artist" => "Beethoven" }
    schedule => "* * * * *"
    statement => "SELECT * from songs where artist = :favorite_artist"
  }
}
```

Input PostgreSQL

Download jdbc driver: https://jdbc.postgresql.org/download/

Sample definition:

```
input {
    jdbc {
        jdbc_driver_library => "D:/postgresql-42.2.5.jar"
        jdbc_driver_class => "org.postgresql.Driver"
        jdbc_connection_string => "jdbc:postgresql://127.0.0.1:57610/mydb"
        jdbc_user => "myuser"
        jdbc_password => "mypw"
        statement => "select * from mytable"
    }
}
```

Input CEF

The common event format (CEF) is a standard for the interoperability of event or log generating devices and applications. The standard defines a syntax for log records. It comprises of a standard prefix and a variable extension that is formatted as key-value pairs.

```
input {
  tcp {
    codec => cef { delimiter => "\r\n" }
    port => 12345
  }
}
```

This setting allows the following character sequences to have special meaning:

- `\r` (backslash “r”) - means carriage return (ASCII 0x0D)
- `\n` (backslash “n”) - means newline (ASCII 0x0A)

Input OPSEC

FW1-LogGrabber is a Linux command-line tool to grab logfiles from remote Checkpoint devices. It makes extensive use of OPSEC Log Export APIs (LEA) from Checkpoint’s OPSEC SDK 6.0 for Linux 50.

Build FW1-LogGrabber

FW1-LogGrabber v2.0 and above can be built on Linux x86/amd64 platforms only.

If you are interested in other platforms please check FW1-LogGrabber v1.11.1 website

Download dependencies

FW1-LogGrabber uses API-functions from Checkpoint’s OPSEC SDK 6.0 for Linux 50.

You must take care of downloading the Checkpoint OPSEC SDK and extracting it inside the `OPSEC_SDK` folder.

You also need to install some required 32-bit libraries.

If you are using **Debian or Ubuntu**, please run:

```
sudo apt-get install gcc-multilib g++-multilib libelf-dev:i386 libpam0g:i386␣
↪zlib1g-dev:i386
```

If you are using **CentOS or RHEL**, please run:

```
sudo yum install gcc gcc-c++ make glibc-devel.i686 elfutils-libelf-devel.i686␣
↪zlib-devel.i686 libstdc++-devel.i686 pam-devel.i686
```

Compile source code

Building should be as simple as running GNU Make in the project root folder:

```
make
```

If the build process complains, you might need to tweak some variables inside the `Makefile` (e.g. `CC`, `LD` and `OPSEC_PKG_DIR`) according to your environment.

Install FW1-LogGrabber

To install FW1-LogGrabber into its default location `/usr/local/fw1-loggrabber` (defined by `INSTALL_DIR` variable), please run

```
sudo make install
```

Set environment variables

FW1-LogGraber makes use of two environment variables, which should be defined in the shell configuration files.

- `LOGGRABBER_CONFIG_PATH` defines a directory containing configuration files (`fw1-loggrabber.conf`, `lea.conf`). If the variable is not defined, the program expects to find these files in the current directory.
- `LOGGRABBER_TEMP_PATH` defines a directory where FW1-LogGrabber will store temporary files. If the variable is not defined, the program stores these files in the current directory.

Since the binary is dynamically linked to Checkpoint OPSEC libraries, please also add `/usr/local/fw1-loggrabber/lib` to `LD_LIBRARY_PATH` or to your dynamic linker configuration with

```
sudo echo /usr/local/fw1-loggrabber/lib > /etc/ld.so.conf.d/fw1-loggrabber.
↪conf
sudo ldconfig
```

Configuration files

lea.conf file

Starting with version 1.11, FW1-LogGrabber uses the default connection configuration procedure for OPSEC applications. This includes server, port and authentication settings. From now on, all this parameters can only be configured using the configuration file `lea.conf` (see `--leaconfigfile` option to use a different LEA configuration file) and not using the command-line as before.

- `lea_server ip <IP address>` specifies the IP address of the FW1 management station, to which FW1-LogGrabber should connect to.
- `lea_server port <port number>` is the port on the FW1 management station to which FW1-LogGrabber should connect to (for unauthenticated connections only).
- `lea_server auth_port <port number>` is the port to be used for authenticated connection to your FW1 management station.
- `lea_server auth_type <authentication mechanism>` you can use this parameter to specify the authentication mechanism to be used (default is `sslca`); valid values are `sslca`,

sslca_clear, sslca_comp, sslca_rc4, sslca_rc4_comp, asym_sslca, asym_sslca_comp, asym_sslca_rc4, asym_sslca_rc4_comp, ssl, ssl_opsec, ssl_clear, ssl_clear_opsec, fwn1 and auth_opsec.

- opsec_sslca_file <p12-file> specify the location of the PKCS#12 certificate, when using authenticated connections.
- opsec_sic_name <LEA client SIC name> is the SIC name of the LEA client for authenticated connections.
- lea_server opsec_entity_sic_name <LEA server SIC name> is the SIC name of your FW1 management station when using authenticated connections.

fw1-loggrabber.conf file

This paragraph deals with the options that can be set within the configuration file. The default configuration file is fw1-loggrabber.conf (see --configfile option to use a different configuration file). The precedence of given options is as follows: command line, configuration file, default value. E.g. if you set the resolve-mode to be used in the configuration file, this can be overwritten by command line option --noresolve; only if an option isn't set neither on command line nor in the configuration file, the default value will be used.

- DEBUG_LEVEL=<0-3> sets the debug level to the specified value; zero means no output of debug information, and further levels will cause output of program specific as well as OPSEC specific debug information.
- FW1_LOGFILE=<name of log file> specifies the name of the FW1 logfile to be read; this can be either done exactly or using only a part of the filename; if no exact match can be found in the list of logfiles returned by the FW-1 management station, all logfiles which contain the specified string are processed; if this parameter is omitted, the default logfile fw.log will be processed.
- FW1_OUTPUT=<files|logs> specifies whether FW1-LogGrabber should only display the available logfiles (files) on the FW11 server or display the content of these logfiles (logs).
- FW1_TYPE=<ng|2000> choose which version of FW1 to connect to; for Checkpoint FW-1 5.0 you have to specify NG and for Checkpoint FW-1 4.1 you have to specify 2000.
- FW1_MODE=<audit|normal> specifies whether to display audit logs, which contain administrative actions, or normal security logs, which contain data about dropped and accepted connections.
- MODE=<online|online-resume|offline> when using online mode, FW1-LogGrabber starts retrieving logging data from the end of the specified logfile and displays all future log entries (mainly used for continuously processing); the online-resume mode is similar to the online mode, but if FW1-LogGrabber is stopped and started again, it resumes processing from where it was stopped; if you instead choose the offline mode, FW1-LogGrabber quits after having displayed the last log entry.
- RESOLVE_MODE=<yes|no> with this option (enabled by default), IP addresses will be resolved to names using FW1 name resolving behavior; this resolving mechanism will not cause the machine running FW1-LogGrabber to initiate DNS requests, but the name resolution will be done directly on the FW1 machine; if you disable resolving mode, IP addresses will be displayed in log output instead of names.
- RECORD_SEPARATOR=<char> can be used to change the default record separator | (pipe) into another character; if you choose a character which is contained in some log data, the occurrence within the logdata will be escaped by a backslash.
- LOGGING_CONFIGURATION=<screen|file|syslog> can be used for redirecting logging output to other destinations than the default destination STDOUT; currently it is possible to redirect output to a file or to the syslog daemon.
- OUTPUT_FILE_PREFIX=<prefix of output file> when using file output, this parameter defines a prefix for the output filename; default value is simply fw1-loggrabber.

- OUTPUT_FILE_ROTATESIZE=<rotatesize in bytes> when using file output, this parameter specifies the maximum size of the output files, before they will be rotated with suffix -YYYY-MM-DD-hhmmss[-x].log; default value is 1048576 bytes, which equals 1 MB; setting a zero value disables file rotation.
- SYSLOG_FACILITY=<USER|LOCAL0|...|LOCAL7> when using syslog output, this parameter sets the syslog facility to be used.
- FW1_FILTER_RULE="<filterexpression1>[;<filterexpression2>]" defines filters for normal log mode; you can find a more detailed description of filter rules, along with some examples, *in a separate chapter below*.
- AUDIT_FILTER_RULE="<filterexpression1>[;<filterexpression2>]" defines filters for audit log mode; you can find a more detailed description of filter rules, along with some examples, *in a separate chapter below*.

Command line options

In the following section, all available command line options are described in detail. Most of the options can also be configured using the file fw1-loggrabber.conf (see --configfile option to use a different configuration file). The precedence of given options is as follows: command line, configuration file, default value. E.g. if you set the resolve-mode to be used in the configuration file, this can be overwritten by command line option --noresolve; only if an option isn't set neither on command line nor in the configuration file, the default value will be used.

Help

Use --help to display basic help and usage information.

Debug level

The --debuglevel option sets the debug level to the specified value. A zero debug level means no output of debug information, while further levels will cause output of program specific as well as OPSEC specific debug information.

Location of configuration files

The -c <configfilename> or --configfile <configfilename> options allow to specify a non-default configuration file, in which most of the command line options can be configured, as well as other options which are not available as command line parameters.

If this parameter is omitted, the file fw1-loggrabber.conf inside $LOGGRABBER_CONFIG_PATH will be used. *See above* for a description of all available configuration file options.

Using -l <leaconfigfilename> or --leaconfigfile <leaconfigfilename> instead, it's possible to use a non-default LEA configuration file. In this file, all connection parameters such as FW1 server, port, authentication method as well as SIC names have to be configured, as usual procedure for OPSEC applications.

If this parameter is omitted, the file lea.conf inside $LOGGRABBER_CONFIG_PATH will be used. *See above* for a description of all available LEA configuration file options.

Remote log files

With -f <logfilename|pattern|ALL> or --logfile <logfilename|pattern|ALL> you can specify the name of the remote FW1 logfile to be read.

This can be either done exactly or using only a part of the filename. If no exact match can be found in the list of logfiles returned by the FW1 management station, all logfiles which contain the specified string are processed.

A special case is the usage of `ALL` instead of a logfile name or pattern. In that case all logfiles that are available on the management station, will be processed. If this parameter is omitted, only the default logfile `fw.log` will be processed.

The first example displays the logfile `2003-03-27_213652.log`, while the second one processes all logfiles which contain `2003-03` in their filename.

```
--logfile 2003-03-27_213652.log
--logfile 2003-03
```

The default behavior of FW1-LogGrabber is to display the content of the logfiles and not just their names. This can be explicitely specified using the `--showlogs` option.

The option `--showfiles` can be used instead to simply show the available logfiles on the FW1 management station. After the names of the logfiles have been displayed, FW1-LogGrabber quits.

Name resolving behavior

Using the `--resolve` option, IP addresses will be resolved to names using FW1 name resolving behavior. This resolving mechanism will not cause the machine running FW1-LogGrabber to initiate DNS requests, but the name resolution will be done directly on the FW1 machine.

This is the default behavior of FW1-LogGrabber which can be disabled by using `--no-resolve`. That option will cause IP addresses to be displayed in log output instead of names.

Checkpoint firewall version

The default FW1 version, for which this tool is being developed, is Checkpoint FW1 5.0 (NG) and above. If no other version is explicitly specified, the default version is `--ng`.

The option `--2000` has to be used if you want to connect to older Checkpoint FW1 4.1 (2000) firewalls. You should keep in mind that some options are not available for non-NG firewalls; these include `--auth`, `--showfiles`, `--auditlog` and some more.

Online and Online-Resume modes

Using `--online` mode, FW1-LogGrabber starts output of logging data at the end of the specified logfile (or `fw.log` if no logfile name has been specified). This mode is mainly used for continuously processing FW1 log data and continues to display log entries also after scheduled and manual log switches. If you use `--logfile` to specify another logfile to be processed, you have to consider that no data will be shown, if the file isn't active anymore.

The `--online-resume` mode is similar to the above online mode, but starts output of logging data at the last known processed position (which is stored inside a cursor).

In contrast to online mode, when using `--offline` mode FW1-LogGrabber quits after having displayed the last log entry. This is the default behavior and is mainly used for analysis of historic log data.

Audit and normal logs

Using the `--auditlog` mode, the content of the audit logfile (`fw.adtlog`) can be displayed. This includes administrator actions and uses different fields than normal log data.

The default `--normallog` mode of FW1-LogGrabber processes normal FW1 logfiles. In contrast to the `--auditlog` option, no administrative actions are displayed in this mode, but all regular log data is.

Filtering

Filter rules provide the possibility to display only log entries that match a given set of rules. There can be specified one or more filter rules using one or multiple `--filter` arguments on the command line.

All individual filter rules are related by OR. That means a log entry will be displayed if at least one of the filter rules matches. You can specify multiple argument values by separating the values by `,` (comma).

Within one filter rule, there can be specified multiple arguments that have to be separated by `;` (semi-colon). All these arguments are related by AND. That means a filter rule matches a given log entry only, if all of the filter arguments match.

If you specify `!=` instead of `=` between the name and value of the filter argument, you can negate the name/value pair.

For arguments that expect IP addresses, you can specify either a single IP address, multiple IP addresses separated by `,` (comma), or a network address with netmask (e.g. `10.0.0.0/255.0.0.0`). Currently, it is not possible to specify a network address and a single IP address within the same filter argument.

Supported filter arguments

Normal mode:

```
action=<ctl|accept|drop|reject|encrypt|decrypt|keyinst>
dst=<IP address>
endtime=<YYYYMMDDhhmmss>
orig=<IP address>
product=<VPN-1 & FireWall-1|SmartDefense>
proto=<icmp|tcp|udp>
rule=<rulenumber|startrule-endrule>
service=<portnumber|startport-endport>
src=<IP address>
starttime=<YYYYMMDDhhmmss>
```

Audit mode:

```
action=<ctl|accept|drop|reject|encrypt|decrypt|keyinst>
administrator=<string>
endtime=<YYYYMMDDhhmmss>
orig=<IP address>
product=<SmartDashboard|Policy Editor|SmartView Tracker|SmartView␣
↪Status|SmartView Monitor|System Monitor|cpstat_monitor|SmartUpdate|CPMI␣
↪Client>
starttime=<YYYYMMDDhhmmss>
```

Example filters

Display all dropped connections:

```
--filter "action=drop"
```

Display all dropped and rejected connections:

```
--filter "action=drop,reject"
--filter "action!=accept"
```

Display all log entries generated by rules 20 to 23:

```
--filter "rule=20,21,22,23"
--filter "rule=20-23"
```

Display all log entries generated by rules 20 to 23, 30 or 40 to 42:

```
--filter "rule=20-23,30,40-42"
```

Display all log entries to `10.1.1.1` and `10.1.1.2`:

```
--filter "dst=10.1.1.1,10.1.1.2"
```

Display all log entries from `192.168.1.0/255.255.255.0`:

```
--filter "src=192.168.1.0/255.255.255.0"
```

Display all log entries starting from `2004/03/02 14:00:00`:

```
--filter "starttime=20040302140000"
```

Checkpoint Device Configuration for SSLCA

Modify `$FWDIR/conf/fwopsec.conf` and define the port to be used for authenticated LEA connections (e.g. 18184):

```
lea_server port 0
lea_server auth_port 18184
lea_server auth_type sslca
```

Restart in order to activate changes:

```
cpstop; cpstart
```

Create a new OPSEC Application Object with the following details:

```
Name: e.g. myleaclient
Vendor: User Defined
Server Entities: None
Client Entities: LEA
```

Initialize Secure Internal Communication (SIC) for recently created OPSEC Application Object and enter (and remember) the activation key (e.g. `def456`).

Write down the DN of the recently created OPSEC Application Object; this is your Client Distinguished Name, which you need later on.

Open the object of your FW1 management server and write down the DN of that object; this is the Server Distinguished Name, which you will need later on.

Add a rule to the policy to allow the port defined above as well as port 18210/tcp (FW1_ica_pull) in order to allow pulling of PKCS#12 certificate by the FW1-LogGrabber machine from the FW1 management server. Port 18210/tcp can be shut down after the communication between FW1-LogGrabber and the FW1 management server has been established successfully.

Finally, install the policy.

FW1-LogGrabber Configuration for SSLCA

Modify `$LOGGRABBER_CONFIG_PATH/lea.conf` and define the IP address of your FW1 management station (e.g. `10.1.1.1`) as well as port (e.g. `18184`), authentication type and SIC names for authenticated LEA connections. You can get the SIC names from the object properties of your LEA client object, respectively the Management Station object (see above for details about Client DN and Server DN).

```
lea_server ip 10.1.1.1
lea_server auth_port 18184
lea_server auth_type sslca
opsec_sslca_file opsec.p12
opsec_sic_name "CN=myleaclient,O=cpmodule..gysidy"
lea_server opsec_entity_sic_name "cn=cp_mgmt,o=cpmodule..gysidy"
```

Get the tool `opsec_pull_cert` either from `opsec-tools.tar.gz` from the project home page or directly from the OPSEC SDK. This tool is needed to establish the Secure Internal Communication (SIC) between FW1-LogGrabber and the FW1 management server.

Get the clients certificate from the management station (e.g. `10.1.1.1`). The activation key has to be the same as specified before in the firewall policy. After that, copy the resulting PKCS#12 file (default name `opsec.p12`) to your FW1-LogGrabber directory.

```
opsec_pull_cert -h 10.1.1.1 -n myleaclient -p def456
```

Authenticated SSL OPSEC connections

Checkpoint Device Configuration for SSL OPSEC

Modify `$FWDIR/conf/fwopsec.conf` and define the port to be used for authenticated LEA connections (e.g. 18184):

```
lea_server port 0
lea_server auth_port 18184
lea_server auth_type ssl_opsec
```

Restart in order to activate changes:

```
cpstop; cpstart
```

Set a password (e.g. `abc123`) for the LEA client (e.g. `10.1.1.2`):

```
fw putkey -ssl -p abc123 10.1.1.2
```

Create a new OPSEC Application Object with the following details:

```
Name: e.g. myleaclient
Vendor: User Defined
Server Entities: None
Client Entities: LEA
```

Initialize Secure Internal Communication (SIC) for recently created OPSEC Application Object and enter (and remember) the activation key (e.g. `def456`).

Write down the DN of the recently created OPSEC Application Object; this is your Client Distinguished Name, which you need later on.

Open the object of your FW1 management server and write down the DN of that object; this is the Server Distinguished Name, which you will need later on.

Add a rule to the policy to allow the port defined above as well as port 18210/tcp (FW1_ica_pull) in order to allow pulling of PKCS#12 certificate from the FW1-LogGrabber machine to the FW1 management server. The port 18210/tcp can be shut down after the communication between FW1-LogGrabber and the FW1 management server has been established successfully.

Finally, install the policy.

FW1-LogGrabber Configuration for SSL OPSEC

Modify `$LOGGRABBER_CONFIG_PATH/lea.conf` and define the IP address of your FW1 management station (e.g. `10.1.1.1`) as well as port (e.g. `18184`), authentication type and SIC names for authenticated LEA connections. The SIC names you can get from the object properties of your LEA client object respectively the Management Station object (see above for details about Client DN and Server DN).

```
lea_server ip 10.1.1.1
lea_server auth_port 18184
lea_server auth_type ssl_opsec
opsec_sslca_file opsec.p12
opsec_sic_name "CN=myleaclient,O=cpmodule..gysidy"
lea_server opsec_entity_sic_name "cn=cp_mgmt,o=cpmodule..gysidy"
```

Set password for the connection to the LEA server. The password has to be the same as specified on the LEA server.

```
opsec_putkey -ssl -p abc123 10.1.1.1
```

Get the tool `opsec_pull_cert` either from `opsec-tools.tar.gz` from the project home page or directly from the OPSEC SDK. This tool is needed to establish the Secure Internal Communication (SIC) between FW1-LogGrabber and the FW1 management server.

Get the clients certificate from the management station (e.g. `10.1.1.1`). The activation key has to be the same as specified before in the firewall policy.

```
opsec_pull_cert -h 10.1.1.1 -n myleaclient -p def456
```

Authenticated OPSEC connections

Checkpoint Device Configuration for Auth OPSEC

Modify `$FWDIR/conf/fwopsec.conf` and define the port to be used for authenticated LEA connections (e.g. 18184):

```
lea_server port 0
lea_server auth_port 18184
lea_server auth_type auth_opsec
```

Restart in order to activate changes

```
fwstop; fwstart
```

Set a password (e.g. `abc123`) for the LEA client (e.g. `10.1.1.2`).

```
fw putkey -opsec -p abc123 10.1.1.2
```

Add a rule to the policy to allow the port defined above from the FW1-LogGrabber machine to the FW1 management server.

Finally, install the policy.

FW1-LogGrabber Configuration for Auth OPSEC

Modify `$LOGGRABBER_CONFIG_PATH/lea.conf` and define the IP address of your FW1 management station (e.g. `10.1.1.1`) as well as the port (e.g. 18184) and authentication type for authenticated LEA connections:

```
lea_server ip 10.1.1.1
lea_server auth_port 18184
lea_server auth_type auth_opsec
```

Set password for the connection to the LEA server. The password has to be the same as specified on the LEA server.

```
opsec_putkey -p abc123 10.1.1.1
```

Unauthenticated connections

Checkpoint Device Configuration for Unauthenticated Connections

Modify `$FWDIR/conf/fwopsec.conf` and define the port to be used for unauthenticated LEA connections (e.g. 50001):

```
lea_server port 50001
lea_server auth_port 0
```

Restart in order to activate changes:

```
fwstop; fwstart  # for 4.1
cpstop; cpstart  # for NG
```

Add a rule to the policy to allow the port defined above from the FW1-LogGrabber machine to the FW1 management server.

Finally, install the policy.

FW1-LogGrabber Configuration for Unauthenticated Connections

Modify `$LOGGRABBER_CONFIG_PATH/lea.conf` and define the IP address of your FW1 management station (e.g. `10.1.1.1`) and port (e.g. `50001`) for unauthenticated LEA connections:

```
lea_server ip 10.1.1.1
lea_server port 50001
```

Input SDEE

> **Note**
>
> The `sdee` input plugin is not shipped with the Network Probe. Install it with `sudo -u logserver-probe /usr/share/logserver-probe/bin/logstash-plugin install logstash-input-sdee` before using this configuration.

This input plugin allows you to call a Cisco SDEE/CIDEE HTTP API, decode the output of it into event(s), and send them on their merry way. The idea behind this plugins came from a need to gather events from Cisco security devices and feed them to Energy Logserver.

SDEE Configuration

You need to import host SSL certificate in Java trust store to be able to connect to Cisco IPS device.

- Get server certificate from IPS device:

```
echo | openssl s_client -connect ciscoips:443 2>&1 | sed -ne '/-BEGIN␣
↪CERTIFICATE-/,/-END CERTIFICATE-/p' > cert.pem
```

- Import it into Java ca certs:

```
$JAVA_HOME/bin/keytool -keystore $JAVA_HOME/lib/security/cacerts -
↪importcert -alias ciscoips -file cert.pem
```

- Verify if import was successful:

```
$JAVA_HOME/bin/keytool -keystore $JAVA_HOME/lib/security/cacerts -list
```

- Setup the Network Probe input config with SSL connection:

```
input {
  sdee {
    interval => 60
    http => {
      truststore_password => "changeit"
      url => "https://10.0.2.1"
      auth => {
        user => "cisco"
        password => "p@ssw0rd"
      }
    }
  }
}
```

Input XML

To download xml files via Network Probe use input “file”, and set the location of the files in the configuration file:

```
file {
       path => [ "/path/to/files/*.xml" ]
       mode => "read"
  }
```

The XML filter takes a field that contains XML and expands it into an actual datastructure.

```
filter {
      xml {
        source => "message"
      }
    }
```

Input WMI

> **Note**
>
> The `wmi` input plugin is not shipped with the Network Probe. Install it with `sudo -u logserver-probe /usr/share/logserver-probe/bin/logstash-plugin install logstash-input-wmi` before using this configuration. For an agentless Windows path that works without an extra plugin, see *Event Collector*.

The input **wmi** allows collecting data from WMI query. This is useful for collecting performance metrics and other data which is accessible via WMI on a Windows host.

WMI Configuration

Configuration example:

```
input {
      wmi {
        query => "select * from Win32_Process"
        interval => 10
      }
      wmi {
        query => "select PercentProcessorTime from Win32_PerfFormattedData_
↪PerfOS_Processor where name = '_Total'"
      }
      wmi { # Connect to a remote host
        query => "select * from Win32_Process"
        host => "MyRemoteHost"
        user => "mydomain\myuser"
        password => "Password"
      }
    }
```

Filter “beats syslog”

This filter processes event data with syslog type:

```
filter {
  if [type] == "syslog" {
    grok {
      match => {
        "message" => [
          # auth: ssh|sudo|su

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}
↪\])?: %{DATA:[system][auth][ssh][event]} %{DATA:[system][auth][ssh][method]}
↪ for (invalid user )?%{DATA:[system][auth][user]} from %
↪{IPORHOST:[system][auth][ssh][ip]} port %{NUMBER:[system][auth][ssh][port]}␣
↪ssh2(: %{GREEDYDATA:[system][auth][ssh][signature]})?",

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
```

(continues on next page)

(continued from previous page)

```
↪{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}
↪\])?: %{DATA:[system][auth][ssh][event]} user %{DATA:[system][auth][user]}␣
↪from %{IPORHOST:[system][auth][ssh][ip]}",

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}
↪\])?: Did not receive identification string from %
↪{IPORHOST:[system][auth][ssh][dropped_ip]}",

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} sudo(?:\[%{POSINT:[system][auth][pid]}
↪\])?: \s*%{DATA:[system][auth][user]} :( %{DATA:[system][auth][sudo][error]}
↪ ;)? TTY=%{DATA:[system][auth][sudo][tty]} ; PWD=%
↪{DATA:[system][auth][sudo][pwd]} ; USER=%{DATA:[system][auth][sudo][user]} ;
↪ COMMAND=%{GREEDYDATA:[system][auth][sudo][command]}",

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} %{DATA:[system][auth][program]}(?:\[%
↪{POSINT:[system][auth][pid]}\])?: %{GREEDYMULTILINE:[system][auth][message]}
↪",

          # add/remove user or group
          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} groupadd(?:\[%
↪{POSINT:[system][auth][pid]}\])?: new group: name=%{DATA:system.auth.
↪groupadd.name}, GID=%{NUMBER:system.auth.groupadd.gid}",

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} userdel(?:\[%
↪{POSINT:[system][auth][pid]}\])?: removed group '%
↪{DATA:[system][auth][groupdel][name]}' owned by '%
↪{DATA:[system][auth][group][owner]}'",

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} useradd(?:\[%
↪{POSINT:[system][auth][pid]}\])?: new user: name=%
↪{DATA:[system][auth][user][add][name]}, UID=%
↪{NUMBER:[system][auth][user][add][uid]}, GID=%
↪{NUMBER:[system][auth][user][add][gid]}, home=%
↪{DATA:[system][auth][user][add][home]}, shell=%
↪{DATA:[system][auth][user][add][shell]}$",

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} userdel(?:\[%
↪{POSINT:[system][auth][pid]}\])?: delete user '%
↪{WORD:[system][auth][user][del][name]}'$",

          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{SYSLOGHOST:[system][auth][hostname]} usermod(?:\[%
↪{POSINT:[system][auth][pid]}\])?: add '%{WORD:[system][auth][user][name]}'␣
↪to group '%{WORD:[system][auth][user][memberof]}'",
```

(continues on next page)

(continued from previous page)

```
          # yum install/erase/update package
          "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %
↪{DATA:[system][package][action]}: %{NOTSPACE:[system][package][name]}"
        ]
      }
      pattern_definitions => {
        "GREEDYMULTILINE"=> "(.|\n)*"
      }
    }

    date {
      match => [ "[system][auth][timestamp]", "MMM  d HH:mm:ss", "MMM dd␣
↪HH:mm:ss" ]
      target => "[system][auth][timestamp]"
    }

    mutate {
      convert => { "[system][auth][pid]" => "integer" }
      convert => { "[system][auth][groupadd][gid]" => "integer" }
      convert => { "[system][auth][user][add][uid]" => "integer" }
      convert => { "[system][auth][user][add][gid]" => "integer" }
    }
  }
}
```

Filter "network"

This filter processes event data with network type:

```
filter {
  if [type] == "network" {
    grok {
      named_captures_only => true
      match => {
        "message" => [

          # Cisco Firewall
          "%{SYSLOG5424PRI}%{NUMBER:log_sequence#}:%{SPACE}%{IPORHOST:device_
↪ip}: (?:.)?%{CISCOTIMESTAMP:log_data} CET: %%{CISCO_REASON:facility}-%
↪{INT:severity_level}-%{CISCO_REASON:facility_mnemonic}:%{SPACE}%
↪{GREEDYDATA:event_message}",

          # Cisco Routers
          "%{SYSLOG5424PRI}%{NUMBER:log_sequence#}:%{SPACE}%{IPORHOST:device_
↪ip}: (?:.)?%{CISCOTIMESTAMP:log_data} CET: %%{CISCO_REASON:facility}-%
↪{INT:severity_level}-%{CISCO_REASON:facility_mnemonic}:%{SPACE}%
↪{GREEDYDATA:event_message}",

          # Cisco Switches
          "%{SYSLOG5424PRI}%{NUMBER:log_sequence#}:%{SPACE}%{IPORHOST:device_
↪ip}: (?:.)?%{CISCOTIMESTAMP:log_data} CET: %%{CISCO_REASON:facility}-%
↪{INT:severity_level}-%{CISCO_REASON:facility_mnemonic}:%{SPACE}%
```

(continues on next page)

(continued from previous page)

```
↪{GREEDYDATA:event_message}",
          "%{SYSLOG5424PRI}%{NUMBER:log_sequence#}:%{SPACE}(?:.)?%
↪{CISCOTIMESTAMP:log_data} CET: %%{CISCO_REASON:facility}-%{INT:severity_
↪level}-%{CISCO_REASON:facility_mnemonic}:%{SPACE}%{GREEDYDATA:event_message}
↪",

          # HP switches
          "%{SYSLOG5424PRI}%{SPACE}%{CISCOTIMESTAMP:log_data} %
↪{IPORHOST:device_ip} %{CISCO_REASON:facility}:%{SPACE}%{GREEDYDATA:event_
↪message}"
        ]

      }
    }

    syslog_pri { }

    if [severity_level] {
      translate {
        dictionary_path => "/etc/logserver-probe/dictionaries/cisco_syslog_
↪severity.yml"
        field => "severity_level"
        destination => "severity_level_descr"
      }
    }

    if [facility] {
      translate {
        dictionary_path => "/etc/logserver-probe/dictionaries/cisco_syslog_
↪facility.yml"
        field => "facility"
        destination => "facility_full_descr"
      }
    }

    #ACL
    if [event_message] =~ /(\d+.\d+.\d+.\d+)/ {
      grok {
        match => {
          "event_message" => [
            "list %{NOTSPACE:[acl][name]} %{WORD:[acl][action]} %
↪{WORD:[acl][proto]} %{IP:[src][ip]}.*%{IP:[dst][ip]}",
            "list %{NOTSPACE:[acl][name]} %{WORD:[acl][action]} %
↪{IP:[src][ip]}",
            "^list %{NOTSPACE:[acl][name]} %{WORD:[acl][action]} %
↪{WORD:[acl][proto]} %{IP:[src][ip]}.*%{IP:[dst][ip]}"
          ]
        }
      }
    }

    if [src][ip] {
```

(continues on next page)

(continued from previous page)

```
      cidr {
        address => [ "%{[src][ip]}" ]
        network => [ "0.0.0.0/32", "10.0.0.0/8", "172.16.0.0/12", "192.168.0.
↪0/16", "fc00::/7", "127.0.0.0/8", "::1/128", "169.254.0.0/16", "fe80::/10",
↪"224.0.0.0/4", "ff00::/8","255.255.255.255/32"  ]
        add_field => { "[src][locality]" => "private" }
      }

      if ![src][locality] {
        mutate {
          add_field => { "[src][locality]" => "public" }
        }
      }
    }


    if [dst][ip] {
      cidr {
        address => [ "%{[dst][ip]}" ]
        network => [ "0.0.0.0/32", "10.0.0.0/8", "172.16.0.0/12", "192.168.0.
↪0/16", "fc00::/7", "127.0.0.0/8", "::1/128", "169.254.0.0/16", "fe80::/10",
↪"224.0.0.0/4", "ff00::/8","255.255.255.255/32" ]
        add_field => { "[dst][locality]" => "private" }
      }

      if ![dst][locality] {
        mutate {
          add_field => { "[dst][locality]" => "public" }
        }
      }
    }

    # date format
    date {
      match => [ "log_data", "MMM dd HH:mm:ss", "MMM  dd HH:mm:ss","MMM dd␣
↪HH:mm:ss.SSS", "MMM  dd HH:mm:ss.SSS", "ISO8601" ]
      target => "log_data"
    }

  }
}
```

Filter “geoip”

This filter processes event data with IP address and check localization:

```
filter {
  if [src][locality] == "public" {

    geoip {
      source => "[src][ip]"
```

(continues on next page)

(continued from previous page)

```
      target => "[src][geoip]"
      database => "/etc/logserver-probe/geoipdb/GeoLite2-City.mmdb"
      fields => [ "city_name", "country_name", "continent_code", "country_
↪code2", "location" ]
      remove_field => [ "[src][geoip][ip]" ]
    }

    geoip {
      source => "[src][ip]"
      target => "[src][geoip]"
      database => "/etc/logserver-probe/geoipdb/GeoLite2-ASN.mmdb"
      remove_field => [ "[src][geoip][ip]" ]
    }

  }

  if [dst][locality] == "public" {

    geoip {
      source => "[dst][ip]"
      target => "[dst][geoip]"
      database => "/etc/logserver-probe/geoipdb/GeoLite2-City.mmdb"
      fields => [ "city_name", "country_name", "continent_code", "country_
↪code2", "location" ]
      remove_field =>  [ "[dst][geoip][ip]" ]
    }

    geoip {
      source => "[dst][ip]"
      target => "[dst][geoip]"
      database => "/etc/logserver-probe/geoipdb/GeoLite2-ASN.mmdb"
      remove_field => [ "[dst][geoip][ip]" ]
    }
  }

}
```

Avoiding duplicate documents

To avoid duplicating the same documents, e.g. if the collector receives the entire event log file on restart, prepare the Network Probe filter as follows:

1. Use the **fingerprint** filter to create consistent hashes of one or more fields whose values are unique for the document and store the result in a new field, for example:

```
fingerprint {
                        source => [ "log_name", "record_number" ]
                        target => "generated_id"
                        method => "SHA1"
                }
```

   - source - The name(s) of the source field(s) whose contents will be used to create the fingerprint

- target - The name of the field where the generated fingerprint will be stored. Any current contents of that field will be overwritten.
- method - If set to `SHA1`, `SHA256`, `SHA384`, `SHA512`, or `MD5` and a key is set, the cryptographic hash function with the same name will be used to generate the fingerprint. When a key set, the keyed-hash (HMAC) digest function will be used.

2. In the **logserver** output set the **document_id** as the value of the **generated_id** field:

```
logserver {
                hosts => ["http://localhost:9200"]
                user => "user"
                password => "password"
                index => "syslog_wec-%{+YYYY.MM.dd}"
                document_id => "%{generated_id}"
        }
```

- document_id - The document ID for the index.

Documents having the same document_id will be indexed only once.

Data enrichment

It is possible to enrich the events that go to the filters with additional fields, the values of which come from the following sources:

- databases, using the `jdbc` plugin;
- Active Directory or OpenLdap, using plugin;
- dictionary files, using the `translate` plugin;
- external systems using their API, e.g. OP5 Monitor/Nagios

Filter `jdbc`

This filter executes a SQL query and store the result set in the field specified as `target`. It will cache the results locally in an LRU cache with expiry.

For example, you can load a row based on an id in the event:

```
filter {
  jdbc_streaming {
    jdbc_driver_library => "/path/to/mysql-connector-java-5.1.34-bin.jar"
    jdbc_driver_class => "com.mysql.jdbc.Driver"
    jdbc_connection_string => "jdbc:mysql://localhost:3306/mydatabase"
    jdbc_user => "me"
    jdbc_password => "secret"
    statement => "select * from WORLD.COUNTRY WHERE Code = :code"
    parameters => { "code" => "country_code"}
    target => "country_details"
  }
}
```

Filter ldap

Configuration

The **ldap** filter will add fields queried from a ldap server to the event. The fields will be stored in a variable called **target**, that you can modify in the configuration file.

If an error occurs during the process the **tags** array of the event is updated with either:

- **LDAP_ERROR** tag: Problem while connecting to the server: bad *host, port, username, password, or search_dn* -> Check the error message and your configuration.
- **LDAP_NOT_FOUND** tag: Object wasn't found.

If error logging is enabled a field called **error** will also be added to the event. It will contain more details about the problem.

Input event

```
{
    "@timestamp" => 2018-02-25T10:04:22.338Z,
    "@version" => "1",
    "myUid" => "u501565"
}
```

Filter

```
filter {
  ldap {
    identifier_value => "%{myUid}"
    host => "my_ldap_server.com"
    ldap_port => "389"
    username => "<connect_username>"
    password => "<connect_password>"
    search_dn => "<user_search_pattern>"
  }
}
```

Output event

```
{
    "@timestamp" => 2018-02-25T10:04:22.338Z,
    "@version" => "1",
    "myUid" => "u501565",
    "ldap" => {
        "givenName" => "VALENTIN",
        "sn" => "BOURDIER"
    }
}
```

Parameters available

Here is a list of all parameters, with their default value, if any, and their description.

Option name	Type	Required	Default value	Description	Example
identifier_value	string	yes	n/a	Identifier of the value to search. If identifier type is uid, then the value should be the uid to search for.	“123456”
identifier_key	string	no	“uid”	Type of the identifier to search	“uid”
identifier_type	string	no	“posixAccount”	Object class of the object to search	“person”
search_dn	string	yes	n/a	Domain name in which search inside the ldap database (usually your userdn or groupdn)	“dc=example,dc=org”
attributes	array	no	[]	List of attributes to get. If not set, all attributes available will be get	[‘givenName’, ‘sn’]
target	string	no	“ldap”	Name of the variable you want the result being stocked in	“myCustomVariableName”
host	string	yes	n/a	LDAP server host adress	“ldapserveur.com”
ldap_port	number	no	389	LDAP server port for non-ssl connection	400
ldaps_port	number	no	636	LDAP server port for ssl connection	401
use_ssl	boole	no	false	Enable or not ssl connection for LDAP server. Set-up the good ldap(s)_port depending on that	true
enable_error_lo	boole	no	false	When there is a problem with the connection with the LDAP database, write reason in the event	true
no_tag_on_fa	boole	no	false	No tags are added when an error (wrong credentials, bad server, . . .) occur	true
username	string	no	n/a	Username to use for search in the database	“cn=SearchUser,ou=person,o=domain”
password	string	no	n/a	Password of the account linked to previous username	“123456”
use_cache	boole	no	true	Choose to enable or not use of buffer	false
cache_type	string	no	“memory”	Type of buffer to use. Currently, only one is available, “memory” buffer	“memory”
cache_memo	number	no	300	Cache duration (in s) before refreshing values of it	3600
cache_memo	number	no	20000	Number of object max that the buffer can contains	100
disk_cache_fi	string	no	nil	Where the cache will periodically be dumped	“/tmp/my-memory-backup”
disk_cache_s	string	no	10m	Cron period of when the dump of the cache should occured. See here for the syntax.	“10m”, “1h”, “every day at five”, “3h10m”

Buffer

Like all filters, this filter treat only 1 event at a time. This can lead to some slowing down of the pipeline speed due to the network round-trip time, and high network I/O.

A buffer can be set to mitigate this.

Currently, there is only one basic **"memory"** buffer.

You can enable / disable use of buffer with the option **use_cache**.

Memory Buffer

This buffer **store** data fetched from the LDAP server **in RAM**, and can be configured with two parameters:

- *cache_memory_duration*: duration (in s) before a cache entry is refreshed if hit.
- *cache_memory_size*: number of tuple (identifier, attributes) that the buffer can contains.

Older cache values than your TTL will be removed from cache.

Persistent cache buffer

For the only buffer for now, you will be able to save it to disk periodically.

Some specificities :

- for *the memory cache*, TTL will be reset

Two parameters are required:

- *disk_cache_filepath*: path on disk of this backup
- *disk_cache_schedule*: schedule (every X time unit) of this backup. Please check here for the syntax of this parameter.

Filter `translate`

A general search and replace tool that uses a configured hash and/or a file to determine replacement values. Currently supported are YAML, JSON, and CSV files. Each dictionary item is a key value pair.

You can specify dictionary entries in one of two ways:

- The dictionary configuration item can contain a hash representing the mapping.

```
filter {
      translate {
        field => "[http_status]"
        destination => "[http_status_description]"
        dictionary => {
          "100" => "Continue"
          "101" => "Switching Protocols"
          "200" => "OK"
          "500" => "Server Error"
        }
        fallback => "I'm a teapot"
      }
    }
```

- An external file (readable by Network Probe) may be specified in the `dictionary_path` configuration item:

```
filter {
  translate {
    dictionary_path => "/etc/logserver-probe/lists/instance_cpu.yml"
    field => "InstanceType"
    destination => "InstanceCPUCount"
```

(continues on next page)

(continued from previous page)

```
    refresh_behaviour => "replace"
  }
}
```

Sample dictionary file:

```
"c4.4xlarge": "16"
"c5.xlarge": "4"
"m1.medium": "1"
"m3.large": "2"
"m3.medium": "1"
"m4.2xlarge": "8"
"m4.large": "2"
"m4.xlarge": "4"
"m5a.xlarge": "4"
"m5d.xlarge": "4"
"m5.large": "2"
"m5.xlarge": "4"
"r3.2xlarge": "8"
"r3.xlarge": "4"
"r4.xlarge": "4"
"r5.2xlarge": "8"
"r5.xlarge": "4"
"t2.large": "2"
"t2.medium": "2"
"t2.micro": "1"
"t2.nano": "1"
"t2.small": "1"
"t2.xlarge": "4"
"t3.medium": "2"
```

External API

A simple filter that checks if an IP (from **PublicIpAddress** field) address exists in an external system. The result is written to the **op5exists** field. Then, using a grok filter, the number of occurrences is decoded and put into the **op5count** field.

```
ruby {
  code => '
    checkip = event.get("PublicIpAddress")
    output=`curl -s -k -u monitor:monitor "https://192.168.1.1/api/filter/
↪count?query=%5Bhosts%5D%28address%20~~%20%22# {checkip}%22%20%29" 2>&1`
    event.set("op5exists", "#{output}")
  '
}
grok {
  match => { "op5exists" => [ ".*\:%{NUMBER:op5count}" ] }
}
```

Mathematical calculations

Using filters, you can perform mathematical calculations for field values and save the results to a new field.

Application example:

```
filter {
   ruby { code => 'event.set("someField", event.get("field1") + event.get(
↪"field2"))' }
}
```

Output to Data Node

This output plugin sends all data to the local Data Node instance and creates indexes:

```
output {
  logserver {
    hosts => [ "127.0.0.1:9200" ]
    index => "%{type}-%{+YYYY.MM.dd}"
    user => "user"
    password => "password"
  }
}
```

naemon beat

This plugin has example of complete configuration for integration with *naemon* application:

```
input {
    beats {
        port => FILEBEAT_PORT
        type => "naemon"
    }
}

filter {
    if [type] == "naemon" {
        grok {
            patterns_dir => [ "/etc/logserver-probe/patterns.d" ]
            match => { "message" => "%{NAEMONLOGLINE}" }
            remove_field => [ "message" ]
        }
        date {
            match => [ "naemon_epoch", "UNIX" ]
            target => "@timestamp"
            remove_field => [ "naemon_epoch" ]
        }
    }
}

output {
  # Single index
#    if [type] == "naemon" {
#        logserver {
```

(continues on next page)

(continued from previous page)

```
#             hosts => ["logserver-ip:logserver-port"]
#             index => "naemon-%{+YYYY.MM.dd}"
#             user => "user"
#             password => "password"
#         }
#     }

  # Separate indexes
  if [type] == "naemon" {
    if "_grokparsefailure" in [tags] {
      logserver {
        hosts => ["logserver-ip:logserver-port"]
        index => "naemongrokfailure"
                user => "user"
        password => "password"
      }
    }
    else {
      logserver {
        hosts => ["logserver-ip:logserver-port"]
        index => "naemon-%{+YYYY.MM.dd}"
        user => "user"
        password => "password"
      }
    }
  }
}
```

perflog

This plugin has an example of a complete configuration for integration with perflog:

```
input {
  tcp {
    port => 6868
    host => "0.0.0.0"
    type => "perflogs"
  }
}

filter {
  if [type] == "perflogs" {
    grok {
      break_on_match => "true"
      match => {
        "message" => [
          "DATATYPE::%{WORD:datatype}\tTIMET::%{NUMBER:timestamp}\tHOSTNAME::%
↪{DATA:hostname}\tSERVICEDESC::%{DATA:servicedescription}\tSERVICEPERFDATA::%
↪{DATA:performance}\tSERVICECHECKCOMMAND::.*?HOSTSTATE::%{WORD:hoststate}\
↪tHOSTSTATETYPE::.*?SERVICESTATE::%{WORD:servicestate}\tSERVICESTATETYPE::%
↪{WORD:servicestatetype}",
          "DATATYPE::%{WORD:datatype}\tTIMET::%{NUMBER:timestamp}\tHOSTNAME::%
```

(continues on next page)

(continued from previous page)

```
↪{DATA:hostname}\tHOSTPERFDATA::%{DATA:performance}\tHOSTCHECKCOMMAND::.*?
↪HOSTSTATE::%{WORD:hoststate}\tHOSTSTATETYPE::%{WORD:hoststatetype}"
        ]
      }
      remove_field => [ "message" ]
    }
    kv {
      source => "performance"
      field_split => "\t"
      remove_char_key => "\.\'"
      trim_key => " "
      target => "perf_data"
      remove_field => [ "performance" ]
      allow_duplicate_values => "false"
      transform_key => "lowercase"
    }
    date {
      match => [ "timestamp", "UNIX" ]
      target => "@timestamp"
      remove_field => [ "timestamp" ]
    }
  }
}

output {
  if [type] == "perflogs" {
    logserver {
      hosts => ["127.0.0.1:9200"]
      index => "perflogs-%{+YYYY.MM.dd}"
          user => "user"
      password => "password"
    }
  }
}
```

LDAP data enrichment

```
```bash
filter {
  ldap {
    identifier_value => "%{[winlog][event_data][TargetUserName]}"
    identifier_key => "sAMAccountName"
    identifier_type => "person"
    host => "10.0.0.1"
    ldap_port => "389"
    username => "user"
    password => "pass"
    search_dn => "OU=example,DC=example"
    enable_error_logging => true
    attributes => ['sAMAccountType','lastLogon','badPasswordTime']
  }
```

(continues on next page)
```

(continued from previous page)

```
}
```
```

6. Fields description

```
identifier_value - Identifier of the value to search. If identifier type␣
↪is uid, then the value should be the uid to search for.
identifier_key - Type of the identifier to search.
identifier_type - Object class of the object to search.
host - LDAP server host adress.
ldap_port - LDAP server port for non-ssl connection.
username - Username to use for search in the database.
password - Password of the account linked to previous username.
search_dn - Domain name in which search inside the ldap database (usually␣
↪your userdn or groupdn).
enable_error_logging - When there is a problem with the connection with␣
↪the LDAP database, write reason in the event.
attributes - List of attributes to get. If not set, all attributes␣
↪available will be get.
```

Multiline codec

The original goal of this codec was to allow joining of multiline messages from files into a single event. For example, joining Java exception and stacktrace messages into a single event.

```
input {
  stdin {
    codec => multiline {
      pattern => "pattern, a regexp"
      negate => "true" or "false"
      what => "previous" or "next"
    }
  }
}
```

```
input {
  file {
    path => "/var/log/someapp.log"
    codec => multiline {
      # Grok pattern names are valid! :)
      pattern => "^%{TIMESTAMP_ISO8601} "
      negate => true
      what => "previous"
    }
  }
}
```
```

Known issues and troubleshooting

Netflow: missing `[netflow][exporter]` field

Symptom: Documents in the `stream-*` index contain the literal string `%{peer_ip_src}` in `[netflow][exporter]` instead of the exporter IP address.

Cause: The shipped `/etc/pmacct/nfacctd-*.conf` uses the short `aggregate:` line which does not include `peer_src_ip`. The pmacct plugin therefore does not emit the `peer_ip_src` field to Kafka, and the Network Probe filter (`20-netflow_filter-pmacct.conf`) falls back to the literal placeholder.

Fix (two steps):

1. Add `peer_src_ip` to the `aggregate:` line in each instance config under `/etc/pmacct/`:

   ```
   aggregate: src_host, dst_host, src_port, dst_port, proto, tos, tcpflags,␣
   ↪src_mac, dst_mac, peer_src_ip
   ```

   Restart the instance: `systemctl restart nfacctd@instance1`.

2. The Kafka `netflow` topic still contains messages in the old schema. Delete the topic so Kafka recreates it with the new schema on the next flush:

   ```
   /opt/kafka/bin/kafka-topics.sh --bootstrap-server localhost:9092 \
     --delete --topic netflow
   ```

Verify the field is now emitted by reading the Kafka topic directly:

```
/opt/kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 \
  --topic netflow
```

Messages should include `"peer_ip_src": "<exporter_ip>"`. After the next indexing cycle the `[netflow][exporter]` field in `stream-*` shows the exporter IP address.

Custom Integrations: Parsers and Pipelines

Built-in *Integrations* cover common vendors with one-click pipelines, dashboards, and alerts. When a device or application is not on that list (a niche appliance, an in-house system, or hardware that emits logs in its own format), you connect it by building a **custom pipeline**: an input that receives the data, a **parser** that turns each raw line into structured fields, and an output that writes the result to the Data Node.

This chapter walks through that process end to end on a single example: a custom syslog device. The same three stages apply to any source; only the input and the parser change.

- *How data flows through a probe*
- *Before you start*
- *Worked example: onboarding a custom syslog device*
  - *1. Receive the logs (input)*
  - *2. Write the parser (grok filter)*
  - *3. Test the parser before deploying*
  - *4. Send to the Data Node (output)*
  - *5. Assemble and deploy the pipeline*

  - *6. Verify*
- *Common patterns and pitfalls*
- *Troubleshooting*

How data flows through a probe

A Network Probe pipeline is a Logstash configuration with three stages:

Stage	Purpose	Reference
input	Receives raw events (syslog, files, HTTP, database, Kafka, and more) and tags them with a `type`.	*Technical Configuration*, the `Input *` sections
filter (parser)	Matches the raw `message` against patterns and extracts named fields. This is the parser.	*Filter "network"*, *Filter "beats syslog"*
output	Sends the structured event to the Data Node and creates the index.	*Output to Data Node*

Pipeline configuration files live on the probe in `/etc/logserver-probe/conf.d/` and use the `.conf` extension. The probe runs them through its bundled Logstash engine at `/usr/share/logserver-probe/bin/logstash`. Built-in integrations keep their files in per-vendor subdirectories and split the stages across numbered files (for example `01.input.conf`, `20.filter.conf`, `30.output.conf`) so that they load in order. A custom pipeline can follow the same convention or use a single combined `.conf` file.

Before you start

1. Check the *Integrations* catalog first. If your device is listed, use One Click Installation instead of building a pipeline by hand. If it is not, you can also let the AI Assistant build the integration from sample logs, see *Creating Integrations with the AI Assistant*.
2. Confirm the probe can receive the data (network path, firewall, credentials).
3. Capture a **sample of the raw logs** exactly as the device sends them. You cannot write a parser without a representative line.

Worked example: onboarding a custom syslog device

The example device is a firewall appliance (`acme-fw01`) that sends syslog in a custom key-value format that no built-in integration recognises:

```
<134>Jul  1 10:15:32 acme-fw01 ACME: user=jsmith action=BLOCK src=10.0.0.5␣
↪dst=8.8.8.8 proto=TCP
```

The goal: receive these lines, extract the fields (`user`, `action`, `src_ip`, `dst_ip`, `proto`), and index them so they are searchable in Discover.

1. Receive the logs (input)

Use the `tcp` and `udp` inputs to listen for syslog, and tag the events with a `type` that identifies this device. The `type` is what the parser and output key off later.

```
input {
  tcp { port => 5514 type => "acme" }
```

(continues on next page)

(continued from previous page)

```
  udp { port => 5514 type => "acme" }
}
```

If the device sends to the standard syslog port 514, redirect it to the collector port with the firewall rules shown in *Input "network"*.

2. Write the parser (grok filter)

The parser is a `grok` filter guarded by `if [type] == "acme"`, so it only runs on events from this device. Each `%{PATTERN:field}` extracts one value into a named field:

```
filter {
  if [type] == "acme" {
    grok {
      match => {
        "message" => "%{SYSLOG5424PRI}%{SYSLOGTIMESTAMP:timestamp} %
↪{SYSLOGHOST:device} ACME: user=%{USERNAME:user} action=%{WORD:action} src=%
↪{IPORHOST:src_ip} dst=%{IPORHOST:dst_ip} proto=%{WORD:proto}"
      }
    }
    date {
      match  => [ "timestamp", "MMM  d HH:mm:ss", "MMM dd HH:mm:ss" ]
      target => "@timestamp"
    }
  }
}
```

The pattern names used here (`SYSLOGTIMESTAMP`, `SYSLOGHOST`, `IPORHOST`, `WORD`, `USERNAME`) are standard grok patterns. For more complex formats you can define your own with `pattern_definitions`, set `named_captures_only => true`, or map values through a dictionary. See the worked filters in *Filter "beats syslog"* and *Filter "network"*.

3. Test the parser before deploying

There is no separate grok-testing UI, but the probe already ships its pipeline engine, so you can test a parser against your sample line in seconds before touching any probe. Create a throwaway config that reads from the keyboard and prints the parsed result:

```
# /tmp/acme-test.conf
input  { stdin { type => "acme" } }
filter {
  grok {
    match => { "message" => "%{SYSLOG5424PRI}%{SYSLOGTIMESTAMP:timestamp} %
↪{SYSLOGHOST:device} ACME: user=%{USERNAME:user} action=%{WORD:action} src=%
↪{IPORHOST:src_ip} dst=%{IPORHOST:dst_ip} proto=%{WORD:proto}" }
  }
}
output { stdout { codec => rubydebug } }
```

Run it with a temporary data directory so it does not clash with the running probe service, paste your sample line, and press `Ctrl+D`:

```
/usr/share/logserver-probe/bin/logstash \
  -f /tmp/acme-test.conf \
  --path.data /tmp/ls-grok-test
```

If the pattern matches, the engine prints the event with the extracted fields (`user`, `action`, `src_ip`, ...). If you see a `_grokparsefailure` tag, adjust the pattern and run again. When the fields look right, validate the full configuration syntax without starting the pipeline:

```
/usr/share/logserver-probe/bin/logstash --config.test_and_exit -f /tmp/acme-
↪test.conf
```

Note

This is a local test only. It never writes to the Data Node and does not change any probe pipeline. Remove `/tmp/acme-test.conf` and `/tmp/ls-grok-test` when you are done.

4. Send to the Data Node (output)

Add the `logserver` output. `index => "%{type}-%{+YYYY.MM.dd}"` creates a daily index named after the `type`, so this device lands in `acme-*`:

```
output {
  logserver {
    hosts    => [ "127.0.0.1:9200" ]
    index    => "%{type}-%{+YYYY.MM.dd}"
    user     => "user"
    password => "password"
  }
}
```

5. Assemble and deploy the pipeline

Combine the input, filter, and output into one `.conf` file (or keep them as separate numbered files) and deploy it to the probes. There are two ways:

- **GUI: New Pipeline Creator.** In the `Pipelines Management` tab of **Network Probe**, click `New pipeline`, name it, and drag in your `.conf` file(s). Each must be `.conf` and no larger than 1 MB. Select the probes to enable it on and `Submit`. The creator rejects a file that cannot be parsed and shows the parser error. See *New Pipeline Creator*.
- **GUI: Files Section.** Create or upload the file directly under `/etc/logserver-probe/conf.d/` on a probe, then enable it. The GUI validates a `.conf` file when you save it and blocks the save on a parsing error. A file copied to the probe outside the GUI is parsed at registration and flagged with a warning icon you can click to read the error. See *Files Section*.

6. Verify

1. In *Pipelines Section*, confirm the pipeline shows a green, enabled status on each assigned probe.
2. In *Files parsing*, confirm the `.conf` has no warning icon.
3. In **Discover**, open the `acme-*` index pattern and confirm events arrive with the extracted fields populated.

Common patterns and pitfalls

- **Guard every parser with its `type`.** `if [type] == "acme"` keeps this device's grok from running on unrelated events and prevents `_grokparsefailure` noise.
- **Reuse patterns and dictionaries.** Translate raw codes to readable values with the `translate` filter and dictionaries in `/etc/logserver-probe/dictionaries/`, as shown in *Filter "network"*.
- **Multiline events.** Stack traces or multi-line records need the multiline codec. See *Multiline codec*.
- **Duplicate documents.** If events are indexed twice, review *Avoiding duplicate documents*.

Troubleshooting

- Parser produces no fields / `_grokparsefailure`: re-run the local test in step 3 against the exact failing line; the pattern does not match the real input.
- Pipeline will not start after deploy: run `--config.test_and_exit` on the deployed file, or read the warning in *Files parsing*.
- Wider probe issues: see *Known issues and troubleshooting*.

Suricata IDS

Suricata is one of the detection engines bundled with the Network Probe (see the *bundled services* table). It inspects live network traffic, matches it against a ruleset, and writes events as JSON. This chapter takes a fresh probe from a masked Suricata service to alerts visible in the Energy Logserver GUI.

- *What Suricata does*
- *What Suricata One Click deploys*
  - *How the pieces fit together*
- *Why manual host configuration is needed*
- *Requirements*
- *Step 1: Deploy the Energy Logserver components (GUI)*
- *Step 2: Unmask the Suricata service (SSH)*
- *Step 3: Identify the capture interface*
- *Step 4: Bind Suricata to the interface with a systemd override*
- *Step 5: Set HOME_NET*
- *Step 6: Download the ruleset*
- *Step 7: Test the configuration*
- *Step 8: Start Suricata*
- *Step 9: Verify that events are flowing*
- *Verify end to end with test traffic*
- *Explore in dashboards and Discover*
  - *Dashboards*
  - *Discover*

- *Troubleshooting*
  - *"certificate signature failure" during One Click*
  - *Suricata will not start*
  - *High CPU or memory use*
  - *Packet drops*
  - *No alerts*
  - *Events reach eve.json but not Discover*
- *Definition of terms*

What Suricata does

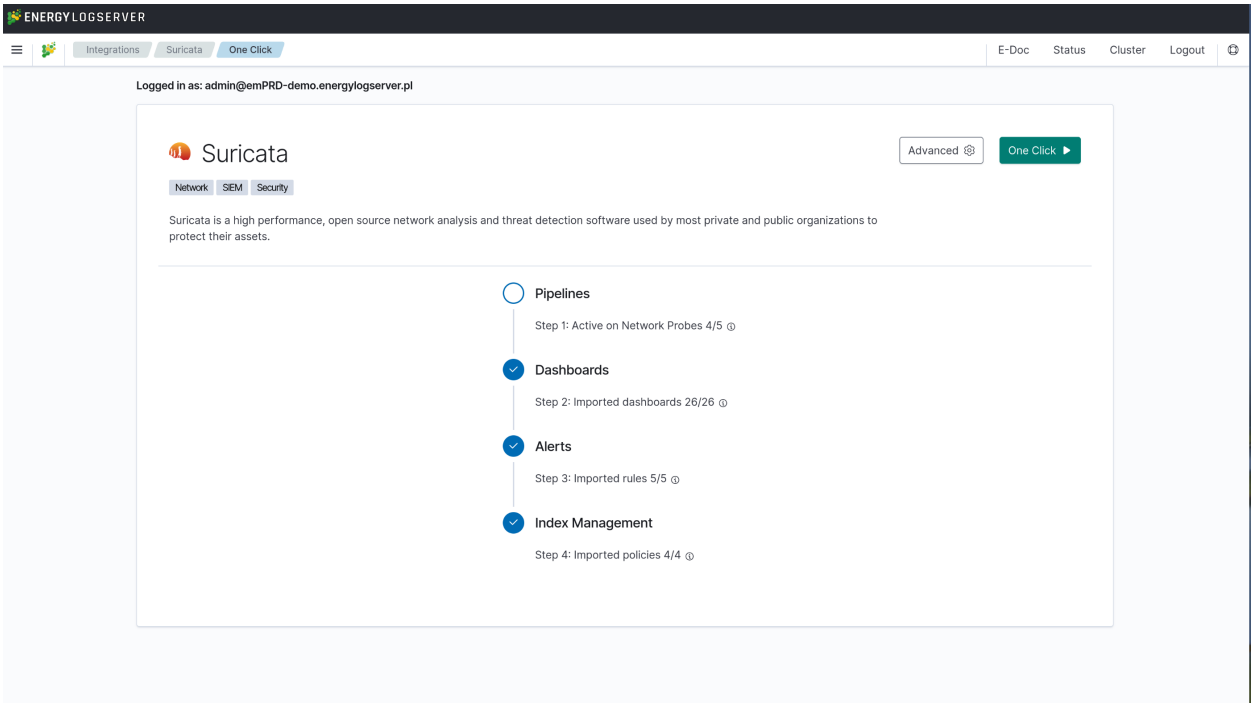
Suricata is a network threat detection engine (IDS and IPS):

- Detects attacks such as SQL injection, cross-site scripting, malware callbacks, and port scans.
- Parses application protocols: HTTP, DNS, TLS, SSH, SMB, NFS.
- Matches traffic against the Emerging Threats ruleset (tens of thousands of signatures).
- Can extract files carried in network traffic.
- Writes events in JSON format to `/var/log/suricata/eve.json`.

What Suricata One Click deploys

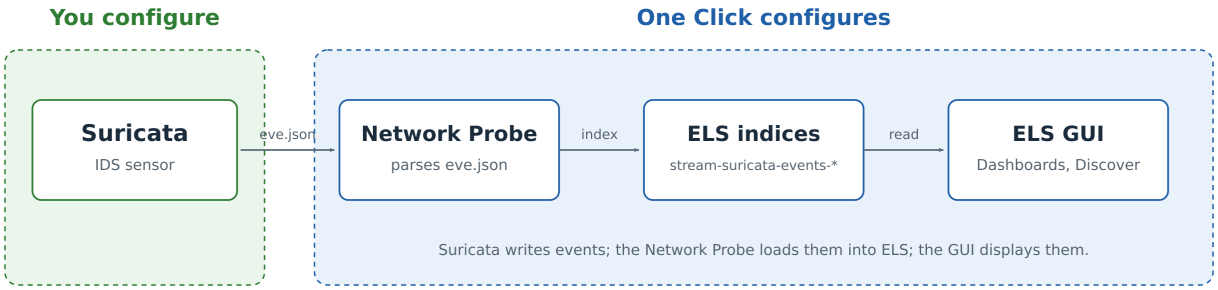
The **Suricata** entry in the *Integrations* catalog installs the Energy Logserver side of the pipeline in a single action. Open it from **Tools** → **Integrations** → **Suricata** and use *One Click Installation*. It sets up four component groups:

Component	Purpose
Pipelines	The Network Probe pipeline that parses `eve.json` and forwards the events.
Dashboards	26 Discover dashboards for alerts, flows, DNS, HTTP, TLS, SSH, SMB, NFS, and threats.
Alerts	5 predefined alert rules.
Index Management	Index lifecycle policies for the Suricata indices.



How the pieces fit together

Before configuring anything, it helps to see the whole chain. Suricata inspects network traffic and writes events to `eve.json`. The Network Probe pipeline reads that file, parses it, and writes the events into Energy Logserver indices. The GUI, through its dashboards and Discover, then reads those indices and shows them to you.



One Click sets up the middle and right of this chain: the pipeline, the indices with their lifecycle policies, and the dashboards. The steps below configure the left side, Suricata itself, because that part depends on your host and network and cannot be set up centrally.

Why manual host configuration is needed

One Click configures only the parts that are identical across every deployment. Three things stay in your hands because they are specific to the host:

- **The Suricata service is left masked.** Masking is a systemd state in which the service unit is linked to `/dev/null`, so the service cannot start until you unmask it. This prevents Suricata from running before it has a capture interface and a ruleset.

- **The capture interface is not auto-detected.** The correct interface depends on which network card actually sees the traffic you want to monitor, which no central process can know.
- **The ruleset is not downloaded.** Rulesets are a policy and licensing decision (ET Open, ET Pro, or custom sources) and are fetched on the host with `suricata-update`.

Requirements

- Root SSH access to the Energy Logserver host running the Network Probe.
- Admin access to the Energy Logserver GUI.
- Port 22 (SSH) reachable.
- Port 443 (GUI) reachable.

Step 1: Deploy the Energy Logserver components (GUI)

1. Log in to the GUI at `https://<IP-ADDRESS-OR-HOSTNAME>`.
2. Open **Tools** → **Integrations** and select the **Suricata** card.
3. Click **One Click** and wait one to two minutes for the deployment to finish.
4. Confirm that every component group reports as installed: the Pipelines, Dashboards, Alerts, and Index Management steps all show their full counts.

Step 2: Unmask the Suricata service (SSH)

Connect to the host:

```
ssh root@<IP-ADDRESS-OR-HOSTNAME>
```

Check the current state:

```
systemctl status suricata
```

Expected output:

```
o suricata.service
   Loaded: masked (Reason: Unit suricata.service is masked.)
   Active: inactive (dead)
```

Unmask the service:

```
systemctl unmask suricata
```

Expected output:

```
Removed /etc/systemd/system/suricata.service.
```

Step 3: Identify the capture interface

List the available interfaces:

```
ip link show | grep -E "^[0-9]+:"
```

Example output:

```
1: lo: <LOOPBACK,UP,LOWER_UP> ...          # loopback, do not use
2: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> ...
```

Inspect the candidate interface:

```
ip addr show <INTERFACE>
```

Choose the interface that sees the traffic you want Suricata to monitor. On a sensor with a dedicated capture port, that is the port connected to the SPAN or mirror source, not the management interface.

Step 4: Bind Suricata to the interface with a systemd override

Create an override so Suricata always starts on the chosen interface:

```
mkdir -p /etc/systemd/system/suricata.service.d

cat > /etc/systemd/system/suricata.service.d/override.conf <<'EOF'
[Service]
# Clear the unit's default ExecStart before setting a new one
ExecStart=
# Start Suricata on the chosen capture interface
ExecStart=/sbin/suricata -c /etc/suricata/suricata.yaml --pidfile /var/run/
↪suricata.pid -i <INTERFACE>
EOF
```

Reload the systemd configuration:

```
systemctl daemon-reload
```

Verify the override took effect:

```
systemctl cat suricata.service | grep "ExecStart"
```

Expected output:

```
ExecStart=/sbin/suricata -c /etc/suricata/suricata.yaml --pidfile /var/run/
↪suricata.pid -i <INTERFACE>
```

> **Note**
>
> Replace `<INTERFACE>` with the name from Step 3 (for example `ens192`). This override is the deployment-specific counterpart to the capture-interface note in *Installation*.

Step 5: Set HOME_NET

`HOME_NET` tells Suricata which address ranges are your internal network, so it can tell inbound from outbound and score direction-sensitive rules correctly.

Back up the configuration first:

```
cp /etc/suricata/suricata.yaml /etc/suricata/suricata.yaml.backup
```

Edit `/etc/suricata/suricata.yaml` and find the `vars` section near the top:

```
vars:
  address-groups:
    HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]"
```

Set HOME_NET to your own ranges, then verify:

```
grep "HOME_NET:" /etc/suricata/suricata.yaml | head -1
```

Expected output (example):

```
HOME_NET: "[10.4.9.0/24]"
```

Step 6: Download the ruleset

```
suricata-update
```

The download takes two to three minutes. Verify that the rules landed:

```
wc -l /var/lib/suricata/rules/suricata.rules
```

Expected output (example):

```
62019 /var/lib/suricata/rules/suricata.rules
```

Step 7: Test the configuration

Validate the configuration before starting the service:

```
suricata -T -c /etc/suricata/suricata.yaml
```

A successful test ends with a notice that the configuration was successfully loaded.

Enable promiscuous mode on the capture interface so it accepts frames not addressed to it:

```
ip link set <INTERFACE> promisc on
ip link show <INTERFACE> | grep PROMISC
```

Expected output (example):

```
2: ens192: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> ...
```

Step 8: Start Suricata

```
systemctl start suricata
systemctl status suricata
```

Expected output (example):

```
* suricata.service - Suricata Intrusion Detection Service
   Loaded: loaded (/usr/lib/systemd/system/suricata.service; ...)
   Active: active (running)
```

Enable start on boot:

```
systemctl enable suricata
```

Step 9: Verify that events are flowing

Check that Suricata is writing events:

```
ls -lh /var/log/suricata/eve.json
sleep 30
ls -lh /var/log/suricata/eve.json
```

The file size should grow between the two listings.

Check the event types being written:

```
tail -50 /var/log/suricata/eve.json | jq -r .event_type | sort | uniq -c
```

Expected output (example):

```
     15 flow
      8 http
      3 stats
      2 dns
      1 alert
```

Verify end to end with test traffic

These tests confirm the whole chain, from the sensor to the dashboards, by generating traffic that trips known rules.

Install a target web server:

```
dnf install httpd -y
systemctl start httpd
systemctl enable httpd
```

Generate traffic that matches Emerging Threats rules:

```
# Web scanner signature
curl -A "nikto/1.0" http://<YOUR-SERVER-IP>/

# SQL injection attempt
curl "http://<YOUR-SERVER-IP>/?id=1' OR '1'='1"
```

Count the alerts after a short wait:

```
sleep 30
grep '"event_type":"alert"' /var/log/suricata/eve.json | wc -l
```

List the alert signatures that fired:

```
grep '"event_type":"alert"' /var/log/suricata/eve.json | jq -r '.alert.
↪signature'
```

Example output:

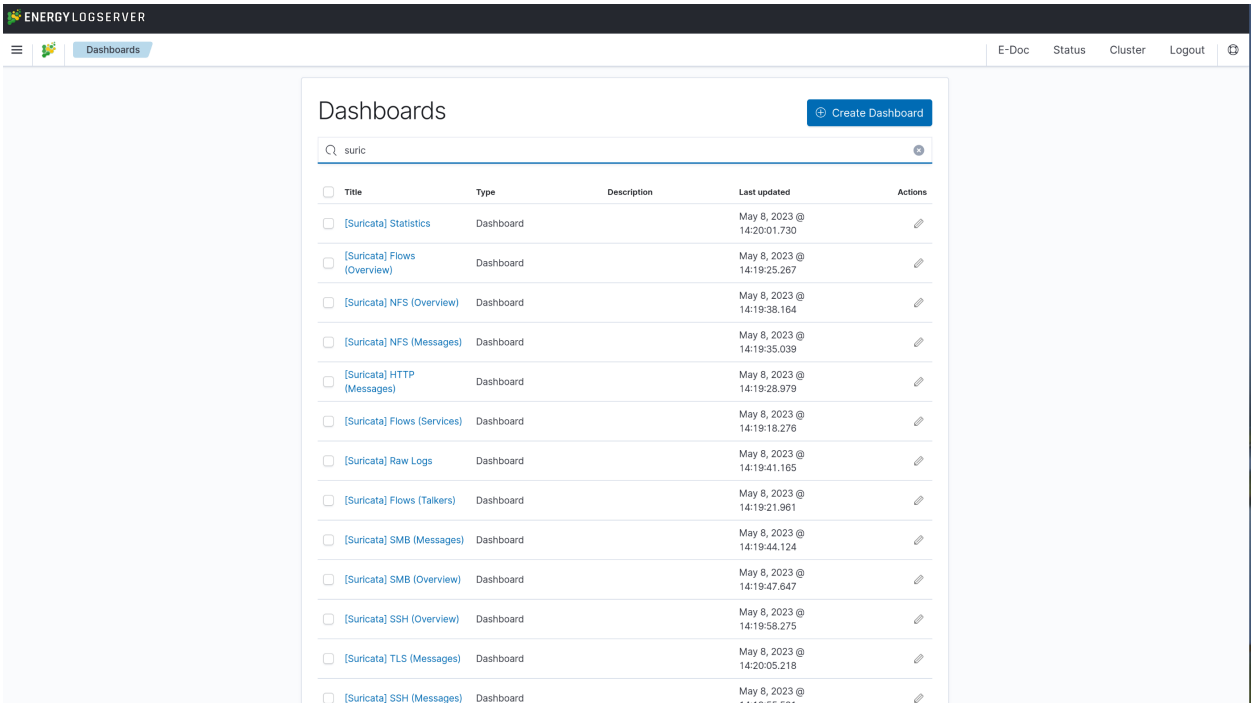
```
ET SCAN Nikto Web Scanner Detected
ET WEB_SPECIFIC_APPS SQL Injection Attempt
```

Alerts in `eve.json` confirm that the rules load and match. The next section confirms that the pipeline carries them into the GUI.

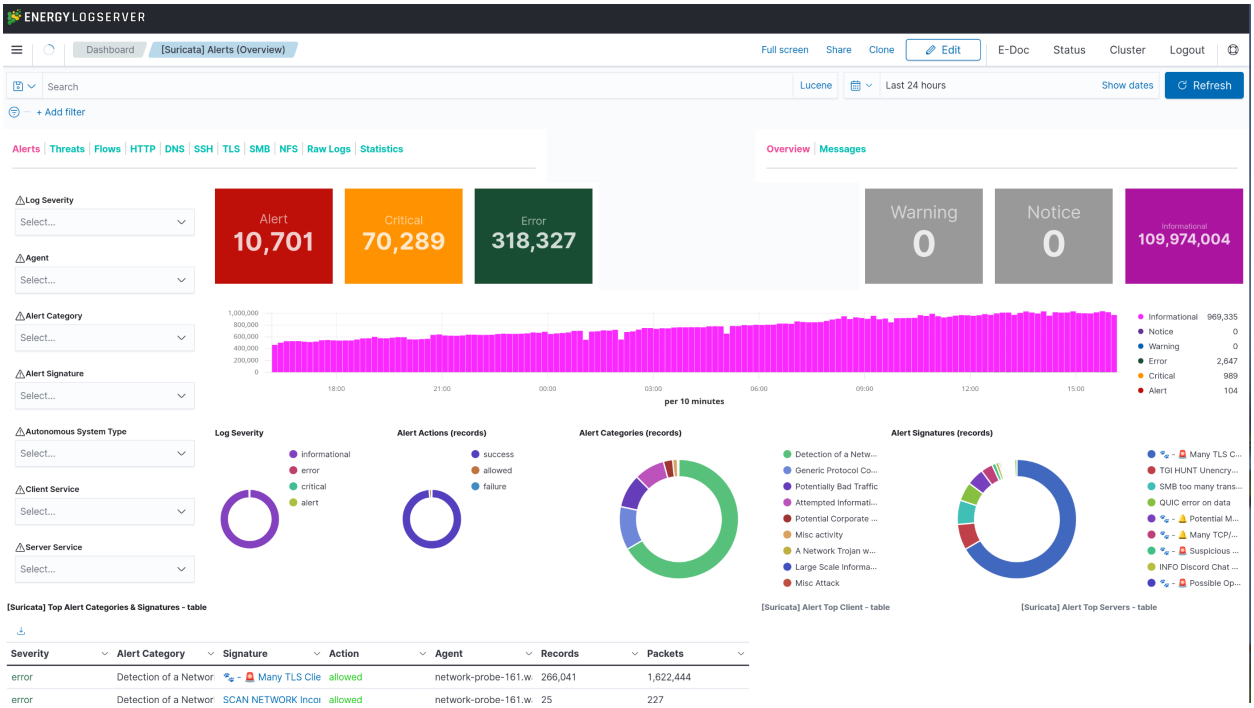
Explore in dashboards and Discover

Dashboards

Open **Menu** → **Dashboards** and search for `suricata`. The One Click deployment imports 26 dashboards.



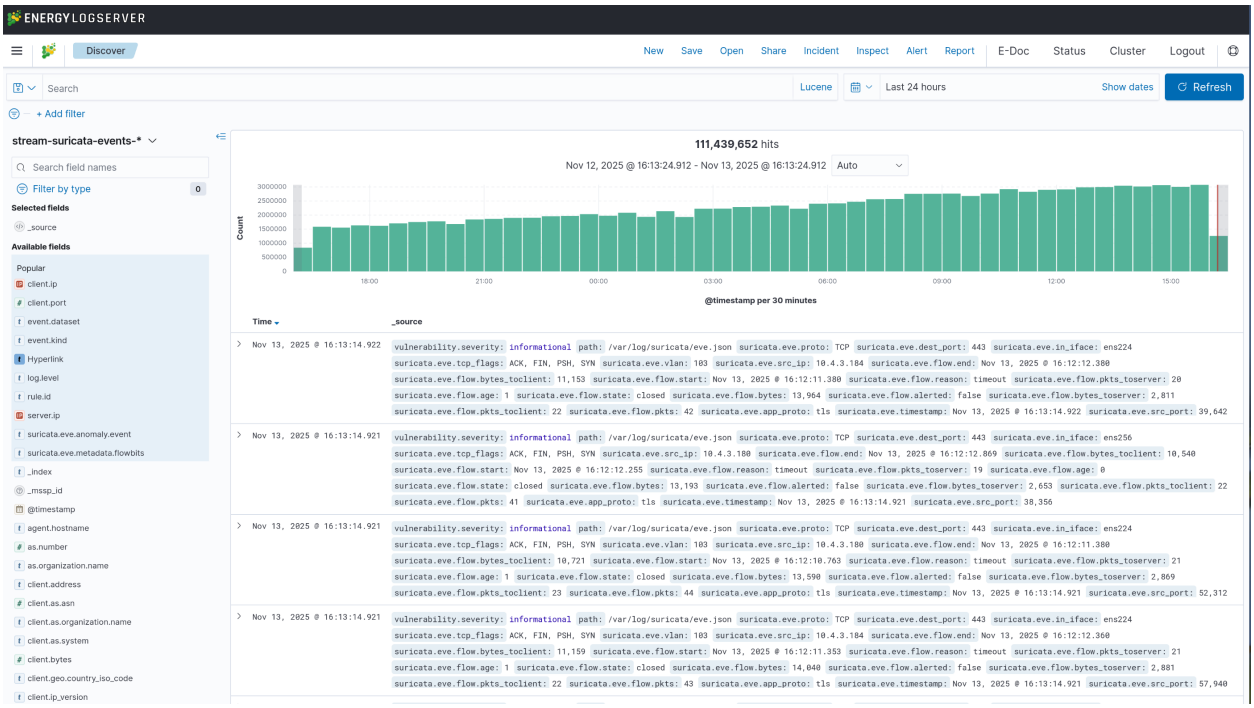
Open **[Suricata] Alerts (Overview)** for a threat-focused summary, then set the time picker (top right) to a range that covers your test traffic.



Discover

Discover is where you read the raw Suricata events. Open **Menu** → **Discover** and select the index pattern:

```
stream-suricata-events-*
```



If you see “No results match your search criteria”, work through these causes in order:

1. **Wrong time range.** The most common cause. Set the time picker to `Last 1 hour` right after starting

Suricata, or `Last 7 days` for older events.

2. **Suricata not running.** Run `systemctl status suricata`. If it is not active, start it (Step 8); if it fails to start, see *Troubleshooting*.
3. **No traffic.** Confirm events exist with `tail -50 /var/log/suricata/eve.json`.
4. **Pipeline not running.** Run `systemctl status logserver-probe`.

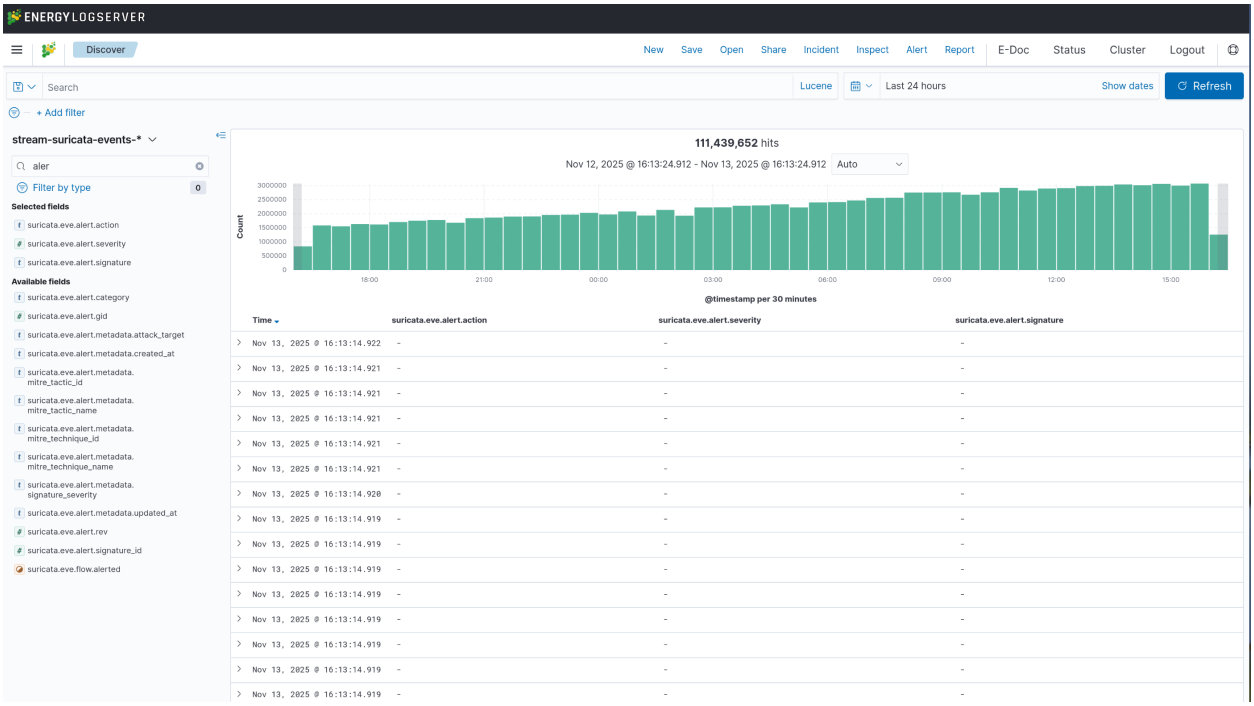
Useful searches

The Discover search bar uses Lucene syntax:

```
event_type:"alert"                                    # all alerts
event_type:"http"                                     # HTTP events
src_ip:"<IP-ADDRESS>"                                 # by source
dest_ip:"<IP-ADDRESS>"                                # by destination
alert.severity:1                                       # 1 = critical, 2 = high,
↪ 3 = medium
alert.signature:"SQL Injection"                       # by signature text
event_type:"alert" AND alert.severity:[1 TO 2]        # high severity only
event_type:"alert" AND NOT alert.category:"Not Suspicious Traffic"
```

Columns worth adding

The default `_source` column is hard to scan. Add these fields from the field list on the left to get a readable table:




```
event_type          # event type
src_ip              # source
dest_ip             # destination
alert.severity      # severity
```

(continues on next page)

(continued from previous page)

```
alert.signature   # alert name
proto             # protocol (TCP, UDP, ICMP)
```

Troubleshooting

"certificate signature failure" during One Click

If One Click fails at the pipeline step with `unable to execute operation! certificate signature failure`, this is not a Suricata problem and is not related to the manual steps above. It comes from inconsistent `license-service` certificates and affects every integration installed through One Click. See *Network Probe Troubleshooting* for diagnosis and resolution.

Suricata will not start

Read the service log:

```
journalctl -u suricata -n 50
```

Common causes:

- **`Unable to find iface '<name>'`**: the interface in the systemd override does not exist. Recheck Step 3 and Step 4.
- **Config validation failed**: run `suricata -T -c /etc/suricata/suricata.yaml` and fix the reported line.
- **Permission denied on `eve.json`**: reset ownership.

```
chown -R suricata:suricata /var/log/suricata
chmod 755 /var/log/suricata
```

High CPU or memory use

Each option below trades some detection or capture for lower load. Read the trade-off before applying it.

1. **Disable rule categories you do not need.**

   Cause: every enabled rule is evaluated against matching traffic, so categories you never look at still cost CPU.

   Trade-off: you stop detecting those categories. Disable only what you knowingly do not monitor.

   ```
   cat >> /etc/suricata/disable.conf <<'EOF'
   group:*games*
   group:*p2p*
   group:*chat*
   EOF
   suricata-update --disable-conf=/etc/suricata/disable.conf
   systemctl restart suricata
   ```

2. **Limit the AF_PACKET worker threads.**

   Cause: by default Suricata runs one worker per CPU core (`threads: auto`). On a many-core host each worker holds its own buffers and flow tables, which inflates CPU and memory even at low traffic.

   Trade-off: fewer workers means less capture throughput. On a busy link, setting this too low causes packet drops (see *Packet drops*). Lower it gradually and watch the drop rate.

Set the thread count in the `af-packet` section of `/etc/suricata/suricata.yaml`:

```
af-packet:
  - interface: <INTERFACE>
    threads: 2
```

3. **Disable file extraction if you do not need it.**

   Cause: with file extraction on, Suricata reassembles files carried in the traffic and writes them to disk, which is extra CPU and I/O unless you are actually collecting files.

   Trade-off: you lose extracted-file capture and any file-hash rules that depend on it.

```
file-store:
  enabled: no
```

Packet drops

A sustained drop rate above roughly 1 percent means Suricata is not keeping up with the traffic and is missing events. Check the statistics:

```
journalctl -u suricata -n 50 | grep "Stats for"
```

If drops are high, apply these in order:

1. **Increase the capture ring buffer** in the `af-packet` section.

   Cause: the ring buffer absorbs bursts between the NIC and Suricata. When it fills, the kernel drops packets before Suricata ever sees them.

   Trade-off: a larger ring pins more memory per interface. Double it and re-measure rather than setting it very high at once.

```
af-packet:
  - interface: <INTERFACE>
    ring-size: 4096
```

2. **Disable NIC offloading.**

   Cause: offloading features (GRO, LRO, TSO, GSO) hand Suricata pre-reassembled segments that no longer match what crossed the wire, which both hides detail and wastes work.

   Trade-off: the CPU now does work the NIC used to, so expect slightly higher CPU. This is the accepted setup for capture interfaces.

```
ethtool -K <INTERFACE> tx off rx off gro off lro off tso off gso off
```

3. **Raise the process priority (last resort).**

   Cause: under contention the scheduler may starve Suricata of CPU while it is trying to keep up with capture.

   Trade-off: a negative `Nice` value takes CPU from the other probe services (Kafka, pmacct, Zeek). Use it only after the ring buffer and offloading changes are not enough.

   Run `systemctl edit suricata` and add a `Nice=-10` line under `[Service]`.

No alerts

1. Confirm the ruleset exists and is not empty:

   ```
   ls -lh /var/lib/suricata/rules/suricata.rules
   ```

   If missing or empty, run `suricata-update` and restart the service.

2. Confirm `HOME_NET` matches your network (Step 5).

3. Confirm rules loaded:

   ```
   grep "rules loaded" /var/log/suricata/suricata.log
   ```

Events reach eve.json but not Discover

The data is stuck between Suricata and the GUI: Suricata writes `eve.json`, but the Network Probe pipeline is not reading it.

Check the pipeline service and its Suricata configuration:

```
systemctl status logserver-probe
ls -l /etc/logserver-probe/conf.d/suricata/
journalctl -u logserver-probe -f
```

Look for JSON parsing errors in the log. Then confirm the events reached the Data Node:

```
curl -X GET "localhost:9200/stream-suricata-events-*/_count?pretty"
```

A `count` of `0` means nothing has been indexed yet: go back and confirm Suricata is running and writing events.

Definition of terms

Term	Meaning
IDS / IPS	Intrusion Detection System and Intrusion Prevention System. Suricata runs here as an IDS: it detects and logs, it does not block.
eve.json	Suricata's JSON event log at `/var/log/suricata/eve.json`, one event per line.
HOME_NET	The address ranges Suricata treats as your internal network.
Emerging Threats (ET)	The open ruleset Suricata matches traffic against.
Masked service	A systemd state in which a unit is linked to `/dev/null` and cannot start until unmasked.
Promiscuous mode	An interface mode that accepts all frames on the wire, not only those addressed to the host.
Index pattern	A wildcard such as `stream-suricata-events-*` that tells the GUI which indices to read.

Zeek

Zeek (formerly Bro) is one of the detection engines bundled with the Network Probe (see the *bundled services* table). It is a Network Security Monitor (NSM): instead of matching traffic against a signature ruleset, it parses every connection and writes detailed, structured logs of what happened on the network. This chapter takes a fresh probe from a masked Zeek service to events visible in the Energy Logserver GUI.

- *What Zeek does*
- *What Zeek One Click deploys*
  - *How the pieces fit together*
- *Why manual host configuration is needed*
- *Requirements*
- *Step 1: Deploy the Energy Logserver components (GUI)*
- *Step 2: Unmask the Zeek service (SSH)*
- *Step 3: Set the capture interface*
- *Step 4: Switch Zeek output to JSON*
- *Step 5: Deploy and verify Zeek*
- *Step 6: Confirm the pipeline is reading the logs*
- *Step 7: Verify events reach the Data Node*
- *Verify end to end with test traffic*
- *Explore in Discover*
  - *Useful searches*
  - *Columns worth adding*
- *Build your own dashboards*
- *Maintenance*
  - *Log rotation*
  - *Zeek updates*
  - *Performance monitoring*
- *Troubleshooting*
  - *"certificate signature failure" during One Click*
  - *Zeek will not start*
  - *Events reach the sensor but not the index*
  - *Packet drops*
  - *High CPU or memory use*
- *Definition of terms*

What Zeek does

Zeek is a passive network analyzer focused on protocol detail and behavior, not on blocking:

- Writes per-protocol logs: connections (`conn.log`), DNS, HTTP, SSL/TLS, SSH, SMB, files, and more.
- Records rich connection metadata: endpoints, duration, bytes, connection state, and detected service.
- Runs a scripting language for custom detection and enrichment.
- Detects protocol anomalies and notable events (`notice.log`).

- Monitors passively from a mirror or SPAN port, so it adds no latency to the traffic it watches.

Zeek complements a signature engine such as *Suricata*: Suricata answers "did this match a known attack", Zeek answers "what exactly happened on the wire".

What Zeek One Click deploys

The **Zeek** entry in the *Integrations* catalog installs the Energy Logserver side of the pipeline in a single action. Open it from **Tools** → **Integrations** → **Zeek** and use *One Click Installation*.

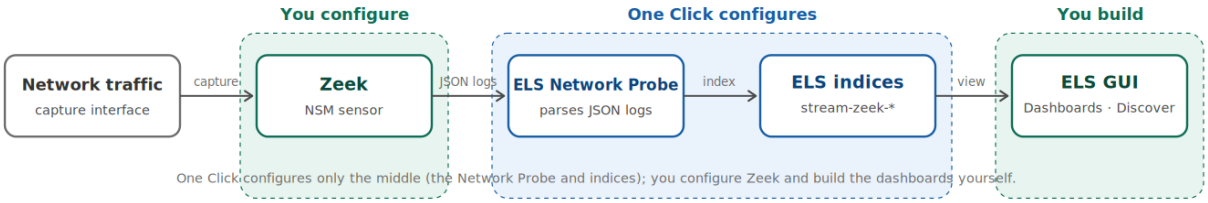
Component	Purpose
Pipeline	The Network Probe pipeline that reads Zeek's JSON logs, parses them, and writes the events to the `stream-zeek-*` indices. The pipeline installs the matching index template.

Note

Unlike the *Suricata* integration, Zeek One Click does not ship dashboards or predefined alert rules. Zeek's value is its detailed logs: you read them in **Discover** and build the dashboards that fit what you monitor. *Explore in Discover* and *Build your own dashboards* cover both.

How the pieces fit together

Before configuring anything, it helps to see the whole chain. Zeek captures traffic on the monitored interface and writes its logs to `/opt/zeek/spool/zeek/`. The Network Probe pipeline reads those logs, parses them, and writes the events into Energy Logserver indices under `stream-zeek-*`. The GUI, through Discover and any dashboards you build, then reads those indices.



One Click sets up the middle of this chain: the pipeline and the indices with their template. Both ends are left to you. On the left you configure Zeek itself, because that depends on your host and network. On the right you explore the data and build dashboards, because what is worth visualizing depends on what you monitor.

Why manual host configuration is needed

One Click configures only the parts that are identical across every deployment. Three things stay in your hands because they are specific to the host:

- **The Zeek service is left masked.** Masking is a systemd state in which the service unit is linked to `/dev/null`, so the service cannot start until you unmask it. This prevents Zeek from running before it has a capture interface and configuration.

- **The capture interface is not auto-detected.** The correct interface depends on which network card actually sees the traffic you want to monitor, which no central process can know.
- **Zeek logs in TSV by default, not JSON.** Zeek writes tab-separated logs out of the box, but the Network Probe pipeline expects JSON. You switch the output format on the host.

Requirements

- Root SSH access to the Energy Logserver host running the Network Probe.
- Admin access to the Energy Logserver GUI.
- A capture interface that sees the traffic you want to monitor (a SPAN or mirror port).
- Port 22 (SSH) reachable.
- Port 443 (GUI) reachable.

Step 1: Deploy the Energy Logserver components (GUI)

1. Log in to the GUI at `https://<IP-ADDRESS-OR-HOSTNAME>`.
2. Open **Tools** → **Integrations** and select the **Zeek** card.
3. Click **One Click** and wait for the deployment to finish.
4. Confirm that the Pipelines step reports its pipeline as activated on every Network Probe you selected.

Step 2: Unmask the Zeek service (SSH)

Connect to the host:

```
ssh root@<IP-ADDRESS-OR-HOSTNAME>
```

Check the current state:

```
systemctl status zeek
```

Expected output:

```
o zeek.service
   Loaded: masked (Reason: Unit zeek.service is masked.)
   Active: inactive (dead)
```

Unmask the service:

```
systemctl unmask zeek
```

The service is now unmasked but not yet running: it starts once the interface and output format are set.

Step 3: Set the capture interface

Zeek reads its node layout from `/opt/zeek/etc/node.cfg`. List the available interfaces:

```
ip link show | grep -E "^[0-9]+:"
```

Example output:

```
1: lo: <LOOPBACK,UP,LOWER_UP> ...          # loopback, do not use
2: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> ...
```

Choose the interface connected to the SPAN or mirror source, not the management interface. Back up the configuration, then set the interface:

```
cp /opt/zeek/etc/node.cfg /opt/zeek/etc/node.cfg.backup
```

Edit `/opt/zeek/etc/node.cfg` and point the standalone node at your interface:

```
[zeek]
type=standalone
host=localhost
interface=<INTERFACE>
```

Note

Replace `<INTERFACE>` with the name from this step (for example `ens192`). This is the deployment-specific counterpart to the capture-interface note in *Installation*.

Optionally, set your internal address ranges in `/opt/zeek/etc/networks.cfg`. Zeek uses them to classify traffic direction, which populates the locality fields (`private` or `public`) on the events.

Step 4: Switch Zeek output to JSON

The Network Probe pipeline parses JSON, so enable Zeek's JSON log policy. Back up the site configuration first:

```
cp /opt/zeek/share/zeek/site/local.zeek /opt/zeek/share/zeek/site/local.zeek.
↪backup
```

Append the JSON policy load to `/opt/zeek/share/zeek/site/local.zeek`:

```
# Write all logs as JSON for the Network Probe pipeline
@load policy/tuning/json-logs.zeek
```

Step 5: Deploy and verify Zeek

Check the configuration, then deploy it. `zeekctl deploy` validates the scripts, installs them, and starts Zeek:

```
zeekctl check
zeekctl deploy
```

Confirm Zeek is running:

```
zeekctl status
```

Expected output (example):

```
Name         Type       Host          Status    Pid    Started
zeek         standalone localhost      running   12345  ...
```

Because the `zeek.service` unit starts Zeek through `zeekctl deploy`, use `zeekctl status`, not `systemctl status zeek`, to read the true state of the running node.

Confirm the logs are now JSON. Give Zeek a moment to see traffic on the interface, then read the current connection log:

```
head -2 /opt/zeek/spool/zeek/conn.log
```

A JSON line confirms the format:

```
{"ts":1731234567.123456,"uid":"CHFgkX3PJpvTcqGvi3","id.orig_h":"10.4.9.10", ..
↪.}
```

If you still see tab-separated columns with `#separator` header lines, the policy did not load: recheck Step 4 and run `zeekctl deploy` again.

Enable start on boot:

```
systemctl enable zeek
```

Step 6: Confirm the pipeline is reading the logs

The One Click deployment placed the Zeek pipeline on the probe. Confirm the files are present and the service is running:

```
ls /etc/logserver-probe/conf.d/zeek/
systemctl status logserver-probe
```

Expected pipeline files:

```
04-zeek-input.conf   10-zeek-filter.conf   30.output.conf   ...
```

The input reads Zeek's per-protocol logs from `/opt/zeek/spool/zeek/` (`conn.log`, `dns.log`, `http.log`, `ssl.log`, and the rest) as JSON, and the output writes them to `stream-zeek-*`. Follow the probe log to confirm it picked up the files:

```
journalctl -u logserver-probe -f
```

Step 7: Verify events reach the Data Node

Check that the index exists and is filling:

```
curl -u <USER>:<PASSWORD> "http://localhost:9200/_cat/indices/stream-zeek-*?v"
```

Expected output (example):

```
health status index                    docs.count  store.size
green  open   stream-zeek-2025.11.15        12345       6.1mb
```

Confirm the document count is climbing:

```
curl -u <USER>:<PASSWORD> "http://localhost:9200/stream-zeek-*/_count?pretty"
```

A `count` of `0` means nothing has been indexed yet: go back and confirm Zeek is running (Step 5) and writing JSON, and that the probe is active (Step 6).

Verify end to end with test traffic

Zeek is passive, so it logs traffic that crosses the monitored link. Generate a little traffic across that link, then confirm it lands as Zeek events.

```
# DNS lookup and an HTTP request over the monitored path
dig example.com
curl -s http://example.com/ > /dev/null
```

After a short wait, look for the matching event types on the host:

```
sleep 30
cat /opt/zeek/spool/zeek/dns.log  | tail -2
cat /opt/zeek/spool/zeek/http.log | tail -2
```

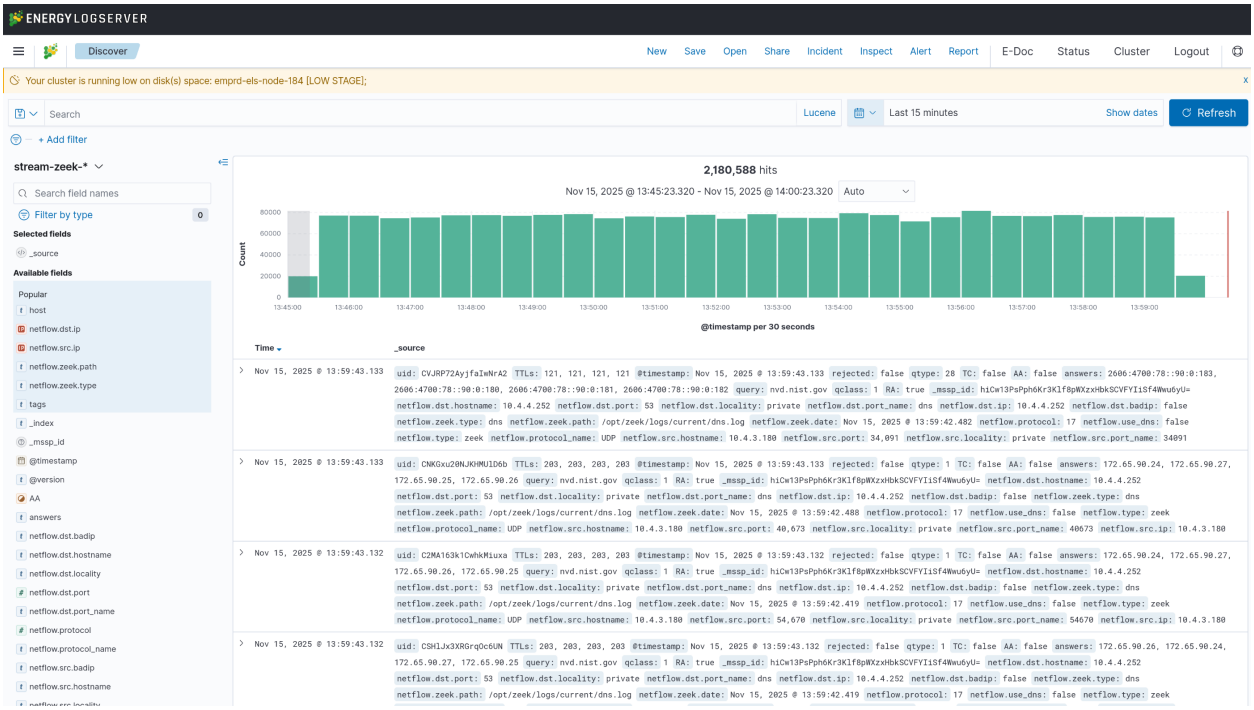
DNS and HTTP entries confirm that Zeek sees the link. The next section confirms the pipeline carried them into the GUI.

Explore in Discover

Discover is where you read the raw Zeek events. Open **Menu** → **Discover** and select the index pattern:

```
stream-zeek-*
```

If the pattern does not exist yet, create it once under **Management** → **Index Patterns** with `stream-zeek-*` and `@timestamp` as the time field.



If you see “No results match your search criteria”, work through these causes in order:

1. **Wrong time range.** The most common cause. Set the time picker to `Last 1 hour` right after starting Zeek, or `Last 7 days` for older events.
2. **Zeek not running.** Run `zeekctl status` on the host. If it is not running, deploy it (Step 5).

3. **No traffic.** Confirm logs exist with `tail -2 /opt/zeek/spool/zeek/conn.log`.
4. **Pipeline not running.** Run `systemctl status logserver-probe`.

Useful searches

Zeek events carry the log type in `netflow.zeek.type` and the connection detail under `netflow.*`. The Discover search bar uses Lucene syntax:

```
netflow.zeek.type:"conn"                              # connection records
netflow.zeek.type:"dns"                               # DNS queries
netflow.zeek.type:"http"                              # HTTP requests
netflow.zeek.type:"ssl"                               # TLS sessions
netflow.src.ip:"<IP-ADDRESS>"                         # by source
netflow.dst.ip:"<IP-ADDRESS>"                         # by destination
netflow.dst.port:443                                  # by destination port
netflow.dst.badip:true                                # destination flagged as a␣
↪known bad IP
```

Columns worth adding

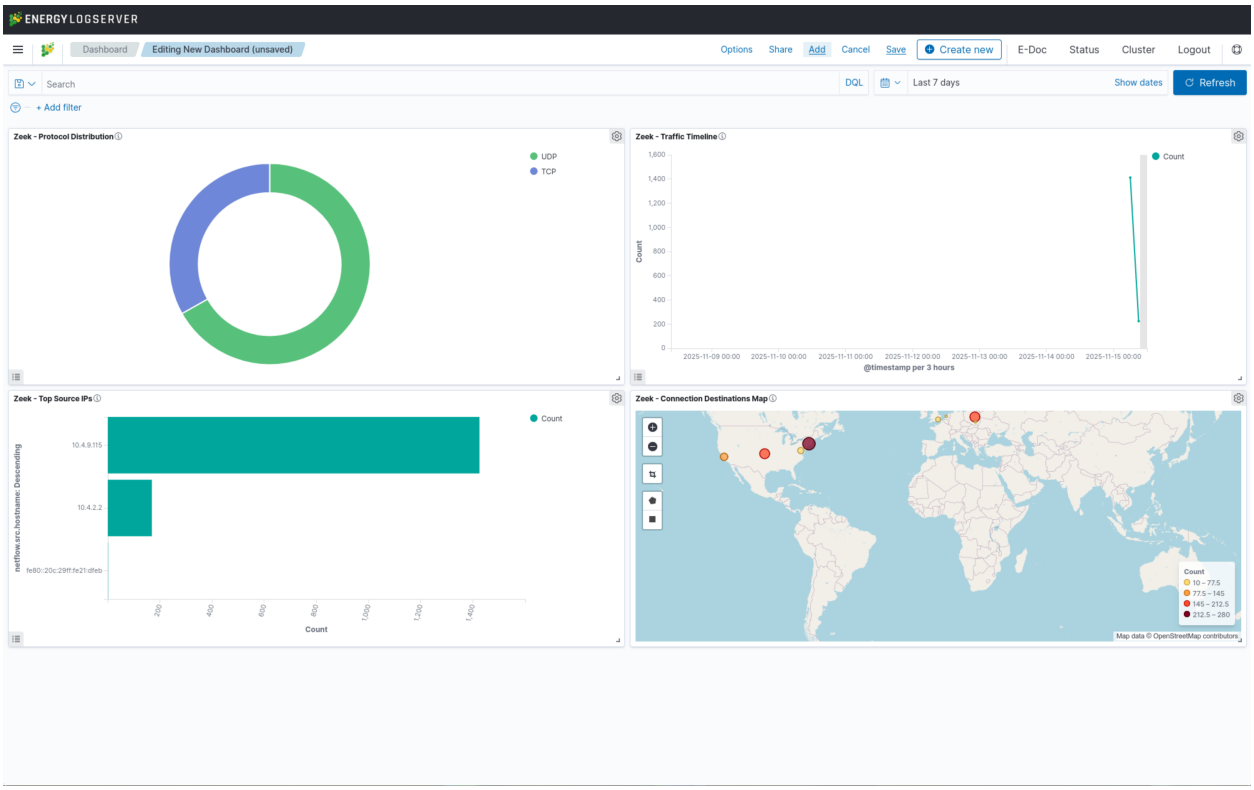
The default `_source` column is hard to scan. Add these fields from the field list on the left to get a readable table:

```
netflow.zeek.type       # log type (conn, dns, http, ssl, ...)
netflow.src.ip          # source
netflow.dst.ip          # destination
netflow.dst.port        # destination port
netflow.zeek.service    # detected service
netflow.protocol_name   # transport protocol
```

Build your own dashboards

Zeek One Click does not import dashboards, so you build the views you need. Create visualizations under **Menu** → **Visualize** → **Create visualization** against the `stream-zeek-*` index pattern, then collect them on a dashboard. A useful starter set:

Visualization	Type	Bucket / field
Top source IPs	Data table	Terms on `netflow.src.ip`
Protocol distribution	Pie	Terms on `netflow.zeek.service`
Traffic timeline	Line	Date histogram on `@timestamp`
Connection destinations map	Coordinate map	Geohash on the destination geo-point field, where geoip enrichment is enabled



Save the visualizations, add them to a new dashboard under **Menu** → **Dashboard** → **Create dashboard**, and set the time picker to a range that covers your traffic.

Maintenance

Log rotation

Zeek's logs exist in two places, with separate retention: as files on the sensor, and as indexed events in Energy Logserver. You manage both, because a full sensor disk stops capture and unbounded indices fill the Data Node.

1. **Zeek rotates its own logs.** Zeek closes the current logs in `/opt/zeek/spool/zeek/` on the interval set by `LogRotationInterval` in `/opt/zeek/etc/zeekctl.cfg`; the `zeekctl cron` maintenance pass then compresses them and moves them to dated directories under `/opt/zeek/logs/`. The probe installs the cron entry `*/5 * * * * /opt/zeek/bin/zeekctl cron` for this. If rotation is not running (an empty `/opt/zeek/logs/`, or `LogRotationInterval=0`), enable it there rather than adding a second rotation mechanism.

2. **Archived logs** (`/opt/zeek/logs/`) are the rotated, compressed logs kept on the sensor. Purge them on a schedule so the disk does not fill:

   ```
   find /opt/zeek/logs/ -mtime +30 -delete
   ```

3. **Indexed events** (`stream-zeek-*`). One Click does not install an index-management policy for these indices, so nothing prunes them by default and they grow until the Data Node fills. Create an Index Management policy (rollover and deletion) for `stream-zeek-*` in the GUI to bound their retention. Deleting log files on the sensor does not remove data already in Energy Logserver, and the reverse also holds.

> **Warning**
>
> Do not point `logrotate` at `/opt/zeek/spool/zeek/`. Zeek keeps those files open while writing, so rotating them underneath it leaves Zeek writing to a deleted file and breaks the Network Probe's read position (the `sincedb`). Let Zeek rotate its own logs, and purge the archive in `/opt/zeek/logs/` instead.

Zeek updates

```
# Back up the configuration first
tar -czf /root/zeek-config-$(date +%Y%m%d).tar.gz /opt/zeek/etc/

systemctl stop zeek
yum update zeek -y
zeekctl deploy
```

Performance monitoring

```
zeekctl status      # is the node running
zeekctl netstats    # received vs dropped packets
du -sh /opt/zeek/spool/zeek/   # current log volume
```

Troubleshooting

"certificate signature failure" during One Click

If One Click fails at the pipeline step with `unable to execute operation! certificate signature failure`, this is not a Zeek problem and is not related to the manual steps above. It comes from inconsistent `license-service` certificates and affects every integration installed through One Click. See *Network Probe Troubleshooting* for diagnosis and resolution.

Zeek will not start

Check the configuration and the diagnostic output:

```
zeekctl check
zeekctl diag
tail -50 /opt/zeek/spool/zeek/stderr.log
```

Common causes:

- **Interface does not exist**: the interface in `node.cfg` is wrong. Recheck Step 3 with `ip link show`.
- **Script error**: `zeekctl check` reports the failing line. Fix it and run `zeekctl deploy` again.
- **No packets arriving**: confirm the mirror or SPAN port is delivering traffic with `tcpdump -i <INTERFACE> -c 10`.

Events reach the sensor but not the index

Zeek writes its logs, but the Network Probe pipeline is not reading them. The usual cause is file permissions: the probe reads Zeek's logs through membership in the `zeek` group, so the log files must be group-readable by `zeek`.

```
ls -l /opt/zeek/spool/zeek/*.log
chown zeek:zeek /opt/zeek/spool/zeek/*.log
chmod 640 /opt/zeek/spool/zeek/*.log
```

`640` with group `zeek` is enough; the probe user (`logserver-probe`) does not need world-readable files. The Network Probe package adds `logserver-probe` to the `zeek` group at install time, so this membership normally already exists. Confirm it, and add it if missing:

```
id logserver-probe | grep -o zeek || gpasswd -a logserver-probe zeek
```

If you just added the membership, restart the pipeline so it takes effect:

```
systemctl restart logserver-probe
```

If the log format is the problem instead, confirm it is JSON (`head -2 /opt/zeek/spool/zeek/conn.log`); if it is still TSV, redo Step 4.

If the pipeline read position is stuck after a bad rotation, clear the `sincedb` and let it re-read (this reprocesses existing logs):

```
systemctl stop logserver-probe
rm -f /var/lib/logserver-probe/plugins/inputs/file/.sincedb_*
systemctl start logserver-probe
```

Packet drops

A sustained drop rate means Zeek is not keeping up with the traffic and is missing events. Check the counters:

```
zeekctl netstats
```

If drops are high, apply these in order:

1. **Spread the load across worker processes.**

   Cause: a single Zeek process is bound to one core, which caps how much traffic it can parse.

   Trade-off: each worker holds its own state and buffers, so more workers use more CPU and memory. Raise the count gradually and re-measure.

   Set `lb_procs` in `/opt/zeek/etc/node.cfg`:

   ```
   [zeek]
   type=standalone
   host=localhost
   interface=<INTERFACE>
   lb_procs=2
   ```

2. **Increase the capture ring buffer.**

   Cause: the ring buffer absorbs bursts between the NIC and Zeek. When it fills, the kernel drops packets before Zeek sees them.

   Trade-off: a larger ring pins more memory per interface. Double it and re-measure rather than setting it very high at once.

   ```
   ethtool -G <INTERFACE> rx 4096
   ```

High CPU or memory use

Cause: Zeek parses and logs every protocol it recognizes, so on a busy link the volume of parsed events drives both CPU and log size.

Trade-off: disabling logs you do not use lowers load but removes that visibility. Turn off only what you knowingly do not need, in `/opt/zeek/share/zeek/site/local.zeek`, then `zeekctl deploy`. Before that, confirm the load is Zeek and not the pipeline:

```
ps -o pid,%cpu,%mem,cmd -C zeek
```

Definition of terms

Term	Meaning
NSM	Network Security Monitor. Zeek observes and records network activity in detail rather than blocking it.
conn.log	Zeek's connection log: one record per TCP, UDP, or ICMP flow, with endpoints, duration, bytes, and state.
zeekctl	Zeek's control utility. `zeekctl deploy` validates, installs, and (re)starts the configuration; `zeekctl cron` runs the log-rotation and maintenance pass.
networks.cfg	`/opt/zeek/etc/networks.cfg`, where you list the address ranges Zeek treats as your internal network. Zeek uses them to classify traffic direction and locality.
Masked service	A systemd state in which a unit is linked to `/dev/null` and cannot start until unmasked.
sincedb	The Network Probe's record of how far it has read each log file, so it does not re-read from the start.
Index pattern	A wildcard such as `stream-zeek-*` that tells the GUI which indices to read.

pmacct Local Traffic Monitoring

- *What pmacct does*
- *How pmacctd fits the Network Probe*
- *Why manual configuration is needed*
- *Requirements*
- *Step 1: Create the configuration and log directories*
- *Step 2: Identify the capture interface*
- *Step 3: Create the pmacctd configuration*
- *Step 4: Create the systemd service*
- *Step 5: Start pmacctd*
- *Step 6: Verify pmacctd is exporting*
- *Verify end to end with test traffic*
- *Explore in Discover*
- *Troubleshooting*

- *Definition of terms*

What pmacct does

pmacct is the flow accounting toolkit bundled with the Network Probe. It ships as several daemons, and Energy Logserver uses them in two distinct roles:

- **Collector** (`nfacctd`, `sfacctd`): receives NetFlow, IPFIX, or sFlow that network devices export to the probe over UDP, and forwards it into the collection pipeline. This is the default role, covered in *Bundled Services* and *Technical Configuration*.
- **Generator** (`pmacctd`): captures packets directly from a host interface in promiscuous mode, aggregates them into flows, and exports those flows as NetFlow. This chapter covers the generator role.

Use the generator when you want to account for the traffic of a host that does not export NetFlow on its own, most commonly the Energy Logserver server itself. It complements *Suricata IDS* and *Zeek*: where those inspect content and protocols, `pmacctd` answers who talked to whom, over which protocol and port, and how much data moved.

How pmacctd fits the Network Probe

`pmacctd` does not write to Energy Logserver directly. It produces NetFlow and hands it to the collector that already runs on the probe:

```
Host interface (promiscuous, libpcap)
        |
        v
pmacctd  -- aggregates packets into flows, nfprobe plugin exports NetFlow v9
        |  UDP 127.0.0.1:2055
        v
nfacctd  -- the bundled NetFlow collector (plugins: kafka)
        |  Kafka topic "netflow"
        v
Network Probe network-stream pipeline  -- /etc/logserver-probe/conf.d/network-
↪stream/
        |
        v
Index  stream-netflow-pmacct-*
```

The right-hand side of this chain already exists on a probe that collects NetFlow: `nfacctd` listens on UDP 2055, publishes to the Kafka `netflow` topic, and the `network-stream` pipeline writes the enriched flows (GeoIP, DNS, bad-IP lookups) to `stream-netflow-pmacct-*`. What you configure here is only `pmacctd`, so it captures on the correct interface and exports to the port the collector is already listening on.

This is why there is no One Click step for the generator. One Click installs the Energy Logserver side of a pipeline; here the Energy Logserver side is the existing NetFlow collection path, and the work is entirely host-side.

Note

The hand-off from `pmacctd` to the collector is a UDP protocol, not a file. `pmacctd` sends flows and receives no acknowledgement. That shapes what goes wrong: if flows do not arrive, the question is whether `pmacctd` is exporting and whether the collector is listening on UDP 2055, not whether a file is readable.

Why manual configuration is needed

The pmacct package ships the `pmacctd` binary but no generator configuration and no service unit. Energy Logserver ships setup scripts for the `nfacctd`/`sfacctd` collector role, not for `pmacctd`. To run the generator you create its configuration file and a systemd unit yourself, described below.

Requirements

- The pmacct package, installed with the Network Probe (`rpm -qa | grep pmacct`).
- A running NetFlow collector on the probe: `nfacctd` listening on UDP 2055, with the `network-stream` pipeline active, so the generated flows have somewhere to go. The installer does not create the collector, see *Setting Up the NetFlow and sFlow Collectors*.
- Root access on the host you want to account for.
- The interface that carries the traffic you want to measure.

Step 1: Create the configuration and log directories

The package ships no generator directories, so create them:

```
mkdir -p /etc/pmacct
chmod 755 /etc/pmacct

mkdir -p /var/log/pmacct
chmod 755 /var/log/pmacct
```

Step 2: Identify the capture interface

Find the interface that carries the traffic you want to account for. You will write its name into the configuration in the next step.

```
ip -br addr show
```

Note the interface name (for example `ens192`, `eth0`, `enp0s3`). The interface does not need an address of its own to be captured, but it must be `UP`.

Step 3: Create the pmacctd configuration

Create `/etc/pmacct/pmacctd.conf`. Replace `INTERFACE_NAME` with the interface from Step 2:

```
! pmacctd generator configuration for Energy Logserver
! Local traffic accounting with NetFlow v9 export to the local collector

! General
daemonize: true
pidfile: /var/run/pmacctd.pid
syslog: daemon

! Capture
pcap_interface: INTERFACE_NAME
promisc: true

! Export flows as NetFlow v9 to the local collector on UDP 2055
```

(continues on next page)

(continued from previous page)

```
plugins: nfprobe
nfprobe_version: 9
nfprobe_receiver: 127.0.0.1:2055
nfprobe_timeouts: tcp=120:maxlife=3600

! Aggregation key
aggregate: src_host, dst_host, src_port, dst_port, proto, tos
```

Note

Do not add `imt_mem_pools_size` or `imt_buckets` here. Those tune the in-memory table of the `memory` plugin. With `plugins: nfprobe` there is no memory table, so `pmacctd` accepts the keys without error but they do nothing. The `memory` plugin is only useful if you query flows locally with the `pmacct` client (`pmacct -s`), which this setup does not.

Step 4: Create the systemd service

`pmacctd` ships without a unit. Create `/etc/systemd/system/pmacctd.service`:

```
[Unit]
Description=pmacctd - Promiscuous Mode Accounting Daemon
Documentation=man:pmacctd(8)
After=network-online.target
Wants=network-online.target

[Service]
Type=forking
PIDFile=/var/run/pmacctd.pid
ExecStart=/usr/sbin/pmacctd -f /etc/pmacct/pmacctd.conf
ExecReload=/bin/kill -HUP $MAINPID
Restart=on-failure
RestartSec=10
CapabilityBoundingSet=CAP_NET_RAW CAP_NET_ADMIN
AmbientCapabilities=CAP_NET_RAW CAP_NET_ADMIN

[Install]
WantedBy=multi-user.target
```

Reload systemd and enable the service:

```
systemctl daemon-reload
systemctl enable pmacctd
```

Step 5: Start pmacctd

```
systemctl start pmacctd
systemctl status pmacctd --no-pager
```

A healthy start reports `active (running)` with two tasks: the core process and the NetFlow probe plugin. `pmacctd` parses the configuration only when it starts, so this is also the check for Step 3. If the service fails, read the log as described in *pmacctd will not start*.

> **Note**
>
> Do not test the configuration by running `pmacctd -f /etc/pmacct/pmacctd.conf -d` by hand. The `-d` option only enables debug output. With `daemonize: true` in the file, the process goes to the background at once, `Ctrl+C` does not stop it, and it keeps running. If you already started it this way, stop it with `kill $(cat /var/run/pmacctd.pid)`.

Step 6: Verify pmacctd is exporting

Confirm the process is up and the log shows flows leaving for the collector:

```
journalctl -u pmacctd -n 20 --no-pager
```

The log line `Exporting flows to [127.0.0.1]:iop` confirms export; `iop` is the service-name notation for port 2055.

Verify end to end with test traffic

`pmacctd` exports a flow only when the flow expires, not on every packet. With the default `nfprobe` timeouts, which `pmacctd` prints to the log at start, an ICMP or UDP flow expires after 300 seconds without packets, and a TCP flow 300 seconds after it closes. Generate traffic and wait at least six minutes before concluding anything is wrong:

```
ping -c 10 8.8.8.8 > /dev/null 2>&1
curl -s https://example.com > /dev/null 2>&1

# wait for flow expiry before the count updates
sleep 360
```

Then confirm documents are landing in the collector's index:

```
curl -s -u admin:PASSWORD \
  "http://localhost:9200/_cat/indices/stream-netflow-pmacct-*?v&h=index,docs.
↪count,store.size"
```

A growing `docs.count` on today's `stream-netflow-pmacct-*` index confirms the whole chain: capture, export, collection, and indexing.

Explore in Discover

The generated flows join the same index pattern as all collected NetFlow. Open **Discover**, select the `stream-*` index pattern, and set a recent time range. To isolate the generator's own flows, filter on the index `stream-netflow-pmacct-*`.

Useful fields:

- `netflow.src.ip`, `netflow.dst.ip`: the endpoints of each flow.
- `netflow.src.port`, `netflow.dst.port`: the ports.
- `netflow.protocol_name`: TCP, UDP, ICMP.
- `netflow.bytes`, `netflow.packets`: the volume of each flow.
- `netflow.exporter`: the address that exported the flow (`127.0.0.1` for the local generator). In 8.1.0 this field holds the literal text `%{peer_ip_src}` until you add `peer_src_ip` to the collector configuration, see *Setting Up the NetFlow and sFlow Collectors*.

Troubleshooting

pmacctd will not start

Read the service log for the cause:

```
journalctl -u pmacctd -n 50 --no-pager
```

`pcap_open_live(): INTERFACE_NAME: No such device` means the interface name is wrong. Either the placeholder from Step 3 was never replaced, or the name does not exist on this host. List the real interfaces with `ip -br link show` and correct `pcap_interface`.

`pcap_open_live(): Permission denied` means the process lacks `CAP_NET_RAW`. Run `pmacctd` as root, or confirm the `CapabilityBoundingSet` and `AmbientCapabilities` lines are present in the unit.

No flows reach the index

`pmacctd` exports over UDP, which is fire-and-forget: it reports no error even when nothing receives the flows. If the collector is not listening, the kernel discards every packet silently while `pmacctd` still logs `Exporting flows`.

Confirm something is bound to the collector port:

```
ss -ulnp | grep 2055
```

If nothing is listening, the NetFlow collector is not running. Start it and confirm the `network-stream` pipeline is active on the probe (see *Setting Up the NetFlow and sFlow Collectors*). If the collector is listening but the index stays empty, generate traffic and wait for flow expiry, then re-check, since flows are exported only when they expire.

High CPU use

`pmacctd` builds an aggregation key for every packet and keeps a flow entry for each unique combination. The more fields in the key, the more distinct flows exist, and the more CPU each packet costs. At high traffic volumes a fine-grained `aggregate` directive is usually what drives the load.

Reduce the aggregation granularity first. Check what you currently aggregate on:

```
grep aggregate /etc/pmacct/pmacctd.conf
```

Drop the fields you do not query on. Source and destination IP, port, and protocol produce far fewer flows than adding ToS, interfaces, and MAC addresses on top. Restart and measure again:

```
systemctl restart pmacctd
top -p "$(pgrep -d, pmacctd)"
```

If trimming the key is not enough, sample the traffic:

```
sampling_rate: 100  ! analyse 1 packet in 100
```

> **Warning**
>
> Sampling trades accuracy for CPU. With `sampling_rate: 100`, byte and packet counters become estimates scaled by 100, and short flows can be missed entirely. That is acceptable for traffic trends and top-talker views, but not for accounting or forensic use where individual flows matter. Trim the aggregation key first; sample only if the sensor still cannot keep up.

Reducing data volume

To lower the number of exported flows and the storage they consume, raise the export timeouts so flows are batched over longer intervals:

```
nfprobe_timeouts: tcp=300:maxlife=7200
```

This reduces the volume of NetFlow records and downstream documents, not the per-packet CPU cost of capture and aggregation. It also delays when data appears: with a long `maxlife`, an active session may not reach Energy Logserver until it ends or the timeout elapses. Use it to control data volume, and keep the timeouts short if you need flows to surface quickly.

Flows land under the wrong date

NetFlow indices are date-based (`stream-netflow-pmacct-YYYY.MM.DD`), and the index a flow lands in follows its timestamp. If the host clock is wrong, flows are written to an index for another date and do not appear under a normal time range in Discover. Confirm the clock and enable time synchronisation:

```
timedatectl
timedatectl set-ntp true
```

List the indices that actually exist to find where the flows went:

```
curl -s -u admin:PASSWORD "http://localhost:9200/_cat/indices/stream-netflow-
↪pmacct-*?v"
```

Many flows to port 53 of the DNS server

The `network-stream` pipeline resolves flow addresses to hostnames with reverse DNS lookups sent to the server set in `DNS_SRV`, `8.8.8.8` by default. `pmacctd` captures these lookups on the interface and exports them as flows, so expect flows from the probe to port 53 of that server. When the same collector also receives flows from a busy network device, such as a firewall exporting NetFlow to the probe, the lookups can reach thousands per minute, and the matching flows keep arriving for several minutes after that traffic stops.

Missing GeoIP data

GeoIP enrichment is applied by the Network Probe `network-stream` pipeline, not by `pmacctd`. If flow records have no `netflow.geoip` fields, confirm the GeoIP databases are installed and current on the probe, and check the probe log for GeoIP errors. Local flows between private addresses have no GeoIP by design.

Definition of terms

- **pmacctd**: the pmacct generator daemon. Captures packets from an interface with libpcap and exports flows. Distinct from `nfacctd`/`sfacctd`, which receive flows over UDP.
- **nfprobe**: the pmacctd plugin that formats flows as NetFlow and exports them to a receiver.
- **Flow**: an aggregate of packets that share the same key (source and destination address, ports, protocol) over a time window.
- **Flow expiry**: the point at which pmacctd stops accumulating a flow and exports it, either when the connection ends or when a timeout elapses.
- **Promiscuous mode**: an interface mode in which the card accepts all packets it sees, not only those addressed to it. Required to account for traffic that passes the interface.

4.8 SIEM Module

Real-time security event management with intelligent alerting, incident response orchestration, and automated risk assessment.

The SIEM Module serves as the operational core of Energy Logserver's security intelligence capabilities. It transforms raw security events into actionable alerts through intelligent correlation, manages incident response workflows with automated playbooks, and provides comprehensive risk scoring to prioritize security operations. From configuring detection rules to orchestrating response actions, this module enables security teams to detect, investigate, and respond to threats efficiently.

4.8.1 Overview

Energy Logserver's SIEM capabilities provide end-to-end security event management for modern security operations. Key features include:

- **Advanced Alerting System**: Configure detection rules with multiple alert types, use pre-built templates, and customize alert actions
- **Incident Management**: Track security incidents from detection through resolution with full audit trails
- **Risk Management**: Automated risk scoring and prioritization for security events and entities
- **Playbook Automation**: Define automated response workflows to accelerate incident handling
- **Performance Optimization**: Fine-tune alert processing and correlation for high-volume environments
- **External Integrations**: Connect with ticketing systems, SOAR platforms, and notification channels

4.8.2 In This Section

This chapter covers the complete SIEM workflow, from alert configuration to operational troubleshooting.

SIEM Interface Overview

The SIEM module consists of two main components: the **SIEM Dashboard** (agent-based security monitoring powered by the Endpoint Security engine) and the **Alerts Interface** (native alerting, incident management, and risk scoring).

SIEM Dashboard

The SIEM Dashboard provides security monitoring through agents installed on monitored hosts. It presents data as a set of dashboard cards organized in four sections.

> **Note**
>
> The SIEM Dashboard requires Endpoint Security agents to be installed and connected on monitored systems. Without active agents, dashboard cards will show no data.

Security Information Management

Card	Description
Security Events	Security alerts and threat events detected across monitored hosts
Integrity Monitoring	File change monitoring: alerts on permission, content, ownership, and attribute changes
Office 365	Security events from Microsoft Office 365 audit logs
Amazon AWS	Security monitoring for Amazon Web Services infrastructure
Google Cloud Platform	Security monitoring for Google Cloud Platform services
GitHub	Security events from GitHub audit logs

Auditing and Policy Monitoring

Card	Description
Policy Monitoring	Security policy compliance verification against a defined baseline
System Auditing	User activity tracking: command execution, file access, and behavior monitoring
OpenSCAP	Automated security compliance assessment using SCAP policies
CIS-CAT	CIS Benchmark configuration assessment scanning
Security Configuration Assessment	Agent-based system configuration scanning for security misconfigurations

Threat Detection and Response

Card	Description
Vulnerabilities	Known vulnerability detection across applications in your environment
VirusTotal	File and URL threat intelligence lookups via VirusTotal integration
Osquery	Endpoint visibility using SQL-based osquery queries
Docker Listener	Security monitoring for Docker container activity
MITRE ATT&CK	Security event mapping to MITRE ATT&CK framework techniques

Regulatory Compliance

Card	Description
PCI DSS	Payment Card Industry Data Security Standard compliance monitoring
NIST 800-53	Federal security controls compliance tracking
TSC	Trust Services Criteria (SOC 2) compliance monitoring
GDPR	EU General Data Protection Regulation compliance monitoring
HIPAA	Healthcare data protection (Protected Health Information) compliance monitoring

Each card links to a dedicated dashboard with detailed views, filters, and visualizations for the respective security domain.

Alerts Interface

Navigation: ELS Console → SIEM

The native Alerts interface provides alerting, incident management, risk scoring, and playbook capabilities. It consists of six tabs:

Tab	Description	Details
Incidents	Manage triggered alert incidents: assign, update status, add notes	See *Incident Management*
Risks	Create risk categories and map field values to risk scores	See *Risk Management*
Playbook	Define response procedures and scripts for alert rules	See *Playbooks*
Alert Rules	Create, manage, enable/disable, and test alert rules	See *Alerting System*
Alert Status	Monitor alert processing: activation status, event counts, run times	See *Alerts Status*
Create Alert Rule	Single-page form for defining new alert rules	See *Alert Creation*

Workflow

The tabs work together in a detection-to-response workflow:

1. **Alert Rules** define detection logic and notification methods.
2. When a rule triggers, an **Incident** is created automatically.
3. **Playbooks** associated with the rule provide response guidance and scripts.
4. **Risk** scoring prioritizes incidents based on entity risk categories.

SIEM Navigation Links

The SIEM section in the sidebar provides quick-access links:

- **Status**: redirects to the main SIEM overview dashboard
- **Vulnerability**: redirects to the vulnerability management dashboard
- **FIM**: redirects to the File Integrity Monitoring dashboard
- **Alerts**: opens the Alerts interface
- **Risk**: opens the Risks tab directly
- **SIEM Engine**: opens the Endpoint Security engine overview (home dashboard)

Alerting System

- *Overview*
- *Alerting Interface Overview*
- *Creating Your First Alert*
- *Alert Rule Types*
- *Risk Framework*

- *Playbooks*
- *Incidents*
- *Working with Pre-built Alert Groups*
- *Escalation and Recovery*

Overview

Alerting in Energy Logserver is the early warning system, automatically monitoring log data and generating notifications when security events occur. It provides continuous surveillance of your environment.

What is Alerting?

Alerting transforms raw log data into actionable security intelligence by detecting patterns, anomalies, and known threat indicators in real-time, supporting compliance and incident response.

Straightforward explanation:

- **What it is:** An automated security event detection system.
- **How it works:** Define rules; the system monitors data and alerts when matches occur.
- **What it does:** Enables early threat detection, incident response, and compliance monitoring.

Alerting Interface Overview

Access: ELS Console → SIEM → Alerts

The Alerts workspace exposes six tabs:

1. **Create Alert Rule**: form for defining a new alert rule.
2. **Alert Rules List**: all existing rules, grouped by the Group Name assigned at creation time.
3. **Alert Status**: last execution status for every rule (module state, hits, matches, duration).
4. **Playbook**: library of response procedures that can be attached to alert rules.
5. **Risks**: risk categories and (field, value) to category assignments that drive the risk score on triggered alerts.
6. **Incidents**: triggered alerts ordered by time, with the calculated risk score per incident.

How the pieces fit together

- **Alert Rules** detect conditions and emit events into the `.siem-*` indices.
- **Incidents** are those triggered events, annotated with a per-record risk score derived from **Risks** (Risk Key value looked up in the Risk List, aggregated, multiplied by the rule's Risk boost).
- **Playbooks** are text and script artefacts kept in the Playbook library; a rule may attach any number of playbooks that are shown next to the triggered incident.

Creating Your First Alert

Energy Logserver allows you to create alerts, i.e., monitoring queries. These are constant queries that run in the background, and when the conditions specified in the alert are met, the specified action is taken.

For example, if you want to know when more than 20 "status:500" response codes from your homepage appear within one hour, then create an alert that checks the number of occurrences of the "status:500" query for a specific index every 5 minutes. If the condition is met, an action is sent, such as a message to an email address or launching a script.

Enabling the Alert Module

SMTP Server Configuration

To configure the SMTP server for email notifications, you should:

- Edit `/opt/alert/config.yaml` and add the following section:

```
# email conf
smtp_host: "mail.example.com"
smtp_port: 587
smtp_ssl: false
from_addr: "siem@example.com"
smtp_auth_file: "/opt/alert/smtp_auth_file.yml"
```

- Add the new `/opt/alert/smtp_auth_file.yml` file:

```
user: "user"
password: "password"
```

- Restart the `alert` service:

```
systemctl restart alert
```

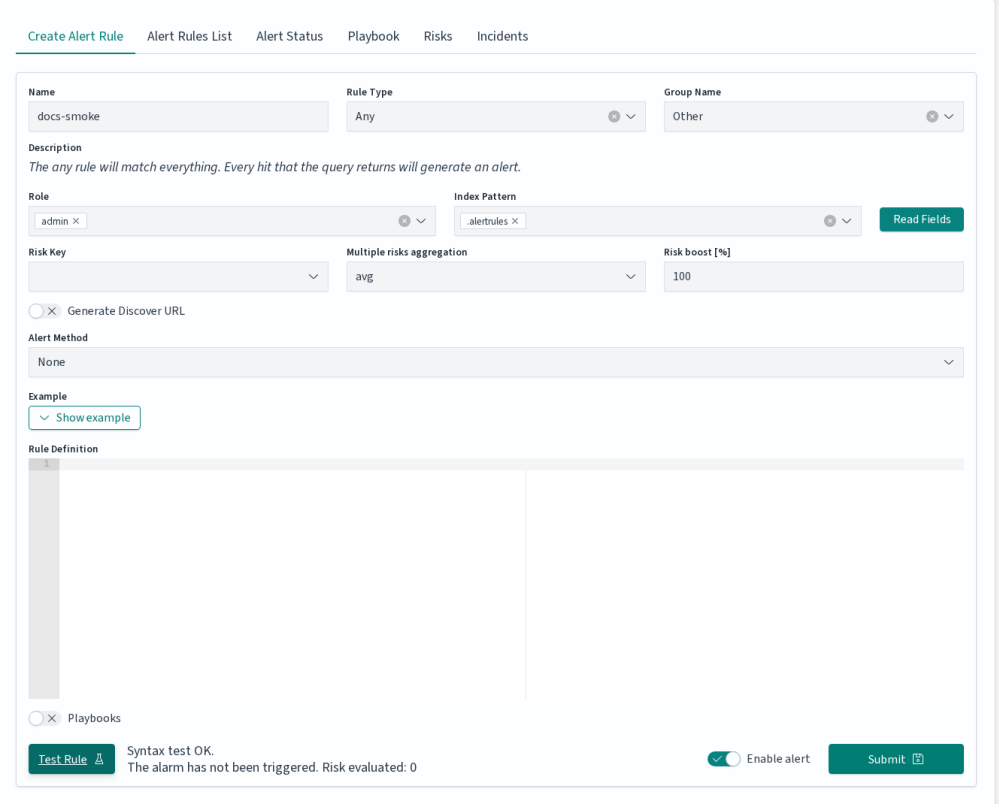
Alert Creation

Navigation: ELS Console → SIEM → Alerts → **Create Alert Rule**

The alert creation form is a single-page form with the following fields:

Basic Identity

- **Name**: descriptive name for the alert rule
- **Rule Type**: type of detection logic. The dropdown lists 19 types: Any, Blacklist, Blacklist-IOC, Cardinality, Chain, Change, Consecutive Growth, Difference, Difference Multi Pattern, Find Match, Flatline, Frequency, Logical, Metric Aggregation, New term, Percentage Match, Spike, Unique Long Term, Whitelist. Selecting a Rule Type updates the **Description** field dynamically with an explanation of the selected type.
- **Group Name**: organizes alerts into groups for management (e.g., Linux, Windows Security, Netflow, Correlated)
- **Description**: read-only field, auto-populated based on selected Rule Type


Create Alert Rule
Alert Rules List
Alert Status
Playbook
Risks
Incidents
Name
docs-smoke
Rule Type
Any
Group Name
Other
Description
The any rule will match everything. Every hit that the query returns will generate an alert.
Role
admin
Index Pattern
.alertrules
Read Fields
Risk Key
Multiple risks aggregation
avg
Risk boost [%]
100
Generate Discover URL
Alert Method
None
Example
Show example
Rule Definition
Playbooks
Test Rule
Syntax test OK.
The alarm has not been triggered. Risk evaluated: 0
Enable alert
Submit


Create Alert Rule
Alert Rules List
Alert Status
Playbook
Risks
Incidents
Name
Rule Type
Any
Group Name
Other
Description
Any
Blacklist
Blacklist-IOC
Cardinality
Chain
Change
Consecutive Growth
Role
Read Fields
Risk Key
Risk boost [%]
100

Scope

- **Role**: user role scope for the alert
- **Index Pattern**: index pattern to query (e.g., `windows-security-*`)
- **Read Fields**: button that loads available fields from the selected index pattern

Risk Settings

- **Risk Key**: field used as the risk key for scoring
- **Multiple risks aggregation**: aggregation method: avg, custom, max, min, sum. Default: avg
- **Risk boost [%]**: percentage multiplier for risk scoring. Default: 100

Discovery

- **Generate Discover URL**: toggle (off by default). When enabled, generates a Discover link attached to the alert notification

> **Note**
>
> If you reference the Discover link directly in the Rule Definition body instead of using this toggle, use the field name `discovery_url`. The earlier name `kibana_discovery_url` was removed in 7.9.0 and no longer resolves.

Notification

- **Alert Method**: delivery method for alert notifications. Available options: Command, Email, Energy SOAR, ITRS OP5 Monitor, Jira, None, ServiceNow, Slack, Syslog, TheHive Project, User, WebHook Connector

Rule Definition

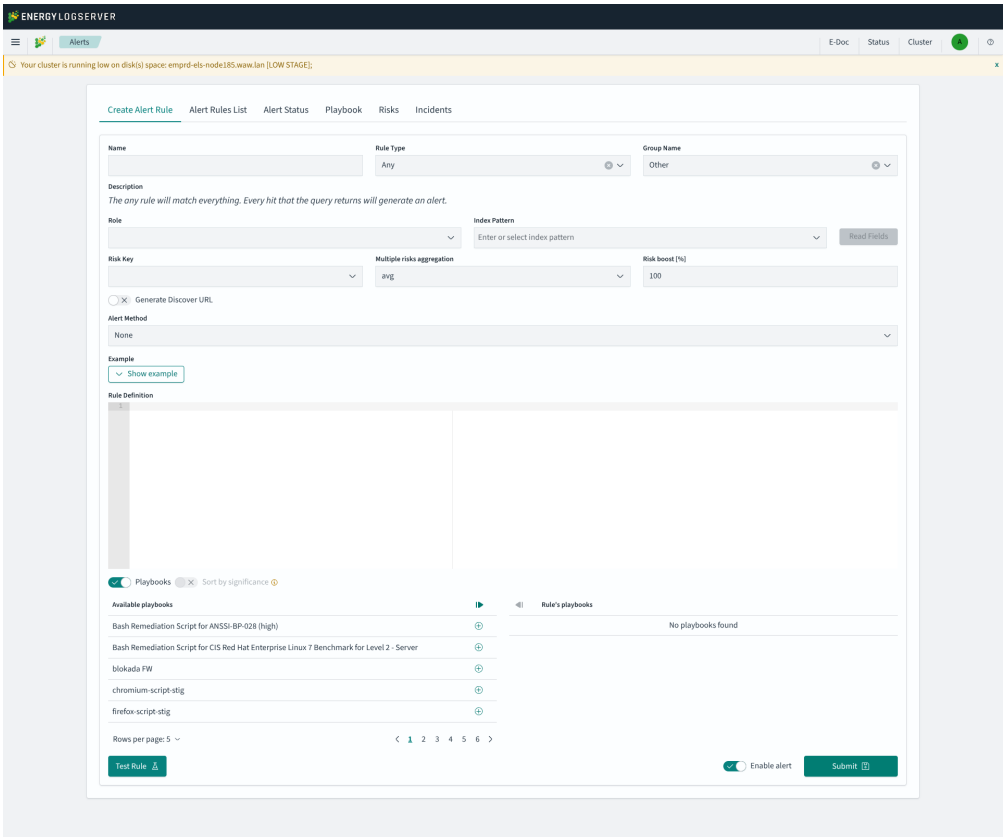
- **Example**: collapsible section with a YAML example for the selected Rule Type. Click **Show example** to expand the example pane with a ready-made snippet matching the current Rule Type; copy the blocks you need into the Rule Definition editor below.



- **Rule Definition**: code editor for writing the alert rule in YAML format

Playbooks

- **Playbooks**: toggle (off by default). When enabled, the form reveals two adjacent lists (**Available playbooks** and **Rule's playbooks**) populated from the Playbook library. Use the arrow icon in each row to move a playbook between lists; the paginated **Available playbooks** list supports 5 rows per page.
- **Sort by significance**: toggle shown next to Playbooks. When enabled, the Rule's playbooks list is reordered by the significance weight stored with each playbook (higher weight first).



Form Actions

- **Test Rule**: runs a syntax test and a dry-run of the rule without saving it. The form is validated first (Name, Role and Index Pattern are required); if anything is missing a toast such as `Form is not properly filled. Missing: alert name; index pattern;` is shown. On success the result appears inline next to the button, for example `Syntax test OK. The alarm has not been triggered. Risk evaluated: 0`. No notification is dispatched and the rule is not persisted.



- **Enable alert**: toggle to set the alert as active upon creation (on by default)
- **Submit**: saves and creates the alert rule

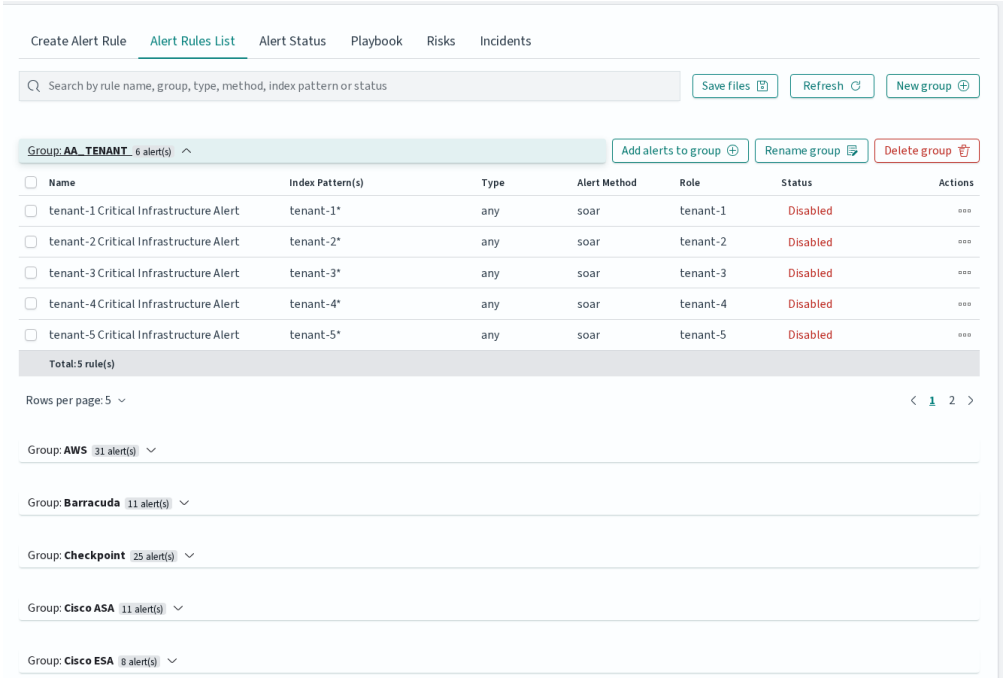
List of Alert Rules

The **Alert Rules List** tab displays all created alert rules organized in collapsible groups. The toolbar provides a search bar for filtering by rule name, group, type, method, index pattern, or status, as well as buttons for saving rule files, refreshing the list, and creating new groups.

Each group can be expanded to show individual rules. Group-level actions include **Add alerts to group**, **Rename group**, and **Delete group**.

Each alert rule row contains the following columns: Name, Group, Index Pattern(s), Type, Alert Method, Role, and Status. The Status column is a clickable toggle that enables or disables the rule directly.

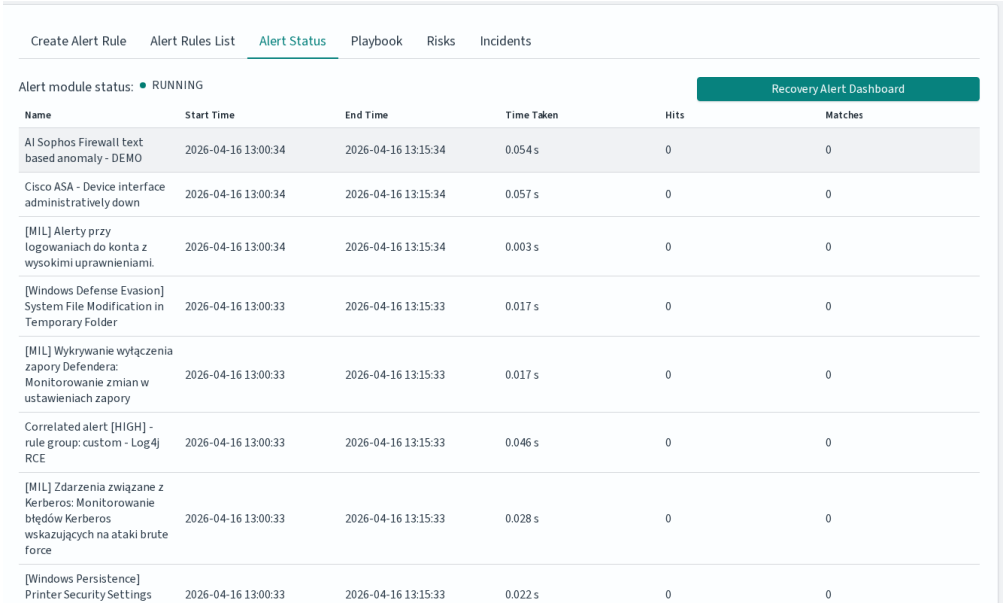
Per-rule actions are available as icon buttons: **Run once** (executes the rule against historical data with a configurable



date range), **Show** (opens read-only view), **Update** (opens the rule for editing), and **Delete** (removes the rule after confirmation).

When multiple rules are selected via checkboxes, a bulk action bar appears with options to **Enable**, **Disable**, **Change roles**, **Change group**, or **Delete** the selected rules.

Alerts Status



The **Alert Status** tab displays the alert module status (Running or Stopped) and a table with the following columns:

Column	Description
Name	Alert rule name
Start Time	When the rule check started
End Time	When the rule check finished
Time Taken	Duration of the check in seconds
Hits	Number of matching events found
Matches	Number of times the rule condition was met

The **Recovery Alert Dashboard** button restores the default alert dashboard if it has been modified or deleted.

Alert Rule Types

Alert rule types define how the system detects events. Available types:

Basic Alerts

- **Any:** Triggers on any matching event.
- **Blacklist:** The blacklist rule will check a certain field against a blacklist, and match if it is in the blacklist.
- **Blacklist-IOC:** The blacklist rule will check a certain field against a blacklist, and match if it is in the blacklist (can be used for large files).
- **Cardinality:** This rule matches when the total number of unique values for a certain field within a time frame is higher or lower than a threshold.
- **Change:** This rule will monitor a certain field and match if that field changes. The field must change with respect to the last event with the same query_key.
- **Consecutive Growth:** Rule matches for value difference between two aggregations calculated for different periods in time.
- **Difference:** This rule calculates percentage difference between aggregations for two non-overlapping time windows.
  - **Required parameters:**
    - `type: difference` - Enable the rule.
    - `compare_key: value` - Field for aggregation calculation.
    - `threshold_pct: 10` - Alert triggers when percentage difference exceeds this value.
    - `delta_min: 3` - Difference in minutes between calculated aggregations.
    - `agg_min: 1` - Aggregation bucket in minutes.
  - **Optional parameters:**
    - `query_key: hostname` - If present, calculates aggregation for each unique query_key (must be of type keyword).
  - **Calculation:**
    - For current time `x`, historical and present time windows are defined as: `<x - agg_min - delta_min; x - delta_min> <= <x - agg_min; x>`, where `x - delta_min <= x - agg_min => delta_min >= agg_min`.

    * Percentage difference: `d = | avg_now - avg_history | / max(avg_now, avg_history) * 100` (if `avg_now - avg_history != 0`, `avg_now != 0`, `avg_history != 0`); otherwise, `d = 0`.
    * `avg_now` is the arithmetic mean of `<x - agg_min; x>`, and `avg_history` is the arithmetic mean of `<x - agg_min - delta_min; x - delta_min>`.

- **Difference Multi Pattern:** The rule matches the difference in values between two aggregations from two index patterns calculated in a unit of time.
- **Find Match:** Rule matches when, in a defined period of time, two correlated documents match certain strings.
- **Flatline:** Triggers when event count drops below threshold.
- **Frequency:** This rule matches when there are at least a certain number of events in a given time frame.
- **Metric Aggregation:** Aggregates metrics and alerts on thresholds.
- **New Term:** Detects new values in a monitored field.
- **Percentage Match:** This rule matches when the percentage of documents in the match bucket within a calculation window is higher or lower than a threshold.
- **Spike:** Detects sudden increases in event frequency.
- **Unique Long Term:** This rule matches when there are values of compare_key in each checked timeframe.
- **Whitelist:** Similar to blacklist, this rule will compare a certain field to a whitelist, and match if the list does not contain the term.

Advanced Alerts

- **Chain:** Combines multiple rules. Rule matches when a complex, logical criteria is met. Rule can be used for alert data correlation.
  - **Example:**
    * Alerts that must occur:
      · Linux - Login Failure: Must appear 10 times.
      · AND
      · Linux - Login Success: Must appear 1 time.
    * If the sequence of these alerts occurs within 5 minutes and the values of the "username" field are related, the alert rule is triggered. The order of occurrence is important.
- **Logical:** Rule matches when a complex, logical criteria is met. Rule can be used for alert data correlation.
  - **Example:**
    * Alerts that must occur:
      · Switch - Port is off-line: Must appear 5 times.
      · OR
      · Switch - Port is on-line: Must appear 5 times.
    * If both alerts are met within 5 minutes and the values of the "port_number" field are related, the alert rule is triggered. Logical connectives such as OR, AND, NOR, NAND, XOR can be used.

The wizard offers exactly the types listed above. **Spike Aggregation** exists in the rule engine and in the option matrix below, but the wizard does not offer it; create such a rule from raw YAML or through the API.

Rule type options reference

Every rule type accepts options from two shared groups, **common options** and **additional options**: plus its own required and rule-specific optional parameters shown in the matrix below (`Req` = required, `Opt` = optional, empty cell = not applicable to this rule type).

Common options (apply to all rule types)

Alerter-specific settings (`alerta_*`, `alertmanager_*`, `jira_*`, `kibana_discover_*`, etc.) are not listed here.

Option	Type	Description
`name`	string	Rule name; defaults to the rule filename
`description`	string	Human-readable description
`type`	string	Rule type (`any`, `frequency`, `spike`, `blacklist_ioc`, `chain`, etc.)
`filter`	array	Query DSL filters applied to every search
`index`	string	Index pattern to query
`priority`	integer	1 (highest) to 5 (lowest), default 2
`owner`	string	Owner of the rule
`query_key`	string or array	Dimension to group matches by
`timestamp_field`	string	Event timestamp field; default `@timestamp`
`realert`	time	Minimum time between alerts for the same rule; default 1 min. Not honored by `Blacklist-IOC` and `Difference` rules, both override it to `0`
`realert_key`	string	Override the key used for realert grouping
`exponential_realert`	time	Maximum realert time when applying exponential backoff
`aggregation`	time	Aggregate matches within this time window into one alert
`buffer_time`	time	Query window size; default from `config.yaml`
`query_delay`	time	Delay before querying newly arrived data
`use_local_time`	boolean	Display timestamps in server local time
`include`	array	Fields to include in matches; default `["*"]`
`include_fields`	array	Explicit subset of fields
`top_count_keys`	array	Fields whose top N values are listed in the alert. The engine queries the field names as written; `raw_count_keys: true` appends `.keyword` to them (default `false` since 8.1.0)

continues on next page

Table 3 – continued from previous page

Option	Type	Description
`top_count_number`	integer	N for `top_count_keys`; default 5
`alert_text`	string	Custom alert message template
`alert_text_args`	array	Fields substituted into `alert_text` via positional args
`alert_text_kw`	object	Named keyword substitutions for `alert_text`
`alert_subject`	string	Custom alert title, used as the e-mail subject or ticket summary; without it the alerter builds a default title
`alert_subject_args`	array	Fields substituted into `alert_subject` via positional args
`alert_subject_max_len`	integer	Longer titles are truncated to this many characters; default 2048
`alert_text_type`	enum	`alert_text_only`, `alert_text_jinja`, `exclude_fields`, or `aggregation_summary_only`
`alert_missing_value`	string	Placeholder for missing fields in alert text; default `<MISSING VALUE>`
`match_enhancements`	array	Python class paths to run on each match
`jinja_filters`	array	Python class paths providing custom Jinja filters
`summary_table_fields`	array	Fields shown in the alert summary table. Each item is a field name or an object with `path` (the field) and an optional `header` (the column label); see *Aggregation summary table*
`summary_table_type`	enum	`ascii`, `html`, or `markdown`
`scan_entire_timeframe`	boolean	Re-scan the full timeframe for every query
`import`	string or array	Rule file(s) to inherit from

Additional options

These options apply to every rule type.

Option	Type	Description
recovery	boolea	When true, fire a recovery alert once the rule's condition clears
recovery_comma	string	Shell command to execute on recovery
escalate_after	time	Escalate the alert if it is not acknowledged within this window
escalate_users	ar-ray	Users to notify on escalation
calendar	ob-ject	Object with a schedule sub-key holding a cron expression; the alert only fires when the match timestamp matches the expression
risk_key	string	Comma-separated event fields whose values look up the risk category
risk_key_aggre	enum	Combine multiple risk values with MIN, MAX, SUM, AVG; any other value (including the GUI's custom) falls through to MAX
rule_importanc	inte-ger	Multiplier applied to the aggregated risk score, in percent (GUI: "Risk boost [%]")

Dynamic alert titles

Set alert_subject in the **Rule Definition** editor to build the alert title from the event that triggered the rule. Every alerter that sends a one line summary uses it, including the e-mail subject, the Slack and Discord message title, the PagerDuty summary and the ticket title in Jira or TheHive. Without it the rule falls back to a generated title, usually the rule name.

Write the title as a Python format string and list the fields that fill it:

```
alert_subject: "Failed login on {} from {}"
alert_subject_args:
  - host.name
  - source.ip
```

Numbered placeholders such as {0} work as well. Field names with dots resolve into nested documents, and "@timestamp" has to be quoted, because YAML rejects a bare token starting with @.

Only the first match populates the arguments, so a rule that aggregates many events shows values from one of them. When a field is missing from the event, the engine looks for a rule option of the same name, and only then substitutes alert_missing_value, by default <MISSING VALUE>. Rules that run with use_count_query hit the same path, because the counted result carries no fields. Titles longer than alert_subject_max_len are truncated, the default limit being 2048 characters.

Rules that render their body with Jinja use the template syntax in the title as well:

```
alert_text_type: alert_text_jinja
alert_subject: "{{ rule_name }}: {{ user.name }} on {{ host.name }}"
```

Fields whose keys contain dots are reachable through the _data dictionary, for example {{ _data['host.name'] }}, and the name of that dictionary can be changed with jinja_root_name.

The same substitution rules apply to alert_text_args, which fills the alert body. Two match fields are easy to misread there: num_matches counts the matches the rule produced in that run, not the events behind a single match, and num_hits counts every hit the query returned for the whole rule. Neither is the per key event count, so a frequency rule cannot print how many events one client sent. Reference the rule option instead, for example num_events, and phrase the line as a threshold.

Two alerters keep their own title setting and ignore alert_subject: OpsGenie reads opsgenie_subject, and Slack overrides the title with slack_title when the Slack notification is configured in the wizard. The Energy SOAR notification fills alert_subject and alert_subject_args from its own **Title** fields, so set the title there rather than in the rule definition.

Aggregation summary table

A rule with `aggregation` can open the alert body with a table that counts the aggregated matches. List the columns in `summary_table_fields`. A plain item uses the field name as the column label, an object sets its own label in `header`:

```
aggregation:
  minutes: 10
summary_table_fields:
  - path: user
    header: User
  - action
summary_table_type: ascii
summary_table_max_rows: 3
```

The engine adds a `count` column and sorts rows by it:

```
Aggregation resulted in the following data for summary_table_fields ==> ['user
↪', 'action', 'count']:

+-----------+--------+-------+
|   User    | action | count |
+===========+========+=======+
| root      | ALLOW  | 5     |
+-----------+--------+-------+
| jsmith    | DROP   | 3     |
+-----------+--------+-------+
| akowalska | ALLOW  | 2     |
+-----------+--------+-------+
```

The first line lists field paths, not the labels. `summary_table_type` also accepts `markdown` and `html`, and `alert_text_type: aggregation_summary_only` sends the table without the match details.

Rule type options reference

`use_keyword_postfix` for **New Term** defaults to `false` since 8.1.0, so the rule queries the field exactly as written. The default index template `default-base-template-0` maps string fields as `keyword` with no `.keyword` sub-field. With `use_keyword_postfix: true` the rule collects no known values and reports every value as new.

* `fields` for **New Term** is technically optional in the schema but the engine enforces at runtime that either `fields` or `query_key` is set, omitting both causes rule load failure.

* `compare_key1` / `compare_key2` for **Diff Multi Pattern** are optional when `pattern_agg_type1` / `pattern_agg_type2` is `count`, but required for any other aggregation type (default `avg`). Rule load fails otherwise.

`corelate_field_names` is spelled with a single *r* in the rule engine. Use this exact spelling, `correlate_field_names` silently disables correlation.

Spike Aggregation (`SpkA`) is defined in source and schema but is not selectable in the Alert Rule Wizard, use raw YAML or the API.

Risk Framework

The Risk Framework assigns a numeric risk score to every incident by mapping field values from the event to user-defined risk **Categories**. The **Risks** tab under Alerts is where these mappings live.

Navigation: ELS Console → SIEM → Alerts → **Risks**

The tab has four sub-tabs:

Sub-tab	Purpose
Create Risk	Assign one or many field values to a category
Risk List	Browse, edit and delete existing (field name, field value, category) assignments
Create Category	Define a new risk category and its numeric value
Category List	Browse and edit existing categories

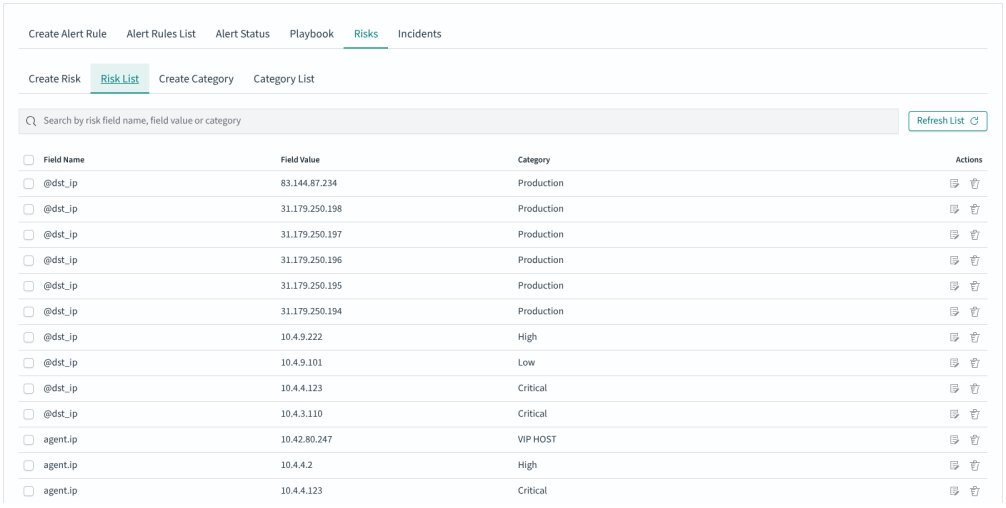
Risk Categories

A Risk Category is a named bucket with a numeric weight. Shipped categories commonly include `Production`, `High`, `Low`, `Critical`, `VIP HOST`; installations can define additional categories under **Create Category**.

Defining risk values

1. Open **Risks** → **Create Risk**.
2. Pick an **Index Pattern** and click **Read Fields** to load the fields.
3. Choose the field whose values should carry risk (for example `agent.ip`, `@dst_ip`, `user.name`).
4. Set the **Time Range** (default: last 15 minutes) and click **Read Values** to load values actually seen in that range.
5. Select one or many values, pick the **Category** (or `uncategorized`), and click **Submit**.

The result appears in **Risk List**:



Scoring incidents

The risk score for a matched event is computed as follows:

1. The rule's **Risk Key** is read from the YAML as `risk_key`. It can be a single field name or a comma-separated list of field names.

2. For every risk key field, the event's value is looked up in the Risk List by field name and value. That list lives in the index set by `risk_index` in `/opt/alert/config.yaml` (default `.risks`). The entry returns a risk category id, which resolves to a numeric `categoryvalue` in the category index set by `category_index` (default `.riskcategories`). A value with no matching entry falls back to the built-in default category.
3. The collected values are aggregated according to **Multiple risks aggregation**, stored in YAML as `risk_key_aggregation` with uppercase values: `MIN`, `MAX`, `SUM`, `AVG`. The GUI also shows a `custom` option, a value not recognised by the aggregator falls through to `MAX`, which is the same behavior as not setting `risk_key_aggregation` at all.
4. The aggregate is divided by 100, multiplied by the rule's **Risk boost [%]** (YAML: `rule_importance`) also divided by 100, then multiplied by 100. In plain terms: `risk = aggregate × rule_importance ÷ 100`. The wizard fills **Risk boost** with 100, so the final score equals the aggregate category value. In a hand-written rule the field matters: with `rule_importance` omitted the engine reads it as 0 and every incident scores 0.

The final number is stored on the incident record (in the `alert` writeback index by default) and shown in the **Risk** column on the Incidents tab.

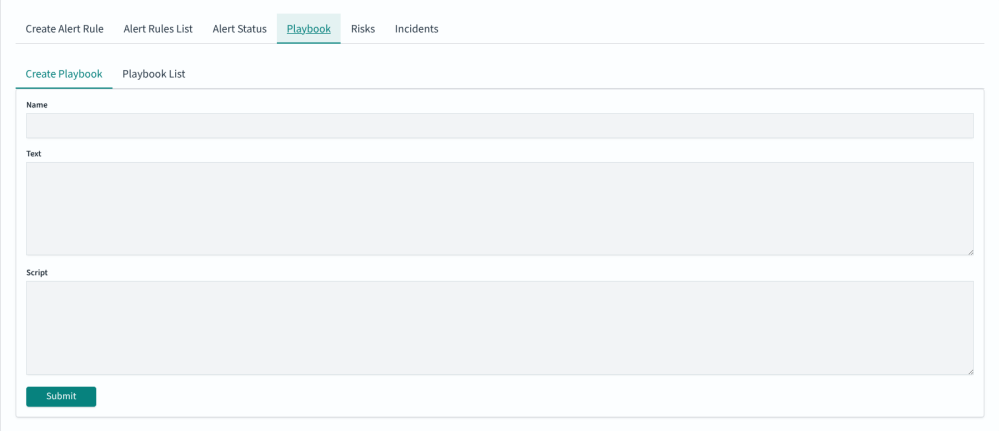
Playbooks

Playbooks are reusable response procedures attached to alert rules. Each playbook pairs a human-readable description with an optional executable script.

Navigation: ELS Console → SIEM → Alerts → **Playbook**

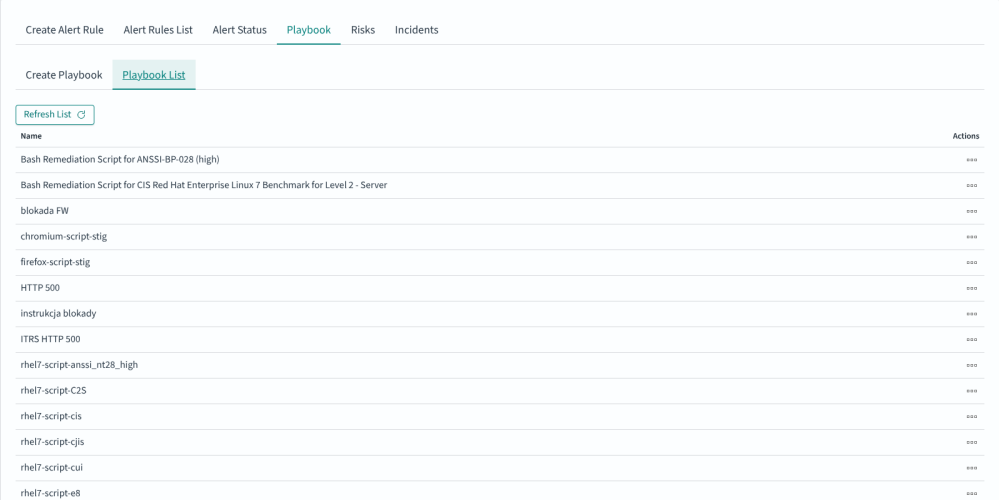
Creating a playbook

The **Create Playbook** sub-tab accepts three fields:



Field	Purpose
Nam	Identifier shown in the Playbook list and in the Available playbooks chooser
Text	Runbook content for the operator (Markdown-friendly, shown verbatim alongside a triggered incident)
Scrip	Optional script body displayed as read-only text alongside the incident. Typical uses include Bash remediation snippets for CIS or ANSSI benchmarks (see the shipped `rhel7-script-*` playbooks for reference). Scripts are **not executed automatically**: an operator copies and runs the script manually.

Click **Submit** to save. The new entry appears in **Playbook List**:



Attaching playbooks to an alert rule

In **Create Alert Rule**, enable the **Playbooks** toggle, then use the arrow icons to move playbooks from **Available playbooks** into **Rule's playbooks**. Multiple playbooks can be attached. When the rule triggers an incident, the attached playbooks are shown alongside the incident in the Incidents tab for the operator to review and act on. Playbook scripts are not executed automatically by the system.

Incidents

The **Incidents** tab lists triggered alerts in reverse chronological order. It is the operator-facing view: what fired, when, who is responsible, and how much risk the incident carries.

Navigation: ELS Console → SIEM → Alerts → **Incidents**

The table columns:

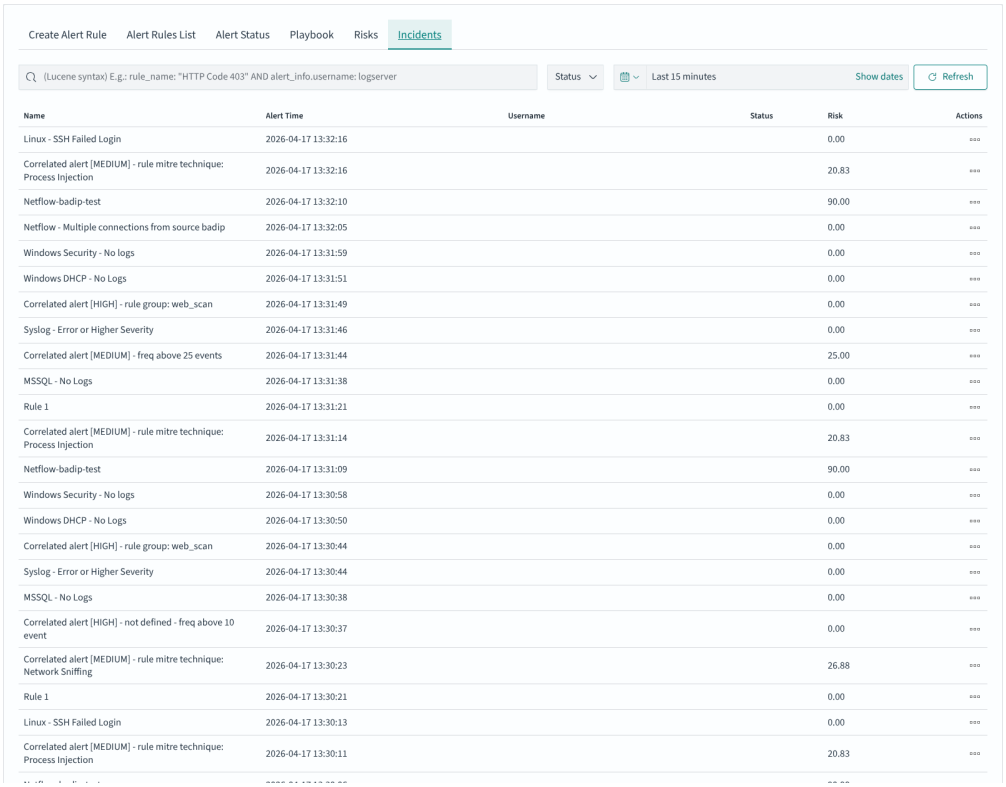
Column	Description
Name	Rule name that triggered the incident
Alert Time	Timestamp of the trigger
Username	Field value from the event used as actor (often `user.name`)
Status	Incident handling state, filter via the Status dropdown at the top
Risk	Calculated risk score (see *Risk Framework*)
Actions	Per-row icon menu: open the incident, view attached playbooks, change status

The search field accepts Lucene syntax and is evaluated against the alert writeback index (default name `alert`, configurable in `/opt/alert/config.yaml` via `writeback_index`); the date-range picker on the right bounds the result set (default: last 15 minutes). Use **Refresh** to re-fetch.

Working with Pre-built Alert Groups

Navigation: ELS Console → SIEM → Alert Rules

Energy Logserver ships with a library of pre-built alert rule groups organized by vendor, platform, and use case. These groups can be enabled selectively based on your environment.



For a rule-by-rule catalog of the alerts each integration installs, including detection type and index pattern, see *Predefined Alert Rules*.

Available Groups

Vendor-specific:

Group	Rules	Description
AWS	33	Amazon Web Services security events
Checkpoint	25	Check Point firewall events
Cisco ASA	11	Cisco ASA firewall events
Cisco ESA	8	Cisco Email Security Appliance events
Cisco VPN	5	Cisco VPN connection events
F5	3	F5 load balancer events
FireEye	16	FireEye threat detection events
Forcepoint	6	Forcepoint web/email security events
Fudo	5	Fudo PAM privileged access events
Infoblox	14	Infoblox DNS/DHCP/IPAM events
Juniper	5	Juniper network device events
McAfee	6	McAfee endpoint security events
Mikrotik	5	MikroTik router events
QualysGuard	6	Qualys vulnerability scanner events

Infrastructure:

Group	Rules	Description
Cluster-Health	25	ELS Data Node cluster health monitoring
HTTP	9	HTTP traffic anomaly detection
Switch	10	Network switch events

Operating systems and services:

Group	Rules	Description
Linux	25	Linux system security events
Microsoft Windows	1	General Windows security events
Microsoft Windows DHCP	11	Windows DHCP server events
Microsoft Windows DNS	12	Windows DNS server events
Exchange	6	Microsoft Exchange mail server events
Microsoft MsSQL	6	Microsoft SQL Server events
MySQL	10	MySQL database events
PostgreSQL	11	PostgreSQL database events
Postfix	6	Postfix mail server events
Squid	7	Squid proxy events

Security and correlation:

Group	Rules	Description
Correlated	30	Cross-source correlated security events
Empowered AI	31	AI-driven anomaly detection rules
Ransomware	8	Ransomware activity detection

Using Pre-built Groups

1. Browse the available groups in the Alert Rules List.
2. Expand a group to review included rules.
3. Enable or disable individual rules as needed.
4. Customize rule thresholds and parameters if required.
5. Save changes and monitor alert activity.

Escalation and Recovery

Escalate

The `escalate_users` parameter assigns alert notifications to a specific user or group of users. The `escalate_after` parameter escalates the notification recipient after a defined period of time.

Example:

```
escalate_users: ["user1", "user2"]
escalate_after:
  days: 2
```

In this example, `user1` and `user2` receive an escalation notification two days after the initial alert.

Recovery

The `recovery` parameter defines an additional action to perform after the conditions that triggered the initial alert are no longer met.

```
recovery: true
recovery_command: "command"
```

Field values can be referenced in the recovery command using two syntaxes:

- `%{field_name}`, pulls the value from the alert match (e.g. `%{@timestamp_recovery}`)
- `${field_name}`, pulls the value from the rule definition (e.g. `${name}`)

The `@timestamp_recovery` variable contains the time at which the recovery was executed.

Example:

```
recovery: true
recovery_command: "echo \"%{@timestamp_recovery};;${name}_${ci};${alert_
↪severity};RECOVERY\" >> /var/log/em_integration/events.log"
```

To send a recovery email notification:

```
recovery: true
recovery_command: "mail -s 'Recovery Alert for rule ${name}' user@example.com
↪< /dev/null"
```

Incident and Risk Management

Incident tracking, risk assessment, and response playbooks within the SIEM module.

Incident Management

Navigation: ELS Console → SIEM → Incidents

The Incidents tab displays all triggered alert incidents in a time-filtered, searchable table. Each incident row contains the following columns:

Column	Description
Name	Alert rule name that triggered the incident (shows a notepad icon if a note is attached)
Alert Time	Time when the alert was triggered
Username	User assigned to handle the incident
Status	Current incident status
Risk	Calculated risk score

Incident Statuses

Each incident has one of the following statuses:

- **New**: incident has been created but not yet reviewed
- **Ongoing**: incident is being actively investigated
- **False**: incident has been identified as a false positive

- **Solved**: incident has been resolved

Incident Actions

For each incident the following actions are available:

- **Show Incident**: view the full incident document
- **Verify**: check incident fields against the blacklist index
- **Preview**: open the incident document in Discover using the alert index pattern
- **Update**: change the incident status, assigned users, and roles
- **Playbooks**: view playbooks associated with the incident's alert rule
- **Note**: add or edit investigation notes and comments

Manual Incident Creation

Navigation: ELS Console → Discover → Incident (top right)

Incidents are typically created automatically when alert rules are triggered. Additionally, you can create incidents manually from the Discover interface using the **Manual Incident** context menu option. The form includes the following fields:

- **Rule Name**: defaults to "Manual Incident"
- **Time**: current time (read-only)
- **Risk**: risk value from 0 to 100 (default: 50)
- **User**: user to assign the incident to
- **Message**: description of the incident

Risk Management

Navigation: ELS Console → SIEM → Risks

Risk management allows you to categorize and score entities (users, hosts, IPs) by mapping field values to risk categories. The Risks tab has four sub-tabs: **Create Risk**, **Risk List**, **Create Category**, and **Category List**.

Risk Categories

A risk category defines a named risk level with a numeric value from 0 to 100.

Creating a category:

1. Go to the **Create Category** sub-tab.
2. Enter a **Category name** (e.g., "Critical Assets", "High Risk Users").
3. Set a **Category value** (0-100), where higher values represent higher risk.
4. Submit the category.

> **Note**
>
> A default `uncategorized` category with value 0 is created automatically on plugin initialization.

Risk Entries

A risk entry maps a specific field value from your data to a risk category.

Creating a risk entry:

1. Go to the **Create Risk** sub-tab.
2. Enter an **Index pattern** and click **Read Fields** to load available fields.
3. Select the **Key field** from the dropdown (e.g., `source.ip`, `user.name`).
4. Select a **Time range** (from "Last 15 minutes" up to "Last 1 year", or "None" for all data).
5. Click **Read Values** to load unique values for the selected field.
6. Map each value to a risk category using the per-row dropdown, or select multiple values and use **Set Global Category** for bulk assignment.
7. Optionally, add custom field values manually using the **+** button.
8. Submit the risk entries.

Risk Scoring on Alert Rules

Each alert rule can include risk scoring configuration:

- **Risk Key**: the field used to calculate risk (e.g., `source.ip`)
- **Multiple risks aggregation**: how to combine multiple risk values: MAX, MIN, AVG, SUM, or CUSTOM
- **Risk boost [%]**: a percentage multiplier applied to the calculated risk score (default: 100)

When an alert triggers, the system looks up the risk value for the entity identified by the Risk Key field and applies the aggregation method and boost percentage to calculate the final risk score. This score is visible in the Incidents tab.

Playbooks

Navigation: ELS Console → SIEM → Playbook

Playbooks provide a way to document response procedures and attach executable scripts to alert rules. The Playbook tab has two sub-tabs: **Create Playbook** and **Playbook List**.

Creating a Playbook

1. Go to the **Create Playbook** sub-tab.
2. Fill in the following fields:
   - **Name**: descriptive name for the playbook
   - **Text**: response procedure description (e.g., investigation steps, escalation instructions)
   - **Script**: executable script content to be run when the playbook is invoked
3. Submit the playbook.

Managing Playbooks

The **Playbook List** sub-tab displays all created playbooks in a table. Available actions for each playbook:

- **Show**: view playbook details
- **Update**: edit the playbook name, text, or script
- **Delete**: remove the playbook

Associating Playbooks with Alert Rules

Playbooks are linked to alert rules through the alert creation form:

1. When creating or editing an alert rule (SIEM → Alert Rules → Create Alert Rule), enable the **Playbooks** toggle.
2. Select one or more playbooks to associate with the rule.
3. When the alert triggers, the associated playbooks are available for the incident responder to reference and execute.

Daily Operations

SIEM Dashboard Operations

The SIEM Dashboard (Endpoint Security) provides agent-driven security monitoring with dedicated dashboards per security domain.

Daily Security Monitoring

Step 1: Review the Overview Dashboard

1. Navigate to the SIEM Overview dashboard.
2. Check the "Total Alerts" chart for unusual spikes.
3. Review "Top 5 Alerts" for immediate priorities.
4. Use the Rule Level slider to filter events by Endpoint Security rule level.

Step 2: Investigate High-Priority Areas

1. Click the relevant dashboard card based on detected issues (e.g., Windows, Vulnerabilities, Audit).
2. Use filters to focus on affected systems or time ranges.
3. Review related events in the MITRE ATT&CK dashboard for attack technique mapping.

Endpoint Security Rule Levels

The SIEM Dashboard uses Endpoint Security rule levels (1-15) to classify event severity:

Level range	Priority	Recommended action
1-3	Informational	Archive for compliance and audit trails
4-7	Low to medium	Review weekly for trends
8-11	High	Investigate promptly
12-15	Critical	Immediate response required

Use the Rule Level slider on the Overview dashboard to filter events by level and reduce noise during daily monitoring.

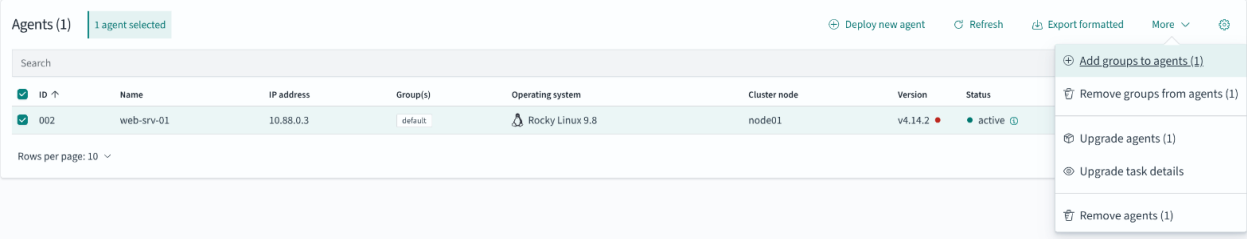
Compliance Monitoring

1. From the main SIEM dashboard, click the relevant compliance card:
   - **PCI DSS**: payment card processing environments
   - **HIPAA**: healthcare and patient data
   - **GDPR**: EU personal data processing
   - **NIST 800-53**: government and federal standards
   - **TSC**: SOC 2 trust services assessments
2. Review the compliance dashboard for current status, violations, and gaps.

Upgrading Agents

You can upgrade connected agents to the manager version from the GUI, without logging in to each host.

1. In the SIEM Engine menu, open **Agents > Summary**. A red dot next to the version marks an outdated agent.
2. Select the agents to upgrade. The checkbox in the table header selects all agents on the page.
3. Click **More > Upgrade agents**, then **Upgrade**.
4. Follow the progress in the **Upgrade tasks** panel. **Task details** lists each task with its status and error message.



The manager downloads the upgrade package (WPK) from `packages.wazuh.com` and sends it to the agent, so the manager node needs HTTPS access to that host.

Alerts Operations

The native Alerts interface provides alert rule management, incident handling, and risk-based prioritization.

Daily Alert Review

Step 1: Check Alert Status

1. Navigate to SIEM → Alert Status.
2. Verify the alert module status is "RUNNING".
3. Review processing times and event counts for anomalies.

Step 2: Review Incidents

1. Navigate to SIEM → Incidents.
2. Filter by status "New" to see unhandled incidents.
3. Prioritize incidents by risk value, higher scores indicate higher risk.
4. Assign incidents to team members and update statuses.

Step 3: Risk-based Prioritization

1. Navigate to SIEM → Risks.
2. Review risk entries for high-value categories.
3. Cross-reference with recent incidents to identify recurring high-risk entities.

Incident Investigation

1. Open the incident from the Incidents tab.
2. Click **Show Incident** to inspect the full incident document, or **Preview** to open it in Discover.
3. Use the Discover URL (if generated by the alert rule) to view related events in context.
4. Add investigation notes to the incident.
5. Verify suspicious fields against the blacklist.
6. Update the incident status as the investigation progresses (New → Ongoing → Solved or False).

Creating Alerts for New Threats

1. Navigate to SIEM → Create Alert Rule.
2. Enter a descriptive name and select the appropriate rule type.
3. Set the index pattern and click "Read Fields" to load available fields.
4. Write the YAML rule definition with detection logic.
5. Configure the alert method (Email, Slack, Jira, etc.).
6. Optionally enable risk scoring and associate playbooks.
7. Use "Test Rule" to validate before saving.
8. Submit the rule.

Performance and Integration

- *Monitoring Alert Performance*
- *Integration with Other Modules*
  - *External System Integration*
  - *Definition-based Notification*
- *Endpoint Security Capabilities*
  - *Enabling the Endpoint Security GUI*
  - *Connecting to the Endpoint Security Manager*
- *Windows Kernel-Level Protection Integration*
  - *Supported Protection Mechanisms*
  - *Integration with Energy Logserver*
  - *Monitoring*

Monitoring Alert Performance

Navigation: ELS Console → SIEM → Alert Status

Check:

- Alert module status (Running or Stopped).
- Processing time per rule (Time Taken column).
- Number of matching events (Hits) and triggered matches (Matches).

Integration with Other Modules

The SIEM module works with data collected by the *Network Probe*, which provides network monitoring, IDS/IPS, and NetFlow analysis capabilities.

External System Integration

UI Notification

- **Email:** Sends information about an alert to defined email addresses. SMTP settings are configured in `/opt/alert/config.yaml` (see *Enabling the Alert Module*). Per-rule email settings include:
  - `email`, list of recipient addresses (required)
  - `cc`, CC recipients
  - `bcc`, BCC recipients
  - `email_reply_to`, reply-to address
  - `email_format`, set to `html` for HTML-formatted emails
- **User:** Sends information about an alert to defined system users.
- **Command:** Performs system tasks, e.g., triggers a script to create a new event in the customer ticket system.

Example of an alert rule definition using the "command" alert method:

```
index: op5-*
name: change-op5-hoststate
type: change

compare_key: hoststate
ignore_null: true
query_key: hostname

filter:
- query_string:
    query: "_exists_: hoststate AND datatype: \"HOSTPERFDATA\" AND _exists_:␣
↪hostname"

realert:
  minutes: 0
alert: "command"
command: ["/opt/alert/send_request_change.sh", "5", "%(hostname)s", "SYSTEM_
↪DOWN", "HOST", "Application Collection", "%(hoststate)s", "%(@timestamp)s"]
```

- **The Hive:** Creates an Incident in The Hive.

Configuration:

1. Edit `/opt/alert/config.yaml`:

```
hive_connection:
  hive_host: https://127.0.0.1/base
  hive_apikey: APIKEY
```

2. Rule definition options:
   - `Alert type`: Type of alert (alert or Case)
   - `Follow`: If enabled, updates set status to Updated and update related case.
   - `Title`: The title of alert.
   - `Description`: Description of the alert.
   - `Type`: The type of the alert.
   - `Source`: The source of the alert.
   - `Status`: The status of the alert (New, Ignored, Updated, Imported).
   - `Severity`: The severity of alert (low, medium, high, critical).
   - `TLP`: The Traffic Light Protocol (white, green, amber, red).
   - `Tags`: Tags attached to alert.
   - `Observable data mapping`: Key and value observable data mapping.
   - `Alert text`: Text content of the alert.

- **RSA Archer:** Forwards alert information to RSA Archer via Command alert method.

Configuration steps:

1. Save scripts to `/opt/alert/bin`:
   - `ucf.sh` (SYSLOG):

```
#!/usr/bin/env bash
base_url="http://localhost/Archer" ##set the appropriate Archer URL

logger -n $base_url -t logger -p daemon.alert -T "CEF:0|LogServer|LogServer|${19}|${18}| TimeStamp=$1 DeviceVendor/Product=$2-$3 Message=$4␣
↪TransportProtocol=$5 Aggregated:$6 AttackerAddress=$7 AttackerMAC=$8␣
↪AttackerPort=$9 TargetMACAddress=${10} TargetPort=${11} TargetAddress=${12}␣
↪FlexString1=${13} Link=${14} ${15} $1 ${16} $7 ${17}"
```

   - `ucf2.sh` (REST API):

```
#!/usr/bin/env bash
base_url="http://localhost/Archer" ##set the appropriate Archer URL
instance_name="Archer"
username="apiuser"
password="Archer"

payload=$(cat <<EOF
{"CEF":"0","Server":"LogServer","Version":"${19}","NameEvent":"${18}",
↪"TimeStamp":"$1","DeviceVendor/Product":"$2-$3","Message":"$4",
```

(continues on next page)

(continued from previous page)

```
↪"TransportProtocol":"$5","Aggregated":"$6","AttackerAddress":"$7",
↪"AttackerMAC":"$8","AttackerPort":"$9","TargetMACAddress":"${10}",
↪"TargetPort":"${11}","TargetAddress":"${12}","FlexString1":"${13}","Link":"$
↪{14}","EventID":"${15}","EventTime":"${16}","RawEvent":"${17}"}
EOF
)

curl -k -u "$username:$password" -H "Content-Type: application/json" -X POST "
↪$base_url:50105/$instance_name" -d "$payload"
```

2. Alert rule definition:

```
index: alert*
name: alert-sent-to-rsa
type: any

filter:
- query:
    query_string:
      query: "_exists_: endTime AND _exists_: deviceVendor AND _exists_:␣
↪deviceProduct AND _exists_: message AND _exists_: transportProtocol AND _
↪exists_: correlatedEventCount AND _exists_: attackerAddress AND _exists_:␣
↪attackerMacAddress AND _exists_: attackerPort AND _exists_:␣
↪targetMacAddress AND _exists_: targetPort AND _exists_: targetAddress AND _
↪exists_: flexString1 AND _exists_: deviceCustomString4 AND _exists_:␣
↪eventId AND _exists_: applicationProtocol AND _exists_: rawEvent"

include:
- endTime
- deviceVendor
- deviceProduct
- message
- transportProtocol
- correlatedEventCount
- attackerAddress
- attackerMacAddress
- attackerPort
- targetMacAddress
- targetPort
- targetAddress
- flexString1
- deviceCustomString4
- eventId
- applicationProtocol
- rawEvent

realert:
  minutes: 0
```

- Alert Method: `command`
- Path to script/command: `/opt/alert/bin/ucf.sh`
- **Jira:** Opens a ticket on Jira when an alert is triggered.

Configuration steps:

1. Create `/opt/alert/jira_auth_file.yml`:

```
user: user.example.com
password: IjP0vVhgrjkotElFf4ig03g6
```

2. Edit `/opt/alert/config.yaml`:

```
jira_auth_file: "/opt/alert/jira_auth_file.yml"
jira_server_url: "https://example.atlassian.net"
```

3. Rule definition options:
   - **Required:**
     - `project`: The name of the Jira project.
     - `issue type`: The type of Jira issue.
   - **Optional:**
     - `Components`: Component(s) to set the ticket to.
     - `Labels`: Label(s) to set the ticket to.
     - `Watchers`: User ID(s) to add as watchers.
     - `Priority`: Issue priority (Lowest, Low, Medium, High, Highest).
     - `Bump tickets`: If true, comments on existing tickets newer than `max_age`.
     - `Bump Only`: Skips ticket creation if no ticket to bump.
     - `Bump in statuses`: Statuses for commenting instead of new ticket.
     - `Ignore in title`: Removes value from Jira subject when searching.
     - `Max age`: Maximum ticket age for bumping (default: 30 days).
     - `Bump not in statuses`: Statuses to exclude for bumping.
     - `Bump after inactivity`: Comments only on inactive tickets (default: 0 days).
     - `Transition to`: Transitions ticket to given status when bumping.

- **Webhook Connector:** Sends POST or PUT request to a web service.
- `URL`: Host of application or web application.
- `Username`: Username used to send alert.
- `Password`: Password of the username.
- `Proxy address`: Proxy address.
- `Headers`: Request headers.
- `Static Payload`: Static payload of the request.
- `Payload`: Payload of the request.

To enable `https`, add to `/opt/alert/config.yaml`:

```
webhook_verify_ssl: true
webhook_ca: /etc/logserver/ssl/rootCA.crt
webhook_cert: /etc/logserver/ssl/logserver.crt
webhook_key: /etc/logserver/ssl/logserver.key
```

- **Slack:** Sends notification to a predefined Slack channel.
- `Webhook URL`: URL with auth data and channel ID (from Slack's Incoming Webhooks).
- `Username`: Username or email address in Slack.
- `Slack channel`: Channel name (default if empty).
- `Message Color`: Message color (default: 'danger').
- `Message Title`: Title of the Slack message.
- **ServiceNow:** Creates a new Incident in ServiceNow.

Configuration steps:

1. Edit `/opt/alert/config.yaml`:

```
servicenow_auth_file: "/opt/alert/servicenow_auth_file.yml"
servicenow_rest_url: "https://dev123.service-now.com/api/now/v1/table/incident
↪"
```

2. Create `/opt/alert/servicenow_auth_file.yml`:

```
username: exampleUser
password: exampleUserPassword
```

- `Short Description`: Incident description.
- `Comments`: Work notes or comments.
- `Assignment Group`: Group to assign the incident.
- `Category`: Incident category (must exist).
- `Subcategory`: Incident subcategory (must exist).
- `CMDB CI`: Configuration item to link.
- `Caller Id`: Caller's email address.
- `Proxy`: Proxy address if needed.
- **Energy SOAR:** Creates a new Incident in Energy SOAR.

Configuration:

1. Edit `/opt/alert/config.yaml`:

```
hive_connection:
  hive_host: https://127.0.0.1/base
  hive_apikey: APIKEY
```

2. Rule definition options:
   - `Alert type`: Type of alert (alert or Case).
   - `Follow`: Updates status to Updated and related case if enabled.

- `Title`: Alert title.
- `Description`: Alert description.
- `Type`: Alert type.
- `Source`: Alert source.
- `Status`: Alert status (New, Ignored, Updated, Imported).
- `Severity`: Alert severity (low, medium, high, critical).
- `TLP`: Traffic Light Protocol (white, green, amber, red).
- `Tags`: Tags attached to alert.
- `Observable data mapping`: Key and value mapping.
- `Alert text`: Alert content text.

- **ITRS OP5 Monitor:** Submits a passive check result to an ITRS OP5 Monitor instance.

Configuration:

1. Edit `/opt/alert/config.yaml`:

```
op5_auth_file: "/opt/alert/op5_auth_file.yml"
op5_url: "https://op5.example.com"
```

2. Create `/opt/alert/op5_auth_file.yml`:

```
username: apiuser
password: apipassword
```

3. Rule definition options:
   - `Host`: Monitored host name in OP5.
   - `Service`: Service description in OP5.
   - `Status`: Check result status (0=OK, 1=WARNING, 2=CRITICAL, 3=UNKNOWN).
   - `Message`: Plugin output message.

- **Syslog:** Forwards alert notifications to a remote syslog server.
- `Host`: Syslog server hostname or IP address.
- `Port`: Syslog server port (default: 514).
- `Protocol`: Transport protocol, `tcp` or `udp` (default: udp).
- `Level`: Syslog severity level, CRITICAL, ERROR, WARNING, INFO, or DEBUG (default: WARNING).
- `Facility`: Syslog facility code (default: 16).

Definition-based Notification

- **Telegram Alerts**

Required settings:

- `telegram_bot_token`: Authorization token (e.g., 110201543:AAHdqTcvCH1vGWJxfSeofSAs0K5PALDsaw) from BotFather.
- `telegram_room_id`: Chat or channel ID (e.g., -xxxxxxxx).

Optional settings:

- `telegram_api_url`: Custom Telegram Bot API domain (default: api.telegram.org).
- `telegram_proxy`: Proxy in hostname:port format if needed.
- **Google Chat Alerts**

Required settings:

- `googlechat_webhook_url`: Webhook endpoint (from Google Chat → Configure Webhooks).

Optional settings:

- `googlechat_format`: Notification layout (basic or card).
- `googlechat_header_title`: Header title for card format.
- `googlechat_header_subtitle`: Header subtitle for card format.
- `googlechat_header_image`: URL to image/icon for card header.
- `googlechat_footer_kibanalink`: URL to Energy Logserver UI for card footer.

Endpoint Security Capabilities

Energy Logserver integrates with the Endpoint Security engine to provide endpoint detection capabilities. The following features are available when the SIEM Plan is active:

- **File Integrity Monitoring (FIM)**: monitors file system changes on monitored endpoints. Results are visible in the SIEM → FIM dashboard.
- **Log Data Collection**: collects logs from endpoints, applications, and system sources via the Endpoint Security agent. Collected data is forwarded to the Energy Logserver Data Node.
- **Active Response**: automatically executes response actions on endpoints when alert conditions are met (e.g., blocking IPs via firewall rules).
- **Vulnerability Detection**: scans endpoints for known vulnerabilities. Results are visible in the SIEM → Vulnerability dashboard.
- **Third-party alert forwarding**: sends alerts to external services such as VirusTotal (file-hash lookup driven by FIM events) or user-supplied scripts that forward alerts to ticketing systems, messaging platforms, or custom APIs.

Configuration is performed in the agent configuration file (`ossec.conf`) on each monitored endpoint.

For agent installation steps, see *SIEM Agents Configuration*.

Enabling the Endpoint Security GUI

The Endpoint Security dashboard plugin is disabled by default. To enable it, set `wazuh.enabled` to `true` in the GUI configuration file and restart the service:

```
# /etc/logserver-gui/logserver-gui.yml
wazuh.enabled: true
```

```
systemctl restart logserver-gui
```

Connecting to the Endpoint Security Manager

The plugin connects to the Endpoint Security Manager API. Configure the connection in:

```
/usr/share/logserver-gui/data/wazuh/config/wazuh.yml
```

The default configuration connects to a local manager instance:

```
hosts:
  - default:
      url: https://localhost
      port: 55000
      username: wazuh-wui
      password: wazuh-wui
      run_as: true
```

Replace `https://localhost` with the manager's address when the Endpoint Security Manager runs on a separate host. The `username` and `password` must match a valid API user account on the manager.

> **Note**
>
> This section covers ELS Endpoint Security Agent capabilities only. For the platform-level integrations catalog (Network Probe pipelines, dashboards, alerts shipped with the product), see *Integrations*.

Windows Kernel-Level Protection Integration

Energy Logserver integrates with a Windows kernel-level protection agent and gives you central monitoring of the file system operations that the agent blocks on your endpoints.

Supported Protection Mechanisms

- **Kernel Write Protection**: blocks unauthorized write operations in protected directories.
- **Kernel Anti-Tamper**: blocks deletion and renaming of protected files.

Integration with Energy Logserver

The agent sends each blocked operation to the Network Probe, which processes the event and indexes it in Energy Logserver. The deployment team sets up the pipeline and the dashboard for this integration.

Monitoring

Open the **Kernel-level Protection** dashboard from **Dashboards** in the sidebar menu. The dashboard shows blocked operations, affected processes, file paths, protection rules, and block status.

Troubleshooting

Dashboard Not Loading

1. Check your browser's console for JavaScript errors.
2. Verify your user has permissions for SIEM dashboards.
3. Try refreshing the page or clearing browser cache.
4. Check if the time range is too large, reduce to last 24 hours.

No Data Showing

1. Verify Endpoint Security agents are connected and reporting, check agent status in the SIEM Overview dashboard.
2. Confirm the selected time range includes the period when events occurred.
3. Check if filters are too restrictive.
4. Verify your user role has access to the relevant data sources.

Alerts Not Triggering

Alerts Not Firing

Diagnostic steps:

1. Check rule syntax - Verify YAML rule definition syntax.
2. Test with smaller dataset - Use Discover to test query logic.
3. Verify index patterns - Ensure data exists in specified indices.
4. Check time ranges - Confirm events fall within detection window.
5. Confirm the engine runs at all:

   ```
   systemctl status alert
   ```

6. Read the engine log. The alert service writes to `/var/log/alert/alert.log`, at level `ERROR`, so a rule that simply never matches leaves no entry there, while a rule that fails to load does:

   ```
   tail -f /var/log/alert/alert.log
   ```

   A rule that raises an error is disabled by the engine, because `/opt/alert/config.yaml` ships with `disable_rules_on_error: true`. Fix the rule and restart `alert` to load it again.

High False Positive Rate

Solutions:

1. Tune alert thresholds - Increase event counts or severity levels.
2. Add exclusions - Filter out known good activities.
3. Refine time windows - Adjust detection time frames.
4. Use risk scoring - Focus on high-risk events only.

Performance Issues

1. Reduce dashboard time ranges to improve loading speed.
2. Use filters to limit data volume.
3. Close unused dashboard tabs.
4. Contact administrator if cluster performance is degraded.

Energy Security Feed

Energy Security Feed is the security content pack shipped with the product. It groups three complementary areas:

- **IOC Threat Lists**: Malware Information Sharing Platform (MISP) indicators of compromise fetched from the Energy Logserver feed repository, indexed and distributed to Network Probe pipelines as translate dictionaries. Requires a **SIEM-PLAN** license (see *IOC Feed*).
- **Windows Events ID Repository**: a reference table of Windows Security Event IDs and the shipped Active Directory dashboards that consume them.
- **Security Rules**: a library of pre-defined correlation rules for the Alerts module, grouped by platform (cluster health, Windows, Cisco ASA, Fortigate, Palo Alto, and more).

IOC Feed (MISP Threat Lists)

The IOC component watches incoming logs and marks documents whose content matches the Malware Information Sharing Platform (MISP) threat lists delivered with the Energy Security Feed. Marked events can be routed to alert rules or used to track incident behavior.

- *How it works*
- *Feed categories*
- *Feed source*
- *Feed file format*
- *Generated dictionary format*
- *Installation paths*
- *Configure feed credentials*
- *Scheduling*
- *Custom exclude and include lists*
- *Using IOC in alert rules*
- *Using IOC in Network Probe pipelines*
- *Verification*
- *Troubleshooting*
- *Further reading*

How it works

1. A shell script (`misp_threat_lists.sh`) downloads the eight blacklist files over HTTPS with Basic authentication.
2. A Network Probe pipeline (`blacklists.conf`) reads the downloaded files and stores each entry in the `.blacklists` index (used as an intermediate store).
3. The same pipeline re-queries `.blacklists` daily and writes YAML dictionary files (`misp_<type>.yml`) consumed by:
    - `translate` filters in integration pipelines, which add a per-event field (for example `[source][badip] => "true"`) when an indicator matches,

- the **Blacklist-IOC** alert rule type, which reads the YAML files directly (see *Alert Rule Types*).

4. At the end of the run, after all eight categories have been fetched, the script issues a single `_delete_by_query` that removes entries in `.blacklists` older than five minutes and tagged `misp_blacklist`, so the intermediate index does not grow indefinitely.
5. The shell script uses `curl` over HTTPS with default TLS certificate validation. Only HTTP 200 responses trigger the rename of the downloaded file to the timestamped form (`misp_<type>-<epoch>.blacklist`) that the Network Probe pipeline picks up, any other response (401, 403, timeout) skips the rename, so the pipeline does not ingest new data for that category.

```
repository.energylogserver.pl/ioc/
             |  misp_threat_lists.sh (HTTPS + Basic auth)
             v
/etc/logserver-probe/lists/misp_<type>.blacklist
             |  blacklists.conf Network Probe pipeline
             v
       .blacklists index (intermediate store)
             |  daily export
             v
/etc/logserver-probe/lists/misp_<type>.yml --+-> Blacklist-IOC alert rules
             |                                 |
             |  translate filter               |
             v                                 |
   Logs tagged "misp_threat_detected"          |
```

Feed categories

Eight category files are fetched from `https://repository.energylogserver.pl/ioc/misp_<type>.blacklist`:

Category file	Indicator type
`misp_domain.blacklist`	Domain names
`misp_email.blacklist`	Email addresses
`misp_filename.blacklist`	File names
`misp_ip.blacklist`	IP addresses
`misp_url.blacklist`	URLs
`misp_filehash.blacklist`	File hashes
`misp_certhash.blacklist`	Certificate hashes
`misp_regkey.blacklist`	Windows registry keys

Feed source

The ELS IOC repository (`repository.energylogserver.pl`) aggregates threat intelligence from the **CIRCL MISP** instance (`misppriv.circl.lu`) operated by the Computer Incident Response Center Luxembourg.

Only attributes that meet all of the following criteria are included:

- marked as IDS-relevant (`to_ids=1`)
- from published events (`published=1`)
- passing the MISP warning list (`enforce_warninglist=1`)
- not tagged with any TLP label (public, unclassified data only)

Indicators are capped by age before being included:

Category	Maximum age
`ip`, `url`	30 days
`domain`	90 days
`filehash`	365 days
`email`, `filename`, `certhash`, `regkey`	no limit

The `source` column in each `.blacklist` file and in the `.blacklists` index records the originating MISP event as `CIRCL MISP (<event-uuid>)`.

Feed file format

Each `.blacklist` file is tab-separated. The `blacklists.conf` Network Probe pipeline renames the five columns as follows:

Column	Renamed to
1	`object`
2	`type`
3	`source`
4	`description`
5	`url`

Note

This is a custom Energy Logserver TSV format. It is not the standard MISP feed format (which supports MISP JSON, CSV or freetext per misp-project.org/feeds).

Generated dictionary format

The pipeline writes one entry per indicator into `/etc/logserver-probe/lists/misp_<type>.yml`:

```
<indicator value>: bad_<type>
```

Values that YAML cannot hold as plain text get quotes, for example `'000709c7506b64557e8115c06065e8292056b4431eb8608bd19ad54e5c5451ac.bin': bad_filename`.

The value is a literal presence label (`bad_domain`, `bad_ip`, `bad_url`, ...) taken from the pipeline tag list, not a threat description. The `source`, `description` and `url` columns from the `.blacklist` file are stored in the `.blacklists` index but are not written into the YAML dictionary.

Note

Shipped translate integrations (Barracuda, Ruckus) therefore use the dictionary as a presence check and attach their own `badip = "true"` field via `add_field`. Custom integrations can read the dictionary value directly if a simple category label is sufficient; for richer threat context (`source`, `description`, `url`), query the `.blacklists` index instead.

Installation paths

On the Network Probe host the IOC components live at the following paths:

Path	Purpose
`/etc/logserver-probe/lists/bin/misp_threat_lists.sh`	Download script
`/etc/logserver-probe/lists/`	Downloaded `.blacklist` files and generated `.yml` dictionaries
`/etc/logserver-probe/conf.d/blacklists/blacklists.conf`	Network Probe pipeline
`/etc/logserver-probe/templates.d/blacklists_template.json`	Index template for `.blacklists`

The index template declares four properties: `object` (keyword), `status` (keyword), `name` (keyword) and `description` (text). The current pipeline stores five fields per entry, `object`, `type`, `source`, `description`, `url`, so `type`, `source` and `url` are mapped dynamically by the data engine, and `status` and `name` from the template are reserved but not populated by the pipeline.

Configure feed credentials

Edit `/etc/logserver-probe/lists/bin/misp_threat_lists.sh` and fill in the variables at the top of the file. A valid SIEM-PLAN license is required; IOC repository credentials are provided with it.

Variable	Purpose	Shipped default
`DEST`	Directory for downloaded blacklist files	`/etc/logserver-probe/lists`
`IOC_USER`	Basic-auth user for `repository.energylogserver.pl`	empty
`IOC_PASS`	Basic-auth password for `repository.energylogserver.pl`	empty
`LOGSERVER_USE`	Local account used to prune `.blacklists`	commented out, default `logserver`
`LOGSERVER_PAS`	Password for `LOGSERVER_USER`	commented out, default `logserver`
`LOGSERVER_HOS`	Data node address	`127.0.0.1`
`LOGSERVER_POR`	Data node port	`9200`
`LOGSERVER_SSL`	Use TLS when querying the data node	`false`

In addition, the pipeline output step uses `${PROBE_USER}` / `${PROBE_PASS}` passed from the Network Probe environment when writing to `.blacklists`.

Scheduling

The `blacklists.conf` pipeline carries its own schedule through the `exec` and `logserver` input plugins, no external cron is required when the pipeline is enabled on the Network Probe host. Cron expressions use the format `m h * * *`.

Cron	Local time	Action
`1 2 * * *`	02:01	Run `misp_threat_lists.sh` to refresh the eight `.blacklist` files
`1 3 * * *`	03:01	Query `.blacklists` and write one `misp_<type>.tmp` file per category
`1 4 * * *`	04:01	Parse each `.tmp` file, drop duplicate indicators and replace `misp_<type>.yml`, the file read by translate filters and Blacklist-IOC rules

When a `.tmp` file fails to parse, the previous `misp_<type>.yml` stays in place and the Network Probe log records `YAML dedup failed, existing yml preserved`.

If the script must run from a host that does not run the pipeline, schedule it externally. See *Installation, Blacklist and Threat Intelligence Setup* for the crontab example and the ELS Console Scheduler configuration.

Custom exclude and include lists

Since 8.1.0 the pipeline no longer reads the `.blacklists-exclude` and `.blacklists-include` indices. Documents written there do not change `misp_domain.yml`.

Using IOC in alert rules

The **Blacklist-IOC** rule type (see *Alert Rule Types*) reads the generated YAML dictionaries directly. The shipped Watchguard alert rules illustrate the expected configuration:

```
blacklist-ioc:
  - "!yaml /etc/logserver-probe/lists/misp_ip.yml"
compare_key: src.ip
realert:
  minutes: 60
```

- `blacklist-ioc:`, list of YAML dictionary files prefixed with `!yaml`
- `compare_key:`, event field compared against the dictionary keys
- `realert:`, minimum interval between repeated alerts for the same match

Point `compare_key` at the field you want to check (for example `src.ip`, `dst.ip`, `dst.domain`) and `blacklist-ioc` at the matching `misp_<type>.yml` file.

Using IOC in Network Probe pipelines

The generated YAML files in `/etc/logserver-probe/lists/` can be consumed by any Network Probe pipeline through the `translate` filter. Two shipped integrations use this pattern: **Barracuda CloudGen Firewall** (`barracuda/conf.d/barracuda/096-filter-misp.conf`) and **Ruckus** (`ruckus/conf.d/ruckus/090-filter-misp.conf`). The Barracuda filter is shown below:

```
filter {
  if "barracuda" in [tags] {
    if [source][locality] == "public" {
      if [source][ip] {
        translate {
          field => "[source][ip]"
          dictionary_path => "/etc/logserver-probe/lists/misp_ip.yml"
```

(continues on next page)

(continued from previous page)

```
          add_field => {
            "[source][badip]" => "true"
          }
          refresh_behaviour => "replace"
          override => true
        }
      }
    }
    if ![source][badip] {
      mutate {
        add_field => { "[source][badip]" => "false" }
      }
    }

    # (symmetric block for [destination][ip] omitted for brevity)

    if [source][badip] == "true" or [destination][badip] == "true" {
      mutate {
        add_tag => [ "misp_threat_detected" ]
      }
    }
  }
}
```

Other integrations can adopt the same approach by pointing `dictionary_path` at the relevant `misp_<type>.yml` file.

> **Note**
>
> The Barracuda example above uses the `field =>` parameter, which is deprecated in the `translate` filter 3.3.0+ (use `source =>` in new pipelines). Existing shipped integrations mix both forms.

Verification

Confirm the feed is flowing after installation:

```
# Index exists and holds documents
curl -sS -u $CREDENTIAL -XGET '127.0.0.1:9200/_cat/indices/.blacklists?v'

# YAML dictionaries were generated
ls -l /etc/logserver-probe/lists/misp_*.yml
```

Troubleshooting

Symptom	What to check
`.blacklists` index missing or empty	Credentials `IOC_USER`/`IOC_PASS` in `misp_threat_lists.sh`, SIEM-PLAN license status
YAML dictionaries not updating	The `blacklists` pipeline is running on the Network Probe host; check pipeline status
HTTP 401/403 from the feed URL	Verify `IOC_USER`/`IOC_PASS` values
`.blacklists` index not refreshing for a category	`misp_threat_lists.sh` only renames the download to the timestamped file on HTTP 200; other codes skip the rename silently and the Network Probe pipeline sees nothing new. Run `bash /etc/logserver-probe/lists/bin/misp_threat_lists.sh` manually, curl prints each HTTP status to stdout

Further reading

The feed uses MISP category naming (`domain`, `email`, `url`, `ip`, `filehash`, `filename`, `certhash`, `regkey`). For background on what a MISP feed is and how public MISP feeds are distributed, see the MISP project public feeds page.

Windows Events ID Repository

This reference table lists Windows Security Event IDs together with the Active Directory dashboards that consume them. The rightmost column lists the equivalent Event IDs for Windows Server 2003.

The content is reproduced from the 7.x SIEM Plan reference.

Note

This table is a reference map of which Event ID feeds which shipped AD dashboard. For correlation logic that consumes these events (filters on `event_id`, `event_data.IpAddress`, `user.role`, `query_key`, timeframes, thresholds), see *MS Windows SIEM rules*.

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
Object	Access	561	AD DNS Changes	Success	Security	Handle Allocated	
System	Security State Change	4608	[AD] Event Statistics	Success	Security	Windows is starting up	512

continues on next page

Table 4 – continued from previous page

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
System	Security System Extension	4610	[AD] Event Statistics	Success	Security	An authentication package has been loaded by the Local Security Authority	514
System	System Integrity	4612	[AD] Event Statistics	Success	Security	Internal resources allocated for the queuing of audit messages have been exhausted, leading to the loss of some audits	516
System	System Integrity	4615	[AD] Event Statistics	Success	Security	Invalid use of LPC port	519
System	Security State Change	4616	[AD] Servers Audit	Success	Security	The system time was changed.	520
Logon/Logoff	Logon	4624	[AD] Total Logins -> AD Login Events	Success	Security	An account was successfully logged on	528, 540
Logon/Logoff	Logon	4625	[AD] Inventory, [AD] Failed Logins -> AD Failed Login Events	Failure	Security	An account failed to log on	529, 530, 531, 532, 533, 534, 535, 536, 537, 539
Object Access	File System, Registry, SAM, Handle Manipulation, Other Object Access Events	4656	[AD] Removable Device Auditing	Success, Failure	Security	A handle to an object was requested	560

continues on next page

Table 4 – continued from previous page

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
Object Access	File System, Registry, Kernel Object, SAM, Other Object Access Events	4663	[AD] Removable Device Auditing	Success	Security	An attempt was made to access an object	567
Object Access	File System, Registry, Policy Change, Authorization Policy Change	4670	[AD] GPO Objects Overview	Success	Security	Permissions on an object were changed	
Account Management	User Account Management	4720	[AD] Accounts Overview -> [AD] A user account was created	Success	Security	A user account was created	624
Account Management	User Account Management	4722	[AD] Accounts Overview -> [AD] A user account was disabled	Success	Security	A user account was enabled	626
Account Management	User Account Management	4723	[AD] Accounts Overview -> [AD] An attempt was made to change an account's password	Success	Security	An attempt was made to change an account's password	627

continues on next page

Table 4 – continued from previous page

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
Account Management	User Account Management	4724	[AD] Accounts Overview -> [AD] An attempt was made to change an account's password	Success	Security	An attempt was made to reset an accounts password	628
Account Management	User Account Management	4725	[AD] Accounts Overview -> [AD] A user account was disabled	Success	Security	A user account was disabled	629
Account Management	User Account Management	4726	[AD] Accounts Overview -> [AD] A user account was deleted	Success	Security	A user account was deleted	630
Account Management	Security Group Management	4727	[AD] Security Group Change History	Success	Security	A security-enabled global group was created	631
Account Management	Security Group Management	4728	[AD] Organizational Unit	Success	Security	A member was added to a security-enabled global group	632
Account Management	Security Group Management	4729	[AD] Organizational Unit	Success	Security	A member was removed from a security-enabled global group	633

continues on next page

Table 4 – continued from previous page

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
Account Management	Security Group Management	4730	[AD] Organizational Unit	Success	Security	A security-enabled global group was deleted	634
Account Management	Security Group Management	4731	[AD] Organizational Unit	Success	Security	A security-enabled local group was created	635
Account Management	Security Group Management	4732	[AD] Organizational Unit	Success	Security	A member was added to a security-enabled local group	636
Account Management	Security Group Management	4733	[AD] Organizational Unit	Success	Security	A member was removed from a security-enabled local group	637
Account Management	Security Group Management	4734	[AD] Organizational Unit	Success	Security	A security-enabled local group was deleted	638
Account Management	User Account Management	4738	[AD] Accounts Overview	Success	Security	A user account was changed	642
Account Management	User Account Management	4740	[AD] Accounts Overview -> AD Account - Account Locked	Success	Security	A user account was locked out	644
Account Management	Computer Account Management	4741	[AD] Computer Account Overview	Success	Security	A computer account was created	645
Account Management	Computer Account Management	4742	[AD] Computer Account Overview	Success	Security	A computer account was changed	646

continues on next page

Table 4 – continued from previous page

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
Account Management	Computer Account Management	4743	[AD] Computer Account Overview	Success	Security	A computer account was deleted	647
Account Management	Distribution Group Management	4744	[AD] Organizational Unit	Success	Security	A security-disabled local group was created	648
Account Management	Distribution Group Management	4746	[AD] Security Group Change History	Success	Security	A member was added to a security-disabled local group	650
Account Management	Distribution Group Management	4747	[AD] Security Group Change History	Success	Security	A member was removed from a security-disabled local group	651
Account Management	Distribution Group Management	4748	[AD] Organizational Unit	Success	Security	A security-disabled local group was deleted	652
Account Management	Distribution Group Management	4749	[AD] Organizational Unit	Success	Security	A security-disabled global group was created	653
Account Management	Distribution Group Management	4751	[AD] Security Group Change History	Success	Security	A member was added to a security-disabled global group	655
Account Management	Distribution Group Management	4752	[AD] Security Group Change History	Success	Security	A member was removed from a security-disabled global group	656

continues on next page

Table 4 – continued from previous page

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
Account Management	Distribution Group Management	4753	[AD] Organizational Unit	Success	Security	A security-disabled global group was deleted	657
Account Management	Security Group Management	4754	[AD] Organizational Unit	Success	Security	A security-enabled universal group was created	658
Account Management	Security Group Management	4755	[AD] Organizational Unit	Success	Security	A security-enabled universal group was changed	659
Account Management	Security Group Management	4756	[AD] Organizational Unit	Success	Security	A member was added to a security-enabled universal group	660
Account Management	Security Group Management	4757	[AD] Organizational Unit	Success	Security	A member was removed from a security-enabled universal group	661
Account Management	Security Group Management	4758	[AD] Organizational Unit	Success	Security	A security-enabled universal group was deleted	662
Account Management	Distribution Group Management	4759	[AD] Security Group Change History	Success	Security	A security-disabled universal group was created	663
Account Management	Distribution Group Management	4761	[AD] Security Group Change History	Success	Security	A member was added to a security-disabled universal group	655

continues on next page

Table 4 – continued from previous page

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
Account Management	Distribution Group Management	4762	[AD] Security Group Change History	Success	Security	A member was removed from a security-disabled universal group	666
Account Management	Security Group Management	4764	[AD] Organizational Unit	Success	Security	A groups type was changed	668
Account Management	User Account Management	4765	[AD] Accounts Overview -> AD Account - Account History	Success	Security	SID History was added to an account	
Account Management	User Account Management	4766	[AD] Accounts Overview -> AD Account - Account History	Failure	Security	An attempt to add SID History to an account failed	
Account Management	User Account Management	4767	[AD] Accounts Overview	Success	Security	A computer account was changed	646
Account Logon	Credential Validation	4776	[AD] Failed Logins	Success, Failure	Security	The domain controller attempted to validate the credentials for an account	680, 681
Account Management	User Account Management	4781	[AD] Accounts Overview	Success	Security	The name of an account was changed	685
Directory Service	Directory Service Changes	5136	[AD] Organizational Unit	Success	Security	A directory service object was modified	566

continues on next page

Table 4 – continued from previous page

Category	Subcategory	Event ID	Dashboard	Type	Event Log	Describe	Event ID for Windows 2003
Directory Service	Directory Service Changes	5137	[AD] Organizational Unit	Success	Security	A directory service object was created	566
Directory Service	Directory Service Changes	5138	[AD] Organizational Unit	Success	Security	A directory service object was undeleted	
Directory Service	Directory Service Changes	5139	[AD] Organizational Unit	Success	Security	A directory service object was moved	
Object Access	File Share	5140	[AD] File Audit	Success	Security	A network share object was accessed	
Directory Service	Directory Service Changes	5141	[AD] Organizational Unit	Failure	Security	A directory service object was deleted	
Object Access	File Share	5142	[AD] File Audit	Success	Security	A network share object was added.	
Object Access	Detailed File Share	5145	[AD] File Audit	Success, Failure	Security	A network share object was checked to see whether client can be granted desired access	
Process Tracking	Plug and Play	6416	[AD] Removable Device Auditing	Success	Security	A new external device was recognized by the system.	

Netflow analysis

The Network Probe collector receives and decodes Network Flows using the provided decoders. During decoding, IP address reputation analysis is performed and the result is added to the event document.

Enable Network Probe pipeline

Edit `/etc/logserver-probe/pipelines.yml`:

```
- pipeline.id: flows
  path.config: "/etc/logserver-probe/conf.d/netflow/*.conf"
```

Data Node template installation

```
curl -XPUT -H 'Content-Type: application/json' -u logserver:logserver \
  'http://127.0.0.1:9200/_template/netflow' \
  -d@/etc/logserver-probe/templates.d/netflow-template.json
```

Enable reverse DNS lookup

To enable reverse DNS lookup, set `USE_DNS:false` to `USE_DNS:true` in `13-filter-dns-geoip.conf`.

Optionally replace the default DNS servers (`${DNS_SRV:8.8.8.8}`) with your local resolvers.

Security Rules

Pre-defined correlation rules grouped by platform. Each row lists the architecture/application, rule name, description, index pattern, optional requirements, log source, rule type, and the YAML rule definition body.

The content is reproduced from the 7.x SIEM Plan reference.

- *Cluster Health rules*
- *MS Windows SIEM rules*
- *Network Switch SIEM rules*
- *Cisco ASA devices SIEM rules*
- *Linux Mail SIEM rules*
- *Linux DNS Bind SIEM Rules*
- *Fortigate Devices SIEM rules*
- *Linux Apache SIEM rules*
- *RedHat / CentOS system SIEM rules*
- *Checkpoint devices SIEM rules*
- *Cisco ESA devices SIEM rule*
- *Forcepoint devices SIEM rules*
- *Oracle Database Engine SIEM rules*
- *Paloalto devices SIEM rules*
- *Microsoft Exchange SIEM rules*
- *Juniper Devices SIEM Rules*
- *Fudo SIEM Rules*

- *Squid SIEM Rules*
- *Microsoft DNS Server SIEM Rules*
- *Microsoft DHCP SIEM Rules*
- *Linux DHCP Server SIEM Rules*
- *Cisco VPN devices SIEM Rules*
- *Netflow SIEM Rules*
- *MikroTik devices SIEM Rules*
- *Microsoft SQL Server SIEM Rules*
- *Postgress SQL SIEM Rules*
- *MySQL SIEM Rules*

Cluster Health rules

Nr.	Architecture/A	Rule Name	Index nam	Description	Rule type	Rule Definition
1	Logtrail	Cluster Services Error Logs	logtra *	Shows errors in cluster services logs.	frequen	# (Optional, any specific) filter: - query_string: query: "log_level:ERROR AND exists:path" # (Optional, any specific) #num_events: 10 #timeframe: # hours: 1 query_key: path timeframe: minutes: 10 num_events: 100
2	Skimmer	Cluster Health Status	skim *	Health status of the cluster, based on the state of its primary and replica shards.	any	timeframe: minutes: 3 filter: - query: query_string: query: cluster_health_status:0
3	Skimmer	Cluster Stats Indices Docs Per Sec	skim *	A single-value metrics aggregation that calculates an approximate count of distinct values.	metric_a	metric_agg_key: "cluster_stats_indices_docs_per_sec" metric_agg_type: "cardinality" doc_type: "_doc" max_threshold: 16000000 buffer_time: minutes: 1
4	Skimmer	Indices Stats All Total Store Size In Bytes	skim *	Size of the index in byte units.	metric_a	metric_agg_key: "indices_stats_all_total_store_size_in_bytes" metric_agg_type: "cardinality" doc_type: "_doc" max_threshold: 60000000000000 buffer_time: minutes: 1
5	Skimmer	Network Probe Stats CPU Load Average 15M	skim *	15m -> Fifteen-minute load average on the system (field is not present if fifteen-minute load average is not available).	metric_a	metric_agg_key: "stats_cpu_load_average_15m" metric_agg_type: "cardinality" doc_type: "_doc" max_threshold: 5 buffer_time: minutes: 1
6	Skimmer	Network Probe Stats Cpu Percent	skim *	Properties of cpu -> percent -> Recent CPU usage for the whole system, or -1 if not supported.	metric_a	metric_agg_key: "stats_cpu_percent" metric_agg_type: "cardinality" doc_type: "_doc" max_threshold: 20 buffer_time: minutes: 1
7	Skimmer	Network Probe Stats Events Queue Push Duration In Millis	skim *	queue_push_duration_in_mi is the accumulative time the input are waiting to push events into the queue.	metric_a	metric_agg_key: "stats_events_queue_push_duration_in_millis" metric_agg_type: "cardinality" doc_type: "_doc" max_threshold: 140000000 buffer_time: minutes: 1
8	Skimmer	Network Probe Stats Mem Heap Used Percent	skim *	Memory currently in use by the heap	any	metric_agg_key: "stats_mem_heap_used_percent" metric_agg_type: "cardinality" doc_type: "_doc" max_threshold: 80 buffer_time: minutes: 1
9	Skimmer	Network Probe Stats Persisted Queue Size	skim *	A Network Probe persistent queue helps protect against data loss during abnormal termination by storing the in-flight message queue to disk.	metric_a	type: metric_aggregation metric_agg_key: node_stats_/var/lib/logserver-probe/queue_disk_usage query_key: source_node_host metric_agg_type: max doc_type: _doc max_threshold: 734003200 realert: minutes: 15
10	Skimmer	Node Stats Expected Data Nodes	skim *	Nodes stats API returns cluster nodes statistics	metric_a	metric_agg_key: "node_stats_expected_data_nodes" metric_agg_type: "cardinality" doc_type: "_doc" min_threshold: 1 buffer_time: minutes: 1
11	Skimmer	Node Stats	skim	flush -> Contains statistics	met-	metric_agg_key:

MS Windows SIEM rules

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
1	Windows	Windows - Admin night logon	Alert on Windows login events when detected outside business hours	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:(4624 OR 1200) AND user.role:admin AND event.hour:(20 OR 21 OR 22 OR 23 0 OR 1 OR 2 OR 3)"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
2	Windows	Windows - Admin task as user	Alert when admin task is initiated by regular user. Windows event id 4732 is verified towards static admin list. If the user does not belong to admin list AND the event is seen than we generate alert. Static Admin list is a Network Probe disctionary file that needs to be created manually. During Network Probe lookup a field user.role:ad is added to an event. 4732: A member was added to a security-enabled local group	winlogbeat-*	winlogbeat Network Probe admin dictionary lookup file	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4732 AND NOT user.role:admin"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
3	Windows	Windows - diff IPs logon	Alert when Windows logon process is detected and two or more different IP addressed are seen in source field. Timeframe is last 15min. Detection is based onevents 4624 or 1200. 4624: An account was successfully logged on 1200: Application token success	winlogbeat-*	winlogbeat	Widnows Security Eventlog	cardinality	max_cardinality: 1 timeframe: minutes: 15 filter: - query_string: query: "event_id:(4624 OR 1200) AND NOT *exists*:user.role AND NOT event_data.IpAddress:"-" " query_key: username
4	Windows	Windows - Event service error	Alert when Windows event 1108 is matched 1108: The event logging service encountered an error	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:1108"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
5	Windows	Windows - file insufficient privileges	Alert when Windows event 5145 is matched 5145: A network share object was checked to see whether client can be granted desired access Every time a network share object (file or folder) is accessed, event 5145 is logged. If the access is denied at the file share level, it is audited as a failure event. Otherwise, it considered a success. This event is not generated for NTFS access.	winlogbeat-*	winlogbeat	Widnows Security Eventlog	frequency	query_key: "event_data.IpAddress" num_events: 50 timeframe: minutes: 15 filter: - query_string: query: "event_id:5145"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
6	Windows	Windows - Kerberos pre-authenticati failed	Alert when Windows event 4625 or 4771 is matched 4625: An account failed to log on 4771: Kerberos pre-authenticati failed	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: “event_id:4625 OR event_id:4771”
7	Windows	Windows - Logs deleted	Alert when Windows event 1102 OR 104 is matched 1102: The audit log was cleared 104: Event log cleared	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: ‘event_desc:“1102 The audit log was cleared”’
8	Windows	Windows - Member added to a security-enabled global group	Alert when Windows event 4728 is matched 4728: A member was added to a security-enabled global group	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: “event_id:4728”

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
9	Windows	Windows - Member added to a security-enabled local group	Alert when Windows event 4732 is matched 4732: A member was added to a security-enabled local group	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4732"
10	Windows	Windows - Member added to a security-enabled universal group	Alert when Windows event 4756 is matched 4756: A member was added to a security-enabled universal group	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4756"
11	Windows	Windows - New device	Alert when Windows event 6414 is matched 6416: A new external device was recognized by the system	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:6416"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
12	Windows	Windows - Package installation	Alert when Windows event 4697 is matched 4697: A service was installed in the system	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4697"
13	Windows	Windows - Password policy change	Alert when Windows event 4739 is matched 4739: Domain Policy was changed	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4739"
14	Windows	Windows - Security log full	Alert when Windows event 1104 is matched 1104: The security Log is now full	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:1104"
15	Windows	Windows - Start up	Alert when Windows event 4608 is matched 4608: Windows is starting up	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4608"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
16	Windows	Windows - Account lock	Alert when Windows event 4740 is matched 4740: A User account was Locked out	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4740"
17	Windows	Windows - Security local group was changed	Alert when Windows event 4735 is matched 4735: A security-enabled local group was changed	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4735"
18	Windows	Windows - Reset password attempt	Alert when Windows event 4724 is matched 4724: An attempt was made to reset an accounts password	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4724"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
19	Windows	Windows - Code integrity changed	Alert when Windows event 5038 is matched 5038: Detected an invalid image hash of a file Information: Code Integrity is a feature that improves the security of the operating system by validating the integrity of a driver or system file each time it is loaded into memory. Code Integrity detects whether an unsigned driver or system file is being loaded into the kernel, or whether a system file has been modified by malicious software that is being run by a user account	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:5038"

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
20	Windows	Windows - Application error	Alert when Windows event 1000 is matched 1000: Application error	winlogbeat-*	winlogbeat	Widnows Application Eventlog	any	filter: - query_string: query: "event_id:1000"
21	Windows	Windows - Application hang	Alert when Windows event 1001 OR 1002 is matched 1001: Application fault bucket 1002: Application hang	winlogbeat-*	winlogbeat	Widnows Application Eventlog	any	filter: - query_string: query: "event_id:1002 OR event_id:1001"
22	Windows	Windows - Audit policy changed	Alert when Windows event 4719 is matched 4719: System audit policy was changed	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:4719"
23	Windows	Windows - Eventlog service stopped	Alert when Windows event 6005 is matched 6005: Eventlog service stopped	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:6005"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
24	Windows	Windows - New service installed	Alert when Windows event 7045 OR 4697 is matched 7045,4697: A service was installed in the system	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:7045 OR event_id:4697"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
25	Windows	Windows - Driver loaded	Alert when Windows event 6 is matched 6: Driver loaded The driver loaded events provides information about a driver being loaded on the system. The configured hashes are provided as well as signature information. The signature is created asynchronously for performance reasons and indicates if the file was removed after loading.	winlogbeat-*	winlogbeat	Widnows System Eventlog	any	filter: - query_string: query: "event_id:6"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
26	Windows	Windows - Firewall rule modified	Alert when Windows event 2005 is matched 2005: A Rule has been modified in the Windows firewall Exception List	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: 'event_desc:"4947 A change has been made to Windows Firewall exception list. A rule was modified"'
27	Windows	Windows - Firewall rule add	Alert when Windows event 2004 is matched 2004: A firewall rule has been added	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:2004"
28	Windows	Windows - Firewall rule deleted	Alert when Windows event 2006 or 2033 or 2009 is matched 2006,2033,2 Firewall rule deleted	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: "event_id:2006 OR event_id:2033 OR event_id:2009"

continues on next page

Table 5 – continued from previous page

Nr.	Architecture/Applic	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
29	Windows	Windows - System has been shutdown	This event is written when an application causes the system to restart, or when the user initiates a restart or shutdown by clicking Start or pressing CTRL+ALT and then clicking Shut Down.	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: 'event_id:"1074"'
30	Windows	Windows - The system time was changed	The system time has been changed. The event describes the old and new time.	winlogbeat-*	winlogbeat	Widnows Security Eventlog	any	filter: - query_string: query: 'event_id:"4616"'

Network Switch SIEM rules

Nr.	Architecture/Appli	Rule Name	Description	Index name	Requirements	Sour	Rule type	Rule definition
1	Switch	Switch - Blocked by LACP	ports: port is Blocked by LACP	syslog *		syslog	any	filter: - query_string: query: “message:“Blocked by LACP””
2	Switch	Switch - Blocked by STP	ports: port is Blocked by STP	syslog *		syslog	any	filter: - query_string: query: “message:“Blocked by STP””
3	Switch	Switch - Port state changed	Port state changed to down or up	syslog *		syslog	any	filter: - query_string: query: “message:“changed state to””
4	Switch	Switch - Configured from console	Configurations changes from console	syslog *		syslog	any	filter: - query_string: query: “message:“Configured from console””
5	Switch	Switch - High collision or drop rate		syslog *		syslog	any	filter: - query_string: query: “message:“High collision or drop rate””
6	Switch	Switch - Invalid login	auth: Invalid user name/password on SSH session	syslog *		syslog	any	filter: - query_string: query: “message:“auth: Invalid user name/password on SSH session””
7	Switch	Switch - Logged to switch		syslog *		syslog	any	filter: - query_string: query: “message:” mgr: SME SSH from””
8	Switch	Switch - Port is offline	ports: port is now off-line	syslog *		syslog	any	filter: - query_string: query: “message:” is now off-line””
9	Switch	Switch - Port is online	ports: port is now on-line	syslog *		syslog	any	filter: - query_string: query: “message:” is now on-line””

Cisco ASA devices SIEM rules

Nr.	Architecture/Ap	Rule Name	Description	Index nam	Requirement	Source	Rule type	Rule definition
1	Cisco ASA	Cisco ASA - Device interface administratively up	Device interface administratively up	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: 'cisco.id:"%ASA-4-411003"'
2	Cisco ASA	Cisco ASA - Device configuration has been changed or reloaded	Device configuration has been changed or reloaded	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: 'cisco.id:("%ASA-5-111007" OR "%ASA-5-111008")'
3	Cisco ASA	Cisco ASA - Device interface administratively down	Device interface administratively down	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: 'cisco.id:"%ASA-4-411004"'
4	Cisco ASA	Cisco ASA - Device line protocol on Interface changed state to down	Device line protocol on Interface changed state to down	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: 'cisco.id:"%ASA-4-411002"'
5	Cisco ASA	Cisco ASA - Device line protocol on Interface changed state to up	Device line protocol on Interface changed state to up	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: 'cisco.id:"%ASA-4-411001"'
6	Cisco ASA	Cisco ASA - Device user executed shutdown	Device user executed shutdown	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: 'cisco.id:"%ASA-5-111010"'
7	Cisco ASA	Cisco ASA - Multiple VPN authentication failed	Multiple VPN authentication failed	syslo *		syslog from Cisco ASA devices	frequen	query_key: "src.ip" num_events: 10 timeframe: minutes: 240 filter: - query_string: query: "cisco.id:"%ASA-6-113005""
8	Cisco ASA	Cisco ASA - VPN authentication failed	VPN authentication failed	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: "cisco.id:"%ASA-6-113005""
9	Cisco ASA	Cisco ASA - VPN authentication successful	VPN authentication successful	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: "cisco.id:"%ASA-6-113004""
10	Cisco ASA	Cisco ASA - VPN user locked out	VPN user locked out	syslo *		syslog from Cisco ASA devices	any	filter: - query_string: query: "cisco.id:"%ASA-6-113006""

Linux Mail SIEM rules

Nr.	Architecture/App	Rule Name	Description	Index name	Requirement	Sour	Rule type	Rule definition
1	Mail Linux	Mail - Flood Connect from	Connection flood, possible DDOS attack	syslo *		sys-log	fre-quen	filter: - query_string: query: “message:“connect from”” query_key: host timeframe: hours: 1 num_events: 50
2	Mail Linux	Mail - SASL LOGIN authentication failed	User authentication failure	syslo *		sys-log	fre-quen	filter: - query_string: query: “message:“SASL LOGIN authentication failed: authentication failure”” query_key: host timeframe: hours: 1 num_events: 30
3	Mail Linux	Mail - Sender rejected	Sender rejected	syslo *		sys-log	fre-quen	filter: - query_string: query: “message:“NOQUEUE: reject: RCPT from”” query_key: host timeframe: hours: 1 num_events: 20

Linux DNS Bind SIEM Rules

Nr.	Architecture/	Rule Name	Description	Index name	Requirement	Sour	Rule type	Rule definition
1	DNS	DNS - Anomaly in geographic region	DNS anomaly detection in geographic region	filebe *		filebeat	spike	query_key: geoip.country_code2 threshold_ref: 500 spike_height: 3 spike_type: “up” timeframe: minutes: 10 filter: - query_string: query: “NOT geoip.country_code2:(US OR PL OR NL OR IE OR DE OR FR OR GB OR SK OR AT OR CZ OR NO OR AU OR DK OR FI OR ES OR LT OR BE OR CH) AND *exists*:geoip.country_code2 AND NOT domain:(*.outlook.com OR *.pool.ntp.org)”
2	DNS	DNS - Domain requests	Domain requests	filebe *		filebeat	frequen	query_key: “domain” num_events: 1000 timeframe: minutes: 5 filter: - query_string: query: “NOT domain:(/.*localdomain/) AND *exists*:domain”
3	DNS	DNS - Domain requests by source IP	Domain requests by source IP	filebe *		filebeat	cardinality	query_key: “src_ip” cardinality_field: “domain” max_cardinality: 3000 timeframe: minutes: 10 filter: - query_string: query: “(NOT domain:(/.*.arpa/ OR /.*localdomain/ OR /.*office365.com/) AND *exists*:domain”
4	DNS	DNS - Resolved domain matches IOC IP blacklist	Resolved domain matches IOC IP blacklist	filebe *		filebeat	blacklist	compare_key: “domain_ip” blacklist-ioc: - “!yaml /etc/logserver-probe/lists/misp_ip.yml”

Fortigate Devices SIEM rules

Nr.	Architecture/A	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
1	FortiOS 6.x	Fortigate virus		fortigate*	FortiOS with Antivirus, IPS, Fortisandbox modules, Network Probe KV filter, default-base-template	syslog from Forti devices	Any	filter: - query_string: query: “subtype:virus and action:blocked”
2	FortiOS 6.x	Fortigate http server attack by destination IP		fortigate*	FortiOS with waf, IPS, modules, Network Probe KV filter, default-base-template	syslog from Forti devices	frequen	query_key: “dst_ip” num_events: 10 timeframe: hours: 1 filter: - query_string: query: “level:alert and subtype:ips and action:dropped and profile:protect_http_server”
3	FortiOS 6.x	Fortigate forward deny by source IP		fortigate*	FortiOS with IPS, modules, Network Probe KV filter, default-base-template	syslog from Forti devices	frequen	query_key: “src_ip” num_events: 250 timeframe: hours: 1 filter: - query_string: query: “subtype:forward AND action:deny”
4	FortiOS 6.x	Fortigate failed login		fortigate*	FortiOS with IPS, modules, Network Probe KV filter, default-base-template	syslog from Forti devices	Any	filter: - query_string: query: “action:login and status:failed”
5	FortiOS 6.x	Fortigate failed login same source		fortigate*	FortiOS with IPS, modules, Network Probe KV filter, default-base-template	syslog from Forti devices	frequen	query_key: “src_ip” num_events: 18 timeframe: minutes: 45 filter: - query_string: query: “action:login and status:failed”
6	FortiOS 6.x	Fortigate device configuration changed		fortigate*	FortiOS with IPS, modules, Network Probe KV filter, default-base-template	syslog from Forti devices	any	filter: - query_string: query:””“Configuration is changed in the admin session””
7	FortiOS 6.x	Fortigate unknown tunneling setting		fortigate*	FortiOS with IPS, modules, Network Probe KV filter, default-base-template	syslog from Forti devices	any	filter: - query_string: query:””“http_decoder: HTTP.Unknown.Tunnelling””
8	FortiOS 6.x	Fortigate multiple tunneling same source		fortigate*	FortiOS with IPS, modules, Network Probe KV filter, default-base-template	syslog from Forti devices	frequen	query_key: “src_ip” num_events: 18 timeframe: minutes: 45 filter: - query_string: query: ““http_decoder: HTTP.Unknown.Tunnelling””
9	FortiOS 6.x	Fortigate firewall configuration changed		fortigate*	FortiOS with IPS, modules, Network Probe KV filter, default-base-template	syslog from Forti de-	any	filter: - query_string: query:”action:Edit”

Linux Apache SIEM rules

Nr.	Architecture/	Rule Name	Description	Index name	Requirement	Source	Rule type	Rule definition
1	Apac	HTTP 1xx peak	Response status 1xx	httpd		Apac logs	spike	threshold_cur: 100 timeframe: hours: 2 spike_height: 5 spike_type: “up” filter: - query: query_string: query: “response.status.code:1*” - type: value: “_doc”
2	Apac	HTTP 2xx response for unwanted URLs	Requests for URLS like: - /phpMyAdmin, /wpadmin, /wp-login.php, /.env, /admin, /owa/auth/logon.aspx, /api, /license.txt, /api/v1/pods, /solr/admin/info/system, /backup/, /admin/config.php, /dana-na, /dbadmin/, /myadmin/, /mysql/, /php-myadmin/, /sqlmanager/, /mysqlmanager/, config.php	httpd		Apac logs	black list	compare_key: http.request ignore_null: true blacklist: - /phpMyAdmin - /wpadmin - /wp-login.php - /.env - /admin - /owa/auth/logon.aspx - /api - /license.txt - /api/v1/pods - /solr/admin/info/system - /backup/ - /admin/config.php - /dana-na - /dbadmin/ - /myadmin/ - /mysql/ - /php-my-admin/ - /sqlmanager/ - /mysqlmanager/ - config.php
3	Apac	HTTP 2xx spike		httpd		Apac logs	spike	threshold_cur: 100 timeframe: hours: 2 spike_height: 5 spike_type: “up” filter: - query: query_string: query: “response.status.code:2*” - type: value: “_doc”
4	Apac	HTTP 3x spike	Response status 3xx	httpd		Apac logs	any	threshold_cur: 100 timeframe: hours: 2 spike_height: 5 spike_type: “up” filter: - query: query_string: query: “response.status.code:3*” - type: value: “_doc”
5	Apac	HTTP 4xx spike	Response status 4xx	httpd		Apac logs	spike	threshold_cur: 100 timeframe: hours: 2 spike_height: 5 spike_type: “up” filter: - query: query_string: query: “response.status.code:4*” - type: value: “_doc”
6	Apac	HTTP 5xx spike	Response status 5xx	httpd		Apac logs	spike	threshold_cur: 100 timeframe: hours: 2 spike_height: 5 spike_type: “up” filter: - query: query_string: query: “response.status.code:5*” - type: value: “_doc”

RedHat / CentOS system SIEM rules

Nr.	Architec-ture/Applic	Rule Name	De-scrip-tion	In-dex name	Re-quire-ments	Sour	Rule type	Rule definition
1	Linux	Linux - Group Change		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:“added by root to group””
2	Linux	Linux - Group Created		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:”new group: “”
3	Linux	Linux - Group Removed		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:”removed group: “ OR mes-sage:”removed shadow group: “”
4	Linux	Linux - Inter-rupted Login		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:“Connection closed by””
5	Linux	Linux -Login Failure		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:“Failed password for””
6	Linux	Linux - Login Success		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:“Accepted password for””
7	Linux	Linux - Out of Memory		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:“killed process””
8	Linux	Linux - Password Change		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:“password changed””
9	Linux	Linux - Process Segfaults		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:segfault”
10	Linux	Linux - Process Traps		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:traps”
11	Linux	Linux - Service Started		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:Started”
12	Linux	Linux - Service Stopped		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:Stopped”
13	Linux	Linux - Software Erased		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:”Erased: “”
14	Linux	Linux - Software Installed		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:”Installed: “”
15	Linux	Linux - Software Updated		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:”Updated: “”
16	Linux	Linux - User Created		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:”new user: “”
17	Linux	Linux - User Removed		syslog-*		Sys-log	any	filter: - query_string: query: “mes-sage:“delete user””

Checkpoint devices SIEM rules

Nr.	Architecture/A	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
1	VPN-1 & FireW 1	Checkpoint - Drop a packet by source IP		check point	Checkpoint devices, fw1-grabber (https://github.com/certego/fw1-loggrabber)	Checkpoint firewall, OPSEC Log Export APIs (LEA)	Frequen	query_key: “src” num_events: 10 timeframe: hours: 1 filter: - query_string: query: “action:drop” use_count_query: true doc_type: doc
2	VPN-1 & FireW 1	Checkpoint - Reject by source IP		check point	Checkpoint devices, fw1-grabber (https://github.com/certego/fw1-loggrabber)	Checkpoint firewall, OPSEC Log Export APIs (LEA)	Frequen	query_key: “src” num_events: 10 timeframe: hours: 1 filter: - query_string: query: “action:reject” use_count_query: true doc_type: doc
3	VPN-1 & FireW 1	Checkpoint - User login		check point	Checkpoint devices, fw1-grabber (https://github.com/certego/fw1-loggrabber)	Checkpoint firewall, OPSEC Log Export APIs (LEA)	Any	query_key: “user” filter: - query_string: query: “auth_status:“Successful Login”” use_count_query: true doc_type: doc
4	VPN-1 & FireW 1	Checkpoint - Failed Login		check point	Checkpoint devices, fw1-grabber (https://github.com/certego/fw1-loggrabber)	Checkpoint firewall, OPSEC Log Export APIs (LEA)	Any	query_key: “user” filter: - query_string: query: “auth_status:“Failed Login”” use_count_query: true doc_type: doc
5	VPN-1 & FireW 1	Checkpoint - Application Block by user		check point	Checkpoint devices, fw1-grabber (https://github.com/certego/fw1-loggrabber)	Checkpoint firewall, OPSEC Log Export APIs (LEA)	Frequen	query_key: “user” num_events: 10 timeframe: hours: 1 filter: - query_string: query: “action:block AND product:“Application Control”” use_count_query: true doc_type: doc
6	VPN-1 & FireW 1	Checkpoint - URL Block by user		check point	Checkpoint devices, fw1-grabber (https://github.com/certego/fw1-loggrabber)	Checkpoint firewall, OPSEC Log Export APIs (LEA)	Frequen	query_key: “user” num_events: 10 timeframe: hours: 1 filter: - query_string: query: “action:block AND product:“URL Filtering”” use_count_query: true doc_type: doc
7	VPN-1 & FireW 1	Checkpoint - Block action with user		check point	Checkpoint devices, fw1-grabber (https://github.com	Checkpoint firewall, OPSEC Log	Any	query_key: “user” filter: - query_string: query: “action:block” use_count_query: true doc_type: doc

Cisco ESA devices SIEM rule

Nr.	Architecture/Appli	Rule Name	Description	Index name	Requirements	Sour	Rule type	Rule definition
1	Cisco ESA	ESA - Attachments exceeded the URL scan		syslog *		Cisco ESA	any	filter: - query_string: query: 'message:"attachments exceeded the URL scan"'
2	Cisco ESA	ESA - Extraction Failure		syslog *		Cisco ESA	any	filter: - query_string: query: 'message:"Extraction Failure"'
3	Cisco ESA	ESA - Failed to expand URL		syslog *		Cisco ESA	any	filter: - query_string: query: 'message:"Failed to expand URL"'
4	Cisco ESA	ESA - Invalid host configured		syslog *		Cisco ESA	any	filter: - query_string: query: 'message:"Invalid host configured"'
5	Cisco ESA	ESA - Marked unscannable due to RFC Violation		syslog *		Cisco ESA	any	filter: - query_string: query: 'message:"was marked unscannable due to RFC Violation"'
6	Cisco ESA	ESA - Message was not scanned for Sender Domain Reputation		syslog *		Cisco ESA	any	filter: - query_string: query: 'message:"Message was not scanned for Sender Domain Reputation"'
7	Cisco ESA	ESA - URL Reputation Rule		syslog *		Cisco ESA	any	filter: - query_string: query: 'message:"URL Reputation Rule"'

Forcepoint devices SIEM rules

Nr.	Architecture/A	Rule Name	Description	Index name	Requirement	Sour	Rule type	Rule definition
1		Forcepoint HIGH	All high alerts	syslo dlp*			any	alert_text_type: alert_text_only alert_text: “Forcepoint HIGH alert\n\n When: {}\n Analyzed by: {}\n User name: {}\n Source: {}\nDestination: {}\n\n{}\n” alert_text_args: - timestamp_timezone - Analyzed_by - user - Source - Destination
2		Forcepoint MEDI	All medium alerts	syslo dlp*			any	alert_text_type: alert_text_only alert_text: “Forcepoint MEDIUM alert\n\n When: {}\n Analyzed by: {}\n User name: {}\n Source: {}\nDestination: {}\n\n{}\n” alert_text_args: - timestamp_timezone - Analyzed_by - user - Source - Destination
3		Forcepoint LOW	All low alerts	syslo dlp*			any	alert_text_type: alert_text_only alert_text: “Forcepoint LOW alert\n\n When: {}\n Analyzed by: {}\n User name: {}\n Source: {}\nDestination: {}\n\n{}\n” alert_text_args: - timestamp_timezone - Analyzed_by - user - Source - Destination
4		Forcepoint blocke email	Email was blocked by forcepoint	syslo dlp*			any	alert_text_type: alert_text_only alert_text: “Email blocked\n\n When: {}\n Analyzed by: {}\n File name: {}\n Source: {}\nDestination: {}\n\n{}\n” alert_text_args: - timestamp_timezone - Analyzed_by - File_Name - Source - Destination
5		Forcepoint removables	Forcepoint blocked data transfer to removeable device	syslo dlp*			any	alert_text_type: alert_text_only alert_text: “Data transfer to removable device blocked\n\n When: {}\n Analyzed by: {}\n File name: {}\n Source: {}\nDestination: {}\n\n{}\n” alert_text_args: - timestamp_timezone - Analyzed_by - File_Name - Source - Destination

Oracle Database Engine SIEM rules

Nr.	Architecture/App	Rule Name	Description	Index name	Requirements	Sourc	Rule type	Rule definition
1	Oracle DB	Oracle - Allocate memory ORA-00090	Failed to allocate memory for cluster database	oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00090”
2	Oracle DB	Oracle logon denied ORA-12317	logon to database (link name string)	oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-12317”
3	Oracle DB	Oracle credential failed ORA-12638	Credential retrieval failed	oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 num_events: 10 filter: - term: oracle.code: “ora-12638”
4	Oracle DB	Oracle client internal error ORA-12643	Client received internal error from server	oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 num_events: 10 filter: - term: oracle.code: “ora-12643”
5	Oracle DB	ORA-00018: maximum number of sessions exceeded		oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00018”
6	Oracle DB	ORA-00019: maximum number of session licenses exceeded		oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00019”
7	Oracle DB	ORA-00020: maximum number of processes (string) exceeded		oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00020”
8	Oracle DB	ORA-00024: logins from more than one process not allowed in single-process mode		oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00024”
9	Oracle DB	ORA-00025: failed to allocate string (out of memory)		oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00025”
10	Oracle DB	ORA-00055: maximum number of DML locks exceeded		oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00055”
11	Oracle DB	ORA-00057: maximum number of temporary table locks exceeded		oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00057”
12	Oracle DB	ORA-00059: maximum number of DB_FILES exceeded		oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00059”
13	Oracle DB	Oracle - Deadlocks ORA - 0060	Deadlock detected while waiting for resource	oracl*	Filebeat	Oracle Alert Log	any	timeframe: minutes: 15 filter: - term: oracle.code: “ora-00060”
14	Oracle DB	ORA-00063: maximum number of log files ex-		oracl*	Filebeat	Oracle	any	timeframe: minutes: 15 filter: - term: ora-

Paloalto devices SIEM rules

Nr.	Architecture/App	Rule Name	Description	Index name	Requirement	Sour	Rule type	Rule definition
1		Paloalto - Configuration changes failed	Config changes Failed	paloa *			frequen	timeframe: minutes: 15 num_events: 10 filter: - term: pan.type: CONFIG - term: result: Failed
2		Paloalto - Flood detected	Flood detected via a Zone Protection profile	paloa *			frequen	timeframe: minutes: 15 num_events: 10 filter: - term: pan.type: THREAT - term: pan.subtype: flood
3		Paloalto - Scan detected	Scan detected via a Zone Protection profile	paloa *			frequen	timeframe: minutes: 15 num_events: 10 filter: - term: pan.type: THREAT - term: pan.subtype: scan
4		Paloalto - Spyware detected	Spyware detected via an Anti-Spyware profile	paloa *			frequen	timeframe: minutes: 15 num_events: 10 filter: - term: pan.type: THREAT - term: pan.subtype: spyware
5		Paloalto - Unauthorized configuration changed	Attepmted Unauthorized configuration changes	paloa *			frequen	timeframe: minutes: 15 num_events: 10 filter: - term: pan.type: CONFIG - term: result: Unathorized
6		Paloalto - Virus detected	Virus detected via an Antivirus profile.	paloa *			frequen	timeframe: minutes: 15 num_events: 10 filter: - term: pan.type: THREAT - terms: pan.subtype: [“virus”, “wildfire-virus”]
7		Paloalto - Vulnerability exploit detected	Vulnerability exploit detected via a Vulnerability Protection profile	paloa *			frequen	timeframe: minutes: 15 num_events: 10 filter: - term: pan.type: THREAT - term: pan.subtype: vulnerability

Microsoft Exchange SIEM rules

Nr.	Architecture/	Rule Name	Description	Index name	Requirement	Sour	Rule type	Rule definition
1	MS Exchang	Exchange - Increased amoun of incoming emails		excha *			spike	metric_agg_key: "exchange.network-message-id" metric_agg_type: "cardinality" doc_type: "_doc" max_threshold: 10 buffer_time: minutes: 1 filter: - query_string: query: "exchange.sender-address:.*company.com AND exchange.event-id:SEND AND exchange.message-subject:*" query_key: ["exchange.message-subject-agg", "exchange.sender-address"]
2	MS Exchang	Exchange - Internal sender sent email to public provid		excha *			white	metric_agg_key: "exchange.network-message-id" metric_agg_type: "cardinality" doc_type: "_doc" max_threshold: 10 buffer_time: minutes: 1 filter: - query_string: query: "NOT exchange.sender-address:(*@company.com) AND exchange.event-id:SEND AND exchange.message-subject:* AND NOT exchange.recipient-address:public@company.com" query_key: ["exchange.message-subject-agg", "exchange.sender-address"]
3	MS Exchang	Exchange - Internal sender sent ethe same title to many recipients		excha *			metric_a	filter: - query_string: query: "NOT exchange.recipient-address:public@company.com AND NOT exchange.sender-address:(*@company.com) AND exchange.event-id:SEND AND exchange.data.atch:[1 TO *] AND_exists_:exchange AND exchange.message-subject:(/.*invoice.*/ OR /.*payment.*/ OR /.*faktur.*/)" query_key: ["exchange.sender-address"]
4	MS Exchang	Exchange - Received email with banned title		excha *			any	threshold_ref: 5 timeframe: days: 1 spike_height: 3 spike_type: "up" alert_on_new_data: false use_count_query: true doc_type: _doc query_key: ["exchange.sender-address"] filter: - query_string: query: "NOT exchange.event-id:(DEFER OR RECEIVE OR AGENTINFO) AND *exists*:exchange"
5	MS Exchang	Exchange - The same title to many re-		excha *			metric_a	compare_key: "exchange.sender-address" ignore_null: true filter: - query_string: query: "NOT exchange.recipient-address:(*@company.com) AND exists:*exchange.recipient-address AND exchange.event-id:AGENTINFO AND NOT exchange.sender-address:(bok@ OR postmaster@*) AND exchange.data.atch:[1 TO *] AND exchange.recipient-count:1 AND exchange.recipient-address:(*@gmail.com OR *@wp.pl OR *@o2.pl OR *@interia.pl OR *@op.pl OR

Juniper Devices SIEM Rules

Nr.	Architecture/Appl	Rule Name	Description	Index name	Requirements	Source	Rule type	Rule definition
1	Junos-IDS	Juniper - IDS attact detection		junos	JunOS devices with IDS module	Syslog from Juniper devices	Any	filter: - query_string: query: "*exists*:attack-name" include: - attack-name
2	Junos-IDS	Junos - RT flow session deny		junos	JunOS devices SRX, RT Fflow	Syslog from Juniper devices	Any	filter: - query_string: query: "category:RT_FLOW AND subcat:RT_FLOW_SESSION_DENY" include: - srcip - dstip
3	Junos-IDS	Junos - RT flow reassemble fail		junos	JunOS devices SRX, RT Fflow	Syslog from Juniper devices	Any	filter: - query_string: query: "category:RT_FLOW AND subcat:FLOW_REASSEMBLE_FAIL" include: - srcip - dstip
4	Junos-IDS	Junos - RT flow mcast rpf fail		junos	JunOS devices SRX, RT Fflow	Syslog from Juniper devices	Any	filter: - query_string: query: "category:RT_FLOW AND subcat:FLOW_MCAST_RPF_FAIL" include: - srcip - dstip

Fudo SIEM Rules

Nr.	Architecture/Ap	Rule Name	Description	Index nam	Requirements	Sour	Rule type	Rule definition
1		Fudo - General Error		fudo*	http://download.wheelsystems.com/documentation/fudo/4_0/online_help/en/reference/en/log_messages.html	Syslog FUD	Any	filter: - query_string: query: "syslog_serverity:error" include: - fudo_message
2		Fudo - Failed to authenticate using password		fudo*	http://download.wheelsystems.com/documentation/fudo/4_0/online_help/en/reference/en/log_messages.html	Syslog FUD	Any	filter: - query_string: query: "fudo_code:FSE0634" include: - fudo_user
3		Fudo - Unable to establish connection		fudo*	http://download.wheelsystems.com/documentation/fudo/4_0/online_help/en/reference/en/log_messages.html	Syslog FUD	Any	filter: - query_string: query: "fudo_code:FSE0378" include: - fudo_connection - fudo_login
4		Fudo - Authentication timeout		fudo*	http://download.wheelsystems.com/documentation/fudo/4_0/online_help/en/reference/en/log_messages.html	Syslog FUD	Any	filter: - query_string: query: "fudo_code:FUE0081"

Squid SIEM Rules

Nr.	Architecture/Appli	Rule Name	Description	Index name	Requirements	Sour	Rule type	Rule definition
1	Squid	Squid - Configuration file changed	Modyfing squid.conf file	syslog *	Audit module	syslog	any	filter: - query_string: query: 'message:"File /etc/squid/squid.conf checksum changed."'
2	Squid	Squid - Cannot open HTTP port	Cannot open HTTP Port	squid-*		squid	any	filter: - query_string: query: 'message:"Cannot open HTTP Port"'
3	Squid	Squid - Unauthorized connection	Unauthorized connection, blocked website entry	squid-*		squid	any	filter: - query_string: query: 'squid_request_status:"TCP_DENIED/403"'
4	Squid	Squid - Proxy server stopped	Service stopped	syslog *		syslog	any	filter: - query_string: query: 'message:"Stopped Squid caching proxy."'
5	Squid	Squid - Proxy server started	Service started	syslog *		syslog	any	filter: - query_string: query: 'message:"Started Squid caching proxy."'
6	Squid	Squid - Invalid request	Invalid request	squid-*		squid	any	filter: - query_string: query: 'squid_request_status:"error:invalid-request"'

Microsoft DNS Server SIEM Rules

Nr.	Architecture/Appl	Rule Name	Description	Index name	Requirements	Sour	Rule type	Rule definition
1.	WINDOWS DNS	WIN DNS - Format Error	Format error; DNS server did not understand the update request	prod-win-dns-*			any	timeframe: minutes: 15 filter: - term: dns.result: SERVFAIL
2.	WINDOWS DNS	WIN DNS - DNS server internal error	DNS server encountered an internal error, such as a forwarding timeout	prod-win-dns-*			any	timeframe: - minutes: 15 filter: - term: dns.result: FORMERR
3.	WINDOWS DNS	WIN DNS - DNS refuses to perform the update	DNS server refuses to perform the update	prod-win-dns-*			any	“timeframe: - minutes: 15 filter: - term: dns.result: REFUSED
4.	WINDOWS DNS	WIN DNS - DNS Zone Deleted	DNS Zone delete	prod-win-dns-*			any	timeframe: minutes: 15 filter: - term: event.id: 513
5.	WINDOWS DNS	WIN DNS - DNS Record Deleted	DNS Record Delete	prod-win-dns-*			any	timeframe: minutes: 15 filter: - term: event.id: 516
6.	WINDOWS DNS	WIN DNS - DNS Node Deleted	DNS Node Delete	prod-win-dns-*			any	timeframe: minutes: 15 filter: - term: event.id: 518
7.	WINDOWS DNS	WIN DNS - DNS Remove Trust Point	DNS Remove trust point	prod-win-dns-*			any	timeframe: minutes: 15 filter: - term: event.id: 546
8.	WINDOWS DNS	WIN DNS - DNS Restart Server	Restart Server	prod-win-dns-*			any	timeframe: minutes: 15 filter: - term: event.id: 548
9.	WINDOWS DNS	WIN DNS - DNS Response failure	Response Failure	prod-win-dns-*			frequen	timeframe: minutes: 5 num_events: 20 filter: - term: event.id: 258
10.	WINDOWS DNS	WIN DNS - DNS Ignored Query	Ignored Query	prod-win-dns-*			frequen	timeframe: minutes: 5 num_events: 20 filter: - term: event.id: 259
11.	WINDOWS DNS	WIN DNS - DNS Recursive query timeout	Recursive query timeout	prod-win-dns-*			frequen	timeframe: minutes: 5 num_events: 20 filter: - term: event.id: 262

Microsoft DHCP SIEM Rules

Nr.	Architecture/A	Rule Name	Description	Index name	Requirement	Sour	Rule type	Rule definition
1	Windows DHCI	MS DHCP low disk space	The log was temporarily paused due to low disk space.	prod-win-dhcp-*			any	timeframe: minutes: 15 filter: - term: dhcp.event.id: 02
2	Windows DHCI	MS DHCP lease denied	A lease was denied	prod-win-dhcp-*			frequen	timeframe: minutes: 15 num_events: 10 filter: - terms: dhcp.event.id: [“15”, “16”] include: - dhcp.event.id - src.ip - src.mac - dhcp.event.descr summary_table_field: - src.ip - src.mac - dhcp.event.descr
3	Windows DHCI	MS DHCP update denied	DNS update failed	prod-win-dhcp-*			frequen	timeframe: minutes: 15 num_events: 50 filter: - term: dhcp.event.id: 31
4	Windows DHCI	MS DHCP Data Corruption	Detecting DHCP Jet Data Corruption	prod-win-dhcp-*			any	timeframe: minutes: 15 filter: - term: dhcp.event.id: 1014
5	Windows DHCI	MS DHCP service shutting down	The DHCP service is shutting down due to the following error	prod-win-dhcp-*			any	timeframe: minutes: 15 filter: - term: dhcp.event.id: 1008
6	Windows DHCI	MS DHCP Service Failed to restore database	The DHCP service failed to restore the database	prod-win-dhcp-*			any	timeframe: minutes: 15 filter: - term: dhcp.event.id: 1018
7	Windows DHCI	MS DHCP Service Failed to restore registry	The DHCP service failed to restore the DHCP registry configuration	prod-win-dhcp-*			any	timeframe: minutes: 15 filter: - term: dhcp.event.id: 1019
8	Windows DHCI	MS DHCP Can not find domain	The DHCP/BINL service on the local machine encountered an error while trying to find the domain of the local machine	prod-win-dhcp-*			frequen	timeframe: minutes: 15 filter: - term: dhcp.event.id: 1049
9	Windows DHCI	MS DHCP Network Failure	The DHCP/BINL service on the local machine encountered a network error	prod-win-dhcp-*			frequen	timeframe: minutes: 15 filter: - term: dhcp.event.id: 1050
10	Windows DHCI	MS DHCP - There are no IP addresses available for lease	There are no IP addresses available for lease in the scope or superscope	prod-win-dhcp-*			any	timeframe: minutes: 15 filter: - term: dhcp.event.id: 1063

Linux DHCP Server SIEM Rules

Nr.	Architecture/App	Rule Name	Description	Index name	Requirement	Source	Rule type	Rule definition
1	DHCP Linux	DHCP Linux - Too many requests	Too many DHCP requests	syslog *		Linux DHCP Server / Syslog	frequency	query_key: "src_mac" num_events: 30 timeframe: minutes: 1 filter: - query_string: query: "DHCPREQUEST" use_count_query: true doc_type: doc
2	DHCP Linux	DHCP Linux - IP already assigned	IP is already assigned to another client	syslog *		Linux DHCP Server / Syslog	any	filter: - query_string: query: "DHCPNAK"
3	DHCP Linux	DHCP Linux - Discover flood	DHCP Discover flood	syslog *		Linux DHCP Server / Syslog	frequency	query_key: "src_mac" num_events: 30 timeframe: minutes: 1 filter: - query_string: query: "DHCPDISCOVER" use_count_query: true doc_type: doc

Cisco VPN devices SIEM Rules

Nr.	Architecture/A	Rule Name	Description	Index nam	Requirement	Sour	Rule type	Rule definition
1	Cisco IOS - Cisco VPN Concentrator	CiscoVPN - VPN authentication failed	Jan 8 09:10:37 vpn.example.com 11504 01/08/2007 09:10:37.780 SEV=3 AUTH/5 RPT=124 192.168.0.1 Authentication rejected: Reason = Unspecified handle = 805, server = auth.example.com, user = testuser, domain =	cisco			any	filter: - query_string: query: “cisco.id:(“AUTH/5” OR “AUTH/9” OR “IKE/167” OR “PPP/9” OR “SSH/33” OR “PSH/23”)”
2	Cisco IOS - Cisco VPN Concentrator	CiscoVPN - VPN authentication successful	jw.	cisco			any	filter: - query_string: query: “cisco.id:(“IKE/52”)”
3	Cisco IOS - Cisco VPN Concentrator	CiscoVPN - VPN Admin authentication successful	jw.	cisco			any	filter: - query_string: query: “cisco.id:(“HTTP/47” OR “SSH/16”)”
4	Cisco IOS - Cisco VPN Concentrator	CiscoVPN - Multiple VPN authentication failures	jw.	cisco			frequen	query_key: “src.ip” num_events: 10 timeframe: minutes: 240 filter: - query_string: query: “cisco.id:(“AUTH/5” OR “AUTH/9” OR “IKE/167” OR “PPP/9” OR “SSH/33” OR “PSH/23”)”
5	Cisco IOS - Cisco ASA	Cisco ASA - VPN authentication failed	jw.	cisco			any	filter: - query_string: query: “cisco.id:“%ASA-6-113005””
6	Cisco IOS - Cisco ASA	Cisco ASA - VPN authentication successful	jw.	cisco			any	filter: - query_string: query: “cisco.id:“%ASA-6-113004””
7	Cisco IOS -	Cisco ASA -	jw.	cisco			any	filter: - query_string: query: “cisco.id:“%ASA-

Netflow SIEM Rules

Nr.	Architecture/A	Rule Name	Description	Index nam	Requirement	Sour	Rule type	Rule definition
1		Netflow - DNS traffic abnormal		stream *			spike	threshold_ref: 1000 spike_height: 4 spike_type: up timeframe: hours: 2 filter: - query: query_string: query: "netflow.dst.port:53" query_key: [netflow.src.ip] use_count_query: true doc_type: "doc"
2		Netflow - ICMP larger than 64b		stream *			any	filter: - query: query_string: query: "netflow.protocol: 1 AND netflow.packet_bytes:>64" query_key: "netflow.dst_addr" use_count_query: true doc_type: "doc"
3		Netflow - Port scan		stream *			cardinality	timeframe: minutes: 5 max_cardinality: 100 query_key: [netflow.src.ip, netflow.dst.ip] cardinality_field: "netflow.dst.port" filter: - query: query_string: query: "*exists*:(netflow.dst.ip AND netflow.src.ip) NOT netflow.dst.port: (443 OR 80)" aggregation: minutes: 5 aggregation_key: netflow.src.ip
4		Netflow - SMB traffic		stream *			any	filter: - query: query_string: query: "netflow.dst.port:(137 OR 138 OR 445 OR 139)" query_key: "netflow.src.ip" use_count_query: true doc_type: "doc"
5		Netflow - Too many req to port 161		stream *			frequenc	num_events: 60 timeframe: minutes: 1 filter: - query: query_string: query: "netflow.dst.port:161" query_key: "netflow.src.ip" use_count_query: true doc_type: "doc"
6		Netflow - Too many req to port 25		stream *			frequenc	num_events: 60 timeframe: minutes: 1 filter: - query: query_string: query: "netflow.dst.port:25" query_key: "netflow.src.ip" use_count_query: true doc_type: "doc"
7		Netflow - Too many req to port 53		stream *			frequenc	num_events: 120 timeframe: minutes: 1 filter: - query: query_string: query: "netflow.dst.port:53" query_key: "netflow.src.ip" use_count_query: true doc_type: "doc"
8		Netflow - Multiple connections from source badip		stream *			frequenc	num_events: 10 timeframe: minutes: 5 filter: - query: query_string: query: "netflow.src.badip:true" query_key: "netflow.src.ip" use_count_query: true doc_type: "doc"
9		Netflow - Multiple connections to destination badip		stream *			frequenc	num_events: 10 timeframe: minutes: 5 filter: - query: query_string: query: "netflow.dst.badip:true" query_key: "netflow.dst.ip" use_count_query: true doc_type: "doc"

MikroTik devices SIEM Rules

Nr.	Architecture/A	Rule Nam	Description	Index name	Requirement	Sour	Rule type	Rule definition
1			All system errors				any	alert_text_type: alert_text_only alert_text: "System error\n\n When: {}\n Device IP: {}\n From: {}\n\n{}\n" alert_text_args: - timestamp_timezone - host - login.ip
2			All errors connected with logins to the administrative interface of the device eg wrong password or wrong login name				any	alert_text_type: alert_text_only alert_text: "Login error\n\n When: {}\n Device IP: {}\n From: {}\n by: {}\n to account: {}\n\n{}\n" alert_text_args: - timestamp_timezone - host - login.ip - login.method - user
3			All errors connected with wireless eg device is banned on access list, or device had poor signal on AP and was disconected				any	alert_text_type: alert_text_only alert_text: "Wifi auth issue\n\n When: {}\n Device IP: {}\n Interface: {}\n MAC: {}\n ACL info: {}\n\n{}\n" alert_text_args: - timestamp_timezone - host - interface - wlan.mac - wlan.ACL
4			Dhcp offering fail				any	alert_text_type: alert_text_only alert_text: "Dhcp offering fail\n\n When: {}\n Client lease: {}\n for MAC: {}\n to MAC: {}\n\n{}\n" alert_text_args: - timestamp_timezone - dhcp.ip - dhcp.mac - dhcp.mac2

Microsoft SQL Server SIEM Rules

Nr.	Architecture/Ap	Rule Nam	Description	Index nam	Requirement	Sour	Rule type	Rule definition
1			Logon errors, alert any					Rule definition alert_text_type: alert_text_only alert_text: “Logon error\n\n When: {}\n Error code: {}\n Severity: {}\n\n{}\n” alert_text_args: - timestamp_timezone - mssql.error.code - mssql.error.severity
2			Login failed for users, alert any					alert_text_type: alert_text_only alert_text: “Login failed\n\n When: {}\n User login: {}\n Reason: {}\n Client: {}\n\n{}\n” alert_text_args: - timestamp_timezone - mssql.login.user - mssql.error.reason - mssql.error.client
3			server is going down, alert any					alert_text_type: alert_text_only alert_text: “Server is going down\n\n When: {}\n\n{}\n” alert_text_args: - timestamp_timezone
4			NET stopped, alert any					alert_text_type: alert_text_only alert_text: “NET Framework runtime has been stopped.\n\n When: {}\n\n{}\n” alert_text_args: - timestamp_timezone
5			Database Mirroring stopped, alert any					alert_text_type: alert_text_only alert_text: “Database Mirroring endpoint is in stopped state.\n\n When: {}\n\n{}\n” alert_text_args: - timestamp_timezone

Postgress SQL SIEM Rules

Nr.	Architecture/Ap	Rule Name	Description	Index nam	Requirements	Sour	Rule type	Rule definition
1	PostgreSQL	PostgreSQL - New user created		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"LOG: CREATE USER"'
2	PostgreSQL	PostgreSQL - User selected database		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"LOG: SELECT d.datname FROM pg_catalog.pg_database"'
3	PostgreSQL	PostgreSQL - User password changed		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"ALTER USER WITH PASSWORD"'
4	PostgreSQL	PostgreSQL - Multiple authentication failures		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	frequen	query_key: "src_ip" num_events: 5 timeframe: seconds: 25 filter: - query_string: query: 'message:"FATAL: password authentication failed for user"' use_count_query: true doc_type: doc
5	PostgreSQL	PostgreSQL - Granted all privileges to user		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"LOG: GRANT ALL PRIVILEGES ON"'
6	PostgreSQL	PostgreSQL - User displayed users table	User displayed users table	postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"LOG: SELECT r.rolname FROM pg_catalog.pg_roles"'
7	PostgreSQL	PostgreSQL - New database created		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"LOG: CREATE DATABASE"'
8	PostgreSQL	PostgreSQL - Database shutdown		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"LOG: database system was shut down at"'
9	PostgreSQL	PostgreSQL - New role created		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"LOG: CREATE ROLE"'
10	PostgreSQL	PostgreSQL - User deleted		postg *	Filebeat, Network Probe, PostgreSQL	pg_lc	any	filter: - query_string: query: 'message:"LOG: DROP USER"'

MySQL SIEM Rules

Nr.	Architecture/Ap	Rule Name	Description	Index nam	Requirements	Sourc	Rule type	Rule definition
1	MySQL	MySQL - User created		mysq *	Filebeat, Network Probe, MySQL	mysql genera	any	filter: - query_string: query: 'message:"CREATE USER"'
2	MySQL	MySQL - User selected database		mysq *	Filebeat, Network Probe, MySQL	mysql genera	any	filter: - query_string: query: 'message:"Query show databases"'
3	MySQL	MySQL - Table dropped		mysq *	Filebeat, Network Probe, MySQL	mysql genera	any	filter: - query_string: query: 'message:"Query drop table"'
4	MySQL	MySQL - User password changed		mysq *	Filebeat, Network Probe, MySQL	mysql genera	any	filter: - query_string: query: 'message:"UPDATE mysql.user SET Password=PASSWORD" OR message:"SET PASSWORD FOR" OR message:"ALTER USER"'
5	MySQL	MySQL - Multiple authentication failures		mysq *	Filebeat, Network Probe, MySQL	mysql genera	fre-quen	query_key: "src_ip" num_events: 5 timeframe: seconds: 25 filter: - query_string: query: 'message:"Access denied for user"' use_count_query: true doc_type: doc
6	MySQL	MySQL - All priviliges to user granted		mysq *	Filebeat, Network Probe, MySQL	mysql genera	any	filter: - query_string: query: 'message:"GRANT ALL PRIVILEGES ON"'
7	MySQL	MySQL - User displayed users table	User displayed users table	mysq *	Filebeat, Network Probe, MySQL	mysql genera	any	filter: - query_string: query: 'message:"Query select * from user"'
8	MySQL	MySQL - New database created		mysq *	Filebeat, Network Probe, MySQL	mysql genera	any	filter: - query_string: query: 'message:"Query create database"'
9	MySQL	MySQL - New table created		mysq *	Filebeat, Network Probe, MySQL	mysql genera	any	filter: - query_string: query: 'message:"Query create table"'

Predefined Alert Rules

- *Overview*
- *Static alert rules by group*
- *Adaptive rules (Empowered AI)*
- *Coverage*

Overview

Every Energy Logserver integration package ships a set of predefined alert rules that detect common threats and operational problems for the data source it handles. When you install an integration, its installer imports these rules into the **Alert Rules List**, so you do not have to write detection logic from scratch.

This chapter catalogs those rules. For how alerting works and how to build your own, see *Alerting System*.

Where the rules come from

Access: ELS Console → SIEM → Integrations → *(select an integration)* → **Advanced** → Step 3 **Alerts**

The Advanced wizard lists the rules bundled with an integration before you install them. After installation they appear in **SIEM** → **Alerts** → **Alert Rules List**, grouped by the value in each rule's *Group Name*. This catalog uses the same grouping.

Two classes of rule

The rules split into two kinds, and the difference matters when you plan detection:

- **Static rules** install exactly as written: a fixed query against a fixed index pattern. Behavior is identical on every deployment. These are cataloged below by group.
- **Adaptive rules (Empowered AI)** use a machine-learning model trained on your own data. The rule definition is fixed, but the anomaly threshold is specific to your environment, so behavior is not identical across installations. These are listed in their own section.

Reading the tables

Each static table lists the rules in one group:

- **Alert rule**: the name as it appears in the Alert Rules List.
- **Type**: the detection method (`frequency`, `any`, `flatline`, `spike`, `cardinality`, `new_term`, `metric_aggregation`).
- **Index pattern**: the indices the rule queries.

All predefined rules install **disabled**. Review each rule, adjust its filters and notification method for your environment, then enable it. The default importance is 100 unless a rule states otherwise.

> **Note**
>
> This catalog reflects the integration packages that ship their rule definitions in the Energy Logserver integrations repository. Rules are versioned with their package, so names and counts can change between releases. See *Coverage* for what is included and what is not.

Static alert rules by group

151 static rules across 9 groups. Each installs exactly as listed; adjust filters and notification method for your environment, then enable it.

Microsoft Windows

36 rules, from `beats`.

Alert rule	Type	Index pattern
Windows - Account lock	any	`windows-winlogbeat-*`
Windows - Admin night logon	any	`windows-winlogbeat-*`
Windows - Admin task as user	any	`windows-winlogbeat-*`
Windows - Application error	any	`windows-winlogbeat-*`
Windows - Application hang	any	`windows-winlogbeat-*`
Windows - Audit policy changed	any	`windows-winlogbeat-*`
Windows - Code integrity changed	any	`windows-winlogbeat-*`
Windows - Diff IPs logon	cardinality	`windows-winlogbeat-*`
Windows - Driver loaded	any	`windows-winlogbeat-*`
Windows - Event service error	any	`windows-winlogbeat-*`
Windows - Eventlog service stopped	any	`windows-winlogbeat-*`
Windows - file insufficient privileges	frequency	`windows-winlogbeat-*`
Windows - Firewall rule add	any	`windows-winlogbeat-*`
Windows - Firewall rule deleted	any	`windows-winlogbeat-*`
Windows - Firewall rule modified	any	`windows-winlogbeat-*`
Windows - Kerberos pre-authentication failed	any	`windows-winlogbeat-*`
Windows - Login with explicit credentials	any	`windows-winlogbeat-*`
Windows - Logs deleted	any	`windows-winlogbeat-*`
Windows - Member added to a security-enabled global group	any	`windows-winlogbeat-*`
Windows - Member added to a security-enabled local group	any	`windows-winlogbeat-*`
Windows - Member added to a security-enabled universal group	any	`windows-winlogbeat-*`
Windows - New device	any	`windows-winlogbeat-*`
Windows - New service installed	any	`windows-winlogbeat-*`
Windows - No Logs	flatline	`windows-winlogbeat-*`
Windows - Package installation	any	`windows-winlogbeat-*`
Windows - Password policy change	any	`windows-winlogbeat-*`
Windows - Reset password attempt	any	`windows-winlogbeat-*`
Windows - Security local group was changed	any	`windows-winlogbeat-*`
Windows - Security log full	any	`windows-winlogbeat-*`
Windows - Start up	any	`windows-winlogbeat-*`
Windows - SUNBURST Fingerprint	any	`windows-winlogbeat-*`
Windows - System has been shutdown	any	`windows-winlogbeat-*`
Windows - The system time was changed	any	`windows-winlogbeat-*`

continues on next page

Table 6 – continued from previous page

Alert rule	Type	Index pattern
Windows TaskScheduler - Task created	any	`windows-winlogbeat-*`
Windows TaskScheduler - Task deleted	any	`windows-winlogbeat-*`
Windows TaskScheduler - Task executed	any	`windows-winlogbeat-*`

Microsoft Windows Security

31 rules, from `beats`.

Alert rule	Type	Index pattern
Windows Security - Connection initiated via certutil_exe	any	`windows-winlogbeat-*`
Windows Security - Detect outbound rdp connections over non standard tools	any	`windows-winlogbeat-*`
Windows Security - Detect RDP file creation from suspicious application	any	`windows-winlogbeat-*`
Windows Security - Detect rootkit Moriya	any	`windows-winlogbeat-*`
Windows Security - Detection of relevant antivirus events	any	`windows-winlogbeat-*`
Windows Security - Detects execution of the bash shell	any	`windows-winlogbeat-*`
Windows Security - Detects potential suspicious winget package installation from a suspicious	any	`windows-winlogbeat-*`
Windows Security - Detects the execution of Rundll32.exe	any	`windows-winlogbeat-*`
Windows Security - HackTool - Hashcat Password Cracker Execution	any	`windows-winlogbeat-*`
Windows Security - Malicious - New DLL Registered Via Odbcconf.EXE	any	`windows-winlogbeat-*`
Windows Security - Malicious - Odbcconf.EXE Suspicious DLL Location	any	`windows-winlogbeat-*`
Windows Security - Malicious - Response File Execution Via Odbcconf.EXE	any	`windows-winlogbeat-*`
Windows Security - Malicious - Uncommon Child Process Spawned By Odbcconf.EXE	any	`windows-winlogbeat-*`
Windows Security - Microsoft Malware Protection Engine Crash	any	`windows-winlogbeat-*`
Windows Security - Msi installation from web	any	`windows-winlogbeat-*`

continues on next page

Table 7 – continued from previous page

Alert rule	Type	Index pattern
Windows Security - Mstsc_exe execution from uncommon parent	any	`windows-winlogbeat-*`
Windows Security - New ODBC Driver Registered	any	`windows-winlogbeat-*`
Windows Security - No logs	flatline	`windows-winlogbeat-*`
Windows Security - Potential Access Token Abuse	any	`windows-winlogbeat-*`
Windows Security - Potential netcat reverse shell execution	any	`windows-winlogbeat-*`
Windows Security - Potential Perl Reverse Shell Execution	any	`windows-winlogbeat-*`
Windows Security - Potential PHP Reverse Shell	any	`windows-winlogbeat-*`
Windows Security - Potential Python Reverse Shell	any	`windows-winlogbeat-*`
Windows Security - Potential Ruby Reverse Shell	any	`windows-winlogbeat-*`
Windows Security - Potential SolidPDFCreator.DLL Sideloading	any	`windows-winlogbeat-*`
Windows Security - Potential Xterm Reverse Shell	any	`windows-winlogbeat-*`
Windows Security - Potentially Suspicious Network Connection To Notion API	any	`windows-winlogbeat-*`
Windows Security - Rorschach Ransomware Execution Activity	any	`windows-winlogbeat-*`
Windows Security - Suspicious Chromium Browser Instance Executed With Custom Extensions	any	`windows-winlogbeat-*`
Windows Security - Suspicious Driver/DLL Installation Via Odbcconf.EXE	any	`windows-winlogbeat-*`
Windows Security - Windows Defender Real-Time Protection Failure/Restart	any	`windows-winlogbeat-*`

Oracle

18 rules, from `oracle`.

Alert rule	Type	Index pattern
Oracle - Allocate memory ORA-00090	any	`oracle-*`
Oracle - Client internal error ORA-12643	any	`oracle-*`
Oracle - Credential failed ORA-12638	any	`oracle-*`
Oracle - Deadlocks ORA-00060	any	`oracle-*`
Oracle - Failed to allocate string oom ORA-00025	any	`oracle-*`
Oracle - Incorrect role password ORA-12670	any	`oracle-*`
Oracle - Login failure ORA-12672	any	`oracle-*`
Oracle - Logon denied ORA-12317	any	`oracle-*`
Oracle - Max db_files ORA-00059	any	`oracle-*`
Oracle - Max DML locks ORA-00055	any	`oracle-*`
Oracle - Max licenses exceeded ORA-00019	any	`oracle-*`
Oracle - Max log files ORA-00063	any	`oracle-*`
Oracle - Max processes exceeded ORA-00020	any	`oracle-*`
Oracle - Max sessions exceeded ORA-00018	any	`oracle-*`
Oracle - Max temp locks ORA-00057	any	`oracle-*`
Oracle - No Logs	flatline	`oracle-*`
Oracle - Object too large ORA-00064	any	`oracle-*`
Oracle - Single process login ORA-00024	any	`oracle-*`

Fortigate

17 rules, from `fortigate`.

Alert rule	Type	Index pattern
Fortigate - Attack detected	any	`fortigate-*`
Fortigate - Attack dropped	any	`fortigate-*`
Fortigate - Device configuration changed	any	`fortigate-*`
Fortigate - Failed login	frequency	`fortigate-*`
Fortigate - Firewall configuration changed	any	`fortigate-*`
Fortigate - Forward deny by source IP	frequency	`fortigate-*`
Fortigate - HTTP server attack by destination IP	frequency	`fortigate-*`
Fortigate - Multiple SSL VPN login failed by source IP	frequency	`fortigate-*`
Fortigate - Multiple tunneling by source IP	frequency	`fortigate-*`
Fortigate - Multiple URL blocked by source IP	frequency	`fortigate-*`
Fortigate - No Logs	flatline	`fortigate-*`
Fortigate - SSL VPN login fail	any	`fortigate-*`
Fortigate - Suspicious Traffic	any	`fortigate-*`
Fortigate - Suspicious traffic by source_IP	frequency	`fortigate-*`
Fortigate - Unknown tunneling settings	any	`fortigate-*`
Fortigate - URL blocked	any	`fortigate-*`
Fortigate - Virus Detected and Blocked	any	`fortigate-*`

Netflow

15 rules, from `netflow`.

Alert rule	Type	Index pattern
Netflow - DNS traffic abnormal	spike	`stream-*`
Netflow - First Time Seen Remote Named Pipe - Zeek	frequency	`stream-*`
Netflow - ICMP larger then 64b	any	`stream-*`
Netflow - Multiple connections from source badip	frequency	`stream-*`
Netflow - Multiple connections to destination badip	frequency	`stream-*`
Netflow - No Logs	flatline	`stream-*`
Netflow - Port scan	cardinality	`stream-*`
Netflow - Possible Impacket SecretDump Remote Activity - Zeek	frequency	`stream-*`
Netflow - Remote Task Creation via ATSVC Named Pipe - Zeek	frequency	`stream-*`
Netflow - SMB traffic	any	`stream-*`
Netflow - Suspicious Access to Sensitive File Extensions - Zeek	frequency	`stream-*`
Netflow - Too many req to port 161	frequency	`stream-*`
Netflow - Too many req to port 25	frequency	`stream-*`
Netflow - Too many req to port 53	frequency	`stream-*`
Netflow - Transferring Files with Credential Data via Network Shares - Zeek	frequency	`stream-*`

Ransomware

11 rules, from `beats`.

Alert rule	Type	Index pattern
Ransomware - Base64 encoded shellcode	any	`windows-winlogbeat-*`
Ransomware - File changes	frequency	`windows-winlogbeat-*`
Ransomware - Modification file short time	frequency	`windows-winlogbeat-*`
Ransomware - Modification registry	any	`windows-winlogbeat-*`
Ransomware - Netsh rdp port forwarding	any	`windows-winlogbeat-*`
Ransomware - New process detection	any	`windows-winlogbeat-*`
Ransomware - NotPetya activity	frequency	`winlogbeat*`
Ransomware - Operation file	any	`windows-winlogbeat-*`
Ransomware - Suspicious command execution	any	`windows-winlogbeat-*`
Ransomware - Suspicious script execution	any	`windows-winlogbeat-*`
Ransomware - Suspicious use of calc.exe event_id	any	`windows-winlogbeat-*`

Barracuda

9 rules, from `barracuda`.

Alert rule	Type	Index pattern
Barracuda Firewall - DNS Tunneling Detection	any	barracuda-*
Barracuda Firewall - Failed Connection Attempts	frequency	barracuda-*
Barracuda Firewall - High Volume Data Transfer	metric_aggregation	barracuda-*
Barracuda Firewall - Idle Session Timeout Pattern	cardinality	barracuda-*
Barracuda Firewall - Malicious IP Communication	any	barracuda-*
Barracuda Firewall - New Device Detection	new_term	barracuda-*
Barracuda Firewall - Protocol Anomaly	any	barracuda-*
Barracuda Firewall - Suspicious Country Communication	any	barracuda-*
Barracuda Firewall - Unusual DNS Query Volume	frequency	barracuda-*

Paloalto

8 rules, from `paloalto`.

Alert rule	Type	Index pattern
Paloalto - Configuration changes failed	frequency	syslog-net-fw-palo-*
Paloalto - Flood detected	frequency	syslog-net-fw-palo-*
Paloalto - No Logs	flatline	syslog-net-fw-palo-*
Paloalto - Scan detected	frequency	syslog-net-fw-palo-*
Paloalto - Spyware detected	frequency	syslog-net-fw-palo-*
Paloalto - Unauthorized configuration changed	frequency	syslog-net-fw-palo-*
Paloalto - Virus detected	frequency	syslog-net-fw-palo-*
Paloalto - Vulnerability exploit detected	frequency	syslog-net-fw-palo-*

Waystream

6 rules, from `waystream`.

Alert rule	Type	Index pattern
Waystream - High CPU Load	any	waystream-*
Waystream - High Fan Speed	any	waystream-*
Waystream - High Host Temperature	any	waystream-*
Waystream - High Host Voltage - High Threshold	any	waystream-*
Waystream - High Host Voltage - Low Threshold	any	waystream-*
Waystream - No Logs	flatline	waystream-*

Adaptive rules (Empowered AI)

Note

These rules use a machine-learning model trained on your own deployment's data. The rule definition is fixed, but the anomaly threshold is specific to your environment, so detection behavior is not identical across installations. Train and review the model before enabling.

6 adaptive rules.

Alert rule	Package	Index pattern
AI Windows-winlogbeat Text Anomaly message - All Anomalies	beats	windows-winlogbeat*
AI Windows-winlogbeat Text Anomaly message - Log anomaly score	beats	windows-winlogbeat*
AI Windows-winlogbeat Text Anomaly message - No. of rare words	beats	windows-winlogbeat*
AI Windows-winlogbeat Text Anomaly message - Word anomaly score	beats	windows-winlogbeat*
Barracuda Firewall - Received Bytes Anomaly (Empowered AI)	barracuda	barracuda-*
Barracuda Firewall - Sent Bytes Anomaly (Empowered AI)	barracuda	barracuda-*

Coverage

This catalog lists the static rules and Empowered AI rules that ship with the following integration packages in the Energy Logserver integrations repository: **Beats** (Microsoft Windows, Microsoft Windows Security, Ransomware, Empowered AI), **Oracle, FortiGate, NetFlow, Barracuda, Paloalto, and Waystream**. Every entry is taken from the package source, so it matches what the installer imports.

It does not cover:

- **Other vendor integrations** available in the product but maintained outside the shared integrations repository, for example Check Point, Cisco, FireEye, Suricata, Tenable, Watchguard, and Wazuh. Each ships its own rules. To see the exact set for one of them, open its **Advanced** wizard, Step 3 **Alerts**, on your instance.
- **Rules declared in a package manifest but not distributed with its source** (AWS and Cisco ASA). Install the package through the Advanced wizard to see the rules it imports.

For a deployment's authoritative rule set, the Alert Rules List on that instance is always the source of truth. For a higher-level overview of the alert groups the product bundles, see *Working with Pre-built Alert Groups* in the Alerting System chapter.

4.9 UEBA

User and Entity Behavior Analytics for anomaly detection and insider threat identification.

The User and Entity Behavior Analytics (UEBA) module detects anomalous user activity by aggregating Windows Security Event Logs collected via Winlogbeat or Endpoint Security agents. A Python-based data pipeline, installed as a Network Probe integration package, runs on a configurable schedule (default: every 30 minutes), queries source indices, and writes per-user event summaries to the `uba*` index. Six pre-built Empowered AI rules then analyse these summaries for potential threats such as DDoS, APT, and ransomware patterns. Results are presented across three dashboards: **Events**, **Empowered AI**, and **Raw Logs**.

4.9.1 Prerequisites

The UEBA integration is an optional Network Probe package. Before using UEBA dashboards, ensure the following:

1. **Network Probe** is installed and running.
2. The `uba` integration package is installed from the Network Probe integrations store.
3. A Windows Security Event Log source is available, either `windows-winlogbeat-*` (Winlogbeat) or `wazuh-alerts-*` (Endpoint Security agents).

Configuration

After installation the integration is configured through `config.json` located in the pipeline directory. The file uses the following structure:

```
{
  "input_logservers": [
    {
      "hosts": ["127.0.0.1:9200"],
      "username": "login",
      "password": "password",
      "sources": {
        "winlogbeat": {
          "index_pattern": "windows-winlogbeat-*"
        },
        "wazuh": {
          "index_pattern": "wazuh-alerts-*"
        }
      }
    }
  ],
  "output_logserver": {
    "hosts": ["127.0.0.1:9200"],
    "username": "login",
    "password": "password",
    "uba_index": "uba"
  },
  "interval": {
    "type": "minutes",
    "value": 30
  }
}
```

Both `winlogbeat` and `wazuh` sources can be active at the same time. Remove a source block to disable it.

Key fields:

Field	Description	Default
`input_logservers[].sources.winlogbeat.index_pattern`	Source index for Winlogbeat data	`windows-winlogbeat-*`
`input_logservers[].sources.wazuh.index_pattern`	Source index for Endpoint Security agent data	`wazuh-alerts-*`
`output_logserver.uba_index`	Destination index name	`uba`
`interval.value`	Aggregation interval	`30`
`interval.type`	Interval unit (`minutes` or `hours`)	`minutes`
`ignore_users`	Filters out system accounts: machine accounts ending with `$`, service prefixes `DWM-`/`UMFD-`, and exact matches `SYSTEM`, `USŁUGA LOKALNA`, `-`, `MSSQL$MICROSOFT##WID`	pre-configured

4.9.2 Events

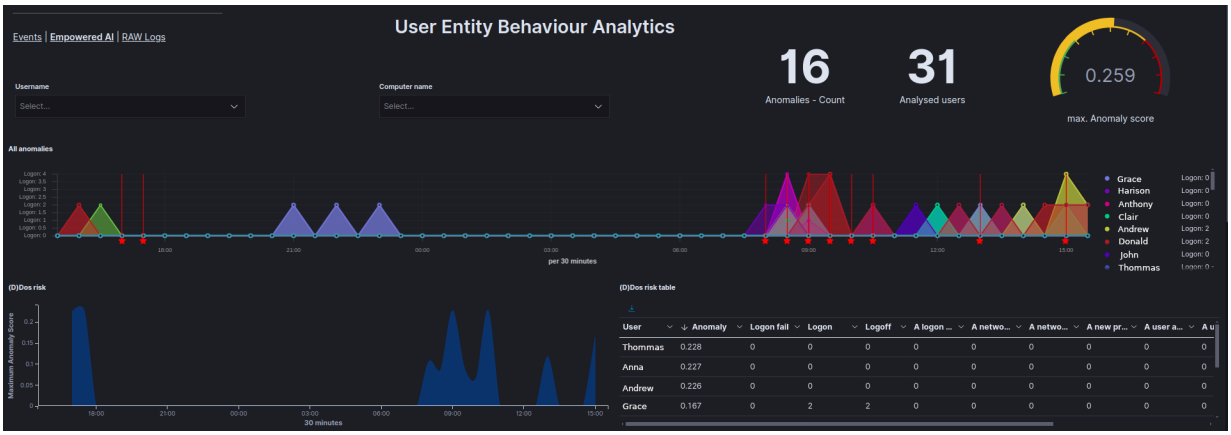
The `[UEBA] Overview` dashboard presents aggregated event data from the `uba*` index. Data can be filtered by `User`, `Computer name`, `Event`, and `Source`.



Data is visualized using the following graphs:

- **Count of Logon/off** - Total count of `Logon`, `Logoff`, and `Failed to Logon` events within a specified time range.
- **Login and authentication** - Count of login and authentication events as a function of time.
- **Access and privilege management** - Count of access and privilege management events as a function of time.
- **Configuration and system registry management** - Count of configuration and system registry management events as a function of time.
- **Service and process management** - Count of service and process management events as a function of time.
- **Management of facilities and access to resources** - Count of facility management and access-to-resource events as a function of time.
- **Account and group management** - Count of account and group management events as a function of time.
- **Top 10 reported events** - Ranking of the most frequently occurring events.
- **Top 10 users by events count** - Ranking of users generating the most events.
- **Top 5 docs count** - Table showing the top 5 users by document count.
- **User - Events metrics** - Number of actions for each event based on the user.
- **Matrix User - Computer Name** - A matrix showing which users are using which devices.

4.9.3 Empowered AI

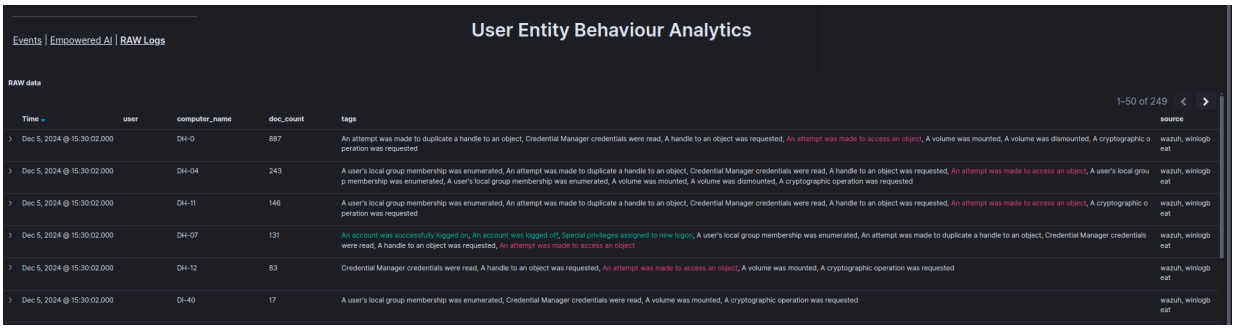


The `[UEBA] Empowered AI` dashboard displays results from six pre-built AI rules that target the `uba*` index. The Empowered AI service creates them on start, so they appear in **Empowered AI** → **Use Cases** as `[UBA]-(D)Dos-Probability`, `[UBA]-APT-Probability`, `[UBA]-Ransomware-Probability`, `[UBA]-all-events-Probability`, `[UBA]-logon-anomaly`, and `[UBA]-service-installation-anomaly`. They arrive with the schedule set to **Run Once**, which means the dashboard stays empty until you run each rule or give it a schedule. See *Default AI Rules*. Each rule analyses per-user event summaries and produces an `Anomaly_score`. The dashboard contains the following visualizations:

- **Anomalies timeline** - Displays all anomalies on a timeline per user.
- **(D)DoS attack probability** - Shows the results of analyses checking whether a (D)DoS attack has occurred.
- **(D)DoS risk table** - Displays the results of the analysis for each user in the `Anomaly_score` field and allows for the review of values for each field analyzed.
- **APT attack probability** - Displays the results of analyses checking for potential APT attacks.
- **APT risk table** - Displays the results of the analysis for each user in the `Anomaly_score` field and allows for the review of values for each field analyzed.
- **Ransomware attack probability** - Displays the results of analyses checking for potential ransomware attacks.
- **Ransomware risk table** - Displays the results of the analysis for each user in the `Anomaly_score` field and allows for the review of values for each field analyzed.
- **All events anomaly** - Displays the results of analyses for all fields provided by the UBA data pipeline to identify abnormal user behavior.
- **All events anomaly table** - Displays the results of the analysis for each user in the `Anomaly_score` field and allows for the review of values for each field analyzed.
- **Services installation anomaly** - Displays the results of analyses related to service installation anomalies.
- **Logon anomaly** - Displays the results of analyses related to `Logon`, `Logoff`, and `Failed to Logon` events.
- **Users risk scores** - Displays the highest and average `Anomaly_score` for each user and the type of analysis in which that score occurred.
- **High risk users** - Shows a pie chart highlighting the users most affected by anomalies.

4.9.4 Raw Logs

The `[UEBA] RAW Logs` dashboard displays raw records from the `uba*` index in a table, enabling verification of specific events, including the time, user, and device involved.



See also *AI and Analytics* for additional machine learning capabilities including anomaly detection rules and clustering.

4.10 AI and Analytics

Advanced AI and machine learning capabilities for anomaly detection, clustering, and intelligent analytics.

The AI and Analytics module provides powerful machine learning capabilities integrated directly into Energy Logserver. From text anomaly detection to multivariate analysis and clustering, this module empowers security teams with AI-driven insights for proactive threat detection and operational intelligence.

4.10.1 Overview

Energy Logserver's AI and Analytics capabilities enable proactive threat detection and security intelligence through machine learning. Key features include:

- **Text Anomaly Detection**: NLP-based analysis to identify unusual text patterns in logs and events
- **Univariate Analysis**: Single-variable anomaly detection for time-series data
- **Multivariate Analysis**: Multi-dimensional correlation for complex pattern detection
- **Clustering**: Automated grouping of similar events and behaviors for pattern analysis
- **Model Library**: Pre-built AI models and access to AI Store for additional capabilities
- **AI Assistant**: Intelligent guidance and recommendations for security operations

For user and entity behavioral analysis, see the *UEBA* module.

4.10.2 In This Section

This chapter covers the complete AI and Analytics implementation, from basic use cases to advanced model management.

Overview

`Empowered AI` is an advanced module of Energy Logserver platform, designed to enhance event detection, correlation, and data analysis across IT and OT environments. It combines mathematical data-analysis methods together with language-model-based detection, creating a powerful and comprehensive analytical engine.

The module uses statistical techniques to identify anomalies, recognize behavioral patterns, and detect deviations that may indicate security incidents or system failures.

A key component of Empowered AI is the use of Large Language Models (LLMs), including our dedicated on-site extension "AI on Prem". This technology enables local execution of language models within the customer's environment, ensuring no data leaves the organization. As a result, Empowered AI delivers advanced semantic detection, event classification, and automated analytical support while maintaining full data privacy and security compliance.

By combining mathematical precision with contextual understanding from language models, Empowered AI provides a modern, multi-layered approach to detection and analytics in complex log environments.

Important to note that Use Cases can work in batch or in realtime - connected to Network Probe pipeline.

`Empowered AI` is an ongoing project, continuously improved by a team of mathematicians, data scientists, and security analysts.

Use Cases

In the `Empowered AI` section you will find a summary of the existing use cases, connecting rules and data to work with. At the top, you'll find the total number of configured cases and the number of scheduled and unscheduled cases. Here is the search field and buttons `Refresh`, `Create` and `Upload`.

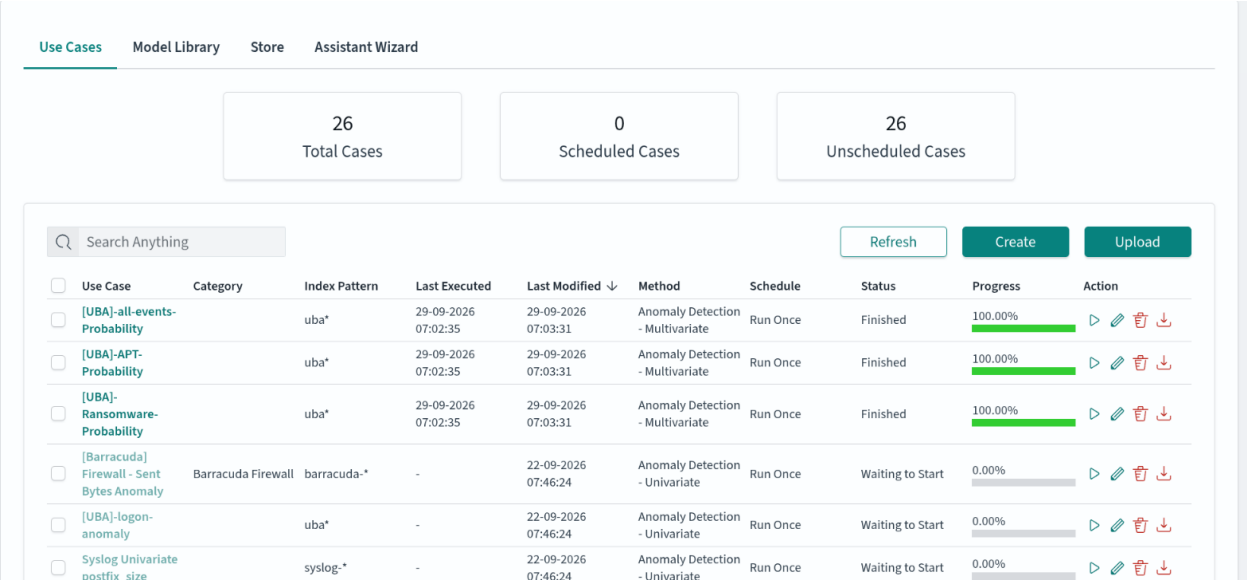


Table contains the following columns:

- `Use Case` - unique name for the rule running on selected data
- `Category` - given category name upon create
- `Index Pattern` - data to work with
- `Last Executed` - date when last use case was Executed
- `Last Modified` - when last use case configuration was changed
- `Method` - selected algorithm
- `Schedule` - configuration of scheduling options
- `Status` - current calculation status
- `Progress` - progress bar in %
- `Action` - additional use case management

Status

The rule has one of the following statuses:

- Waiting to Start - `Run Once` rule starts by clicking symbol play
- Scheduled - the scheduled rule starts automatically
- Scoring
- Building
- Finished
- Error, Build Error, Score Error - click the use case name to see the error details in the Performance view

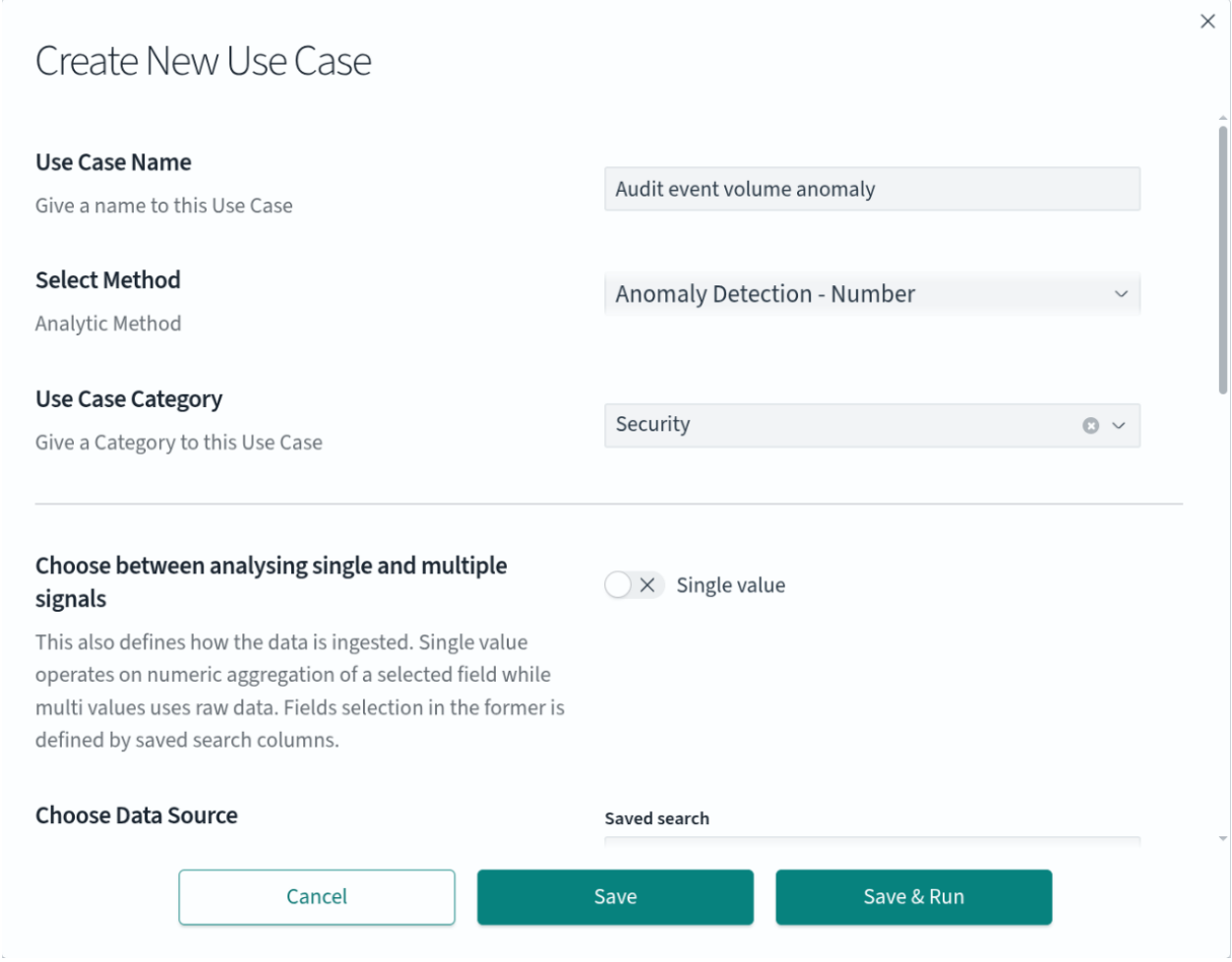
Actions

Icons of actions:

- Play - run or rerun the rule
- Stop - shown while the use case is running or scheduled; stops the running process or unschedules a periodic rule, after this action the rule type changes to Run Once
- Pencil - edit the rule's configuration
- Bin - delete the rule
- Download - get the full usecase, rule definition and its config

Create Use Case

To create a new use case, click the **Create** button in the Use Cases tab. The **Create New Use Case** dialog will open.

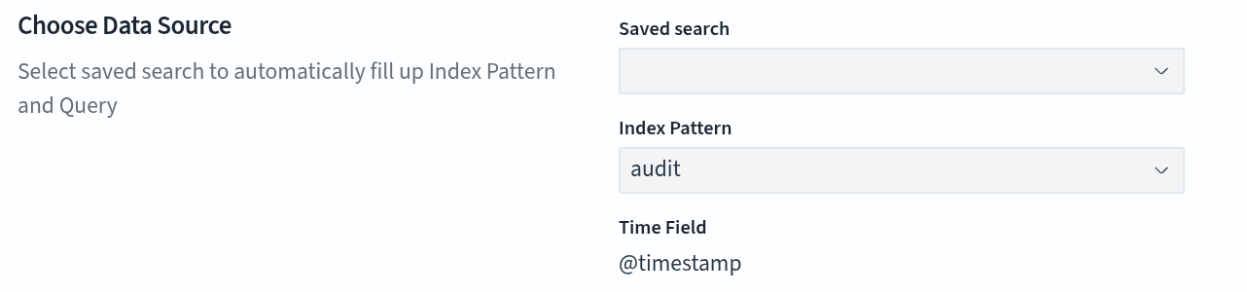


Configuring the Use Case

1. Enter a name in the **Use Case Name** field.
2. Select the analytical method from the **Select Method** dropdown: **Anomaly Detection - Number**, **Anomaly Detection - Text**, or **Clustering**.
3. Select or enter a category in the **Use Case Category** field.
4. For **Anomaly Detection - Number**, use the **Single value / Multi values** switch to choose between univariate and multivariate analysis.

Data Source

In the **Choose Data Source** section, select the index pattern in the **Index Pattern** field. The **Time Field** is taken from the index pattern. You can narrow down the data with a JSON query (Query DSL) in the **Query** field.



To fill in the Index Pattern and Query fields from Discover, select a saved search in the **Saved search** field. The use case takes over the Lucene query and the filters of the saved search. A query written in DQL, the default Discover language, is dropped without a warning, so switch the query language to Lucene before you save the search in Discover.

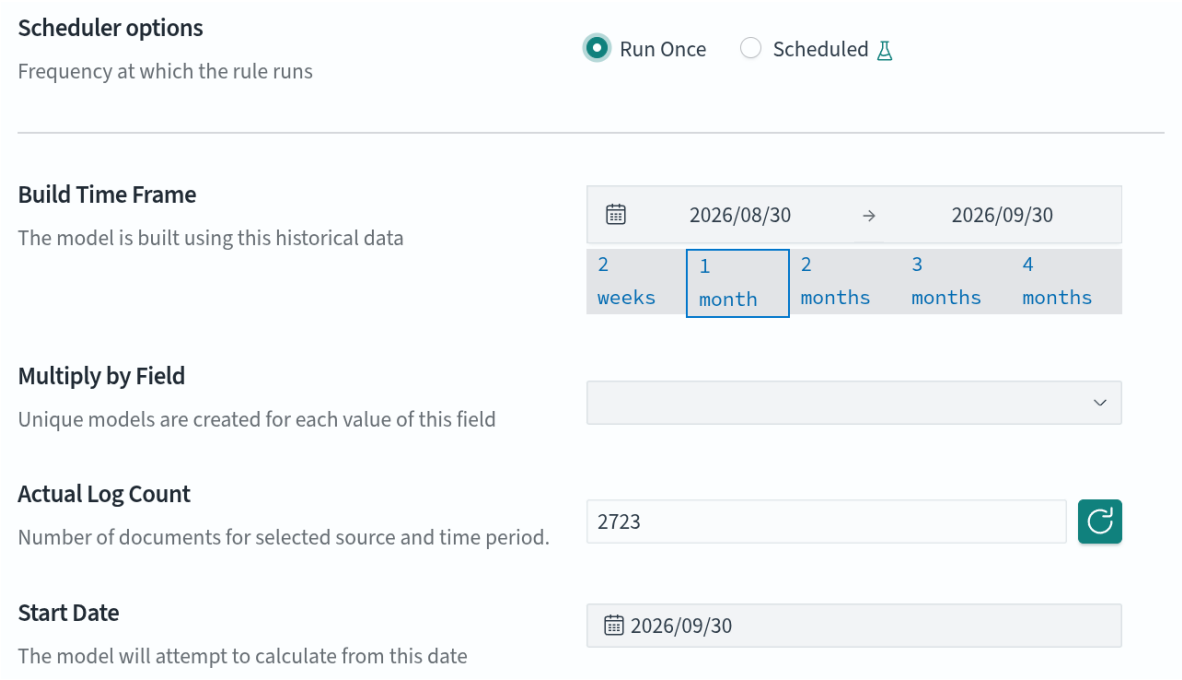
In the **Field to Analyse** section, select the field to analyse. The list shows the index pattern fields that match the selected method; select **Show all fields** to list every field. For **Anomaly Detection - Number** with a single value, you can select **Take 'log count' itself as a signal to analyse** to analyse the number of logs instead of a field.



Configuring the Scheduler

The use case can be run once or on a schedule:

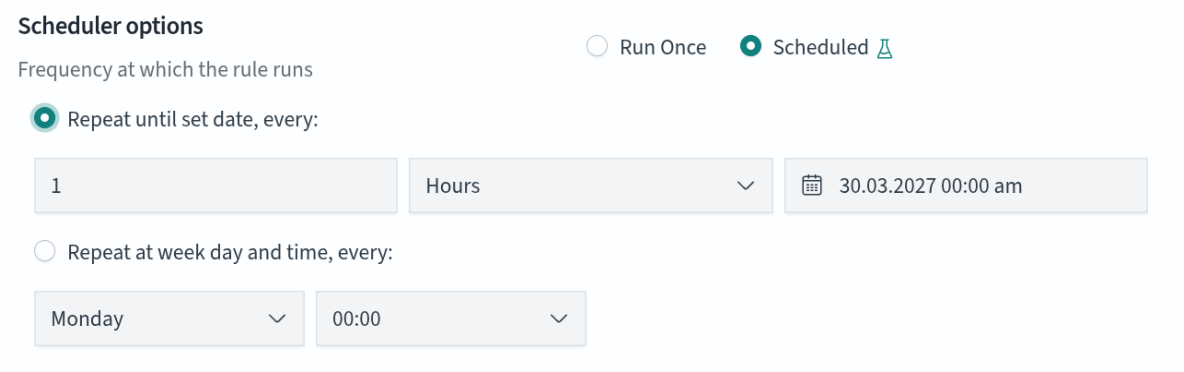
- **Run Once**: runs the analysis once. Provide the **Build Time Frame** (the historical data used to build the model, with shortcuts from 2 weeks to 4 months) and the **Start Date** from which the model calculates results. **Actual Log Count** shows the number of documents for the selected source and time period.



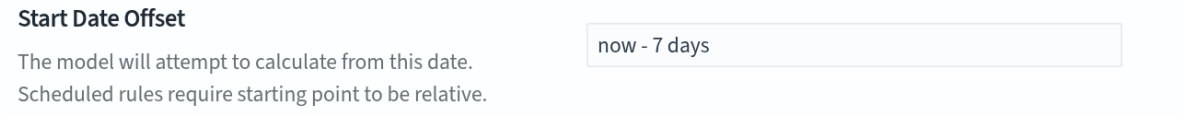
- **Scheduled**: runs the analysis repeatedly. This option is experimental, which the GUI marks with a beaker icon.

Choose one of the repeat modes:

- **Repeat until set date, every:** runs every given number of hours, days, weeks, or months until the selected date,
- **Repeat at week day and time, every:** runs on the selected day of the week at the selected time.



A scheduled use case also uses the **Build Time Frame**. Instead of a fixed start date, it takes a **Start Date Offset** relative to the run time, for example `now - 7 days`.



The remaining fields depend on the selected method and are described in *Anomaly Detection* and *Advanced Analytics*.

Click **Save** to save the use case, or **Save & Run** to save it and start it immediately.

Accessing Performance Results

To view the results of a completed use case, click on its name in the Use Cases list. The Performance view will open, displaying the Use Case Configuration and Model Performance sections.

Anomaly Detection

Text Anomaly Detection focuses on analyzing words in log data, searching for anomalies. For each stream of logs algorithm will store statistical model containing calculated probability of word occurrence. Each source produce different type of log typical for the vendor. Using statistical approach we can easily identify uncommon entries and detect suspicious behavior in logs. Trained models can be saved in Model Library and used in real time in Network Probe pipeline.

- *Options*
- *Performance Tab for Text Anomaly Detection*
  - *Understanding the Performance Graphs*
  - *Using the Performance Graphs*
  - *Anomaly Detection Table*
  - *Information in the Anomaly Detection Table:*
  - *Expanding a Row*

- *Filtering by Rare Words*
- *Actions in the Table*
- *Distinct Anomaly List Table*

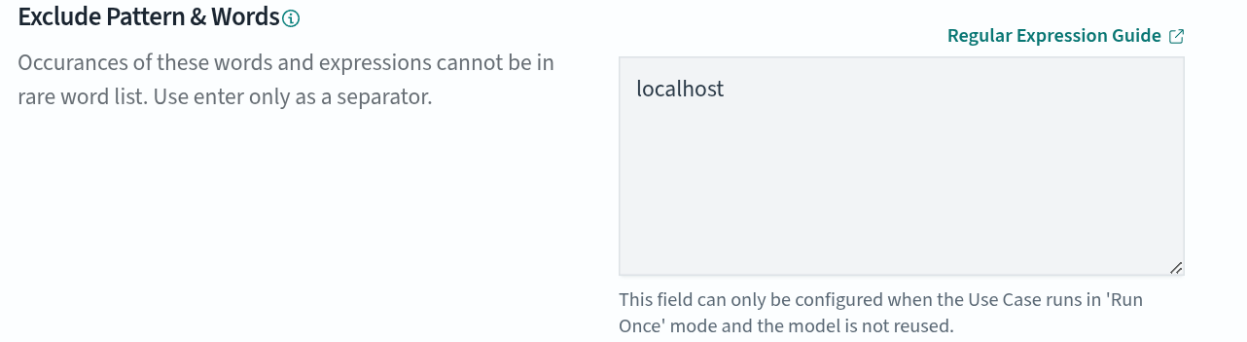
- *Univariate Anomaly Detection*
  - *Step 1: Configuring the Univariate Anomaly Detection Rule*
  - *Step 2: Running the Univariate Rule*
- *Performance Tab for Univariate Anomaly Detection*
  - *Understanding the Performance Graph*
- *Multivariate Anomaly Detection*
  - *Step 1: Configuring the Multivariate Anomaly Detection Rule*
  - *Step 2: Running the Multivariate Rule*
- *Performance Tab for Multivariate Anomaly Detection*
  - *Visualizing Anomalies Over Time*

Options

Select **Anomaly Detection - Text** in the **Select Method** field and set the common fields described in *Create Use Case*. The method adds the following fields:

- **AI Model**: **Build New From Scratch** trains a new model, **Take From Model Library** reuses a model saved in the *Model Library*.
- **Exclude Pattern & Words**: words and regular expressions that cannot appear in the rare word list.
- **Skip numbers**: omits standalone numbers from the analysis.
- **Sampling Rate** and **Sampled Log Count**: the percentage of documents to use and the resulting number of documents.
- **Rareness Threshold**: the share of logs below which a word counts as rare.
- **Create Alert**: creates an alert rule for the detected anomalies.

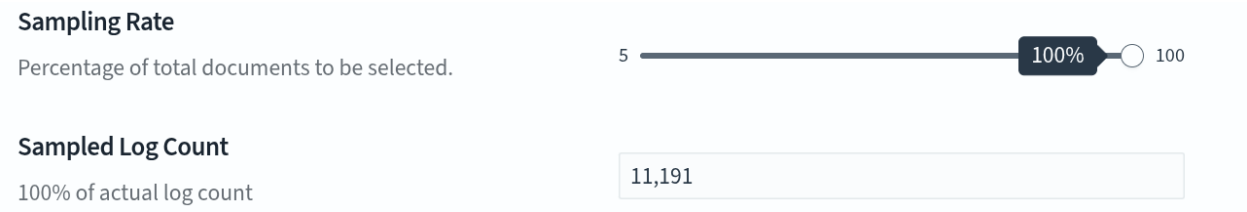
Exclude Pattern & Words: Enter one word or regular expression per line; the **Regular Expression Guide** link opens a syntax reference. You can set this field only when the use case runs in **Run Once** mode and does not reuse a model from the Model Library.



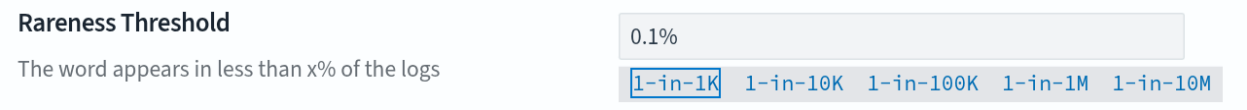
Skip numbers: When selected, standalone numbers are ignored in the analysis. Each number can significantly boost false-positive ratio. The option is off by default.

Actual Log Count: Gives information about number of documents for the selected source and build time frame. If the number becomes huge, consider using sampling.

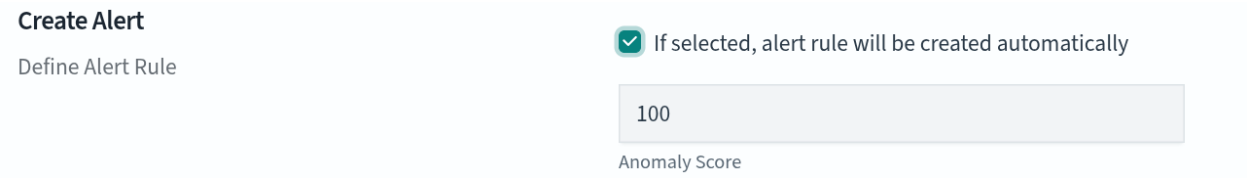
Sampling Rate: Set the percentage of documents to use, from 5% to 100% in steps of 5%. **Sampled Log Count** shows the resulting number of documents.



Rareness Threshold: A word counts as rare when it appears in less than the given percentage of logs. The shortcuts set values from **1-in-1K** (0.1%) to **1-in-10M** (0.00001%). The field is required.

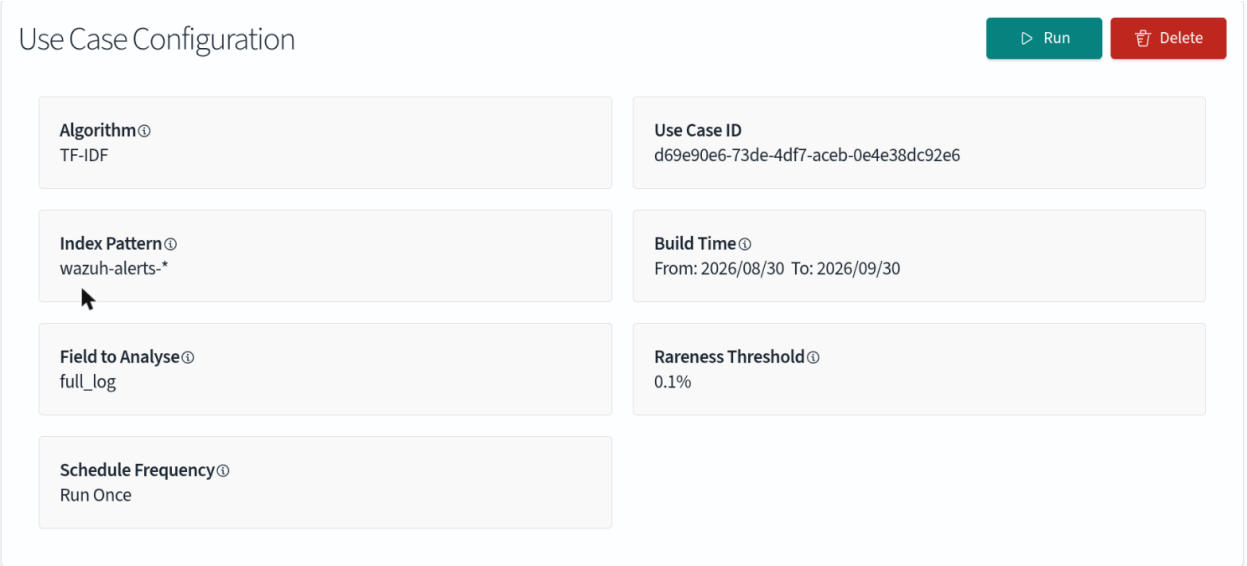


Create Alert: When selected, saving the use case also creates an alert rule named `AI <use case name>` in the **Empowered AI** group. The rule fires for documents of this use case whose log anomaly score is higher than the value in the **Anomaly Score** field (100 by default). Deleting the use case does not delete this alert rule; delete it in **Alerts > Alert Rules List**.



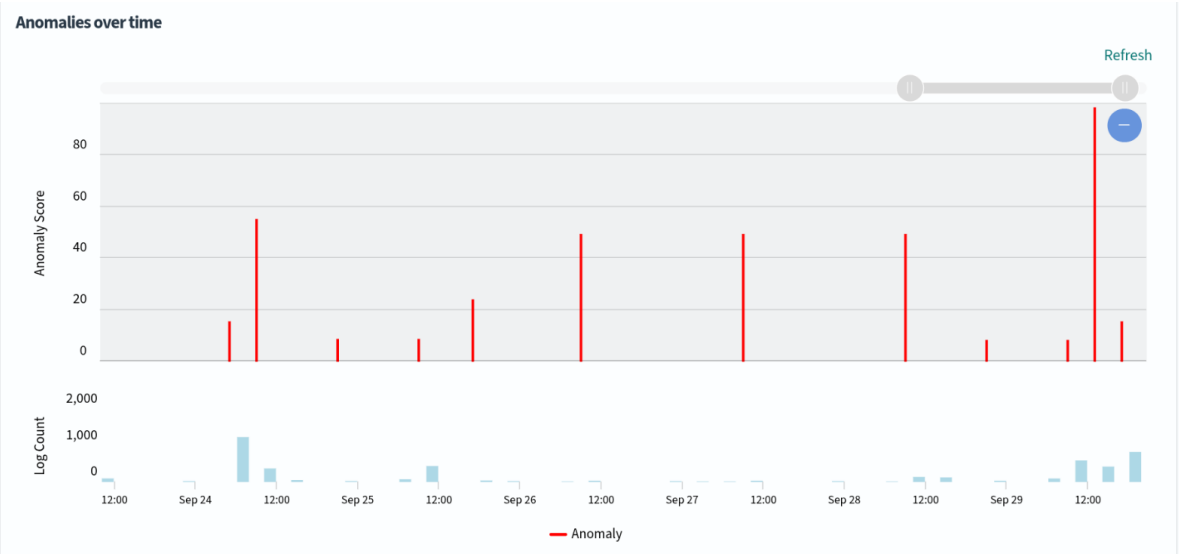
Performance Tab for Text Anomaly Detection

The **Use Case Configuration** section shows the settings of the use case, together with the **Run** and **Delete** buttons.



Understanding the Performance Graphs

- **Anomalies over time**: The red bars show the log anomaly score of each anomaly over time. The lower **Log Count** panel shows the number of logs per hour. Drag the slider handles above the chart to zoom in on a period.



- **Spread of Anomalies**: Shows the distribution of anomaly scores from 0 to 100. Each dot represents the score of one anomalous log. Drag the slider handles to limit the table below to a score range.

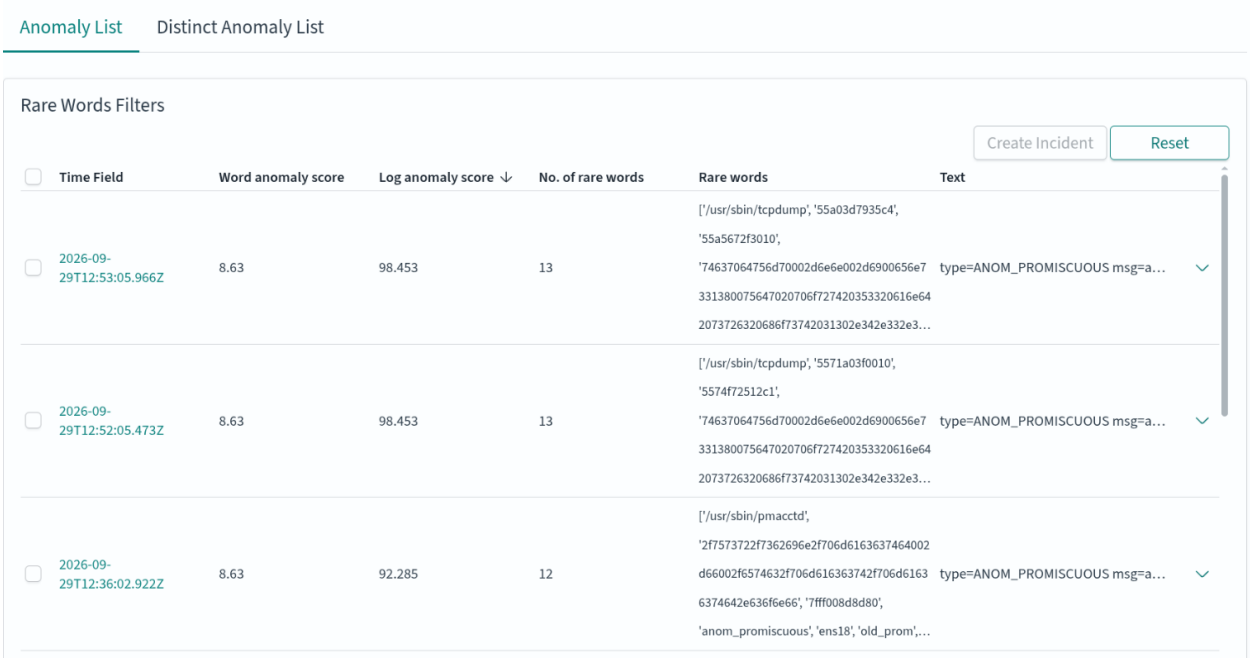


Using the Performance Graphs

1. **Monitoring Trends**: Allows you to monitor trends and identify periods of abnormal activity.
2. **Investigating Anomalies**: By examining the anomaly markers, you can investigate the corresponding timestamps to understand the context.
3. **Adjusting Parameters**: If too many or too few anomalies are detected, consider adjusting the rareness threshold or sampling rate.

Anomaly Detection Table

At the bottom of the **Performance** tab, the **Anomaly List** tab lists the detected anomalies, sorted by log anomaly score in descending order.



Information in the Anomaly Detection Table:

- **Time Field**: The time of the anomalous log. Click it to open the log in Discover.
- **Word anomaly score**: The highest score among the rare words in the log.
- **Log anomaly score**: The anomaly score for the entire log.
- **No. of rare words**: The number of rare words detected in the log.
- **Rare words**: The list of rare words detected in the log.
- **Text**: A fragment of the text containing the detected anomalies.

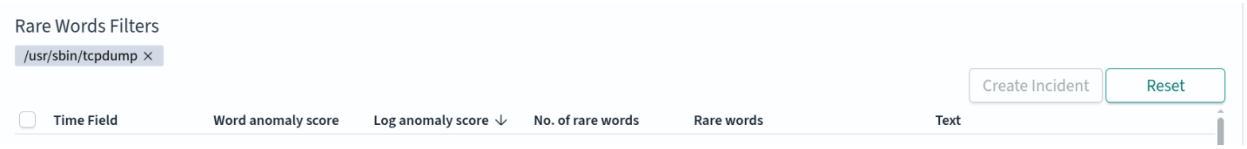
Expanding a Row

Click the arrow icon at the end of a row to expand it. The expanded row shows the full text of the log and the **Rare Words Filtering** list with every rare word found in it.



Filtering by Rare Words

In the **Rare Words Filtering** list of an expanded row, click **+** next to a word to show only the logs that contain it, or **-** to hide them. Each filter appears as a badge in the **Rare Words Filters** section at the top of the table. Click the badge to remove the filter.



Actions in the Table

- **Create Incident**: Creates an incident for each selected row. The button is active when at least one row is selected.
- **Reset**: Clears the row selection and the score range selected on the Spread of Anomalies chart. Rare word filters stay in place.

Distinct Anomaly List Table

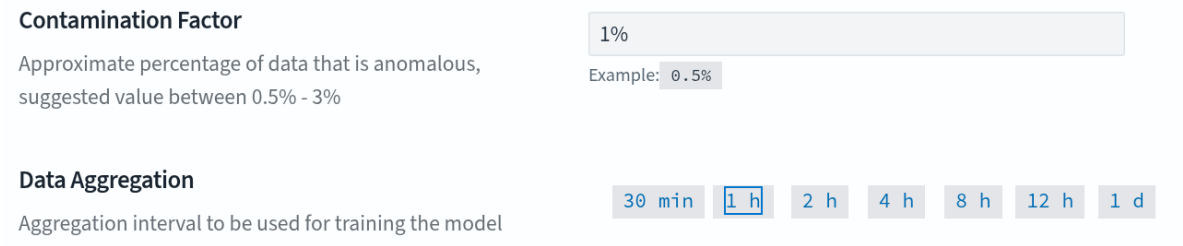
The **Distinct Anomaly List** tab contains the same anomaly data as the first table but without duplicates. If the message is the same, it is displayed only once, and the **Number of occurrences** column shows how many times it was found. In the first table, all analyzed documents detected as anomalies are displayed, even if one document appears 200 times.

Univariate Anomaly Detection

Univariate Anomaly Detection identifies anomalies in a single number field. Select **Anomaly Detection - Number** in the **Select Method** field and leave the switch at **Single value**.

Step 1: Configuring the Univariate Anomaly Detection Rule

- **Field to Analyse**: Select a numeric field, or select **Take 'log count' itself as a signal to analyse** to analyse the number of logs.
- **Multiply by Field**: Optionally select a field; the use case builds a separate model for each value of this field.
- **Contamination Factor**: The approximate percentage of data considered anomalous. The suggested value is between 0.5% and 3% (for example, `0.5%`).
- **Data Aggregation**: Select the aggregation interval to be used for training the model. Available intervals are 30 minutes, 1 hour, 2 hours, 4 hours, 8 hours, 12 hours, and 1 day.



- Complete the configuration of other settings, such as **Start Date**, **Build Time Frame**, and **Scheduler options**. These settings are described in *Create Use Case*.

Step 2: Running the Univariate Rule

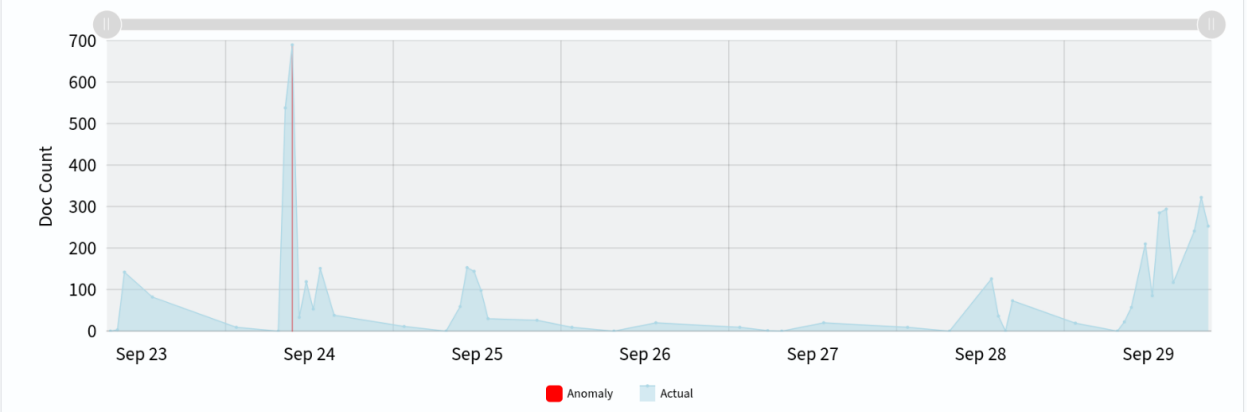
Click **Save & Run** to start the anomaly detection immediately, or click **Save** and start the use case later with the play icon in the Use Cases list.

Performance Tab for Univariate Anomaly Detection

The Performance Tab provides a visual representation of the model's performance, particularly focusing on the detected anomalies in a single data column over time.

Understanding the Performance Graph

The performance graph displays the aggregated values of the analysed field and highlights detected anomalies. The x-axis represents the timeline, while the y-axis shows the values of the analysed field, or **Doc Count** when the use case analyses the log count.



Key Elements of the Performance Graph

1. **Actual (blue area)**
   - The blue area represents the actual values of the monitored data over time. It helps in visualizing the normal behavior of the data and identifying patterns or trends.
2. **Anomaly (red lines)**
   - Red vertical lines mark the points where anomalies have been detected, showing the specific times when the data deviated from the normal pattern.

Drag the slider handles above the chart to zoom in on a period.

Interpreting the Graph

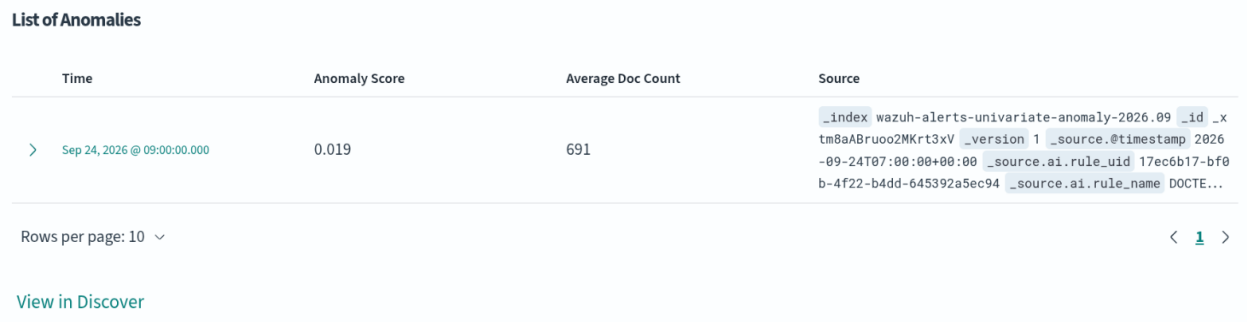
- **Spikes and Drops**: Significant spikes or drops in the blue area could indicate unusual activity or anomalies in the data. Red lines on these spikes or drops confirm the detection of anomalies by the model.
- **Consistency**: Consistent patterns without red lines suggest stable data behavior with no detected anomalies.

List of Anomalies

Below the chart, the **List of Anomalies** table lists the detected anomalies with the following columns:

- **Time**: The start of the aggregation interval in which the anomaly was detected.
- **Anomaly Score**: The score of the anomaly.
- The aggregated value of the analysed field, for example **Average Doc Count**.
- **Source**: The result document stored by the use case.

Click **View in Discover** to open the results in Discover.



Using the Performance Graph

1. **Monitoring Trends**: The graph allows you to monitor trends and identify periods of abnormal activity.
2. **Investigating Anomalies**: By examining the red lines, you can investigate the corresponding timestamps to understand the context of the anomalies.
3. **Adjusting Parameters**: If too many or too few anomalies are detected, consider adjusting the contamination factor or other model parameters to improve detection accuracy.

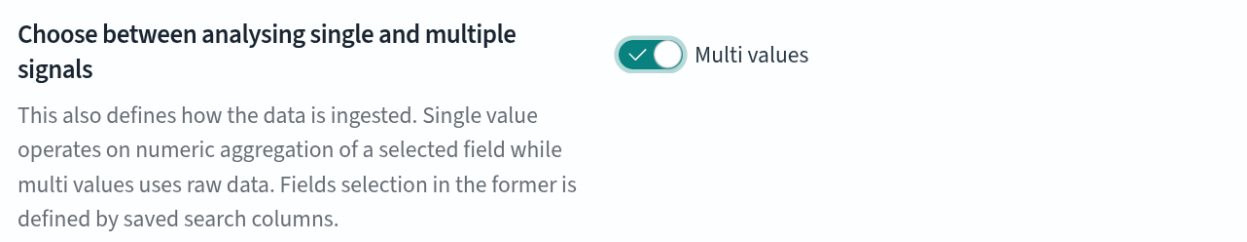
Multivariate Anomaly Detection

Multivariate Anomaly Detection identifies anomalies across multiple data columns. Below is a guide on configuring this use case.

Step 1: Configuring the Multivariate Anomaly Detection Rule

Choose Between Analyzing Single and Multiple Signals

- **Multi Values**: Select **Anomaly Detection - Number** in the **Select Method** field and turn the switch to **Multi values**. Then select several numeric fields in **Field to Analyse**; the use case analyses them together.



- **Contamination Factor**: The approximate percentage of data considered anomalous. The suggested value is between 0.5% and 3% (for example, `0.5%`).

- Complete the configuration of other settings, such as **Start Date**, **Build Time Frame**, and **Scheduler options**. These settings are described in *Create Use Case*.

Step 2: Running the Multivariate Rule

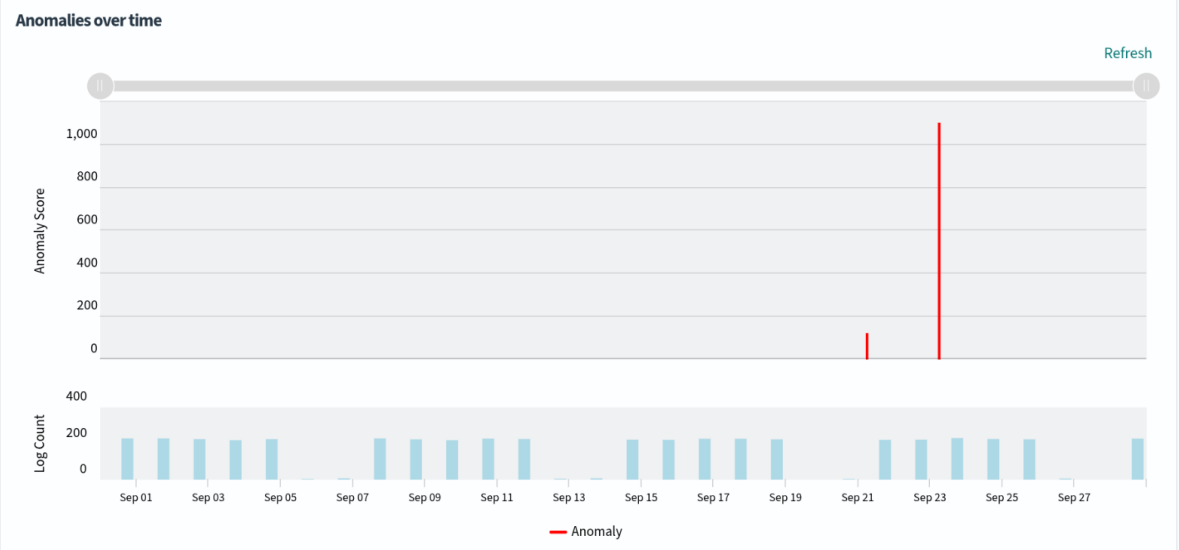
Click **Save & Run** to start the anomaly detection immediately, or click **Save** and start the use case later with the play icon in the Use Cases list.

Performance Tab for Multivariate Anomaly Detection

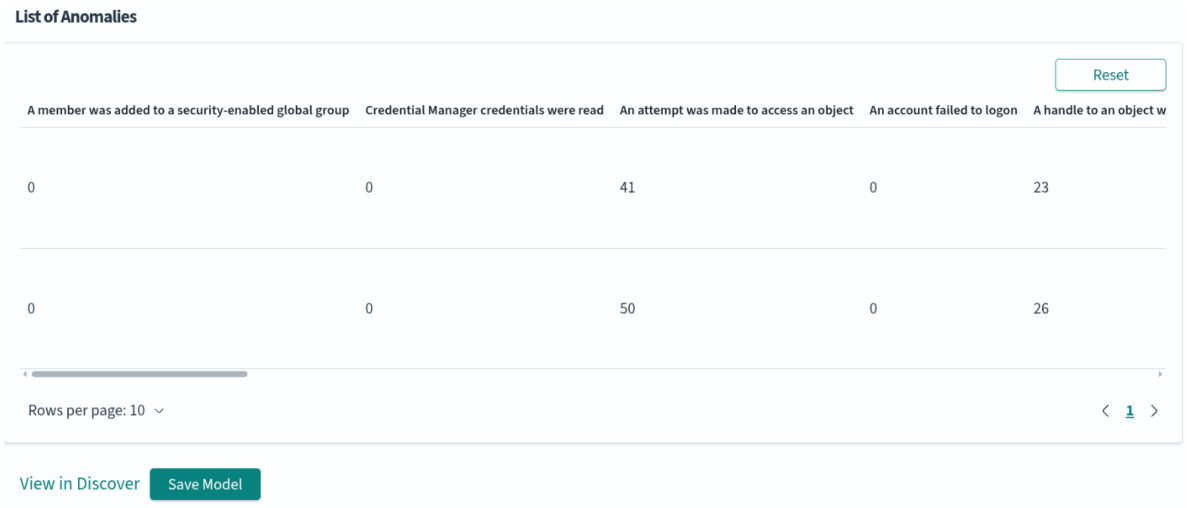
The Performance Tab provides comprehensive visualizations and detailed information on detected anomalies over time. When the use case uses **Multiply by Field**, select the field value at the top of the page to see its results.

Visualizing Anomalies Over Time

- **Anomalies over time**: The X-axis represents time, while the Y-axis represents the anomaly score. Each red vertical line indicates an anomaly detected by the model at a specific time. The lower **Log Count** panel shows the number of logs. This visualization helps in identifying periods of high anomaly activity.



- **Spread of Anomalies**: Shows the distribution of anomaly scores across the entire dataset. Each dot represents an anomaly score, and this chart helps to understand the spread and severity of anomalies in the data.
- **List of Anomalies**: Below the charts, a table lists the values of the analysed fields for each anomaly. The columns are the analysed fields. **Reset** clears the selection, **View in Discover** opens the results in Discover, and **Save Model** saves the trained model in the *Model Library*.



Advanced Analytics

Clustering

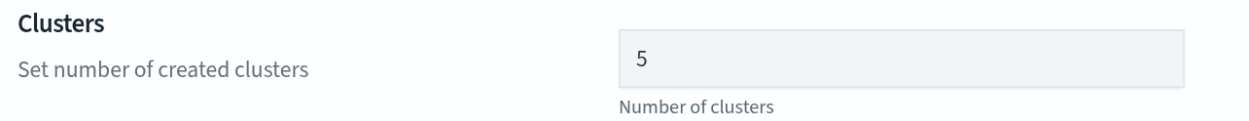
Clustering groups similar data into clusters with the k-means algorithm. Below is a guide on configuring clustering in the *Empowered AI* module.

Step 1: Configuring the Clustering Rule

Select **Clustering** in the **Select Method** field and select one or more fields in **Field to Analyse**. The method accepts fields of any type.

Number of clusters

- In the **Clusters** field, specify the number of clusters to be created (5 by default, at least 2). This number depends on the nature of your data and the objectives of your analysis.



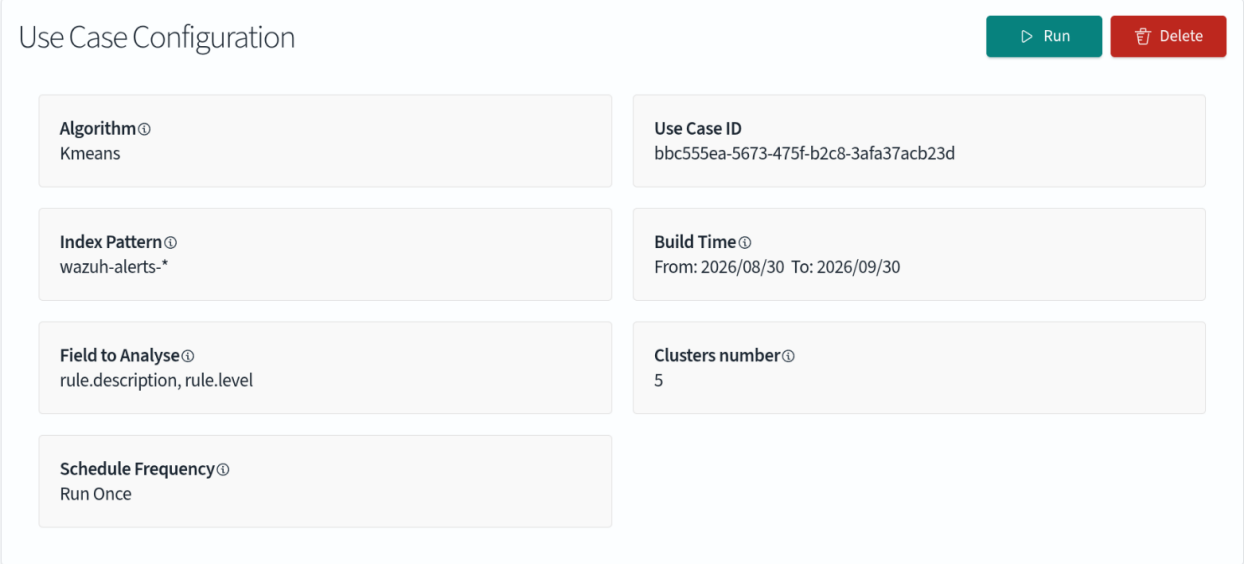
- In **AI Model**, choose **Build New From Scratch** to train a new model or **Take From Model Library** to reuse a saved one.
- Complete the configuration of other settings, such as **Start Date**, **Build Time Frame**, and **Scheduler options**. These settings are described in *Create Use Case*.

Step 2: Running the Rule

Click **Save & Run** to start the clustering immediately, or click **Save** and start the use case later with the play icon in the Use Cases list.

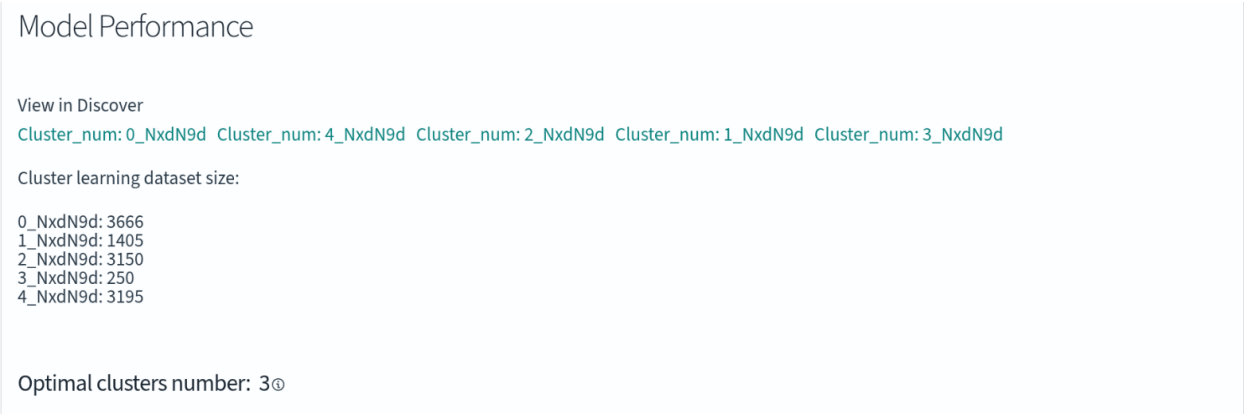
Performance Tab for Clustering

The **Use Case Configuration** section shows the algorithm (**Kmeans**), the analysed fields, and the number of clusters, together with the **Run** and **Delete** buttons.



At the top of the **Model Performance** section:

- **View in Discover** lists one link per cluster; each link opens the documents of that cluster in Discover.
- **Cluster learning dataset size** shows how many documents of the learning dataset fall into each cluster.
- **Optimal clusters number** shows the number of clusters suggested by the elbow method.



Elbow Method and Optimal Number of Clusters

The elbow method is a commonly used technique in clustering analysis to determine the optimal number of clusters. It involves plotting cluster quality against the number of clusters and identifying the point where the quality improvement slows down, forming an "elbow" shape on the graph. This point indicates the optimal number of clusters.

Cluster Quality Chart

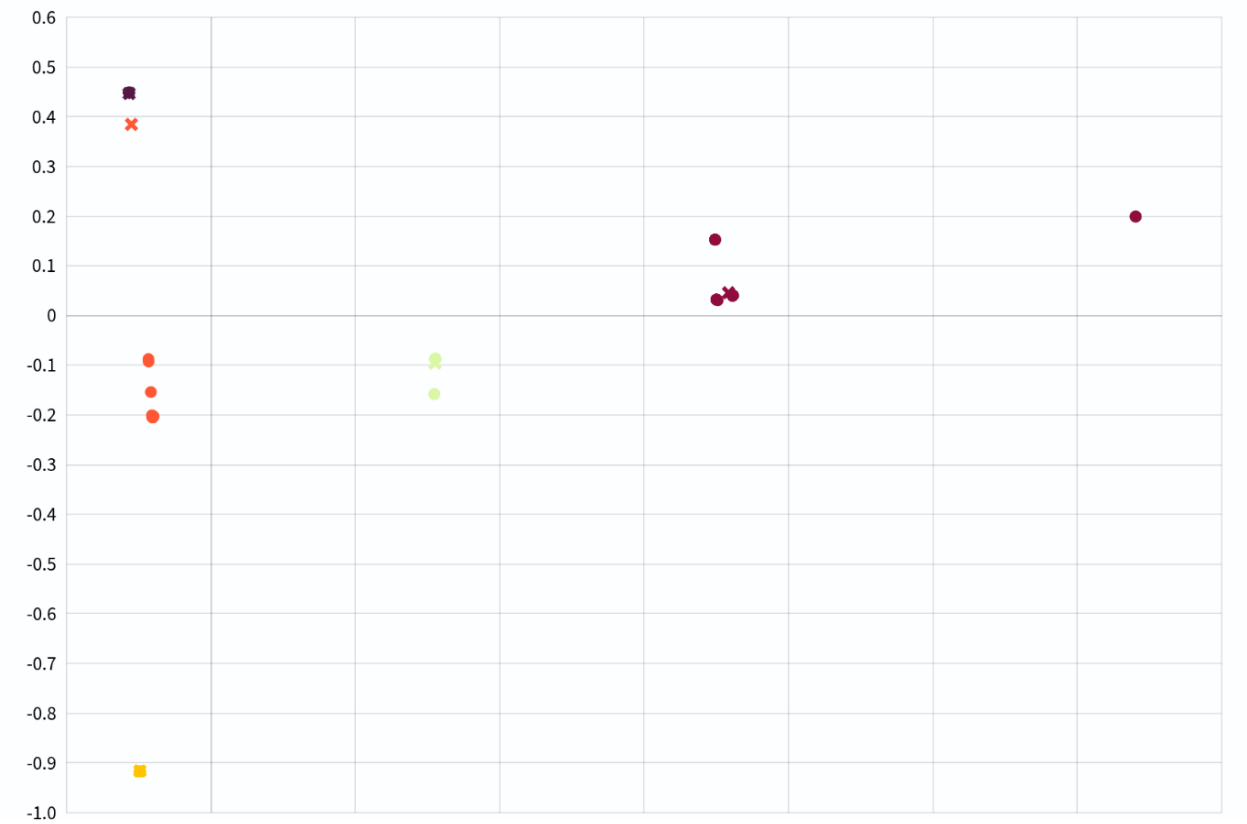
The **Cluster Quality Chart** visualizes the relationship between the number of clusters and the cluster quality. The x-axis represents the number of clusters, from 2 to 9, and the y-axis represents the cluster quality score. The goal is to identify the point where adding more clusters does not significantly improve quality, which is the optimal number of clusters.



The suggested value appears above the chart as **Optimal clusters number**. If it differs from the number you set in **Clusters**, consider changing the setting and running the use case again.

Cluster Distribution Chart

Below the elbow method chart in the **Performance** tab, there is a cluster distribution chart. This chart shows the distribution of documents among clusters along with their centers. Each dot on the chart represents a document assigned to a specific cluster, and the cross marks the center of the cluster in the same color.



- **Clicking on a Dot**: Clicking on any dot on the chart opens the **Preview** window with the values of the analysed fields in that document.



- **Indicative Number of Documents**: The chart shows up to 100 documents per cluster, not the entire dataset. This is done to avoid overcrowding the chart and to provide better readability and data analysis.

Benefits of the Cluster Distribution Chart

The cluster distribution chart helps in:

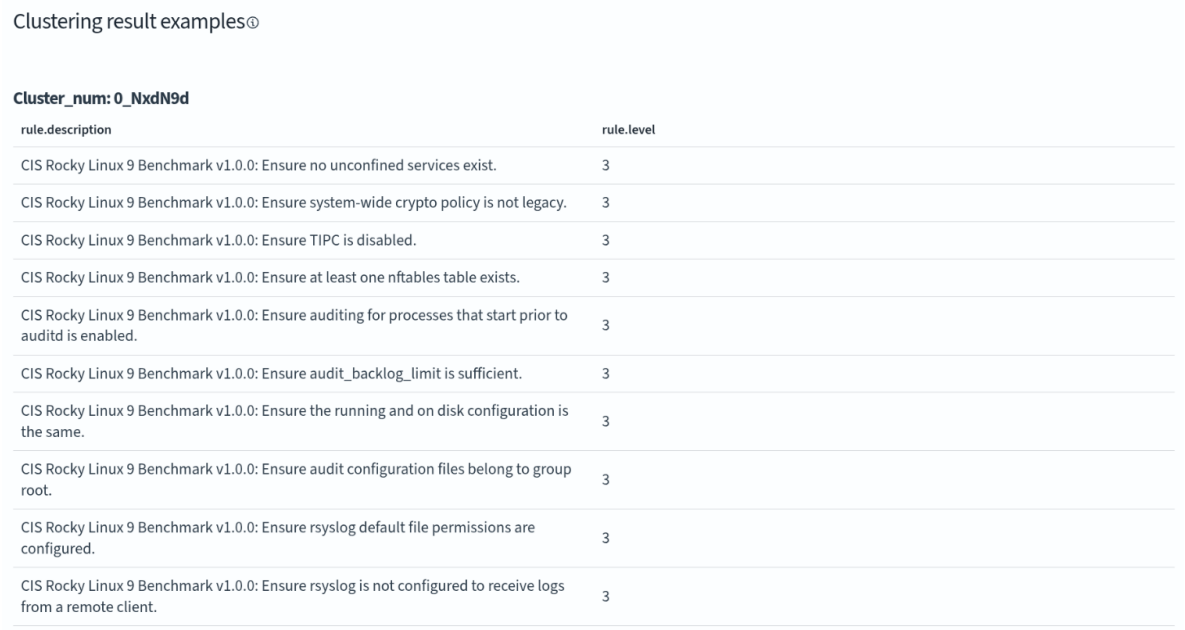
- **Visualizing Document Groupings**: It enables understanding how documents are grouped into clusters and their distributions.
- **Quick Anomaly Identification**: By analyzing the document distribution, unusual groupings can be quickly identified, which may indicate anomalies.
- **Analyzing Cluster Centers**: Cluster centers help evaluate which features are most representative of a given cluster.

Clustering Result Examples

Below the cluster distribution chart, the **Clustering result examples** section displays up to 10 example documents from each cluster. These documents are not a full set but a representative sample.

Understanding the Clustering Result Examples

- **Document Representation**: Each cluster section lists example documents with the values of the analysed fields.



Clustering result examples

Cluster_num: 0_NxdN9d

rule.description	rule.level
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure no unconfined services exist.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure system-wide crypto policy is not legacy.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure TIPC is disabled.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure at least one nftables table exists.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure auditing for processes that start prior to auditd is enabled.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure audit_backlog_limit is sufficient.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure the running and on disk configuration is the same.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure audit configuration files belong to group root.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure rsyslog default file permissions are configured.	3
CIS Rocky Linux 9 Benchmark v1.0.0: Ensure rsyslog is not configured to receive logs from a remote client.	3

- **Sample Data**: The documents shown are not exhaustive. Only a sample is displayed to give an idea of the type of data grouped into each cluster. This helps in understanding the nature and characteristics of the clusters without overwhelming the user with too much data.

At the bottom of the section, **Save Model** saves the trained model in the *Model Library*.

Benefits of Viewing Clustering Result Examples

The clustering result examples help in:

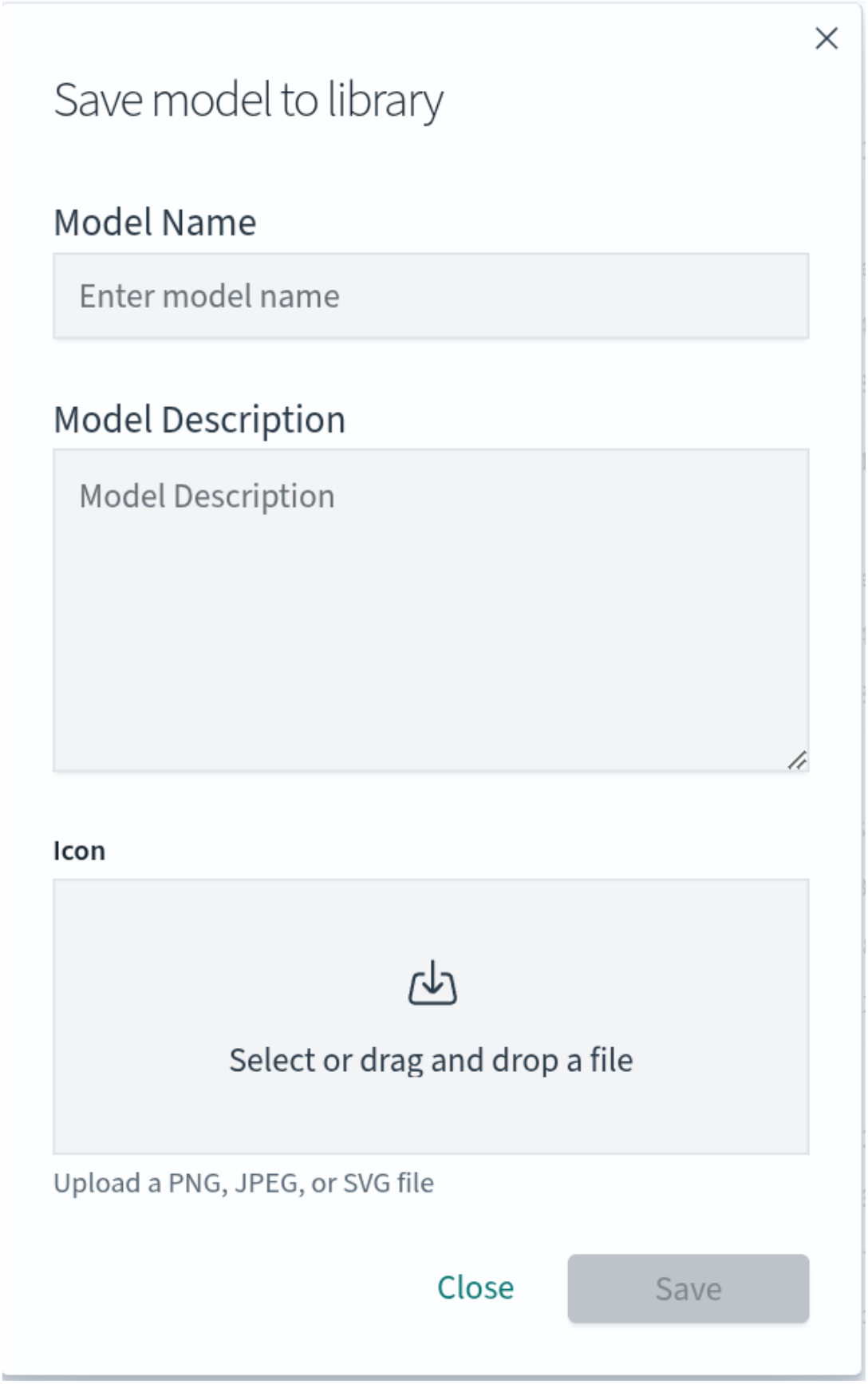
- **Validating Cluster Quality**: By looking at the example documents, you can quickly assess whether the clustering algorithm has grouped the documents meaningfully.
- **Identifying Patterns**: Seeing sample documents from each cluster allows you to identify common patterns or features within each cluster.
- **Quick Reference**: The representative samples provide a quick reference to the kinds of documents in each cluster, aiding in faster analysis and decision-making.

Model Library

Each use case relies on a trained model. When you create a new use case, the selected algorithm builds and trains a model from your data. Models can serve a single use case or be saved to the library for reuse across multiple use cases. Models can also be exported and imported between Energy Logserver instances.

Saving a Model

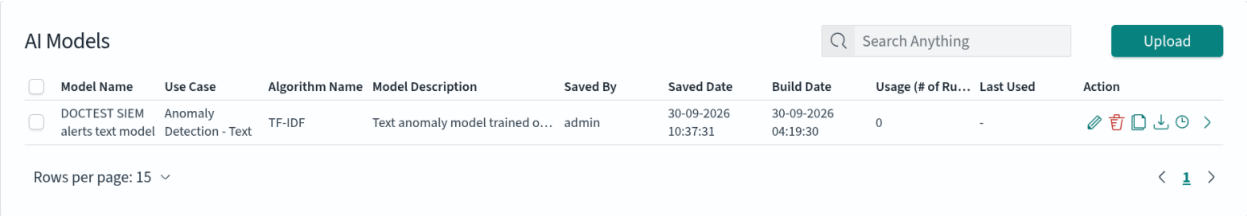
To save a trained model, open the use case from the Use Cases list and click **Save Model** in the Performance view. The button is available for **Anomaly Detection - Text**, multivariate **Anomaly Detection - Number** (**Multi values**), and **Clustering**; univariate models cannot be saved to the library. In the **Save model to library** window, enter the **Model Name**, optionally a **Model Description** and an **Icon** (PNG, JPEG, or SVG), and click **Save**.


Save model to library
Model Name
Enter model name
Model Description
Model Description
Icon
Select or drag and drop a file
Upload a PNG, JPEG, or SVG file
Close
Save

A saved model can be selected in a new use case with **AI Model > Take From Model Library**.

Model Library Tab

The **Model Library** tab lists the saved models in the **AI Models** table.

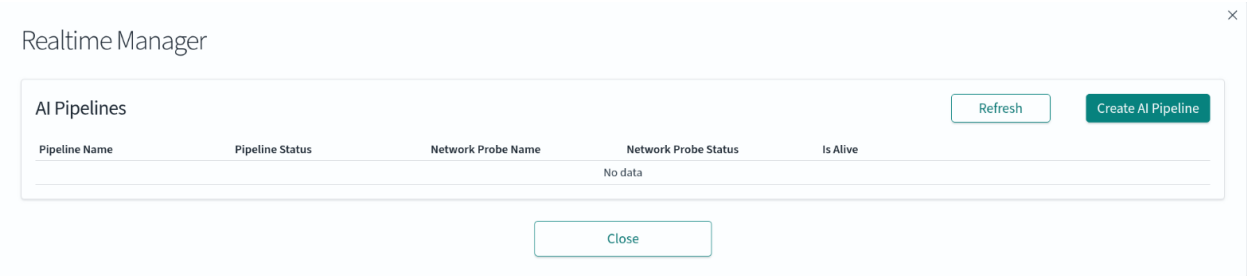


The table shows the **Model Name**, the **Use Case** method, the **Algorithm Name**, the **Model Description**, who saved the model and when (**Saved By**, **Saved Date**), the **Build Date**, the number of use cases using the model (**Usage (# of Rules)**), and **Last Used**. The **Action** column offers:

- **Edit**: changes the model details.
- **Delete**: deletes the model. If use cases use the model, the confirmation lists them, and they are deleted together with the model.
- **Make a copy**: creates a copy of the model.
- **Download**: saves the model to a file, which you can import in another Energy Logserver instance with the **Upload** button above the table.
- **Realtime** (clock icon): opens the Realtime Manager described below.

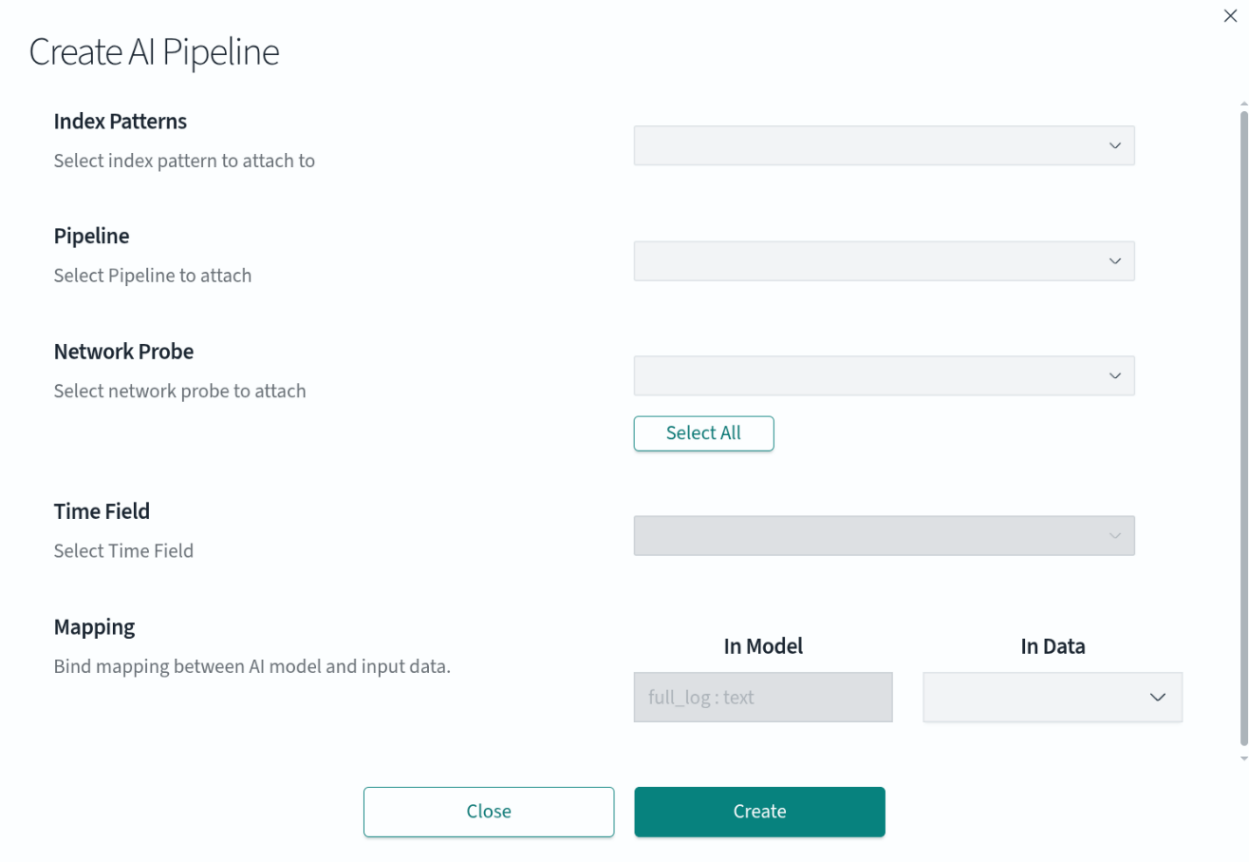
Realtime Manager

Models from the library can run in realtime on Network Probe. Click the clock icon (**Realtime**) next to a model to open the **Realtime Manager**. The **AI Pipelines** table lists the pipelines that already use the model, with the pipeline and Network Probe status.



Click **Create AI Pipeline** to attach the model to an existing Network Probe pipeline:

1. In **Index Patterns**, select the index pattern of the data.
2. In **Pipeline**, select the pipeline to attach the model to.
3. In **Network Probe**, select the probes, or click **Select All**.
4. In **Time Field**, select the time field.
5. In **Mapping**, bind each field used by the model (**In Model**) to a field of the realtime data (**In Data**). A model trained on one dataset can be reused on another, so field names in the realtime stream may differ from those used during training.
6. Click **Create**.



Default AI Rules

Default Rules automatically deploy a set of rules for the syslog, Endpoint Security, HTTPD, UBA, and Barracuda indices at startup, enabling users to quickly start analyzing data.

Default rules are loaded automatically on every service start, 26 of them in total. The service creates only the rules that are missing: a default rule that you changed keeps your changes, and a default rule that you deleted comes back after the next restart. Each one is created with the schedule set to **Run Once**, so it produces no results until it is started from the Use Cases list or given a schedule. A default rule needs the matching index pattern in Energy Logserver. They are grouped by data source:

Syslog (`syslog-*`):

- Syslog Text Anomaly message
- Syslog Univariate network.bytes
- Syslog Univariate network.ttl
- Syslog Univariate postfix_delay
- Syslog Univariate postfix_delay_transmission
- Syslog Univariate postfix_size
- Syslog Univariate count
- Syslog Clustering message
- Syslog Forecast network.bytes
- Syslog Forecast network.ttl

- Syslog Forecast postfix_delay
- Syslog Forecast postfix_delay_transmission
- Syslog Forecast postfix_size
- Syslog Forecast count

The Syslog Forecast rules use the **Forecasting** method, which is not available in the **Select Method** field for new use cases.

Endpoint Security (`wazuh-alerts-*`):

- Wazuh-alerts Text Anomaly full_log
- Wazuh-alerts Text Anomaly data.win.eventdata.data

HTTPD (`httpd*`):

- Httpd Text Anomaly message
- Httpd Clustering message

UBA (`uba*`):

- [UBA]-(D)Dos-Probability (Multivariate)
- [UBA]-APT-Probability (Multivariate)
- [UBA]-Ransomware-Probability (Multivariate)
- [UBA]-all-events-Probability (Multivariate)
- [UBA]-logon-anomaly (Univariate)
- [UBA]-service-installation-anomaly (Multivariate)

Barracuda (`barracuda-*`):

- [Barracuda] Firewall - Received Bytes Anomaly (Univariate)
- [Barracuda] Firewall - Sent Bytes Anomaly (Univariate)

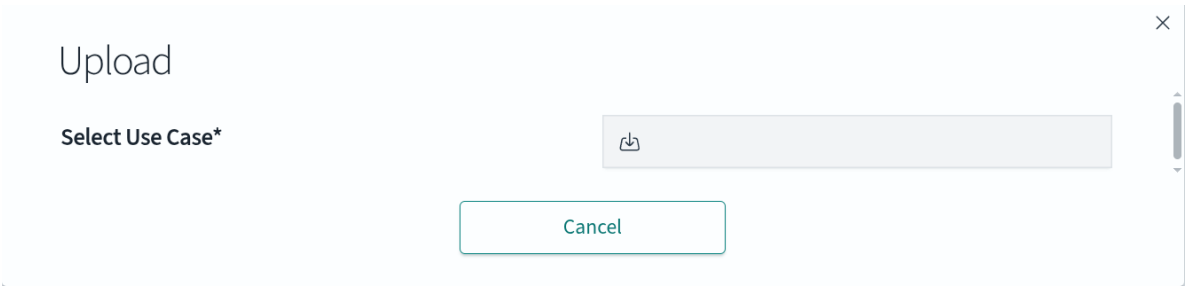
AI Store

The **AI Store** offers ready-made **AI Use Cases** that you can fetch and run on your own data. The store needs access to the Energy Logserver repository and the repository credentials from your license.

AI Use Case models can be accessed through the Energy Logserver webpage and the Energy Logserver app on the **Store** tab of **Empowered AI**.

To upload the selected model through webpage, follow the steps below:

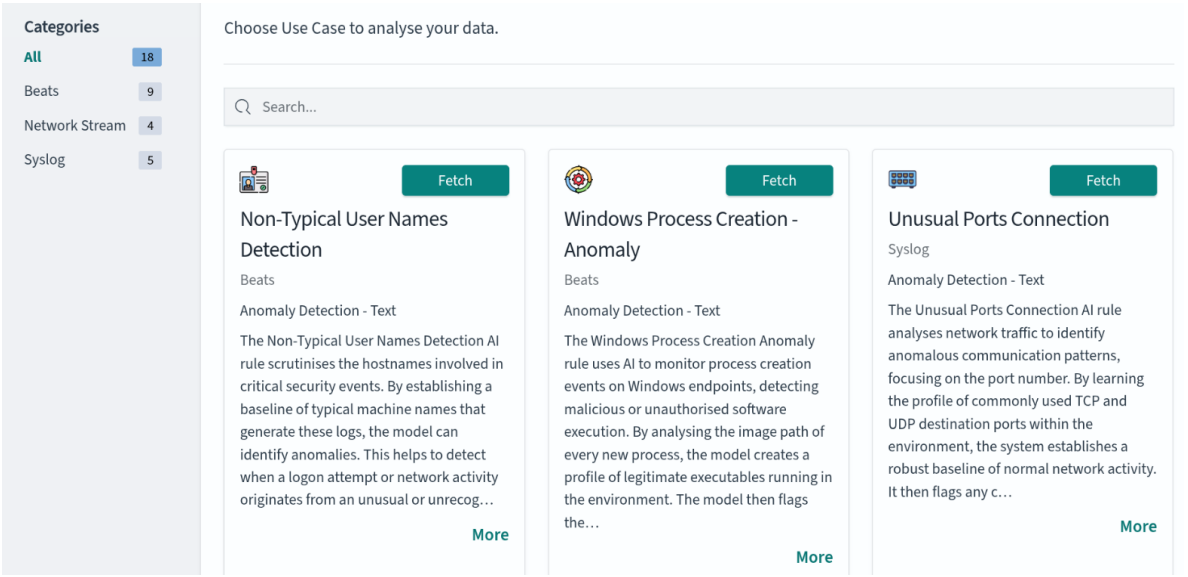
1. `Download` the model you are interested in from the webpage. The file has the `.use_case` extension.
2. Open the Energy Logserver app and navigate to the `Use Cases` tab.
3. Click `Upload` and select the downloaded `.use_case` file in the **Select Use Case** field.



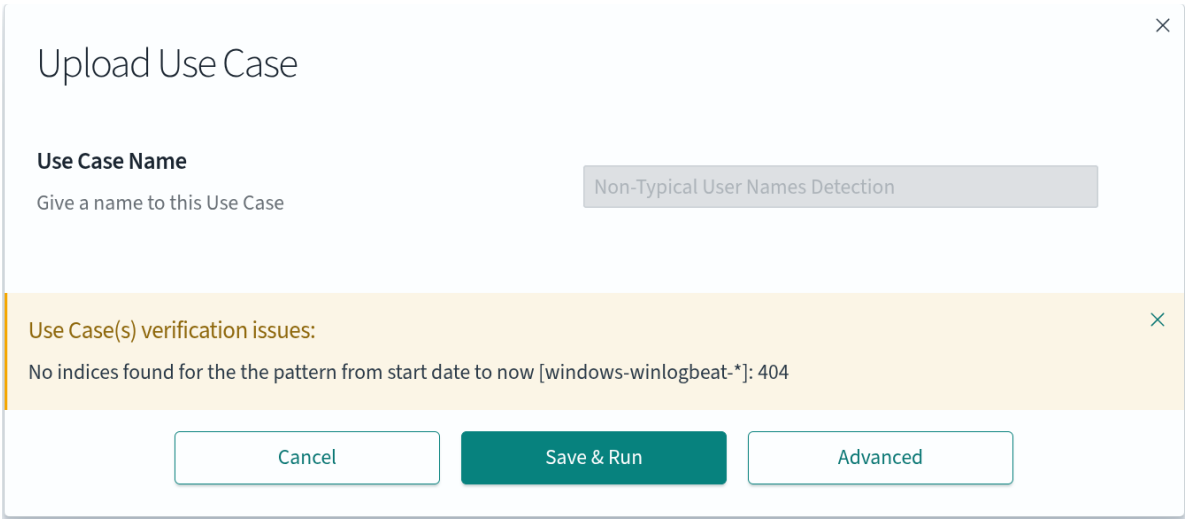
4. In the **Upload Use Case** window, press `Save & Run` to start the use case. **Advanced** opens the full use case form, where you can change the category, the data source, and the other settings before saving; the method and the name stay fixed.

To upload the selected model via the Energy Logserver app, follow the steps below:

1. Navigate to the `Store` tab. Each tile shows the use case name, category, method, and a short description; use the **Categories** list and the search field to narrow the list.



2. Press the `Fetch` button on the use case that you are interested in.
3. In the **Upload Use Case** window, check the **Use Case(s) verification issues** section, if present. For example, it reports that no indices match the index pattern of the use case.



4. Press `Save & Run` to start the use case, or **Advanced** to adjust the settings first.

AI Assistant

This section covers the integration of Energy Logserver with Large Language Models (LLMs). Configuration here manages the AI Assistant and the AI Agent, both available from the Discover tab. Each LLM provider requires a GPU-capable endpoint where semantic operations can run.

Energy Logserver strongly recommends using local AI resources or trusted providers. Remember, security is always local!

AI on Prem is a dedicated hardware appliance that can be deployed in your local environment.

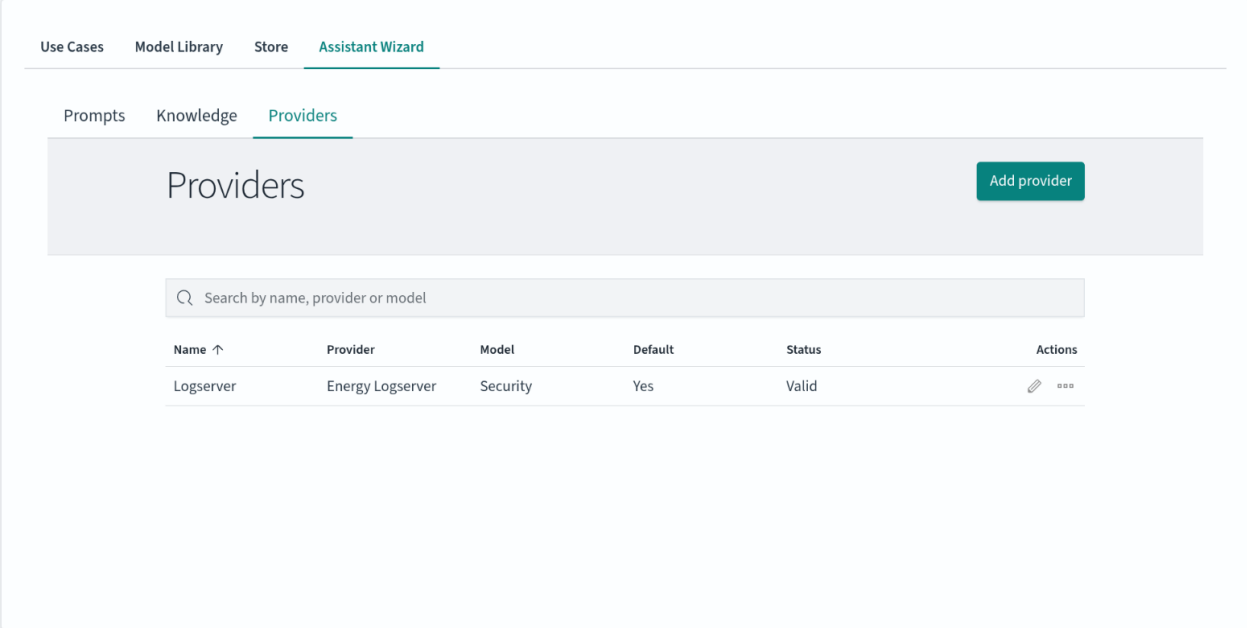
Everything is configured from **Empowered AI** → **Assistant Wizard**, which is split into three tabs: **Prompts**, **Knowledge**, and **Providers**. The sections below follow this order.

- *Providers*
  - *Configuring AI on Prem Connection*
- *Prompts*
- *Knowledge Chapters*
  - *Uploading a PDF Chapter*
  - *Creating a Chapter from Discover*
  - *How Vectorization Works*
- *AI Assistant in Discover*
- *AI Agent in Discover*
  - *Attaching Chapters*
  - *Sending Messages*
  - *When to Use Which*
- *Data Processing and Privacy*
  - *What Is Sent to the Provider*

- *What Stays in Your Cluster*
- *Providers and Security*
- *Availability*
- *Important*

Providers

Energy Logserver ships with a predefined LLM provider hosted in the Energy Logserver data center. This provider is available at no additional cost under an active support agreement, and the connection to it uses mutual TLS with the client certificate from your license. For fully local processing, ***AI on Prem*** is a hardware appliance ready to operate from your server rack. The following external providers are also supported: ***Ollama***, ***OpenAI***, ***Fireworks***, and ***HTTP*** endpoints. Configure each provider in the Providers section with its model name and either its URL or, for OpenAI and Fireworks, an API key.



Configuring AI on Prem Connection

AI on Prem is a local hardware solution that allows you to run Large Language Models within your infrastructure, ensuring complete data privacy and compliance. The following steps describe how to configure a connection to AI on Prem using the Energy Logserver provider.

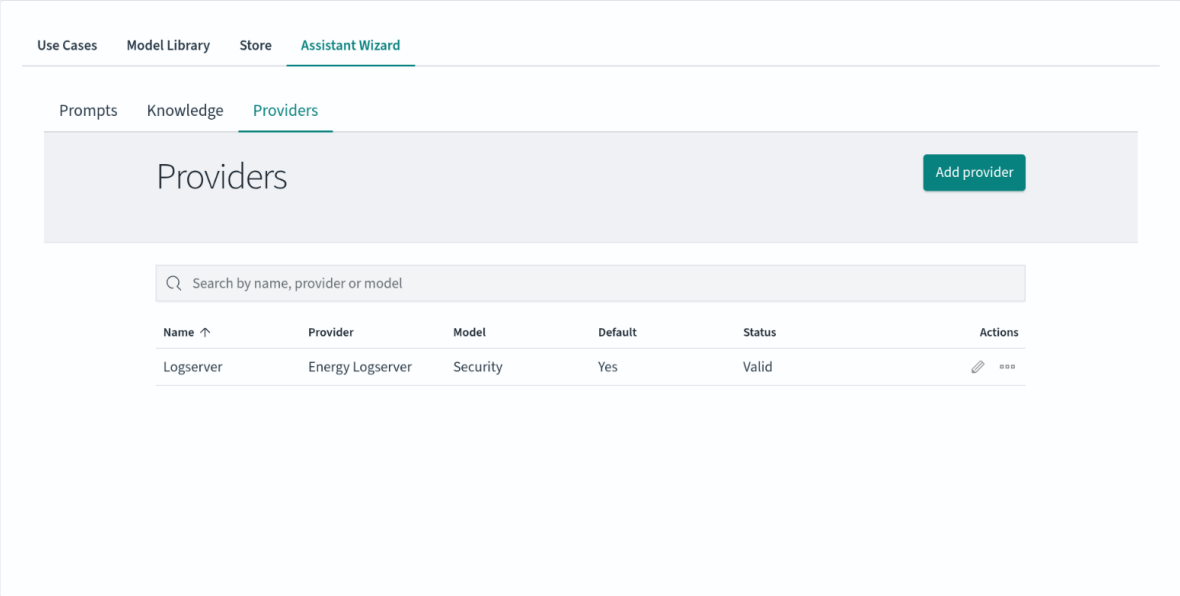
Prerequisites:

Before configuring the connection, ensure that:

- AI on Prem hardware is properly connected to your network
- The AI on Prem device has been assigned an IP address
- Network connectivity between the Energy Logserver GUI node and AI on Prem is allowed in your network firewall rules

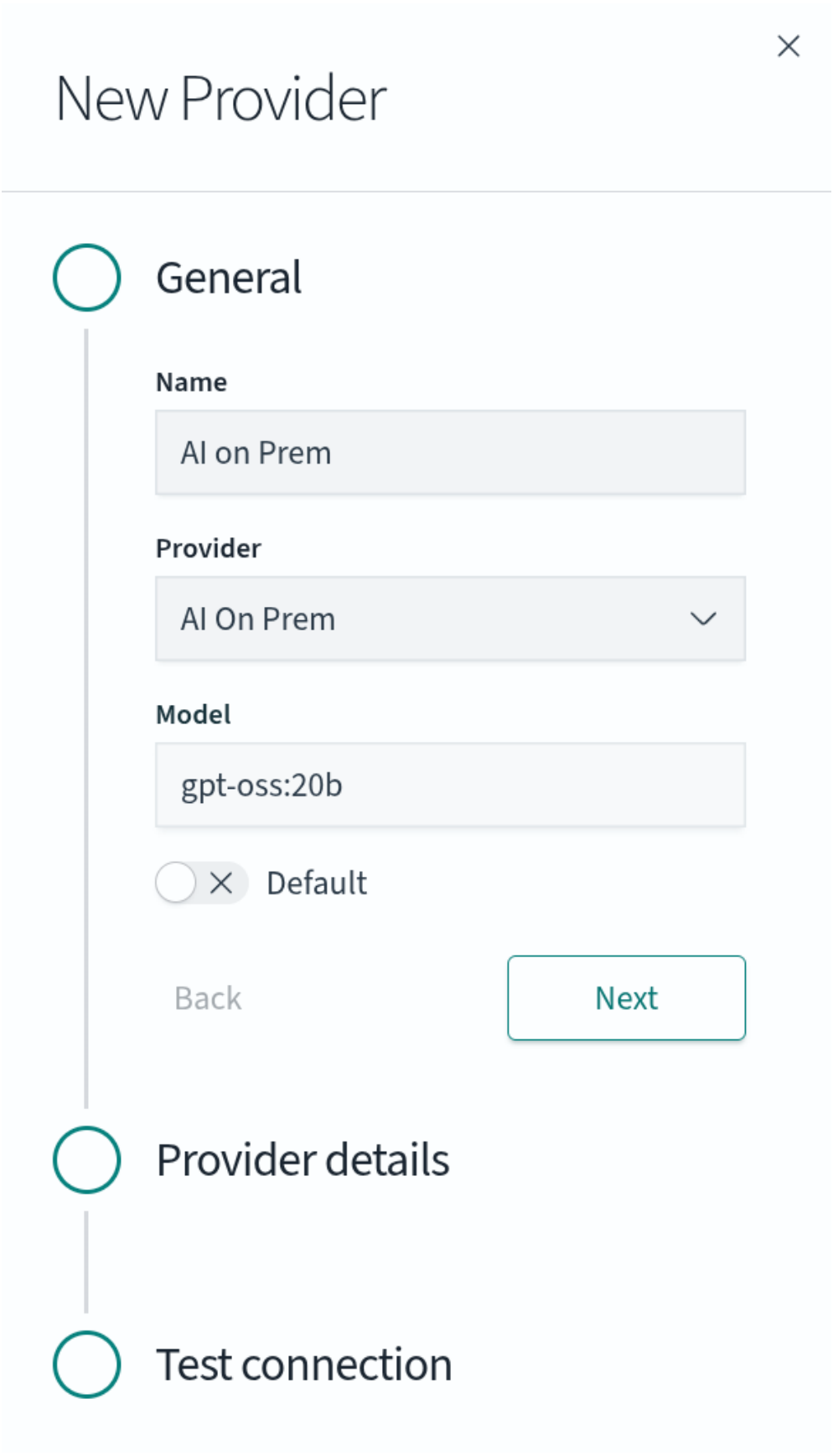
Configuration Steps:

1. Navigate to the **Empowered AI** module and select the **Assistant Wizard** tab:



2. Click **Add provider** button to create a new provider configuration. In the **General** section, configure the following:

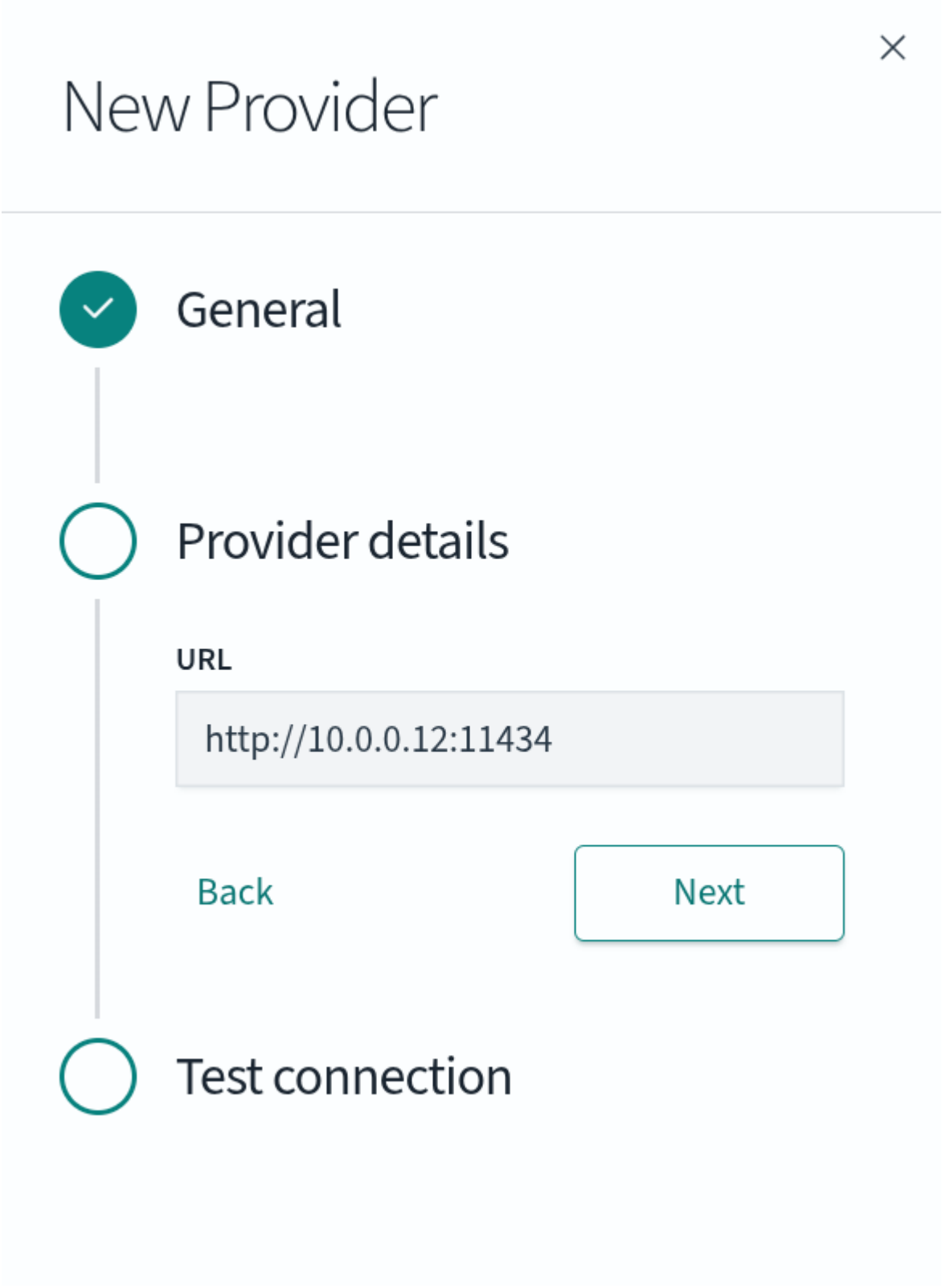
   - **Name**: Provide a descriptive name for your AI on Prem connection (e.g., "AI on Prem")
   - **Provider**: Select **AI On Prem** from the dropdown menu. Do not use **Ollama** for the appliance: the AI Agent uses only **AI On Prem** and **Energy Logserver** connections, so with an **Ollama** connection the Agent sends its requests to the Energy Logserver Provider instead. The dropdown also lists OpenAI, Ollama, HTTP, Energy Logserver, and Fireworks for other endpoints.
   - **Model**: Enter the name of the model loaded on the appliance (e.g., `gpt-oss:20b`)


New Provider
General
Name
AI on Prem
Provider
AI On Prem
Model
gpt-oss:20b
Default
Back
Next
Provider details
Test connection

Click **Next** to proceed to provider details.

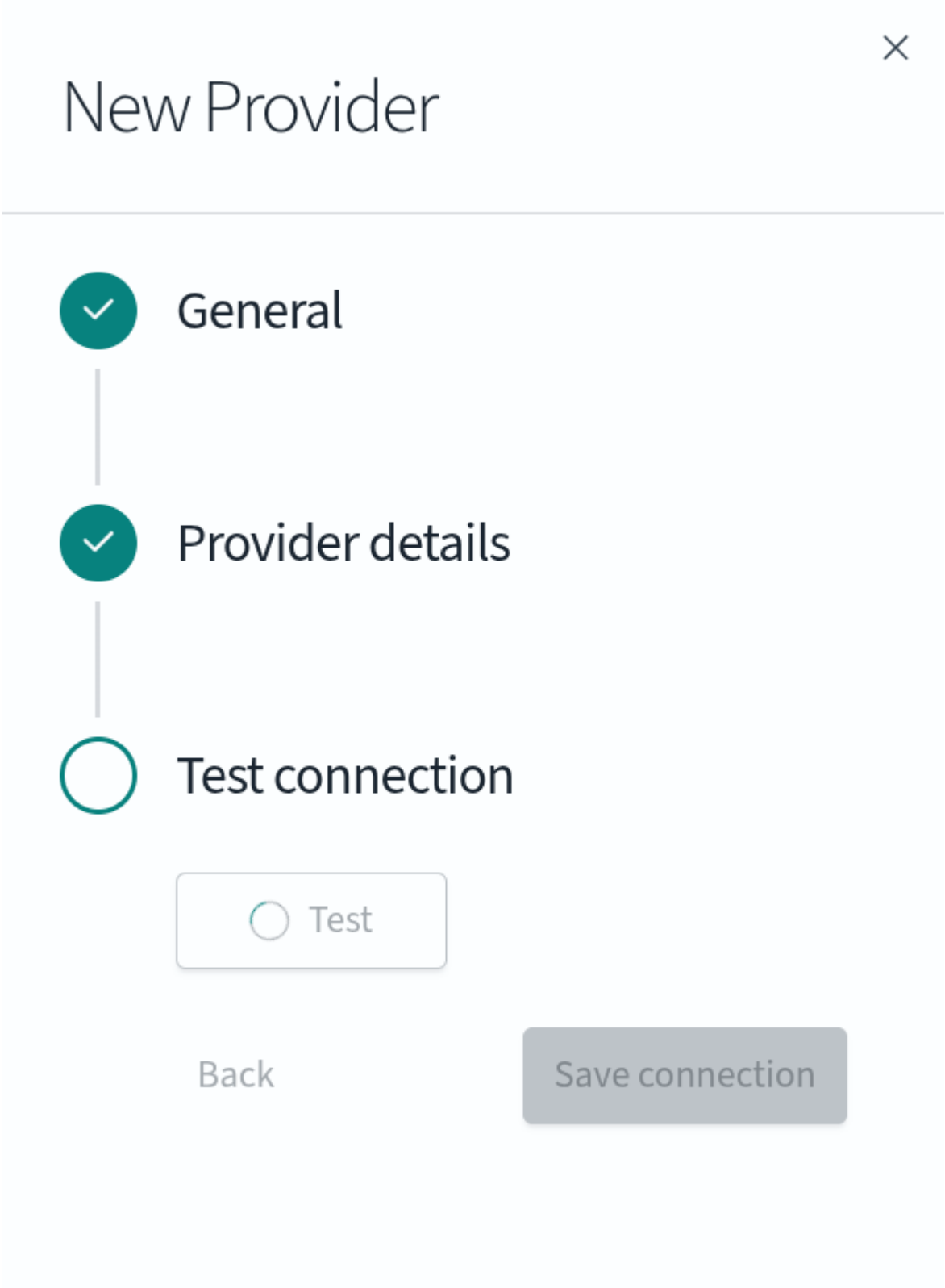
3. In the **Provider details** section, enter the connection information:

   - **URL**: Enter the address of the appliance, including the API port, for example `http://10.0.0.12:11434`. Port 11434 is the Ollama default.



Click **Next** to proceed to connection testing.

4. The **Test connection** step checks the connection to your AI on Prem device as soon as it opens; click **Test** to run the check again. **Save connection** becomes available only after a successful test.



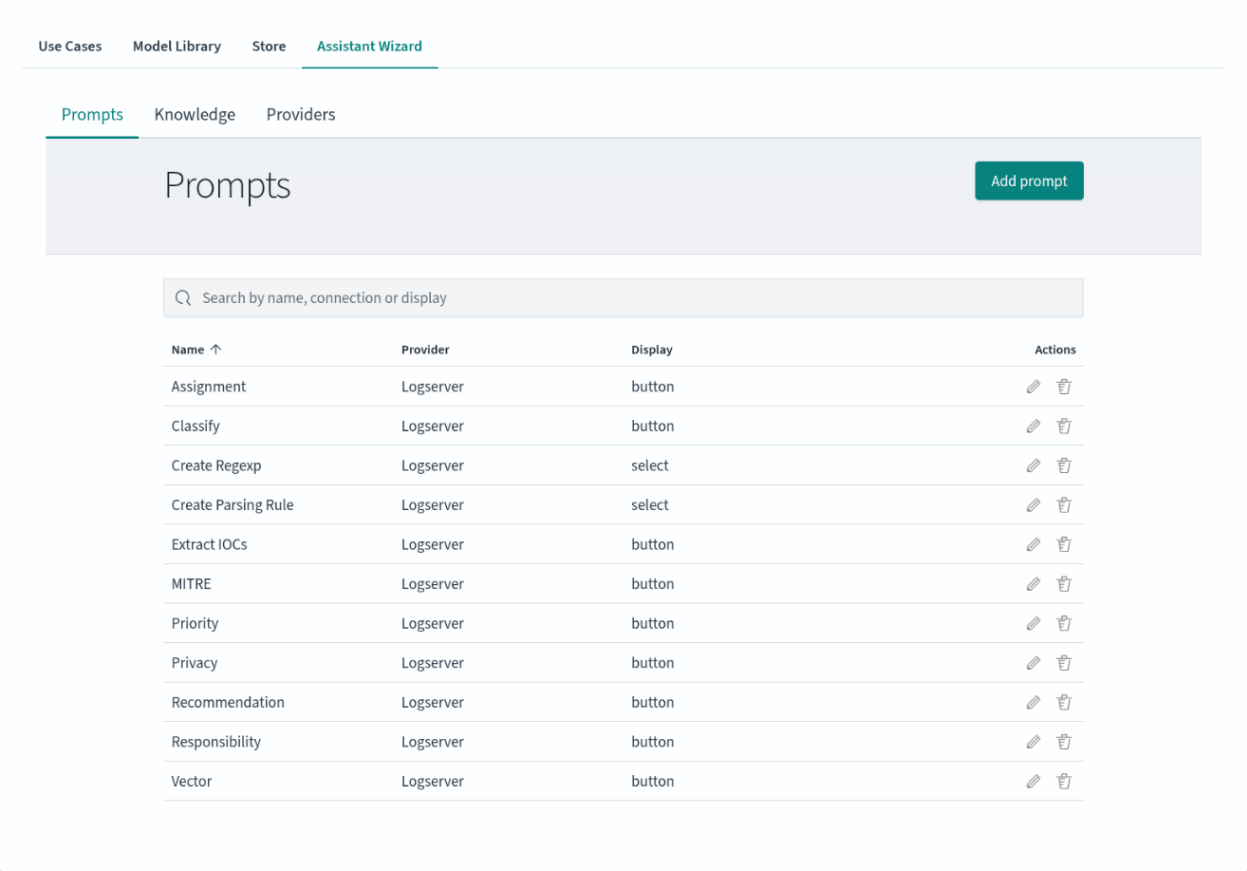
Troubleshooting:

If the connection test fails, or keeps running for minutes because the address does not answer, verify the following:

- The AI on Prem device is powered on and connected to the network
- The URL is correct and reachable from the Energy Logserver GUI node
- Network routing is properly configured between VLANs or network segments if applicable

Once configured, the AI on Prem provider will be available for use in AI Assistant prompts and can be selected as the default provider for your organization.

Prompts

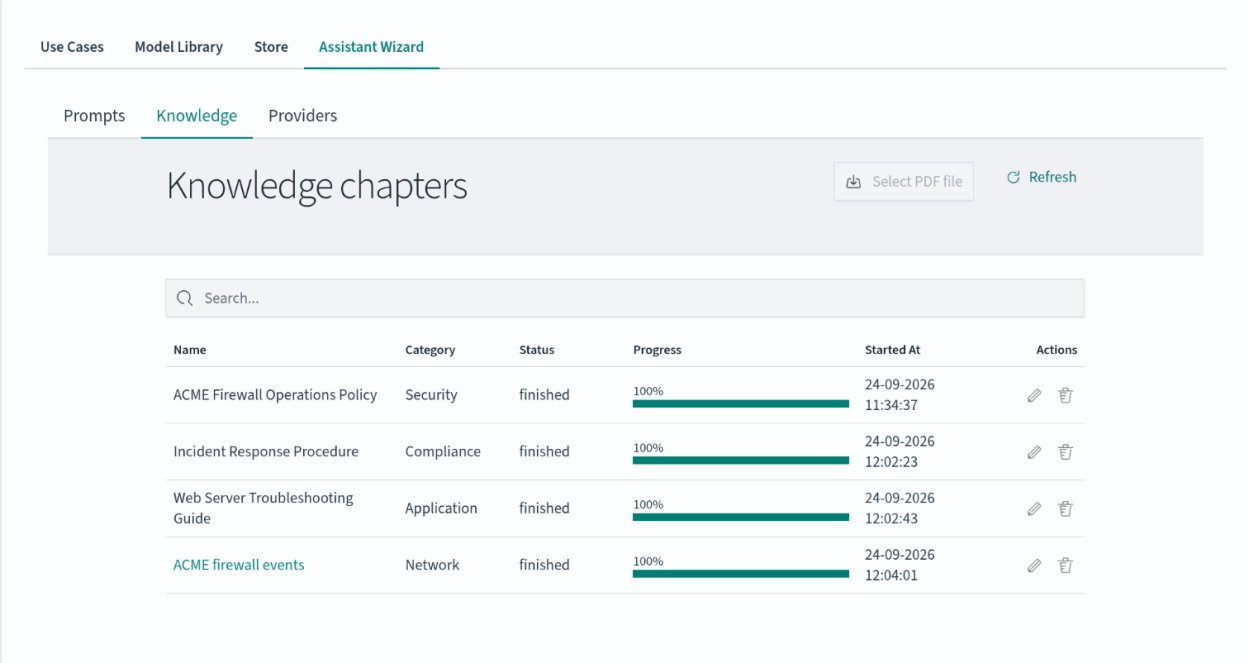


Energy Logserver allows you to create custom prompts for use in the AI Assistant. A prompt describes a problem to solve using a provided log entry, it can be any analytical task or technical inquiry that the model can address. The list above shows the prompts Energy Logserver ships with; they are described in *AI Assistant in Discover*.

Click **Add prompt** and fill in **Name**, **Version**, **Provider** (the connection the prompt uses), **System prompt**, and **Display as**: **Button**, **Select list**, or **Draft** to keep the prompt out of Discover for now.

Knowledge Chapters

Chapters are datasets that the AI Agent can read during a conversation. They turn arbitrary reference material, PDFs of internal policies, compliance documents, vendor manuals, or a slice of log data from Discover, into a searchable knowledge base for the model. Chapters are managed in the **Knowledge** tab of the Assistant Wizard.



The Knowledge table lists every chapter with:

Column	Description
`Name`	Chapter name. Names must be unique.
`Category`	Chapter category: Generic, Security, Application, Network, Compliance, or Performance.
`Status`	`new`, then `running` while vectorization is in progress, then `finished` when the chapter is ready to use, or `error` if vectorization fails.
`Progress`	Vectorization progress bar. Reaches 100% when the chapter becomes usable.
`Started At`	Timestamp of the chapter creation / upload.
`Actions`	**Edit** (name and category) and **Delete**. **Stop** is shown until the chapter reaches `finished`.

Uploading a PDF Chapter

To add a PDF as a chapter:

1. In the **Knowledge** tab click **Select PDF file** and pick a file from your workstation.
2. In the **Upload PDF file** window, set the **Name** (the file name by default) and the **Category** (Generic by default), then click **Upload**. If a chapter with the same name exists, the upload stops with *Chapter with this name already exists*.
3. Wait for the status to change to `finished`. The chapter is then available to attach in AI Agent conversations.

Use **Refresh** to update the status of in-progress uploads.

> **Note**
>
> The file is sent base64-encoded, which makes the request about a third larger than the PDF. With the default `server.maxPayloadBytes: 2097152` in `/etc/logserver-gui/logserver-gui.yml`, the largest PDF you can upload is about 1.5 MB; larger files fail with *Payload content length greater than maximum allowed: 2097152*. To accept larger files, raise this value and restart the `logserver-gui` service.

> **Warning**
>
> In 8.1.0 **Stop** ends the vectorization process, but the chapter stays in `running` status and the partial index remains. Delete the chapter instead and upload it again.

Creating a Chapter from Discover

Log data can be turned into a chapter directly from the Discover tab, so that a Discover query becomes a persistent knowledge source for the AI Agent. Click **Create chapter** in the Knowledge chat panel, enter a **Name** and pick a **Category**. The chapter captures the current index pattern, time range, query, and filters, and is indexed in the same way as a PDF. The time range is saved as fixed dates, so the chapter does not pick up newer events.

Chapters created this way appear in the Knowledge table alongside PDF uploads and can be attached to conversations in the same manner.

How Vectorization Works

When a chapter is uploaded or created, Energy Logserver runs a **vectorization job** in the background. The job splits the source material into segments and converts each segment into a numerical vector with a local embedding model, `sentence-transformers/all-MiniLM-L6-v2`, shipped in `/opt/intelligence/models`. Vectorization does not call the LLM provider, and the model is loaded only from the local files, never downloaded at run time. These vectors are stored in a dedicated index per chapter, named `.intelligence_knowledge_pdf_<file>_<timestamp>` for PDF chapters and `.intelligence_knowledge_index_<index-pattern>_<timestamp>` for chapters created from Discover.

> **Note**
>
> The splitting strategy differs between PDF and Discover chapters:
>
> - **PDF chapters** are split into overlapping text chunks (default 1300 characters with 300-character overlap, recursive character splitter). A single PDF therefore produces many segments, and the `(N)` count next to the chapter in the Attach chapters panel reflects the vectorized segment count.
> - **Discover chapters** do not run through a text splitter. Each matched document becomes exactly one segment, the `(N)` count equals the number of source documents. Records longer than the embedding model's input window (256 tokens for the default `all-MiniLM-L6-v2` model) are truncated at indexing time; semantic search over the trailing part of such records will not match.

During an AI Agent conversation, the model performs **semantic search** against the attached chapters: the user's question, translated to English, is vectorized with the same model, and the most relevant segments are pulled from the chapter indices and included in the model's context. A chapter that is still `running` can already be attached, but the Agent searches only the segments indexed so far; the `(N)` count grows during vectorization.

Progress and estimated execution time are visible in the Knowledge table. Document throughput is reported while the job runs.

AI Assistant in Discover

To analyze a single log entry, expand it in Discover and open the **Assistant** tab. The answer appears on the right.



Quick Actions are built in and cannot be changed:

- **Explain Log**: explains what the entry means.
- **Detect Threat**: assesses whether the entry points to a security threat.
- **Create Alert**: generates a query from the entry and opens **Create alert rule** with that query and the current index pattern. Enter the rule name and type, then save. The action needs the `alert` role and an index pattern with a time field.

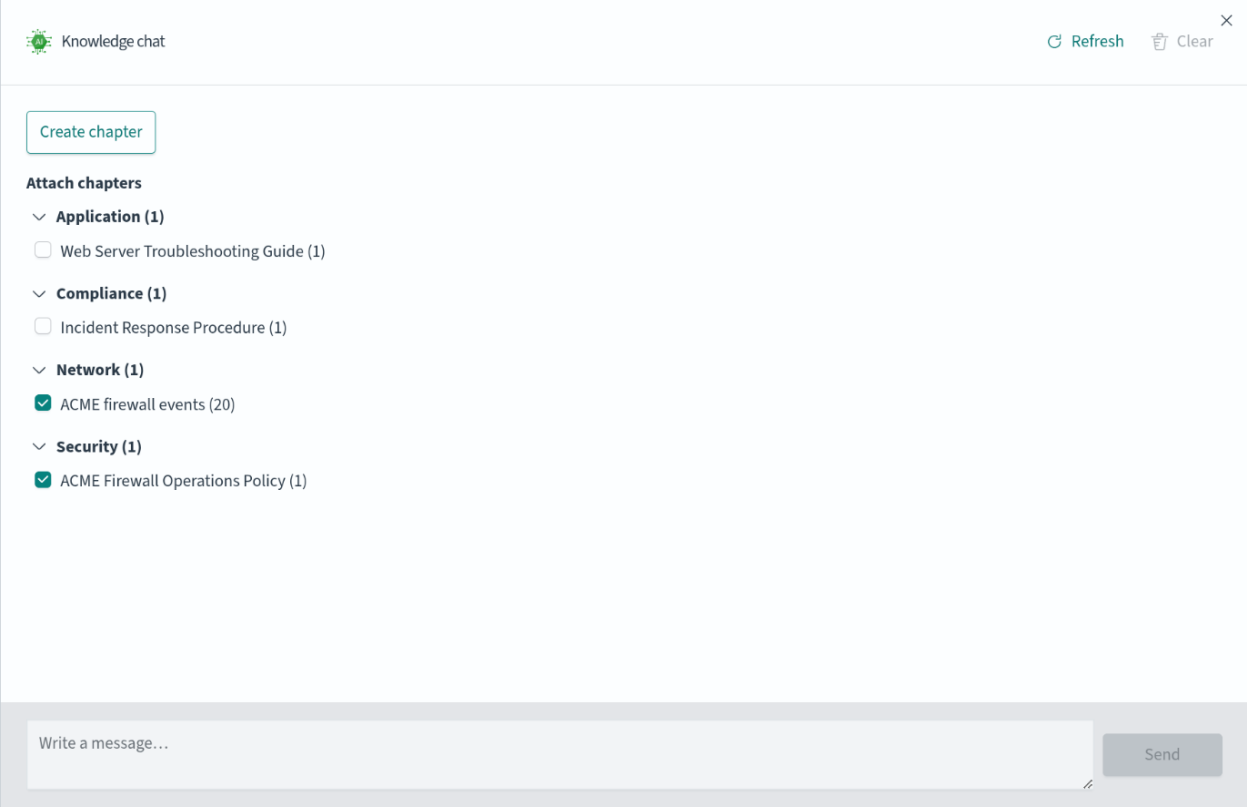
Prepared Prompts come from the **Prompts** tab of the Assistant Wizard: prompts set as buttons appear as buttons, the others in the **Select a prompt** list with **Run**. Energy Logserver ships with these prompts:

Prompt	What it does
Assignment	Routes the entry to the IT team that should handle it
Classify	Classifies the event
Extract IOCs	Extracts indicators of compromise
MITRE	Maps the event to MITRE ATT&CK
Priority	Assigns a severity
Privacy	Looks for personal or sensitive data
Recommendation	Recommends what to do next
Responsibility	Names the responsible team
Vector	Places the event in the Lockheed Martin Kill Chain and the Diamond Model
Create Parsing Rule (list)	Writes a Network Probe parsing rule for the `message` field
Create Regexp (list)	Writes a regular expression for the `message` field

AI Agent in Discover

AI Assistant runs a single prompt against a single log entry. The **AI Agent** is the conversational counterpart: an open chat that can reason about large datasets using selected Knowledge Chapters as context. The Agent does not remember earlier messages: each question is answered on its own, so write every question in full.

Open the AI Agent by clicking the **Assistant** button in the Discover toolbar. A **Knowledge chat** panel appears on the right side of the screen.



Attaching Chapters

The top of the Knowledge chat panel lists **Attach chapters**, grouped by category. Each group shows the number of chapters in it; chapters without a category are listed under **Other**. Expand a group to see its chapters, each with a checkbox followed by a number in parentheses. That number shows how much material the chapter holds, for PDF chapters, the count of vectorized segments produced from the file; for chapters built from Discover, the number of source documents captured when the chapter was created. A value of `0` means that no segment has been indexed yet, so the Agent has nothing to search in that chapter.

Select one or more chapters before sending a message, the Agent will consult all selected chapters during the conversation and combine them into its answers.

This is the recommended pattern when the same question needs both formal context (a PDF policy document, a compliance requirement) and live context from the cluster (a chapter created from a Discover query over recent logs).

Create chapter at the top opens the chapter creation flow directly from the chat, without leaving Discover. New chapters appear in Knowledge and become attachable once vectorization finishes.

Sending Messages

Type the question in the **Write a message…** field at the bottom of the panel and click **Send**. The Agent performs semantic search across the attached chapters, retrieves the relevant segments and uses them to compose an answer.

> **Note**
>
> The Agent does not read whole chapters. With up to 6 chapters attached, it passes the model at most 8 of the segments most relevant to the question, taken from all chapters together. With more chapters, it summarizes up to 3 segments from each chapter in a separate request and answers from those summaries. Segments that match the

question poorly are skipped. Questions that need every record, such as counts or "is there any event that…", can get incomplete answers. Check such answers in Discover.

In the panel header, **Refresh** reloads the list of chapters and **Clear** resets the conversation while keeping the attached chapters. Closing the panel (**X**) also keeps the selection of attached chapters.

When to Use Which

Feature	Best for
AI Assistant	A single action on a single log entry, explain, classify, extract IOCs, escalate. Uses predefined prompts, returns immediately.
AI Agent	Questions about policies, manuals, or a set of records captured in a chapter. Uses attached chapters for grounded answers.

Both features use the connections from the Providers tab: AI Assistant prompts use the connection assigned to the prompt or the default one, the built-in **Explain Log**, **Detect Threat**, and **Create Alert** always use the default one, and the AI Agent uses the AI on Prem connection if one exists, otherwise the Energy Logserver Provider. The AI Agent does not use other providers and does not work without one of these two.

Data Processing and Privacy

AI Assistant and the AI Agent send text to the LLM provider. Vectors are used only to find the relevant material in your cluster; the answer is always generated by the provider from text.

What Is Sent to the Provider

Feature	Sent to the provider
AI Assistant (a prompt or a built-in action run on a log entry in Discover)	The whole log document (`_source`) as JSON and the text of the prompt. **Create Alert** also sends the field mapping of the log's index.
Integration creator (**AI Assistant** tile in Integrations)	The sample logs selected in the creator, at most 10 and up to 16,000 characters each. The Dashboards step also sends up to 5 documents from the new index with statistics and example values of each field, and the Alerts step sends one document from the new index with its field mapping. See *Creating Integrations with the AI Assistant*.
AI Agent (Knowledge chat)	The question, first to translate it to English, then with the answer request, together with the text of the selected segments of the attached chapters (see the note in *Sending Messages*). With more than 6 chapters attached, one extra request per chapter carries the question and up to 3 of its segments.

Requests to the Energy Logserver Provider also carry the customer name from the license and the product version in the `User-Agent` header.

What Stays in Your Cluster

- Chapters are vectorized locally with `all-MiniLM-L6-v2`; the vectors are stored in the `.intelligence_knowledge_*` indices together with a copy of the source text.
- Each AI Agent question is written to `/opt/intelligence/logs/intelligence.log` with the user name: up to 80 characters of the question and up to 60 characters of its English translation. The log rotates daily.

Providers and Security

Choose the provider according to where the log content may go:

- **AI on Prem** runs on your own hardware, so the text sent to the model stays in your network.
- The **Energy Logserver Provider** runs in the Energy Logserver data center. Energy Logserver connects to it over mutual TLS with the client certificate delivered with your license.
- **External public providers** such as OpenAI and Fireworks receive the text under their own terms. They are **not recommended** for production environments handling sensitive security data. Use them for testing and evaluation only.

Availability

AI Assistant is available at no additional cost for customers with an active support agreement. The Assistant tab in Discover, the Knowledge chat, and the Assistant Wizard require the **Assistant** module in the license; without it, Energy Logserver shows a message that the feature is only available if your license includes the "Assistant" module. AI on Prem is a separate hardware offering, contact your Energy Logserver representative for details. Integration with public providers is handled directly between you and the provider.

Important

AI Assistant is a supporting tool. Results from the LLM should be treated as one input and always verified by a security analyst before making operational decisions.

Troubleshooting

Q: What data can I use in saved searches? A: Various types of data can be used, but different analysis models expect specific data types. For example, Text Anomaly detection requires textual values.

Q: Can I edit saved searches? A: Yes, but be cautious as changes affect all rules using the same saved search.

Q: How often should I run the rule? A: The frequency depends on the nature of your data and how often you need to monitor it. For continuous monitoring, consider setting a scheduled analysis at regular intervals.

Q: How many clusters should I choose? A: The number of clusters depends on the nature of your data and the objectives of your analysis. It is recommended to test different values to find the optimal number of clusters for your use case.

Q: What is the purpose of the Data Aggregation setting? A: The Data Aggregation setting defines how data will be grouped over time. It determines the interval at which data points are aggregated.

Q: What types of texts in the Text Anomaly use case can be analyzed? A: The system is capable of analyzing various types of textual data, including log messages, emails, reports, and other textual documents.

Q: Does the system support multiple languages? A: Yes, the text analysis system supports multiple languages, making it a versatile tool on an international scale.

Where to look first

The Empowered AI engine runs as two services and writes its own logs:

```
systemctl status intelligence intelligence-scheduler
tail -f /opt/intelligence/logs/*.log
```

The log level is set in `/opt/intelligence/configuration/engine.yml` (`logging.level`), together with the connection details the engine uses to reach the Data Node. The engine accepts `DEBUG`, `INFO`, `WARNING`,

ERROR, and CRITICAL; the shipped value WARN is not one of them, so the engine logs at INFO. Set DEBUG there when a rule fails without an explanation in the GUI.

Troubleshooting Common Issues

- **Problem: I can't find saved searches in the Empowered AI module.**
  - Ensure the search is saved in the Discover module and you have the appropriate permissions.
- **Problem: The rule does not analyze data as expected.**
  - Check the rule configuration, time range, and search criteria.
- **Problem: The rule does not detect anomalies as expected.**
  - Check the contamination factor and data aggregation settings. Adjust these parameters to better fit your data characteristics.
- **Problem: Too many anomalies are detected.**
  - Fine-tune the contamination factor to reduce the number of false positives.
- **Problem: No anomalies are detected despite unusual data patterns.**
  - Adjust the contamination factor or data aggregation settings to refine the model's sensitivity.
- **Problem: Charts do not display data correctly.**
  - Ensure the data source and configuration settings are correct. Refresh the page or rerun the analysis if necessary.
- **Problem: Anomaly results appear incorrect.**
  - Review the contamination factor and other model settings to ensure they are appropriately configured for your data.
- **Problem: Clustering does not work as expected.**
  - Check the rule configuration and ensure that all fields are correctly set and that the data is appropriate for clustering.
- **Problem: The Cluster Quality Chart does not display correctly.**
  - Ensure that the data is appropriately prepared and that the clustering rule is correctly configured. Check for errors in the data source or clustering parameters.
- **Problem: I cannot view clusters in the Discover module.**
  - Ensure you have the necessary permissions to access the Discover module and that the data is correctly indexed.
- **Problem: The system does not analyze the exact number of logs specified.**
  - Ensure that the number of logs is correctly set and that the available data meets the analysis criteria.
- **Problem: Delays in text analysis.**
  - Verify system resources and configuration to increase processing efficiency.
- **Problem: Incorrect mapping for indices when starting a rule.**
  - When you start a use case, the **Are you sure you want to start Use Case(s)?** window lists `Incorrect mapping for indices [<index pattern>]` under **Use Case(s) verification issues:**, and the use case form shows **Warning: incorrect mapping**.

  - This indicates that the new mapping required for analysis has not yet been applied. The new mapping will be applied when a new index is created, which occurs on the next day for daily indices (e.g., index_name-2024.03.03) or on the next month for monthly indices (e.g., index_name-2024.03). You can also move the Start Date forward to exclude the older indices, as the warning suggests.

- **Problem: The rule shows `Error`, `Build Error`, or `Score Error` after the run or the results of the analysis are unsatisfactory.**
  - This probably means that it is necessary to adjust the learning parameters. We recommend the following steps:

  1. Press the `Edit` button
  2. Set new `Build Time Frame` to a period where the data occurs (7 days recommended)
  3. Set the new `Start Date` as the start date of the analysis (we recommend setting this parameter to the day when the `Build Time Frame` starts).
  4. Click `Save & Run` to start the process.

4.11 Reporting

The Reports module provides tools for exporting data and generating graphical reports. It consists of three tabs: **Data Export**, **Report Export**, and **Settings**.

4.11.1 Data Export

The **Data Export** tab allows you to export raw data from indices to CSV or HTML files. To access it, click the **Reports** icon in the top menu and go to the **Data Export** tab.

The tab contains two sub-tabs: **Create Task** and **Task List**.

Create Task

The form is divided into the following sections:

General

Enter a task name in the **Task name** field and select the export type: **CSV** or **HTML**. The name accepts letters, digits and spaces, up to 32 characters.

Data

Select the data source for the export. You can use a saved search to quickly access a predefined query and index pattern, or select an index pattern directly from the **Index pattern** dropdown. Use the **Query** field to filter the data that will be exported.

Time range

Choose the date range for the export. Enable the time range toggle and select the period using the date picker. The **Time field** defaults to `@timestamp`. If neither the time field nor the index pattern is selected, `@timestamp` will be used by default. To export all available data without a time filter, disable the toggle.

Fields

By default all fields are exported. Use the **Retrieved fields** selector to limit the export to specific fields. Additional options:

- **Include unmapped fields**: includes fields that do not appear in filters or dashboards. Enabling this option may significantly increase export duration.
- **Include meta fields**: includes metadata fields. Disabled by default.

Limits

- **Max document count**: maximum number of documents to export. Default: 10,000, maximum: 100,000. Set to 0 to disable the limit.
- **Fast mode**: skips intermediate file creation for faster processing. Recommended for large exports.

Scheduling

Exported data can be sent automatically via email to configured recipients. To enable scheduling, click the **Disabled** toggle, it will switch to **Enabled** and reveal the following fields:

Field	Purpose
Email topic	Subject line of the notification email.
Email addresses	One or more recipient addresses. Invalid entries are rejected.
Select role	Role assigned to the scheduled task. See *Role selection for scheduled tasks*.
Schedule type	`Daily`, `Weekly`, `Monthly`, or `Cron Tab Format`.
Cron format	Custom cron expression, shown only when `Cron Tab Format` is selected.

Note

Scheduling requires a one-time setup that creates a trusted token for the scheduler service. When scheduling is not yet enabled, the Data Export tab shows the message *"Scheduling is disabled right now. Please request your Admin to enable this feature."* and, for admin users, an **Enable Scheduling** button that performs the setup. Non-admin users need an administrator to click it.

Scheduled tasks also appear in the *Scheduler* with the plugin **Reports**.

After configuring all options, click **Submit** to create the task.

Role selection for scheduled tasks

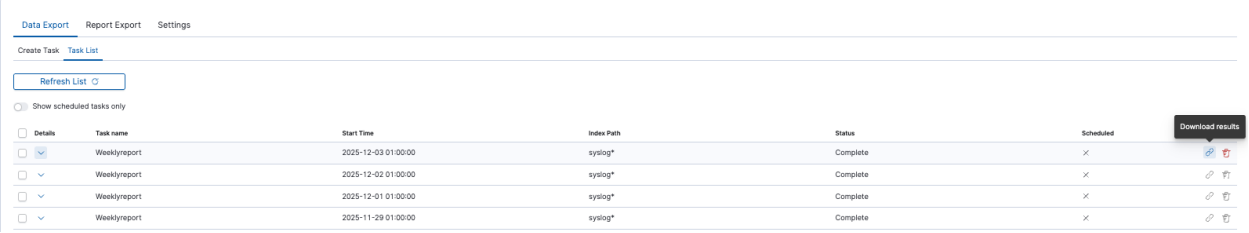
The **Select role** field controls two independent checks:

- **Task visibility.** Non-admin users only see tasks whose assigned role matches one of their own roles; admins see all tasks.
- **Dashboard and data access.** The assigned role must have access to the dashboard (for Report Export) or to the index pattern and fields (for Data Export). A role that cannot open the dashboard in the GUI cannot be used for a recurring report against it.
- **Data Export downloads.** For Data Export specifically, a user whose roles do not overlap with the role assigned to the task will not be able to download the generated file from the Task List.

Task List

The **Task List** sub-tab displays all created export tasks with their current status. When the status changes to **Complete**, you can download the file using the **Download** button or remove it with the **Delete** button.

A task reports one of these statuses: `Pending`, `Enabled`, `Disabled`, `Running`, `Complete`, `Aborted`, `No Data`, `Expired`, `Error`, `Logging Error`, `PDF Creation Failed`, `DOCX Creation Failed`, `Image Creation Failed`. `No Data` means the query matched nothing, and `Expired` means the retention period from the Settings tab has passed and the file was deleted.



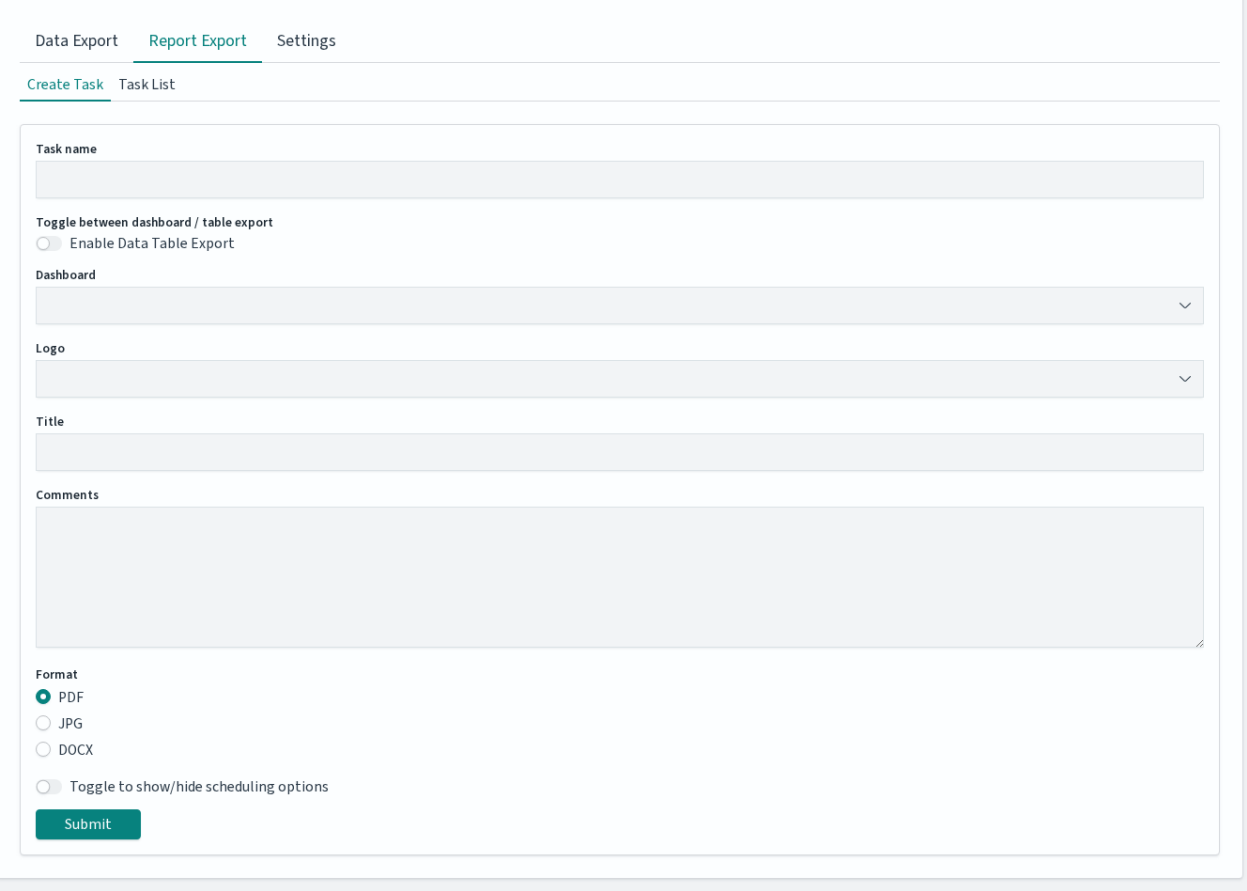
The refresh symbol  updates the task list status.

Quick Report from Discovery

You can initiate a data export directly from the **Discover** tab. After configuring your index pattern, query, fields, and time range in Discover, use the **Data Export Wizard** button to open the Data Export form pre-filled with your current Discover settings. This avoids manually re-entering the same parameters in the Reports module.

4.11.2 Report Export

The **Report Export** tab allows you to generate graphical reports from dashboards in PDF, JPG, or DOCX format.



To create a report, go to the **Report Export** tab and fill in the **Create Task** form:

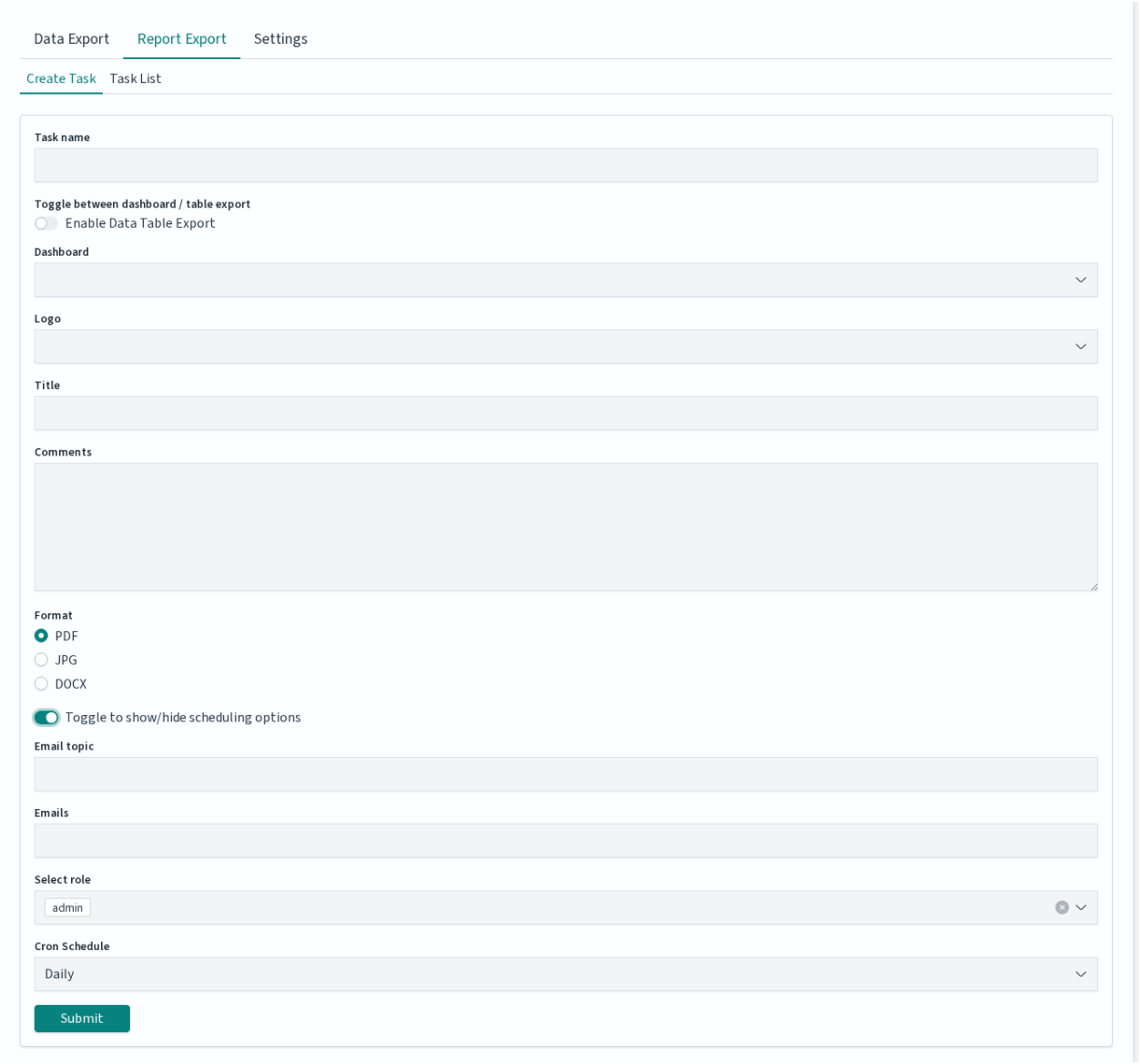
1. Enter a task name in the **Task name** field.
2. To export a table visualization instead of a dashboard, enable the **Enable Data Table Export** toggle.

3. Select the dashboard from the **Dashboard** dropdown.
4. Optionally select a logo from the **Logo** list.
5. Optionally add a report title in the **Title** field.
6. Optionally add comments in the **Comments** field.
7. Select the output format: **PDF**, **JPG**, or **DOCX**.
8. To schedule automatic report delivery via email, enable the **Toggle to show/hide scheduling options** toggle.
9. Click **Submit** to start generating the report.

You can monitor the progress in the **Task List** sub-tab. When the status changes to **Complete**, download the report via the **Download** button.

Scheduling a Report Export

Enable the **Toggle to show/hide scheduling options** to reveal the scheduling fields:



Field	Purpose
Email topic	Subject line of the notification email.
Emails	Recipient addresses for the generated report.
Select role	Role assigned to the scheduled task. Same semantics as in Data Export, see *Role selection for scheduled tasks*.
Cron Schedule	`Daily`, `Weekly`, `Monthly`, or a custom cron expression.

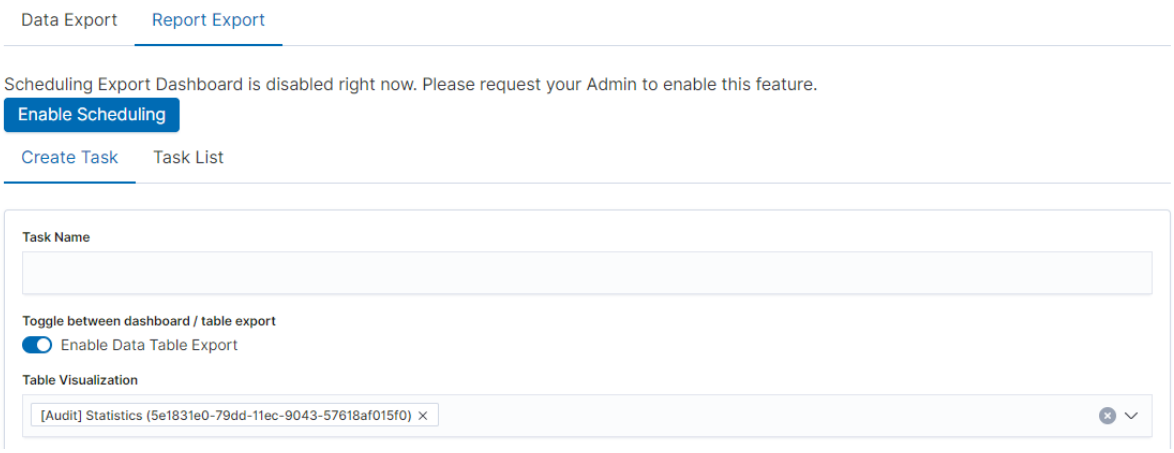
Below is an example PDF report generated from a dashboard template:



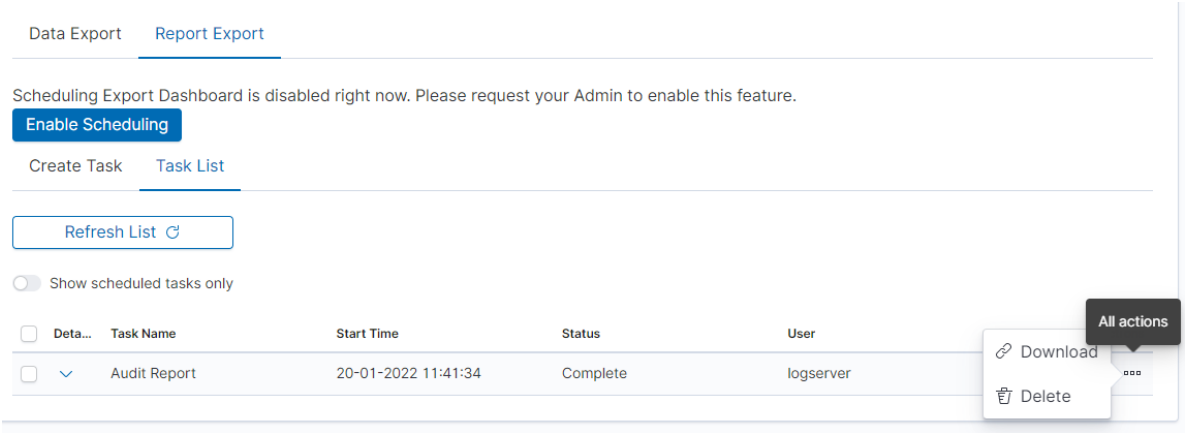
PDF report from table visualization

Data from a table visualization can be exported as a PDF or JPG report:

1. Go to the **Report Export** tab.
2. Enter a task name in the **Task Name** field.
3. Enable the **Enable Data Table Export** toggle:

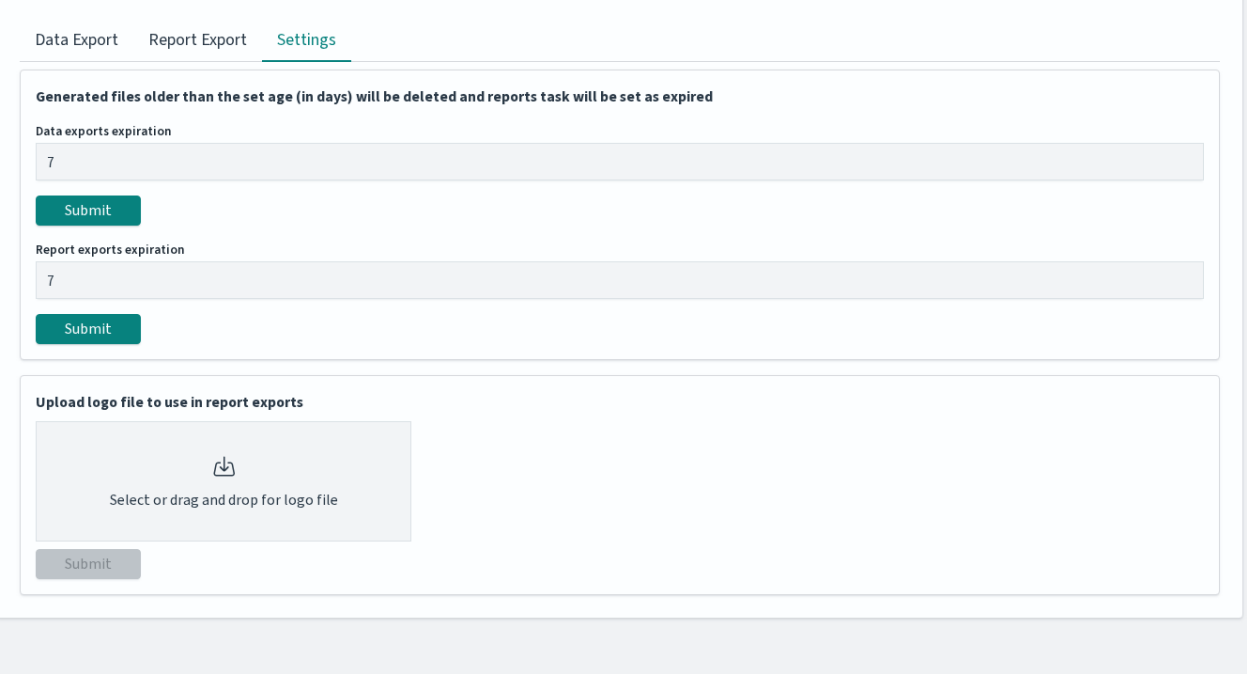


4. Select the table from the **Table Visualization** list.
5. Select the time range for the report.
6. Optionally select a logo, title, and comments.
7. Click **Submit** to start the task.
8. After the task completes, download the PDF via **Action** → **Download**:



4.11.3 Settings

The **Settings** tab allows you to configure retention periods for generated files and upload a logo for use in report exports.



- **Data exports expiration**: number of days after which generated data export files are deleted and tasks are marked as expired. Default: 30 days.
- **Report exports expiration**: number of days after which generated report files are deleted and tasks are marked as expired. Default: 30 days.
- **Upload logo file**: upload a logo image (drag and drop or select file) to use in Report Export tasks.

Confirm each setting with the **Submit** button.

4.12 CMDB

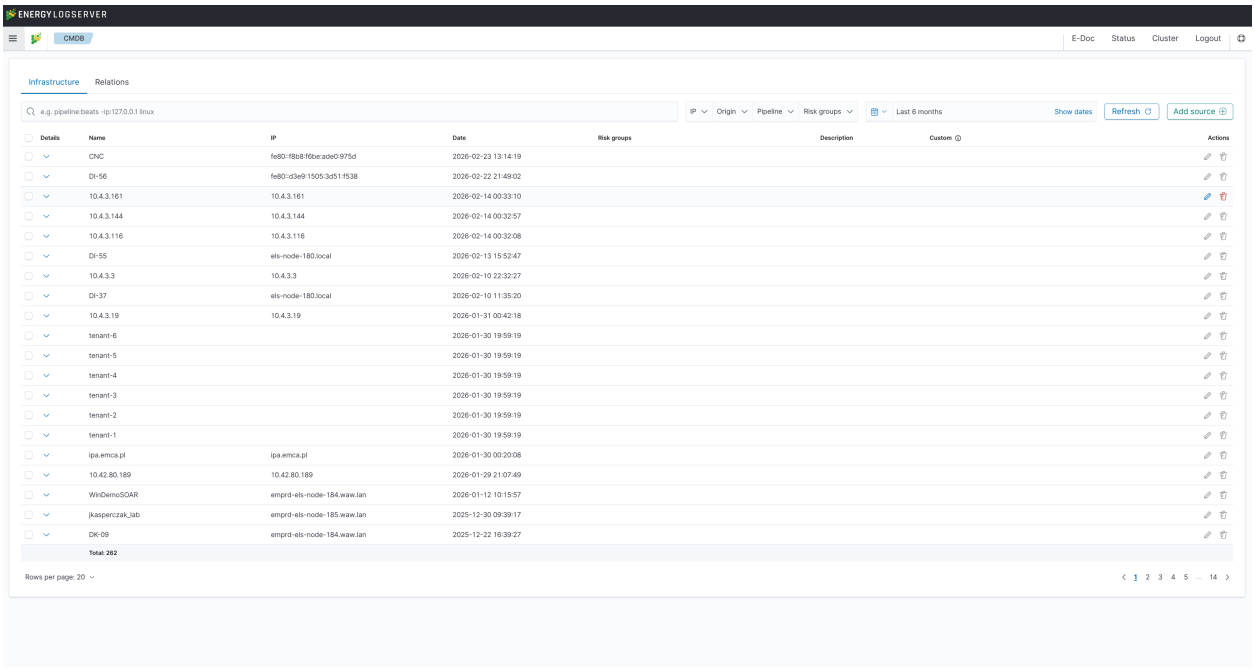
The CMDB (Configuration Management Database) module provides centralized asset inventory management and infrastructure relationship mapping. It is accessible from the sidebar under **Tools** → **CMDB**.

The module contains two tabs: **Infrastructure** and **Relations**.

> **Note**
>
> CMDB is available to the `admin` role only. A user with any other role who opens the module sees *"This page is accessible for administrator users only."* instead of the tabs. The same restriction applies to Agents, Archive, Integrations, Network Probe, CRUD, XLSX Import, Index Management, Tasks, and Scheduler.

4.12.1 Infrastructure

The Infrastructure tab displays all registered data sources in a paginated table. Use this view to maintain an inventory of hosts, agents, and other assets sending data to the system.



The table contains the following columns:

Column	Description
Details	Expand the row to view additional attributes (pipeline, origin)
Name	Asset name or hostname
IP	IP address of the source
Date	Timestamp of the last recorded activity
Risk groups	Assigned risk classification groups
Description	Free-text description of the asset
Actions	Edit or delete the entry

The toolbar above the table provides the following filters:

- **Search bar**: free-text search (e.g., `pipeline:beats -ip:127.0.0.1 linux`)
- **IP**: filter by IP address
- **Origin**: filter by data origin
- **Pipeline**: filter by processing pipeline
- **Risk groups**: filter by assigned risk group

- **Date range**: restrict results to a specific time window (with **Show dates** toggle)

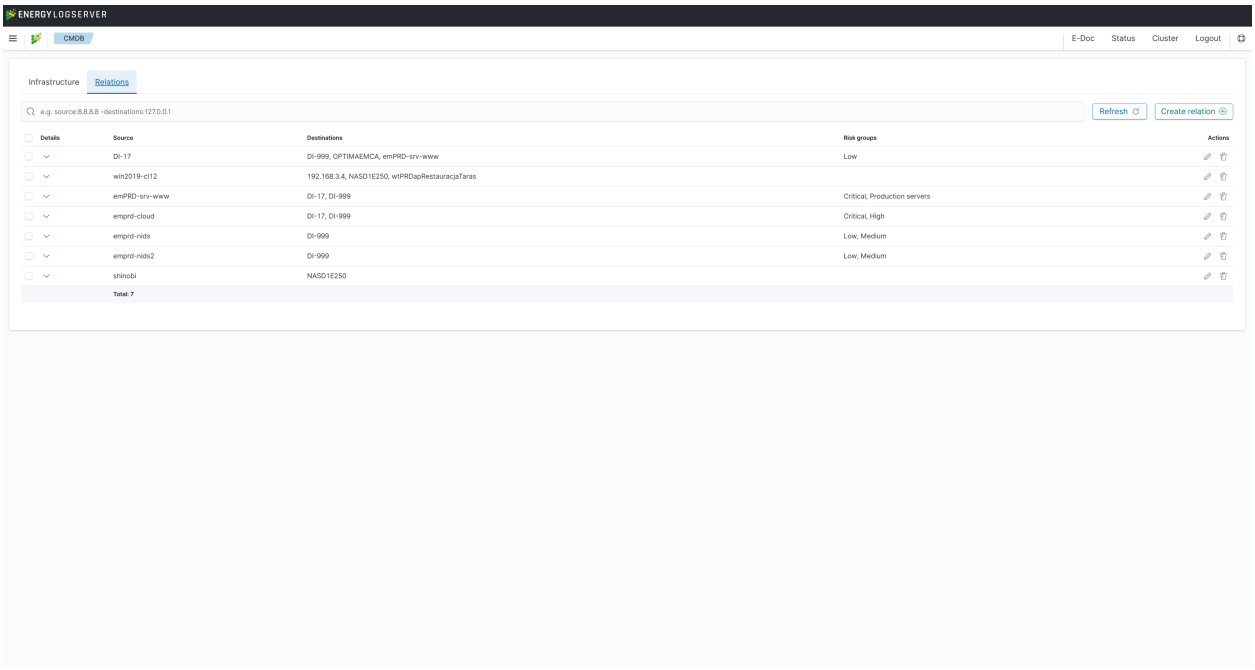
Available actions:

- **Refresh**: reload the table with current data
- **Add source**: manually register a new infrastructure asset by providing its name, IP, pipeline, origin, description, and risk groups

Each row provides **Edit** (pencil icon) and **Delete** (trash icon) actions in the Actions column.

4.12.2 Relations

The Relations tab maps connections between infrastructure assets. Each relation defines a source, one or more destinations, and an associated risk group.



The table contains the following columns:

Column	Description
Details	Expand the row to view additional attributes
Source	The originating asset in the relation
Destinations	One or more target assets connected to the source
Risk groups	Risk classification assigned to the relation
Actions	Edit or delete the relation

Use the search bar to filter relations by source or destination (e.g., `source:8.8.8.8 -destinations:127.0.0.1`).

Available actions:

- **Refresh**: reload the relations table
- **Create relation**: define a new relation by selecting a source, one or more destinations, and a risk group

4.13 E-doc

E-doc is a knowledge base and wiki portal built on Wiki.js. It provides a collaborative documentation platform integrated with Energy Logserver through single sign-on (SSO) authentication.

4.13.1 Service Details

Property	Value
Service name	`e-doc`
Install path	`/opt/e-doc/`
Config file	`/opt/e-doc/config.yml`
Database	`/opt/e-doc/database.sqlite` (SQLite)
Default port	5603 (HTTPS)
TLS certificates	`/etc/logserver-gui/ssl/logserver-gui.key`, `/etc/logserver-gui/ssl/logserver-gui.crt`

4.13.2 Configuration

The main configuration file is `/opt/e-doc/config.yml`. Key settings:

Parameter	Description	Default
`ssl.enabled`	Enable HTTPS	`true`
`ssl.port`	Listening port	`5603`
`ssl.key` / `ssl.cert`	TLS certificate paths	`/etc/logserver-gui/ssl/logserver-gui.key` / `.crt`
`logserverGuiSSLE`	Whether the GUI uses HTTPS	`true`
`logserverGuiCrec`	Credentials for GUI API authentication (format: `user:password`)	`e-doc:e-doc`
`logserverGuiHost`	GUI address (only if non-default)	`127.0.0.1:5601`
`db.type`	Database backend	`sqlite`
`db.storage`	Database file path	`/opt/e-doc/database.sqlite`

4.13.3 Authentication

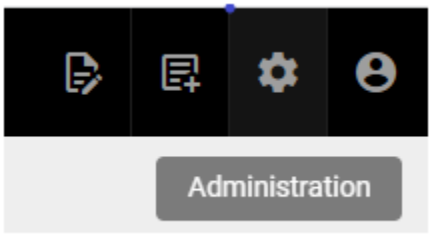
E-doc uses SSO with the Energy Logserver GUI. When a user clicks the **E-Doc** button in the top navigation bar:

1. The GUI verifies the user session and generates a one-time UUID token.
2. The browser is redirected to `https://<host>:5603/login/logserver/callback?uuid=<token>`.
3. E-doc calls back to the GUI API (`GET /api/admin/wikiuser?uuid=<token>`) to retrieve user details and roles.
4. A Wiki.js session is created with the user's Energy Logserver identity.

> **Note**
>
> The `e-doc` system user must exist in Energy Logserver with the `e-doc` role assigned. This user is used by the E-doc service to authenticate API calls to the GUI.

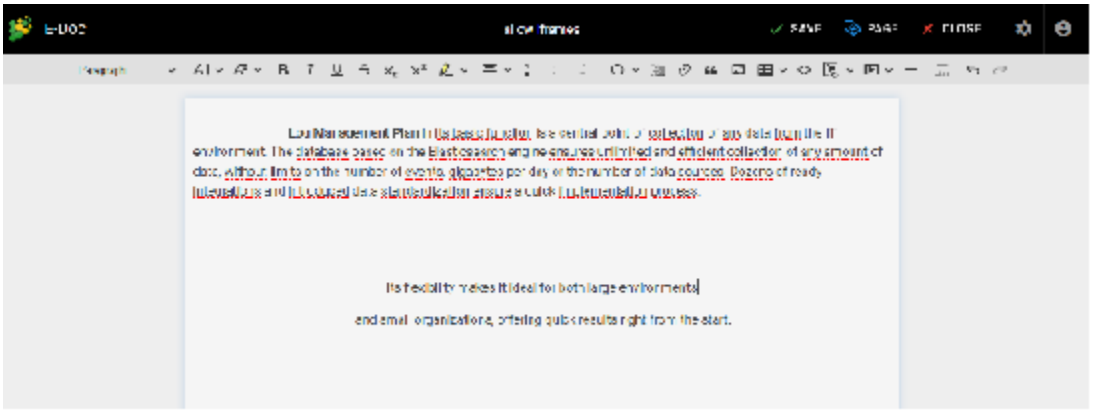
4.13.4 Document Editing

E-doc provides a full wiki editor for creating and organizing documentation pages.



4.13.5 Document Conversion

E-doc supports converting documents between formats.



4.13.6 Search Integration

E-doc indexes page content (title, description, body text, tags) in the Energy Logserver search engine using the `.e-doc` index. This enables full-text search and suggestions across all wiki pages.

4.13.7 Backup

To back up E-doc, preserve the following files:

- `/opt/e-doc/config.yml`, service configuration
- `/opt/e-doc/database.sqlite`, page content and user data
- `.e-doc` index in Energy Logserver, search index (can be re-indexed from the database)

4.14 System Administration

Comprehensive platform management covering monitoring, auditing, backup procedures, maintenance operations, and cluster administration.

The System Administration module provides essential tools and procedures for maintaining Energy Logserver in production environments. From real-time health monitoring and performance optimization to disaster recovery planning and cluster management, this module ensures system reliability, security compliance, and operational efficiency. Administrators will find guidance on routine maintenance, troubleshooting common issues, and managing distributed cluster deployments.

4.14.1 Overview

Energy Logserver's administration capabilities enable proactive system management and rapid issue resolution. Key features include:

- **Monitoring and Performance**: Skimmer installation for health monitoring, performance dashboards, and system metrics
- **Audit and Compliance**: Authorization plugins, configuration auditing, and audit trail management
- **Backup and Recovery**: Configuration backup procedures, disaster recovery planning, and capacity management
- **Troubleshooting Tools**: Diagnostic utilities, common issue resolution, and log analysis for system issues
- **Cluster Management**: Multi-node cluster configuration, task management, Cluster Console health monitoring, and cross-cluster operations

4.14.2 In This Section

This chapter covers the complete system administration lifecycle, from initial monitoring setup to advanced cluster management.

Monitoring and Performance

- *About Skimmer*
- *Skimmer Installation*
- *Skimmer service configuration*
  - *Kafka monitoring*
  - *Skimmer GUI configuration*
  - *Expected Data Nodes*

About Skimmer

Energy Logserver uses a monitoring module called Skimmer to monitor the performance of its hosts. Skimmer collects metrics from the following sources:

- **ELS Data Node API**: node stats, cluster health, cluster shards, license status, indices stats
- **Network Probe API**: JVM, GC, process, events, CPU load metrics
- **OS statistics**: CPU, filesystem, swap, network interfaces, VM stats, zombie processes, disk usage, TCP port status
- **Systemd services**: status of any configured systemd service (via DBus)
- **Kafka** (optional), consumer group lag metrics
- **Process monitoring**: PID tracking for configured processes

Skimmer logs to the systemd journal. Set the log level with the `SKIMMER_LOG_LEVEL` environment variable (`DEBUG`, `INFO`, `ERROR`).

Skimmer Installation

The RPM package is delivered with the system installer in the `install/agents` directory:

```
cd $install_directory/install/agents
yum install skimmer-1.0.*-x86_64.rpm -y
```

The package installs to `/opt/skimmer/` with the following layout:

Path	Description
`/opt/skimmer/bin/skimmer`	Binary
`/opt/skimmer/skimmer.conf`	Configuration file
`/etc/systemd/system/skimmer.service`	Systemd service unit

Skimmer service configuration

The Skimmer configuration is located in the `/opt/skimmer/skimmer.conf` file. The file uses a flat `key = value` format (lines starting with # are comments).

```
# enable stats collection
main_enabled = true

# index name in Data Node
index_name = skimmer
index_format = %Y.%m

# credentials for Data Node API
elasticsearch_auth = logserver:logserver

# Data Node output address (comma-separated list for fault tolerance)
elasticsearch_address = 127.0.0.1:9200

# Network Probe output address (comma-separated list for fault tolerance)
logstash_address = 127.0.0.1:6110

# Data Node API address for stats retrieval
elasticsearch_api = 127.0.0.1:9200

# Network Probe API address for stats retrieval
logstash_api = 127.0.0.1:9600

# collection interval (minimum 10s)
interval = 1min

# how often to collect shards, license, and indices stats
system_health_check_interval = 4h

# index patterns to monitor (default: all)
# indices_stats_patterns = *
# indices_stats_regex = .*

# Kafka monitoring (optional)
```

(continues on next page)

(continued from previous page)

```
# kafka_path = /usr/share/kafka/
# kafka_server_api = 127.0.0.1:9092
# kafka_monitored_topics = topic1,topic2
# kafka_monitored_groups = group1,group2
# kafka_outdated_version = false

# comma-separated process names to track PIDs
processes = /usr/sbin/sshd,/usr/sbin/rsyslogd

# comma-separated systemd services to monitor status
# extend the list as shown below to cover all 8.x services
systemd_services = logserver,logserver-probe,alert,cerebro,logserver-gui,e-
↪doc,intelligence,intelligence-scheduler,license-service,automation

# comma-separated port numbers to check
port_numbers = 9200,9300,9600,5514,5044,443,5601,5602,5603

# enable SSL for Data Node API connections
# elasticsearch_ssl = true

# specific paths to check disk usage
# check_disk_usage = /var/lib/logserver-probe/queue

# force node IP when Skimmer is not running on a cluster node
# elasticsearch_api_force_node_ip = 10.0.0.1

# redefine field type mappings
# mapping = {"store": "long"}
```

Note

List services in `systemd_services` by their unit names. Skimmer stores the state of each one in a `node_stats_systemd_service_<name>` field. The packaged configuration may still list the previous service aliases, and Skimmer reports them under the same `logserver`, `logserver-probe`, and `logserver-gui` fields. Network Probe stats fields use the `logserver-probe_stats_` prefix.

In the Skimmer configuration file, set the credentials to communicate with Data Node:

```
elasticsearch_auth = $user:$password
```

Kafka monitoring

To monitor the Kafka process and the number of documents in the queues of topics, run Skimmer on the Kafka server and uncomment the following parameters:

```
kafka_path = /usr/share/kafka/
kafka_server_api = 127.0.0.1:9092
kafka_monitored_topics = topic1,topic2
kafka_monitored_groups = group1,group2
kafka_outdated_version = false
```

- `kafka_path`, path to Kafka home directory (requires `kafka-consumer-groups.sh`)
- `kafka_server_api`, IP address and port for Kafka server API (default: `127.0.0.1:9092`)
- `kafka_monitored_groups`, comma-separated list of Kafka consumer groups; if not defined, the command will be invoked with the `--all-groups` parameter
- `kafka_outdated_version`, set to `true` if you use Kafka before v2.4.0

After the changes in the configuration file, restart the service:

```
systemctl restart skimmer
```

Skimmer GUI configuration

To view the collected data by Skimmer in the GUI, you need to add an index pattern.

Go to the "**Management**" -> "**Index Patterns**" tab and press the "**Create Index Pattern**" button. In the "**Index Name**" field, enter the formula `skimmer-*`, and select the "**Next step**" button. In the "**Time Filter**" field, select `@timestamp` and then press "**Create index pattern**".

In the "**Discover**" tab, select the `skimmer-*` index from the list of indexes. A list of collected documents with statistics and statuses will be displayed.

The Skimmer dashboard includes the following monitoring parameters (in the GUI, each visualization title is prefixed with `Skimmer -`):

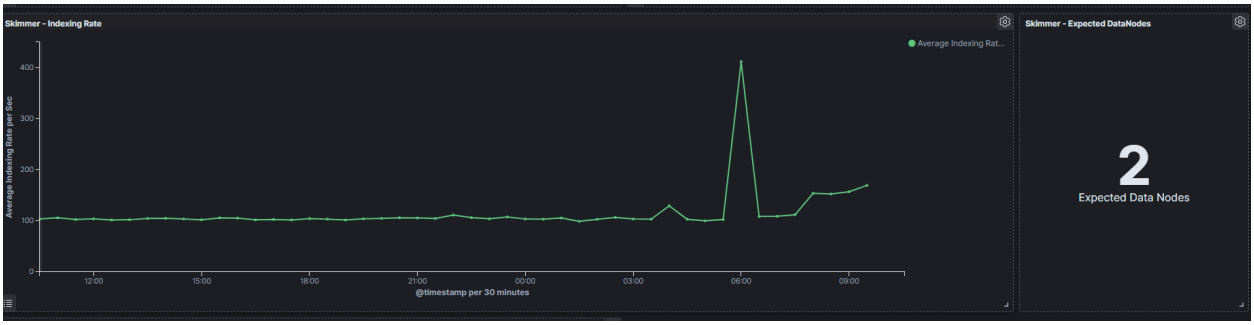
- `Logserver - Heap usage in percent`, total amount of Java heap memory currently used by the JVM Data Node process (in percent)
- `Logserver Probe - Heap usage in percent`, total amount of Java heap memory currently used by the JVM Network Probe process (in percent)
- `Logserver - Process CPU usage`, CPU time used by the Data Node process (in percent)
- `Logserver - Node CPU usage`, CPU usage of the host running the Data Node (in percent)
- `Logserver - Current queries`, current count of search queries to Data Node indices
- `Logserver - Current search fetch`, current count of the fetch phase for search queries to Data Node indices
- `GC Old collection`, duration of Java Garbage Collector Old collection (in milliseconds)
- `GC Young collection`, duration of Java Garbage Collector Young collection (in milliseconds)
- `Flush`, duration of the Data Node flushing process that permanently saves the transaction log to the Lucene index (in milliseconds)
- `Refresh`, duration of the Data Node refresh process that prepares new data for searching (in milliseconds)
- `Indexing`, duration of the Data Node document indexing process (in milliseconds)
- `Merge`, duration of the Data Node merge process that periodically merges smaller segments into larger segments (in milliseconds)
- `Indexing Rate`, growth of the cluster document count per second, measured on primary shards between two collection runs (the closest metric to EPS)
- `Expected DataNodes`, number of data nodes Skimmer considers necessary for the current load, never fewer than the nodes already running
- `Free Space`, total space and free space in bytes on the Data Node cluster
- `Index / Size in Bytes`, size of individual indices in bytes

- `Index / Documents`, document count per index

> **Note**
>
> `Indexing Rate` is a cluster-wide figure sampled at the `interval` set in the Skimmer configuration, one minute by default, and counted on primary shards, so replicas do not inflate it. When documents are removed between two samples the difference is negative, and Skimmer repeats the previous value instead of reporting a drop, which shows as a flat line while retention purges or index deletions run.

Expected Data Nodes

Based on the collected performance data of the Energy Logserver environment, Skimmer automatically indicates the need to run additional data nodes. The value stays at the number of running data nodes while the Data Node process CPU across the cluster is below 50 percent. Above that threshold Skimmer derives it from the total shard count, and it never drops below the number of nodes already running.

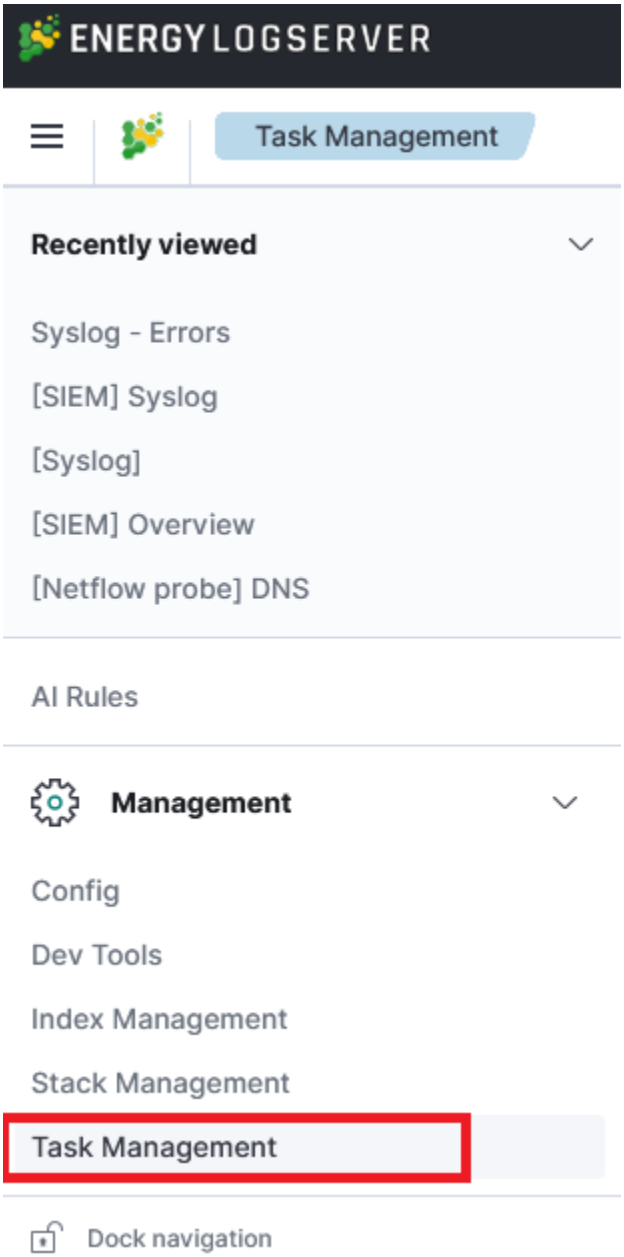


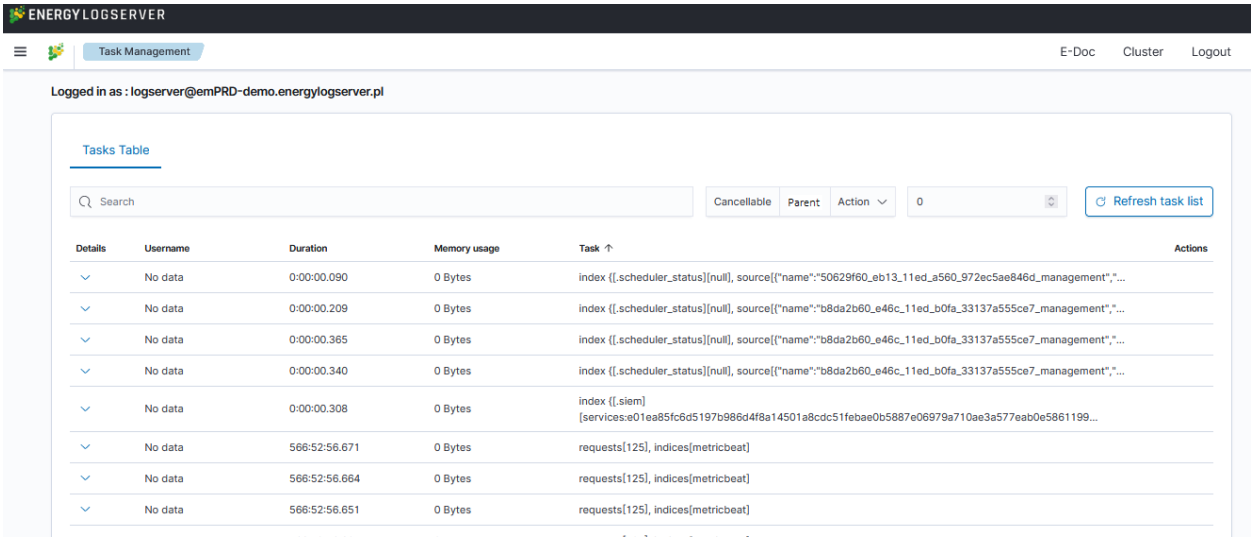
Cluster Management and Tools

- *Task Management*
- *Cluster Console - Cluster Health*
- *Data dump*
- *Data Node index management tool*
- *Cross-cluster Search*
- *Sync/Copy*
- *Separating Data Using Index Aliases*

Task Management

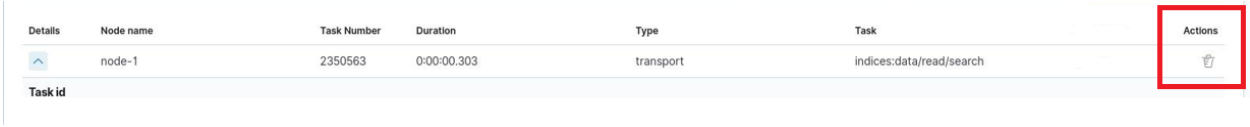
Energy Logserver provides the ability to manage tasks. Table with running tasks is located in `Tasks` section.





Long Running Query

If query is cancellable trash icon is available in Actions column. (Note: Not every query can be canceled)



Searchbar and filtering

It is possible to search for running tasks using the searchbar. The following filters are available:

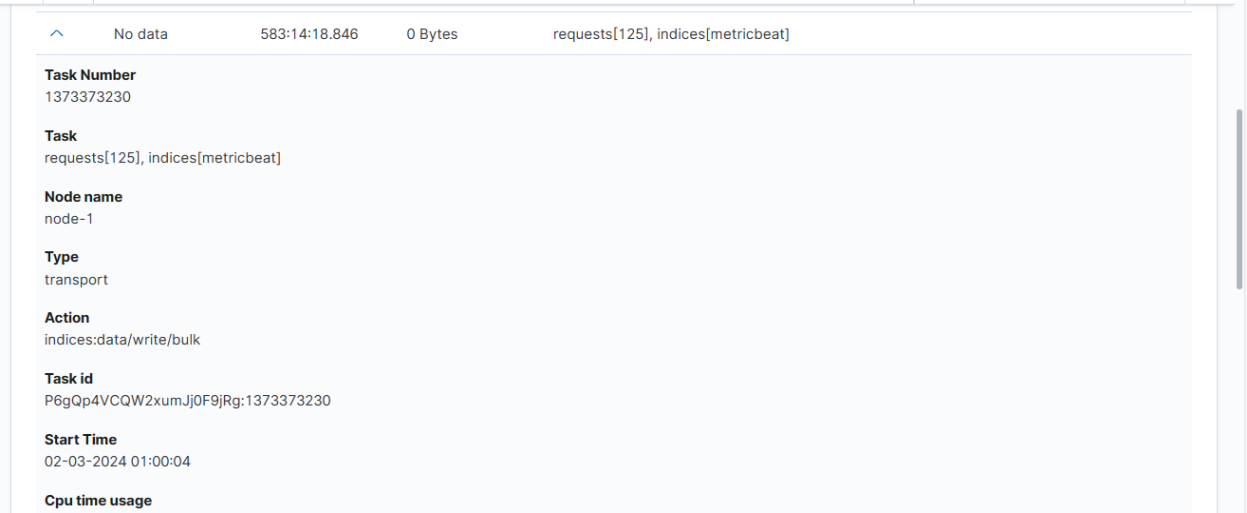
- `Cancellable`, filters only cancellable tasks.
- `Parent`, filters only parent tasks.
- `Action`, filters by action type (e.g., `indices:data/write/bulk`).

Additionally, a **Duration** input field allows you to specify a minimum running time (in seconds) to show only long-running tasks.

The task table displays the following columns: Action, Task ID, Running Time, Username, Memory Usage, and Node.

Task Details

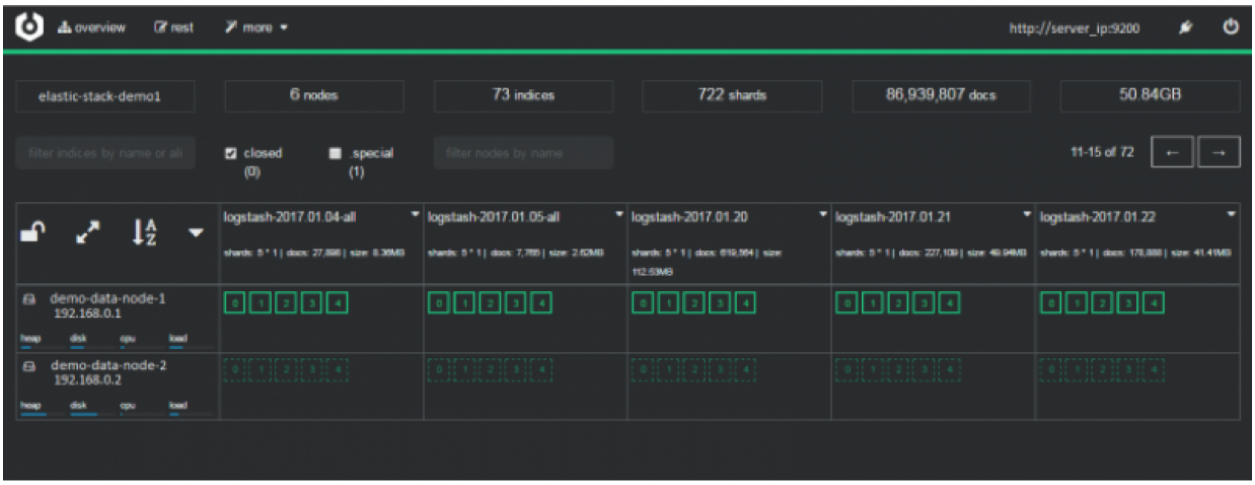
Expanding the details column provides more information about a given task. (Note: Depending on the type of task, different information may be provided.)



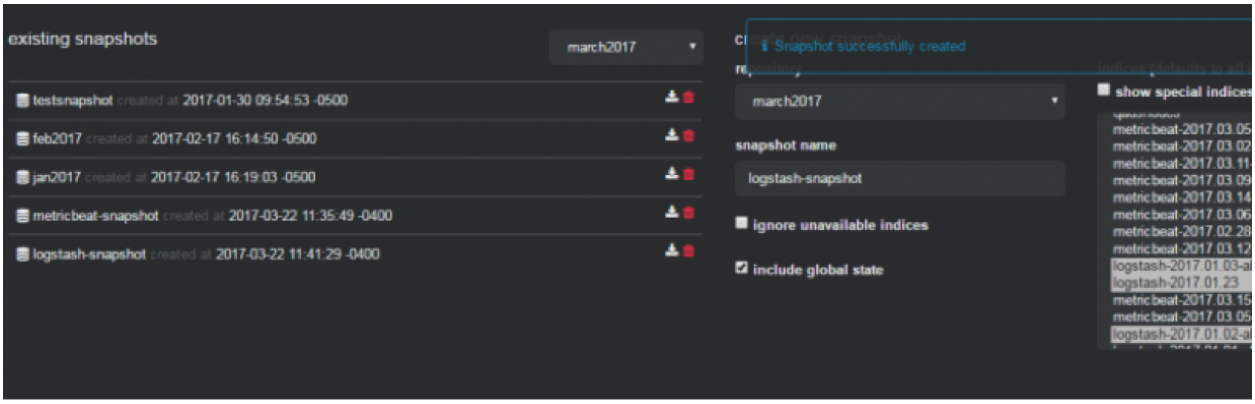
Cluster Console - Cluster Health

Cluster Console (Cerebro-based) is the cluster administration tool that allows you to perform the following tasks:

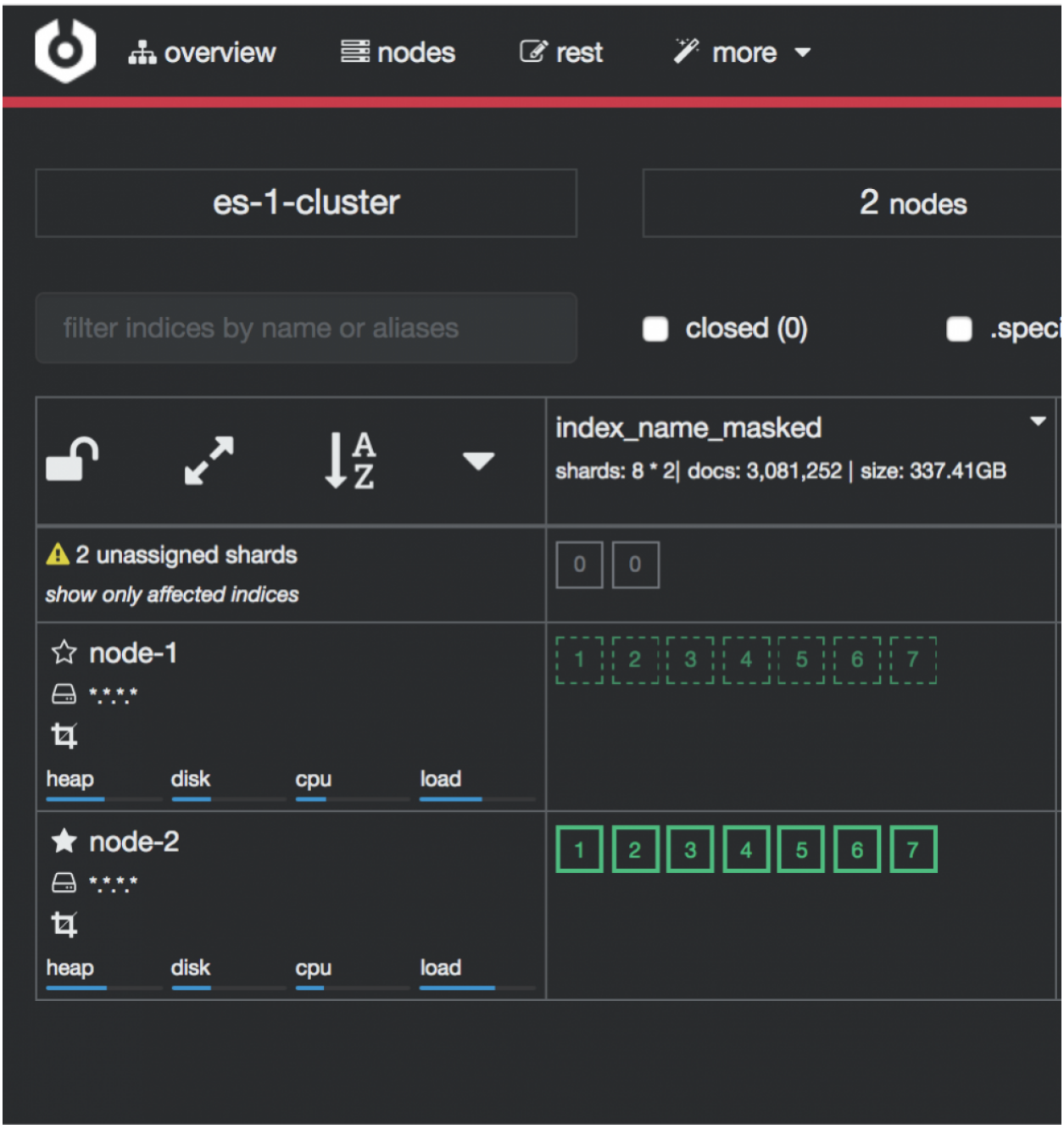
- monitoring and management of indexing nodes, indexes and shards:



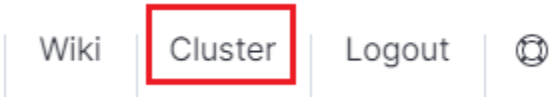
- monitoring and management of index snapshots:



- informing about problems with indexes and shards:



Access to the `Cluster` module is possible through the button in the upper right corner of the main window.



Cluster Console runs on port `5602` by default. Its configuration file is located at `/opt/cerebro/conf/application.conf`.

Data dump

Energy Logserver includes the `elasticdump` utility for exporting and importing index data between clusters or to/from JSON files. The tool is bundled with the system installer.

Data Node index management tool

The Index Management plugin provides GUI-based index operations such as close, delete, force merge, and rollover. Access it from **Management** → **Index Management**. For details on index operations, see the *Index Management* section.

Cross-cluster Search

Cross-cluster search lets you run a single search request against one or more remote clusters. For example, you can use a cross-cluster search to filter and analyze log data stored on clusters in different data centers.

Configuration

1. Use `_cluster` API to add at least one remote cluster:

```
curl -u user:password -X PUT "localhost:9200/_cluster/settings?pretty" -H
↪'Content-Type: application/json' -d'
{
  "persistent": {
    "cluster": {
      "remote": {
        "cluster_one": {
          "seeds": [
            "192.168.0.1:9300"
          ]
        },
        "cluster_two": {
          "seeds": [
            "192.168.0.2:9300"
          ]
        }
      }
    }
  }
}'
```

2. To search data in index `twitter` located on the `cluster_one` use following command:

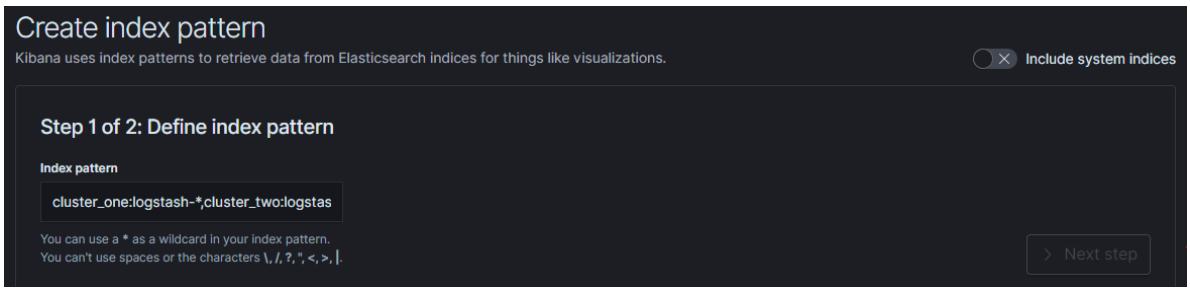
```
curl -u user:password -X GET "localhost:9200/cluster_one:twitter/_search?
↪pretty" -H 'Content-Type: application/json' -d'
{
  "query": {
    "match": {
      "user": "kimchy"
    }
  }
}'
```

3. To search data in index `twitter` located on multiple clusters, use following command:

```
curl -u user:password -X GET "localhost:9200/twitter,cluster_one:twitter,
↪cluster_two:twitter/_search?pretty" -H 'Content-Type: application/json'␣
↪-d'
{
  "query": {
    "match": {
      "user": "kimchy"
    }
  }
}'
```

4. Configure index pattern in GUI to discover data from multiple clusters:

```
cluster_one:syslog-*,cluster_two:syslog-*
```



Security

Cross-cluster search uses the Data Node transport layer (default 9300/tcp port) to exchange data. To secure the transmission, encryption must be enabled for the transport layer.

Configuration is in the `/etc/logserver/logserver.yml` file:

```
# Transport layer encryption
logserverguard.ssl.transport.enabled: true
logserverguard.ssl.transport.pemcert_filepath: "/etc/logserver/ssl/logserver.
↪crt"
logserverguard.ssl.transport.pemkey_filepath: "/etc/logserver/ssl/logserver.
↪key"
logserverguard.ssl.transport.pemkey_password: ""
logserverguard.ssl.transport.pemtrustedcas_filepath: "/etc/logserver/ssl/
↪rootCA.crt"

logserverguard.ssl.transport.enforce_hostname_verification: false
logserverguard.ssl.transport.resolve_hostname: false
```

Encryption must be enabled on each cluster.

Sync/Copy

The Sync/Copy module allows you to synchronize or copy data between two Energy Logserver clusters. You can copy or synchronize selected indexes or indicate index pattern.

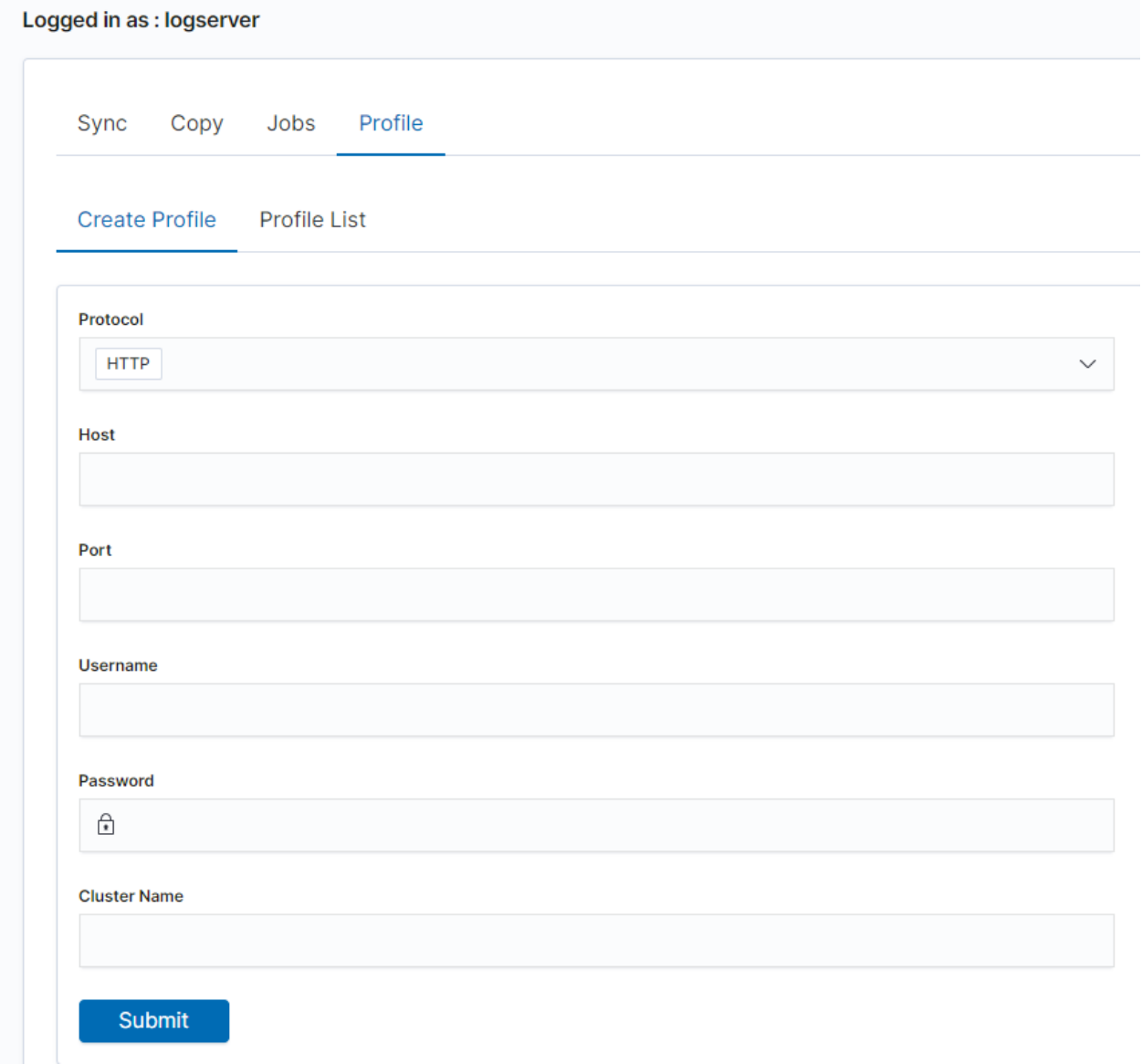
Sync/Copy Configuration

Before starting Sync/Copy, complete the source and target cluster data in the `Profile` and `Create profile`tab:

- Protocol, `http` or `https`
- Host, IP address of the ingest node
- Port, communication port (default `9200`)
- Username, username that has permission to get and save data on the cluster
- Password, password of the above user
- Cluster name

When using `https`, additional SSL/TLS fields are available:

- Key file, path to the client key file
- Certificate file, path to the client certificate file
- Certificate Authority file, path to the CA certificate file
- Passphrase, passphrase for the key file (if applicable)

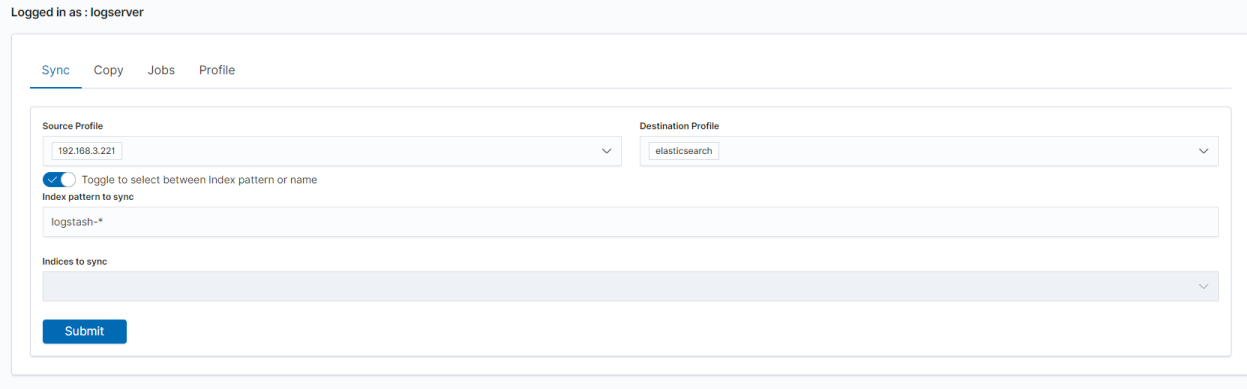


You can view or delete the profile in the `Profile List` tab.

Synchronize data

To perform data synchronization, follow the instructions:

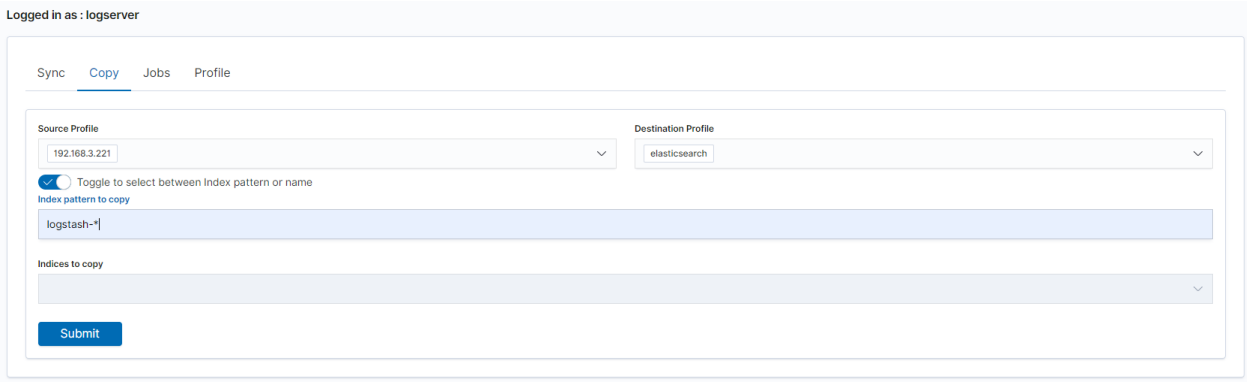
- go to the `Sync` tab;
- select `Source Profile`
- select `Destination Profile`
- enter the index pattern name in `Index pattern to sync`
- or use switch `Toggle to select between Index pattern or name` and enter indices name.
- to create synchronization task, press `Submit` button



Copy data

To perform data copy, follow the instructions:

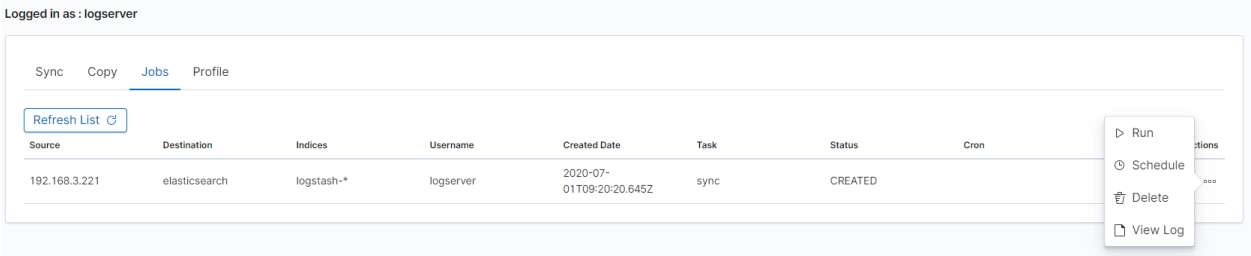
- go to the `Copy` tab;
- select `Source Profile`
- select `Destination Profile`
- enter the index pattern name in `Index pattern to sync`
- or use switch `Toggle to select between Index pattern or name` and enter indices name.
- to start copying data press the `Submit` button



Running Sync/Copy

Prepared Copy/Sync tasks can be run on demand or according to a set schedule. To do this, go to the `Jobs` tab. With each task you will find the `Action` button that allows:

- running the task on demand;
- enabling the scheduler (Cron format);
- disabling the scheduler;
- deleting the task;
- viewing task logs.



Separating Data Using Index Aliases

Index aliases allow you to expose a filtered subset of an index to specific user groups. This is useful when a single index contains data from multiple sources or tenants and you need to restrict access by role.

Procedure

1. **Create an alias on an index**

   In the Management Interface, navigate to the index and create an alias scoped to the appropriate tag or filter condition.

2. **Create an index pattern on the alias**

   In Management → Index Patterns, create a new index pattern using the alias name instead of the original index name.

3. **Assign a role to the index pattern**

   In Management → Roles, create or update a role and assign read permissions to the alias-based index pattern. Users with this role will only see data accessible through the alias.

> **Note**
>
> This approach works with Energy Logserver's role-based access control (RBAC) system. The alias acts as a logical view, users assigned the role cannot access the underlying index directly.

Backup and Recovery

This section covers configuration backup using the built-in backup script. For Energy Logserver data snapshot backup and disaster recovery procedures, see *Backup and Disaster Recovery*. For sizing and storage planning, see *Capacity Planning Guidelines*.

Configuration Backup

Backing up

The backup bash script is located on the hosts with Data Node in the location: `/usr/share/logserver/utils/configuration-backup.sh`.

The script backs up the following data:

- **System indices** (via elasticdump): `.agentsconfig`, `.alertrules`, `.archive`, `.archive_task`, `.authconfig`, `.authuser`, `.cmdb`, `.e-doc`, `.elastfilter`, `.indexmanagement`, `.intelligence`, `.intelligence_models`, `.kibana`, `.kibana_*`, `.networkprobes`, `.playbooks`, `.reports`, `.riskcategories`, `.risks`, `.scheduler`, `.security`, `.sources`,

`.sync_profile`, `.sync_task`, `.taskmanagement`, `.trustedhost`, plus the older `.intelligence_rule_configuration` and `.intelligence_assistant` indices that remain after an upgrade from 8.0. Indices missing on the cluster are skipped.

- **Cluster settings** (via `_cluster/settings` API)
- **Index templates** (via elasticdump)
- **Component configurations**:
  - `/etc/logserver/`: Data Node configuration and license files
  - `/etc/logserver-probe/`: Network Probe configuration
  - `/etc/logserver-gui/`: GUI configuration
  - `/var/ossec/etc/`: SIEM (Endpoint Security) configuration
  - `/usr/share/logserver-gui/data/wazuh/config/`: Endpoint Security GUI configuration
  - `/opt/alert/`: Alert configuration
  - `/opt/cerebro/`: Cluster Console configuration and keystore
  - `/opt/skimmer/`: Skimmer configuration
  - `/opt/intelligence/`: Intelligence engine configuration
  - `/opt/license-service/`: License Service configuration
  - `/opt/e-doc/`: E-Doc configuration and database
  - `/opt/wiki/`: Wiki configuration and database
  - `/etc/logserver-store/`: Logserver keystore

The script uses `/tmp/backup/` as intermediate storage, then packages everything into a `tar.gz` archive named `backup-<hostname>-<datetime>.tar.gz` in the target directory (default: `/root/`). The intermediate files are deleted after packaging.

Note

The script requires the `elasticdump` binary (installed with Logserver GUI at `/usr/share/logserver-gui/elasticdump/elasticdump`). In split deployments where Data Node and GUI run on separate hosts, ensure elasticdump is available on the Data Node host.

Warning

The script reads credentials from `/etc/logserver-gui/logserver-gui.yml` and `/etc/logserver/logserver.yml`. If you have changed the default password or protocol, either update these config files or edit the script to set credentials manually (see the `read_config` section in the script).

Scheduled backup

The RPM package installs a cron job automatically at `/etc/cron.d/configuration-backup` that runs the backup daily at 01:00:

```
0 1 * * * root /usr/share/logserver/utils/configuration-backup.sh
```

To change the backup destination, edit the `backupPath` variable at the top of the script.

Restoration from backup

To restore from a backup archive:

1. Extract the backup archive. The files land in a `<datetime>` subdirectory:

```
mkdir -p /tmp/restore
tar -xzf backup-<hostname>-<datetime>.tar.gz -C /tmp/restore
```

2. Prepare the authorization header for the Data Node, using the account from `opensearch.username` in `/etc/logserver-gui/logserver-gui.yml`:

```
AUTH="{\"Authorization\":\"Basic $(echo -n '<username>:<password>' |␣
↪base64 -w0)\"}"
```

   Without it elasticdump stops with `UNAUTHORIZED: {"status":401,"message":"Request unauthorized"}`.

3. Restore each system index, mapping first, then data. For `.alertrules`:

```
/usr/share/logserver-gui/elasticdump/elasticdump --headers "$AUTH" \
  --input=/tmp/restore/<datetime>/mapping.alertrules.json \
  --output=http://localhost:9200/.alertrules \
  --type=mapping
/usr/share/logserver-gui/elasticdump/elasticdump --headers "$AUTH" \
  --input=/tmp/restore/<datetime>/data.alertrules.json \
  --output=http://localhost:9200/.alertrules \
  --type=data
```

4. Restore index templates:

```
/usr/share/logserver-gui/elasticdump/elasticdump --headers "$AUTH" \
  --input=/tmp/restore/<datetime>/templates.json \
  --output=http://localhost:9200 \
  --type=template
```

5. Restore component configurations by extracting the respective tar.gz sub-archives to their original paths (e.g., `/etc/logserver/`, `/etc/logserver-gui/`, `/opt/alert/`).

6. Restart all affected services after restoring configuration files.

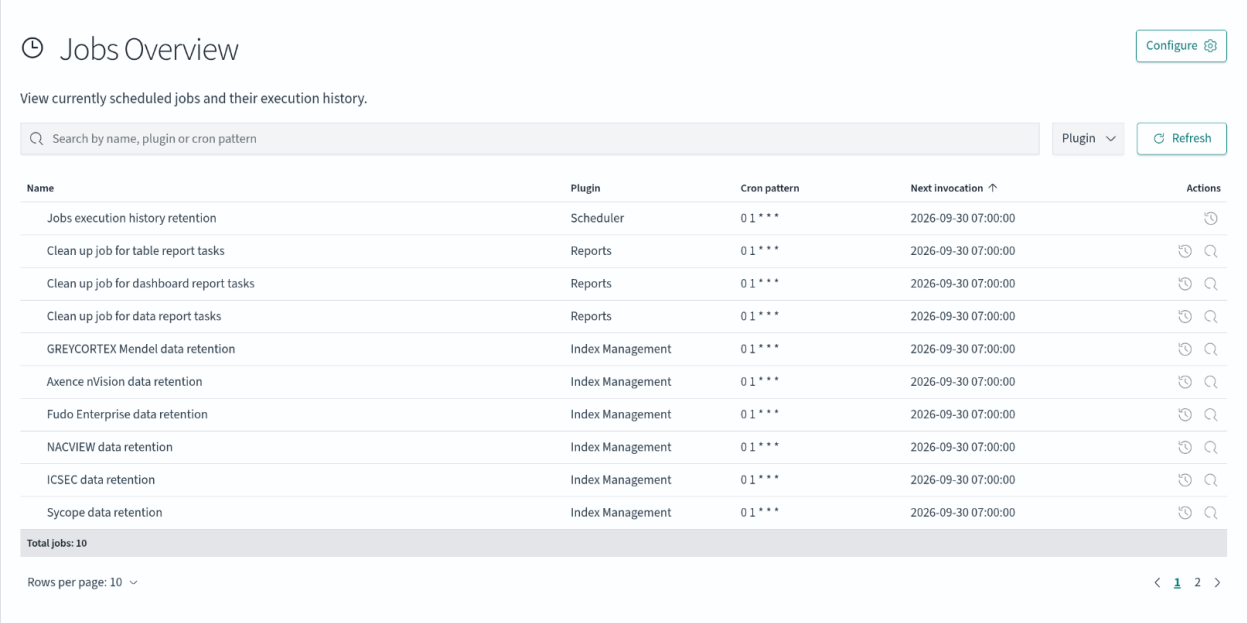
Scheduler

Energy Logserver 8.0 introduces a unified **Scheduler** that collects every scheduled task from across the product, Archive, Index Management, Sync/Copy, into one view, with full execution history for each run. Administrators can see at a glance what runs when, whether past runs succeeded, and how long they took.

Accessing the Scheduler

Open **Sources** from the sidebar (Management group), then switch to the **Scheduler** tab. The scheduler also has a direct URL: `/app/scheduler/jobs`.

The page opens on **Jobs Overview**.



Jobs Overview

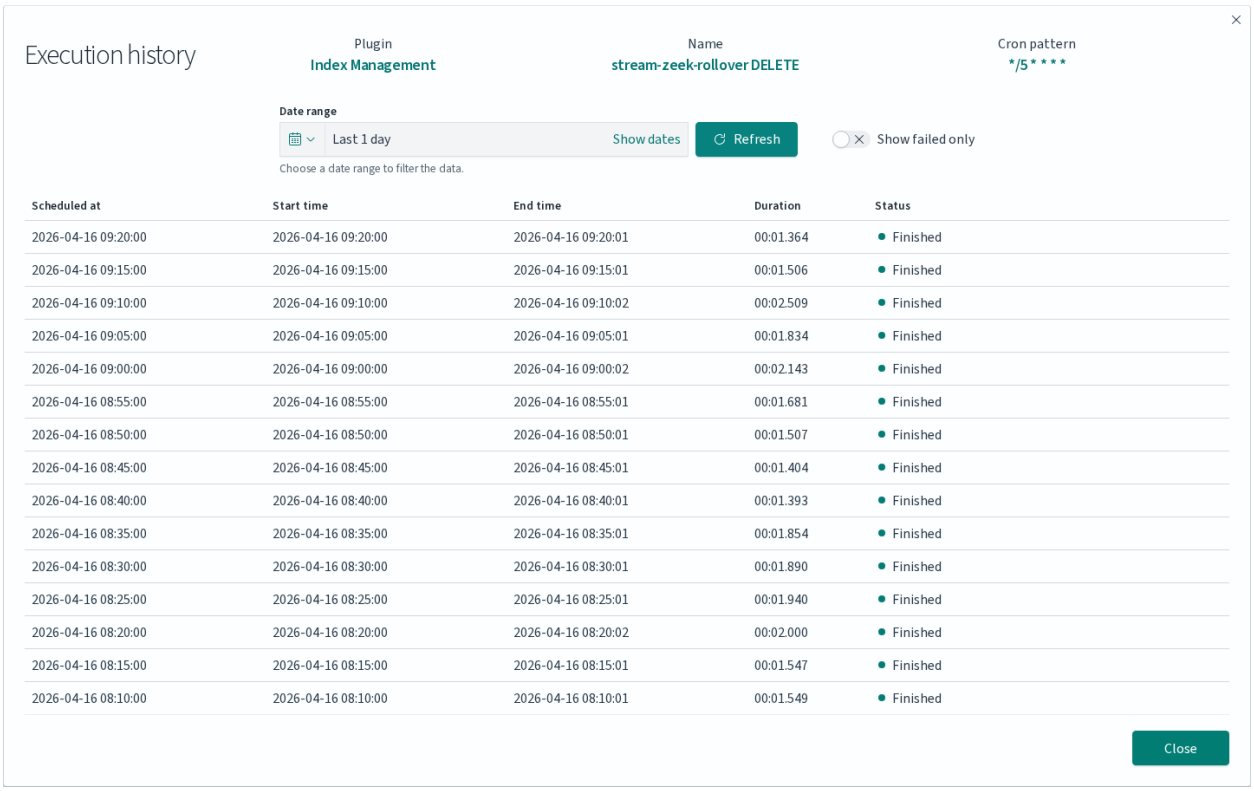
The Jobs Overview table lists every scheduled task currently registered in the cluster, regardless of the module that created it.

Column	Description
`Name`	Task name as defined in the originating module.
`Plugin`	Module that owns the task: **Index Management**, **Archive**, **Reports**, or **Scheduler** (Sync/Copy).
`Cron pattern`	Cron expression controlling when the task runs.
`Next invocation`	Date and time of the next scheduled run.
`Actions`	Open the **execution history** modal for that task, or **Go to plugin** to open the module that owns it.

Use the **Plugin** dropdown in the toolbar to filter the list to a single module. The **Refresh** button reloads the table without leaving the page.

Execution History

Click the action icon on a task row to open the **Execution history** modal.



The modal shows every recorded run of the task within the selected date range:

Column	Description
`Scheduled at`	Time at which the scheduler was supposed to fire the task.
`Start time`	Time at which the task actually started running.
`End time`	Time at which the task finished (or aborted).
`Duration`	Wall-clock time between Start and End.
`Status`	**Finished** for a successful run, **Failed** for an error.

Controls at the top of the modal:

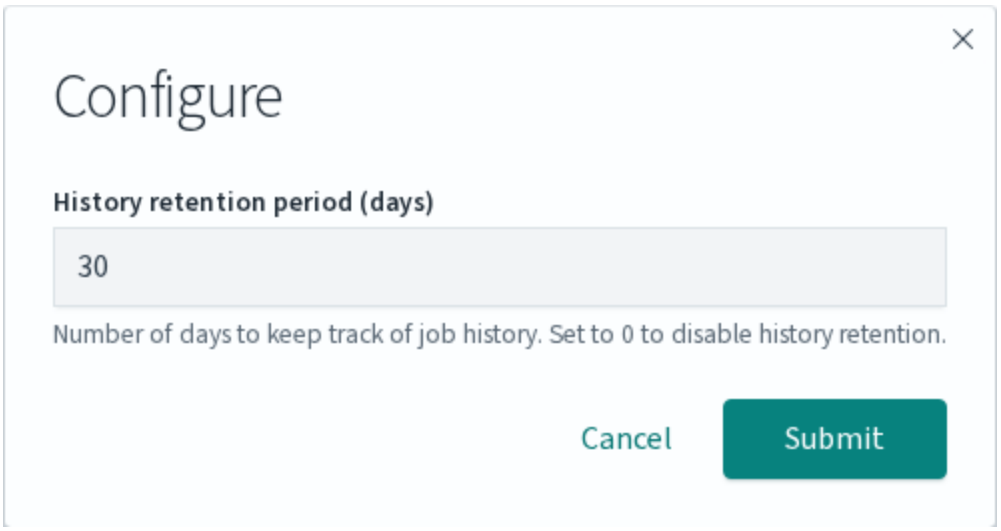
- **Date range picker**: defaults to *Last 1 day*; supports absolute ranges and arbitrary quick options.
- **Show failed only**: hides successful runs, useful when auditing reliability.
- **Refresh**: reloads the history without closing the modal.

The history only goes back as far as the configured retention period (see next section).

Configuring History Retention

Execution history is kept for a fixed number of days, configurable cluster-wide.

Click **Configure** in the Jobs Overview toolbar to open the settings modal.



Field	Description
`History retention period (days)`	Number of days of job history to keep. Defaults to **30**. Set to **0** to disable history retention entirely.

Disabling history (`0`) stops the scheduler from recording new executions but does not remove existing records immediately; older entries are pruned as their retention window expires.

Module Integration

Each module that schedules work exposes its own management view in addition to appearing in the central Jobs Overview. Use the module view to create and edit tasks; use the Scheduler to audit their execution.

Module	Where tasks are created	What appears in Scheduler
Archive	**Archive** app → **Archive** tab → **Create Task** (supports `Schedule task (crontab format)` field)	All scheduled archive tasks with their cron patterns.
Index Management	**Index Management** app → **Create Action** (tasks flagged `Scheduled: Yes`)	Rollover, delete, close and custom index management actions running on a schedule.
Sync/Cop	**Sources** → **Sync/Copy** tab	Index replication and synchronisation tasks.
Reports	**Reports** app → **Data Export** tab → **Create Task** with **Scheduling** enabled	Scheduled exports, named for example `CSV data report "<task name>"`, and the built-in clean-up jobs for table, dashboard and data reports.

Tasks that are defined but not scheduled (for example, Index Management actions flagged `Scheduled: No`) do not appear in Jobs Overview, they run only on demand.

Audit and Core Modules

- *Audit actions*
  - *Audit records*
  - *Authorization Plugin*
  - *Config*
  - *Reports*
  - *Alerts*
  - *Index Management*
  - *Archive*
  - *Sync*
  - *Agents*
  - *Intelligence*
  - *Network Probe*

Audit actions

Energy Logserver logs audit events for security-relevant operations across the platform. Audit actions are grouped by module.

Audit records

Audit documents go to the `audit` alias, which rolls over into indices named `audit-<date>-<number>`. A login record looks like this:

```
{
  "@timestamp": "2026-09-25T08:58:26.883Z",
  "username": "admin",
  "operation": "LOGIN",
  "request": "/_logserver/login",
  "method": "PUT",
  "params": {},
  "indices": [],
  "client": "10.4.2.133"
}
```

Since 8.1.0 `client` holds the address of the browser that sent the request to the GUI. Earlier versions recorded `127.0.0.1` for every GUI action. The GUI takes the address of the incoming TCP connection and ignores `X-Forwarded-For`, so behind a reverse proxy `client` shows the proxy address, and requests sent from the GUI host itself show `127.0.0.1`.

Authorization Plugin

Action Type	Path	From Request
LOGIN	/_logserver/login	username
LOGOUT	/_logserver/logout	username
FAILED_LOGIN	/_logserver/login	username
USER_CREATE	/_logserver/accounts	created user content
USER_UPDATE	/_logserver/accounts	whole body with diff
USER_DELETE	/_logserver/accounts	deleted user content
ROLE_CREATE	/_logserver/constraints	whole body with diff
ROLE_UPDATE	/_logserver/constraints	whole body with diff
ROLE_DELETE	/_logserver/constraints	whole body with diff
BULK	_bulk	whole body if enabled
QUERY	*	whole body if enabled
OBJECTS	/.kibana*	whole body if enabled

Paths excluded from auditing

The following paths are excluded from audit logging:

- /
- /_nodes*
- /_stats*
- /.auth
- /.authconfig*
- /_logserver/configuration/trustedhost
- /_logserver/resolve_token
- /_logserver/reports/checkpass
- /_logserver/tasksList
- /_logserver/configuration/domains
- /_logserver/license
- /_logserver/license/repo_access
- /_logserver/license/mssp
- /_logserver/mssp/*

Config

Action Type	Path	From Request
TOKENS DELETED	post:/api/setting/job/deletealltokens	,
SETTINGS TOKENDELETE	put:/api/setting/tokendelete	payload.value
SETTINGS TIMEOUT	put:/api/setting/ttl	payload.value
SETTINGS AUDIT SELECTION	put:/api/setting/auditselection	payload.value1
SETTINGS AUDIT EXCLUSION	put:/api/setting/auditexclusion	payload
SETTINGS ALERT EXCLUDE FIELDS	put:/api/setting/alert_exclude_fields	payload.value1
SETTINGS AUTH DOMAINS	put:/api/setting/auth_domains	payload.default_domain
MSSP PERMISSIVE MODE	put:/api/setting/mssp_permissive	payload.value2
MSSP SOURCES ALLOWED	post:/api/mssp/sources/allow	payload.sourcesIds
MSSP SOURCE DESCRIPTION EDITED	put:/api/mssp/source	payload.description
MSSP SOURCES DELETED	delete:/api/mssp/sources	payload.sourcesIds

Reports

Action Type	Path	From Request
DATA EXPORT CREATED	post:/api/reports/data/export	payload.taskName
MANUAL DATA EXPORT CREATED	post:/api/reports/data/export_manual	payload.user
DATA EXPORT EDITED	put:/api/reports/data/export	payload.taskName
DASHBOARD REPORT EXPORT CREATED	post:/api/reports/dashboard/export	payload.taskName
DASHBOARD REPORT EXPORT EDITED	put:/api/reports/dashboard/export	payload.taskName
DATA TABLE REPORT EXPORT CREATED	post:/api/reports/table/export	payload.taskName
DATA TABLE REPORT EXPORT EDITED	put:/api/reports/table/export	payload.taskName
SCHEDULED TASK ENABLED	put:/api/reports/scheduler/enable	payload.id
SCHEDULED TASK DISABLED	put:/api/reports/scheduler/disable	payload.id
TASKS DELETED	delete:/api/reports	payload.objs
SETTINGS PDF_EXPIRY	post:/api/reports/settings/pdf	payload.pdfExpiry
SETTINGS CSV_EXPIRY	post:/api/reports/settings/csv	payload.csvExpiry
REPORT UPLOAD LOGO	post:/api/reports/settings/logos	payload.fileName
ONGOING TASK STOP	post:/api/reports/stop/	params.taskId

Alerts

Action Type	Path	From Request
ALERT RULE CREATED	post:/api/alerts/alertrule	payload.alertrulename
ALERT RULE EDITED	put:/api/alerts/alertrule	payload.alertrulename
ALERT RULES SAVED	post:/api/alerts/alertrule/saverules	,
ALERT RULE ENABLED	put:/api/alerts/alertrule/switch	payload.alertsList
ALERT RULE DISABLED	put:/api/alerts/alertrule/switch	payload.alertsList
ALERT RULE RAN ONCE	post:/api/alerts/alertrule/runonce	payload.id
ALERT RULE DELETED	delete:/api/alerts/alertrule/	params.id
ALERT GROUP CREATED	post:/api/alerts/groups	payload.newGroupName
ALERT GROUP RENAMED	put:/api/alerts/groups	payload.groupName, payload.rename
ALERT GROUP DELETED	delete:/api/alerts/groups	payload.groupName
ALERTS ADDED TO GROUP	put:/api/alerts/updatealerts	payload
ALERT CHANGED ROLES	put:/api/alerts/alertrules/changeroles	payload.alertsList
ALERT MANUAL INCIDENT CREATED	post:/api/alerts/incidents/create_manual	payload.data.rule_name
ALERT RULE INCIDENT EDITED	put:/api/alerts/incidents	payload.id

Index Management

Action Type	Path	From Request
ACTION CREATED	post:/api/index_management/action	payload.name
ACTION EDITED	put:/api/index_management/action	payload.name
ACTION START NOW	post:/api/index_management/action/run_action	payload.id
ACTION DELETED	delete:/api/index_management/action/	params.name
SYSTEM INDEX ROLLOVER CONFIGURE	post:/api/index_management/settings/rollover/	payload

Archive

Action Type	Path	From Request
ARCHIVAL TASK CREATED	post:/api/archive/archive_task	payload.id
ARCHIVAL TASK UPDATED	put:/api/archive/archive_task/	params.taskId
ARCHIVAL TASK START NOW	post:/api/archive/archive_task/run/	params.taskId
TASKS DELETED	delete:/api/archive/tasks	payload.objs
SEARCH TASK CREATED	post:/api/archive/search_task/run	payload.searchtext
RESTORE TASK CREATED	post:/api/archive/restore_task/run	payload.destinationIndex
ARCHIVE(S) DELETED	post:/api/archive/retention/archives	payload.archives
ARCHIVE POLICY EXECUTED	post:/api/archive/retention/execute/	params.archiveTaskId
ARCHIVE RETENTION POLICY DELETED	delete:/api/archive/retention/policies	payload.archiveTaskIds

Sync

Action Type	Path	From Request
SYNC PROFILE CREATED	post:/api/sync/clusterprofile	payload.host
SYNC PROFILE DELETED	post:/api/sync/syncTask/delete	payload.id
SYNC SYNCHRONISED	put:/api/sync/syncTask	payload.destination
SYNC COPIED	post:/api/sync/copyTask	payload.destination
SYNC JOB DELETED	post:/api/sync/syncTask/delete	payload.id
SYNC JOB RAN	post:/api/sync/runTask	payload.id
SYNC ACTION ENABLED	*:/api/sync/scheduler/	payload.id
SYNC ACTION DISABLED	*:/api/sync/scheduler/	payload.id

Agents

Action Type	Path	From Request
AGENTS AGENT RELOADED	post:/api/agents/reloadagents/	params.id
AGENTS MASTERAGENT RELOADED	post:/api/agents/reloadmasteragent/	params.id
AGENTS DELETED	delete:/api/agents/	params.uid
AGENTS FILE CREATED	post:/api/agents/templates/file	payload.file.name
AGENTS FILE DELETED	delete:/api/agents/files	,
AGENTS FILE EDITED	put:/api/agents/files	payload.name
AGENTS TEMPLATE CREATED	post:/api/agents/templates	payload.template.name
AGENTS TEMPLATE EDITED	put:/api/agents/templates	payload.template.name
AGENTS TEMPLATE DELETED	delete:/api/agents/templates/	params.id
AGENTS BEATS STARTED	*:/api/agents/manage/	params.command, params.beatname
AGENTS BEATS RESTARTED	*:/api/agents/manage/	params.command, params.beatname
AGENTS BEATS STOPPED	*:/api/agents/manage/	params.command, params.beatname

Intelligence

Action Type	Path	From Request
INTELLIGENCE RULE CREATED	get:/api/intelligence/ intelligence_modelcreation_set_for	query.rule_name
INTELLIGENCE RULE EDITED	get:/api/intelligence/ intelligence_modelcreation_set_for	query.rule_name
INTELLIGENCE RULE STOPPED	get:/api/intelligence/ intelligence_airules_set_stoprule	query.rule_uid
INTELLIGENCE RULE STARTED	get:/api/intelligence/ intelligence_airules_set_startrule	query.rule_uid
INTELLIGENCE RULE DELETED	get:/api/intelligence/ intelligence_airules_set_deleterul	query.rule_uid
INTELLIGENCE USE CASE DOWNLOADED	get:/api/intelligence/ intelligence_download_use_case	query.rule_uid, query.use_case_name
INTELLIGENCE USE CASE UPLOADED	post:/api/intelligence/ intelligence_upload_use_case	payload.rule. rule_name
ASSISTANT CONNECTION CREATED	post:/api/intelligence/ connections	payload.name
ASSISTANT CONNECTION EDITED	put:/api/intelligence/ connections/	params.id, payload. name
ASSISTANT CONNECTION DELETED	delete:/api/intelligence/ connections/	params.id
ASSISTANT PROMPT CREATED	post:/api/intelligence/prompts	payload.name, payload.id, payload. type
ASSISTANT PROMPT EDITED	put:/api/intelligence/prompts/	params.id, payload. name
ASSISTANT PROMPT DELETED	delete:/api/intelligence/ prompts/	params.id
ASSISTANT PROMPT EXECUTED	post:/api/intelligence/prompt	payload. systemPromptID, payload.name
ASSISTANT LOG EXPLAIN REQUESTED	post:/api/intelligence/ explain_log	,
ASSISTANT THREAT DESCRIPTION REQUESTED	post:/api/intelligence/ describe_threat	,
ASSISTANT ALERT CREATION REQUESTED	post:/api/intelligence/ create_alert	,

Network Probe

Action Type	Path	From Request
NETWORK-PROBE FILE CREATED	post:/api/network-probe/files	payload.file.path
NETWORK-PROBE FILE DELETED	delete:/api/network-probe/files	payload.filePath
NETWORK-PROBE UNREGISTERED	delete:/api/network-probe/host	payload.hostId
NETWORK-PROBE FILE REREGISTERED	post:/api/network-probe/files/register	payload.host.ip
NETWORK PROBE SERVICES STOPPED	post:/api/network-probe/services	payload.services.names
NETWORK PROBE SERVICES STARTED	post:/api/network-probe/services	payload.services.names
NETWORK PROBE SERVICES RESTARTED	post:/api/network-probe/services	payload.services.names
NETWORK-PROBE PIPELINES ENABLED	post:/api/network-probe/pipelines/enable	payload.pipelinesIds, params.hostId
NETWORK-PROBE PIPELINES DISABLED	post:/api/network-probe/pipelines/disable	payload.pipelinesIds, params.hostId
NETWORK-PROBE PIPELINES RELOADED	post:/api/network-probe/pipelines/reload	,
NETWORK-PROBE PIPELINE CREATED	post:/api/network-probe/pipeline/	params.name

Troubleshooting

Diagnostic tools, common issues, and resolution procedures.

- *Too many open files*
- *The Logserver-GUI status code 500*
- *Support tool*
  - *Running the support tool*
- *Verification steps and logs*
  - *Verification of Energy Logserver service*
  - *Verification of Logserver-Probe service*
  - *Debugging*
  - *Verification of Energy Logserver GUI service*
- *SIEM Agent - Windows CP1250 decoding problem*
- *My disk is full*
  - *Step 1: free disk space*
  - *Step 2: unblock indices*
  - *Step 3: verify*

Too many open files

If you have a problem with too many open files by the Energy Logserver process, check the values in the following configuration files:

- `/usr/lib/systemd/system/logserver.service`, `LimitNOFILE=262144`
- `/etc/security/limits.d/30-logserver.conf`, `logserver - nofile 262144`
- `/etc/sysconfig/logserver`, `MAX_OPEN_FILES=262144`

If you need to increase these values, edit the files and then reload systemd:

```
systemctl daemon-reload
systemctl restart logserver
```

The Logserver-GUI status code 500

If the login page is displayed in Logserver-GUI, but after the attempt to login, the browser displays “error: 500”, and the logs will show entries:

```
Error: Failed to encode cookie (sid-auth) value: Password string too short␣
↪(min 32 characters required).
```

Generate a new `login.ironsecret` with the following command:

```
echo "login.ironsecret: \"$(</dev/urandom tr -dc _A-Z-a-z-0-9 | head -c32)\""␣
↪>> /etc/logserver-gui/logserver-gui.yml
```

Support tool

Energy Logserver includes a support tool (formerly diagnostic tool) that collects system data necessary for problem analysis by the support team.

The tool is located at: `/usr/share/logserver/utils/support-tool.sh` (a `diagnostic-tool.sh` symlink is also available for backward compatibility).

The support tool collects the following information:

- **Configuration files**: Data Node (`/etc/logserver/`), GUI (`/etc/logserver-gui/`), Network Probe (`/etc/logserver-probe/conf.d/`), Alert (`/opt/alert/config.yaml`), Cerebro (`/opt/cerebro/`), Skimmer (`/opt/skimmer/skimmer.conf`), E-Doc (`/opt/e-doc/config.yml`), SIEM/Endpoint Security (`/var/ossec/etc/`)
- **Service logs** (via journalctl): `logserver`, `logserver-gui`, `logserver-probe`, `alert`, `cerebro`, `skimmer`, `e-doc`
- **Cluster information**: cluster health, cluster settings, indices summary, shard details

Passwords and IP addresses are automatically removed from collected files.

Running the support tool

The tool reads credentials automatically from `/etc/logserver-gui/logserver-gui.yml`. No command-line arguments are needed:

```
/usr/share/logserver/utils/support-tool.sh
```

> **Warning**
>
> If you have changed the default password or protocol, edit the script and update the credentials manually (see the `read_config` section), or disable `read_config` and set `esUsername`, `esPass`, and `protocol` variables directly.

The tool saves the results as an AES-256-CBC encrypted archive at `/root/diagnostics-logs-<hostname>-<date>.tar.gz.enc`.

To decrypt the archive:

```
openssl enc -aes-256-cbc -md sha512 -base64 -d -in diagnostics-logs-*.tar.gz.
↪enc -out diagnostics-logs.tar.gz
```

Verification steps and logs

Verification of Energy Logserver service

To verify the Energy Logserver service you can use the following commands:

- Control of the Energy Logserver system service via **systemd**:

```
systemctl status logserver
```

output:

```
* logserver.service - Logserver
   Loaded: loaded (/usr/lib/systemd/system/logserver.service; enabled; vendor␣
↪preset: disabled)
   Active: active (running) since Tue 2023-11-14 15:17:16 CET; 5 days ago
 Main PID: 58816 (java)
   CGroup: /system.slice/logserver.service
           └─58816 /etc/alternatives/jre/bin/java -Xshare:auto -Dopensearch.
↪networkaddress.cache.ttl=60 -Dopensearch.networkaddress.cache.n.
```

- Control of Logserver instance via **tcp port**:

```
curl -XGET '127.0.0.1:9200/'
```

output:

```
{
  "name" : "node-1",
  "cluster_name" : "logserver",
  "cluster_uuid" : "B5SDCaaKQU2JdJpsKy6quQ",
  "version" : {
    "distribution" : "opensearch",
    "number" : "3.3.2",
    "build_type" : "tar",
    "build_snapshot" : false
  },
  "tagline" : "The OpenSearch Project: https://opensearch.org/"
}
```

The `version.number` field reports the search engine build, `3.3.2` on Energy Logserver 8.1, not the product version. The response also carries `build_hash`, `build_date`, and the Lucene and compatibility versions, omitted above.

- Control of Logserver instance via **log file**:

```
tail -f /var/log/logserver/logserver.log
```

- other control commands via **curl** application:

```
curl -XGET "http://localhost:9200/_cat/health?v"
curl -XGET "http://localhost:9200/_cat/nodes?v"
curl -XGET "http://localhost:9200/_cat/indices"
```

Verification of Logserver-Probe service

To verify the Logserver-Probe service you can use the following command:

- control Logserver-Probe service via **systemd**:

```
systemctl status logserver-probe
```

output:

```
   logserver-probe.service - logserver-probe
      Loaded: loaded (/usr/lib/systemd/system/logserver-probe.service;␣
```

(continues on next page)

(continued from previous page)

```
↪enabled; vendor preset: disabled)
      Active: active (running) since Mon 2025-01-06 09:15:32 CET; 2 months␣
↪ago
    Main PID: 87818 (java)
      CGroup: /system.slice/logserver-probe.service
              └─87818 /usr/bin/java -Xms1g -Xmx1g
```

- control Logserver-Probe service via **port tcp**:

```
curl -XGET '127.0.0.1:9600'
```

output:

```
{
   "host": "logserver-node",
   "version": "7.17.29",
   "http_address": "127.0.0.1:9600"
}
```

The `version` field reports the pipeline engine bundled with the probe, not the Energy Logserver release.

- control Logserver-Probe service via **log file**:

```
tail -f /var/log/logserver-probe/logserver-probe-plain.log
```

Debugging

- dynamically update logging levels through the logging API (service restart not needed):

```
 curl -XPUT 'localhost:9600/_node/logging?pretty' -H 'Content-Type:␣
↪application/json' -d'
 {
     "logger.logserver-probe.outputs.logserver" : "DEBUG"
 }
 '
```

- permanent change of logging level (service need to be restarted):
  - edit file `/etc/logserver-probe/logserver-probe.yml` and set the following parameter:

```
log.level: debug
```

  - restart logserver-probe service:

```
systemctl restart logserver-probe
```

- checking correct syntax of configuration files:

```
/usr/share/logserver-probe/bin/logstash -tf /etc/logserver-probe/conf.d
```

- get information about load of the Logserver-Probe:

```
curl -XGET '127.0.0.1:9600/_node/jvm?pretty=true'
```

output:

```
{
 "host" : "logserver-node",
 "version" : "7.17.29",
 "http_address" : "127.0.0.1:9600",
 "id" : "5a440edc-1298-4205-a524-68d0d212cd55",
 "name" : "logserver-node",
 "jvm" : {
   "pid" : 14705,
   "version" : "21",
   "vm_version" : "21",
   "vm_vendor" : "Eclipse Adoptium",
   "vm_name" : "OpenJDK 64-Bit Server VM",
   "start_time_in_millis" : 1736150132000,
   "mem" : {
     "heap_init_in_bytes" : 268435456,
     "heap_max_in_bytes" : 1056309248,
     "non_heap_init_in_bytes" : 2555904,
     "non_heap_max_in_bytes" : 0
   },
   "gc_collectors" : [ "G1 Young Generation", "G1 Old Generation" ]
 }
}
```


Verification of Energy Logserver GUI service

To verify the Energy Logserver GUI service you can use the following commands:

- Control the Energy Logserver GUI service via **systemd**:

```
systemctl status logserver-gui
```

output:

```
   * logserver-gui.service - Logserver-GUI
      Loaded: loaded (/usr/lib/systemd/system/logserver-gui.service; enabled;
↪ vendor preset: disabled)
      Active: active (running) since Mon 2025-01-06 09:16:45 CET; 2 months␣
↪ago
    Main PID: 1330 (node)
      CGroup: /system.slice/logserver-gui.service
              └─1330 /usr/share/logserver-gui/bin/../node/bin/node --no-
↪warnings /usr/share/logserver-gui/bin/../src/cli -c /etc/logserver-gui/
↪logserver-gui.yml
```

- Control the Energy Logserver GUI via **port tcp/http**:

```
curl -XGET '127.0.0.1:5601/'
```

output:

```
    <script>var hashRoute = '/app/logserver-gui';
    var defaultRoute = '/app/logserver-gui';
    var hash = window.location.hash;
```

(continues on next page)

(continued from previous page)

```
    if (hash.length) {
      window.location = hashRoute + hash;
    } else {
      window.location = defaultRoute;
    }</script>
```

- Control the Energy Logserver GUI via **log file**:

```
journalctl -fu logserver-gui
```

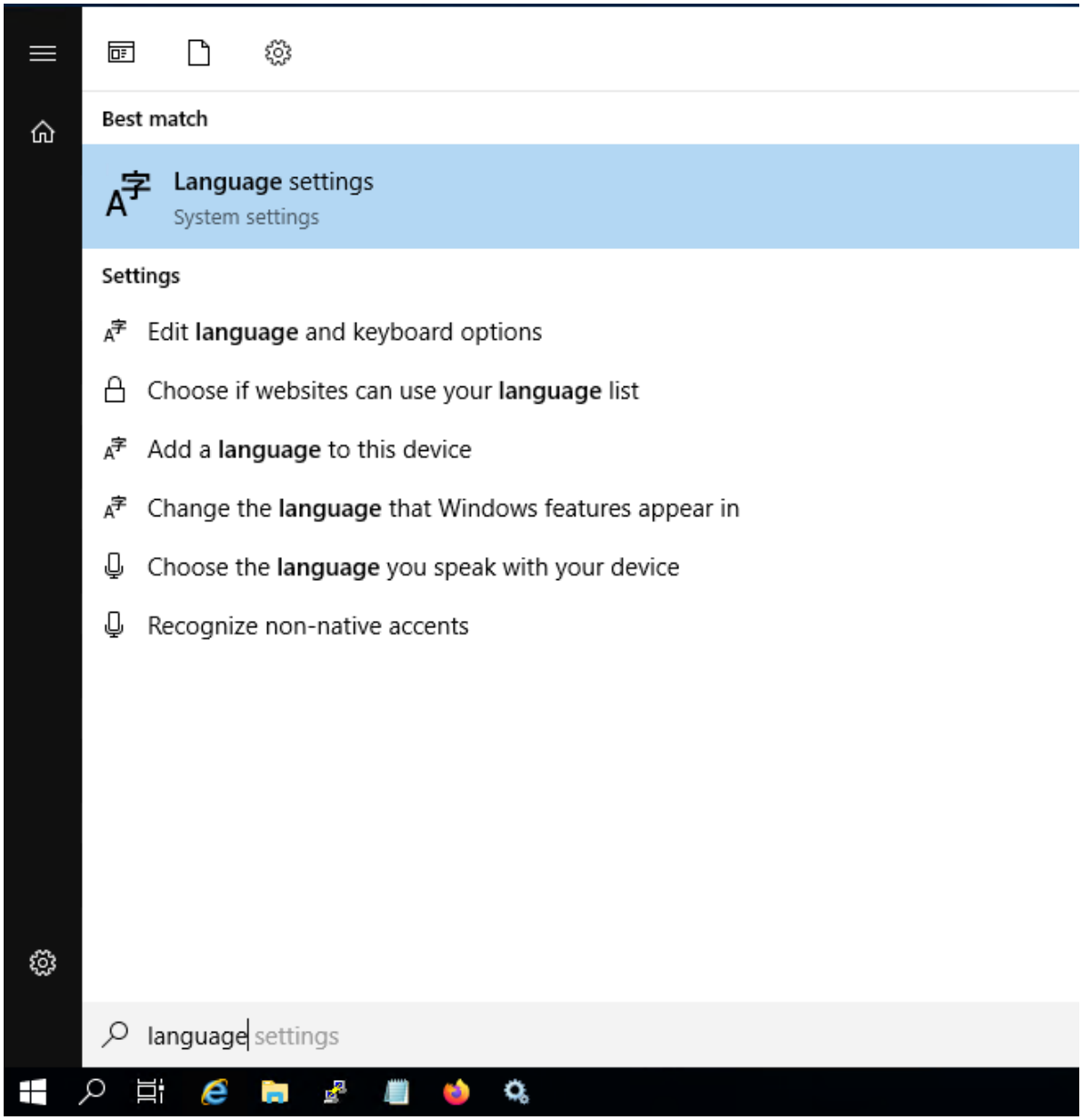
SIEM Agent - Windows CP1250 decoding problem

If the SIEM Agent works on an operating system which uses a non-Latin-script alphabet, the encoding could cause dropping documents by logserver-probe. In the logserver-probe log you can notice lines like the one below.

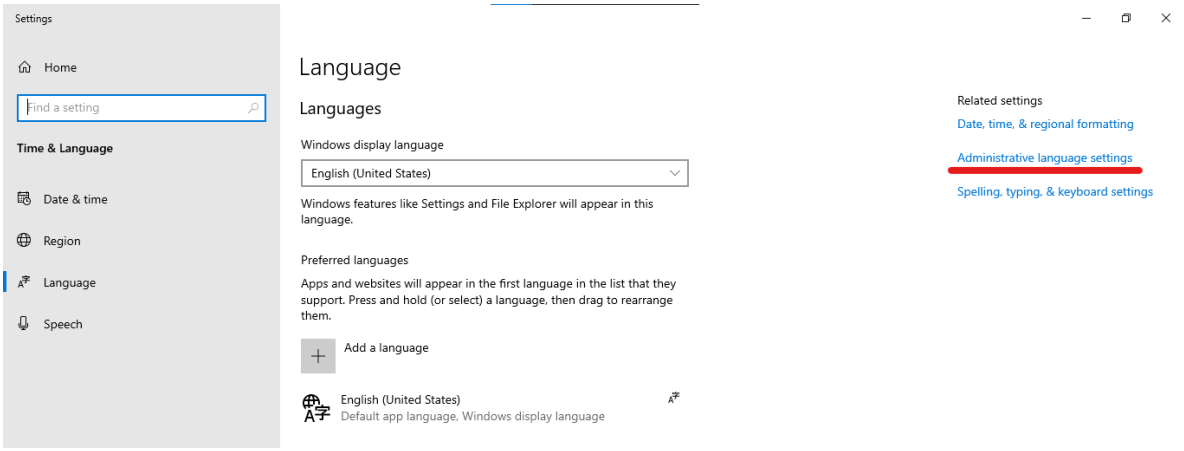
```
[2023-06-01T15:36:02,091][WARN ][logserver-probe.codecs.json     ] Received␣
↪an event that has a different character encoding than you configured.
↪{:text=>"{\\\"timestamp\\\":\\\"2023-06-01T15:36:01.214+0000\\\",\\\"agent\\
↪\":{\\\"id\\\":\\\"002\\\",\\\"name\\\":\\\"win10_laptop\\\"},\\\"manager\\\
↪":{\\\"name\\\":\\\"SiemPlan.local\\\"},\\\"id\\\":\\\"1549035361.0\\\",\\\
↪"full_log\\\":\\\"{\\\\\\\"type\\\\\\\":\\\\\\\"program\\\\\\\",\\\\\\\"ID\\\
↪\\\\\":78741874,\\\\\\\"timestamp\\\\\\\":\\\\\\\"2023/06/01 15:36:00\\\\\\\
↪",\\\\\\\"program\\\\\\\":{\\\\\\\"format\\\\\\\":\\\\\\\"win\\\\\\\",\\\\\\
↪\"name\\\\\\\":\\\\\\\"Skype\\x99 7.34\\\\\\\",\\\\\\\"architecture\\\\\\\
↪":\\\\\\\"i686\\\\\\\",\\\\\\\"version\\\\\\\":\\\\\\\"7.34.102\\\\\\\",\\\\
↪\\\"vendor\\\\\\\":\\\\\\\"Skype Technologies S.A.\\\\\\\",\\\\\\\"install_
↪time\\\\\\\":\\\\\\\"20180212\\\\\\\",\\\\\\\"location\\\\\\\":\\\\\\\"C:\\\
↪\\\\\\\\\\\\\\Program Files (x86)\\\\\\\\\\\\\\\\\\Skype\\\\\\\\\\\\\\\\\\\\\
↪"}}\\\",\\\"decoder\\\":{\\\"name\\\":\\\"syscollector\\\"},\\\"location\\\
↪":\\\"syscollector\\\"}", :expected_charset=>"UTF-8"}
```

This is caused by default Windows encoding `CP1250`. You can change default encoding to `UTF-8` by following this steps:

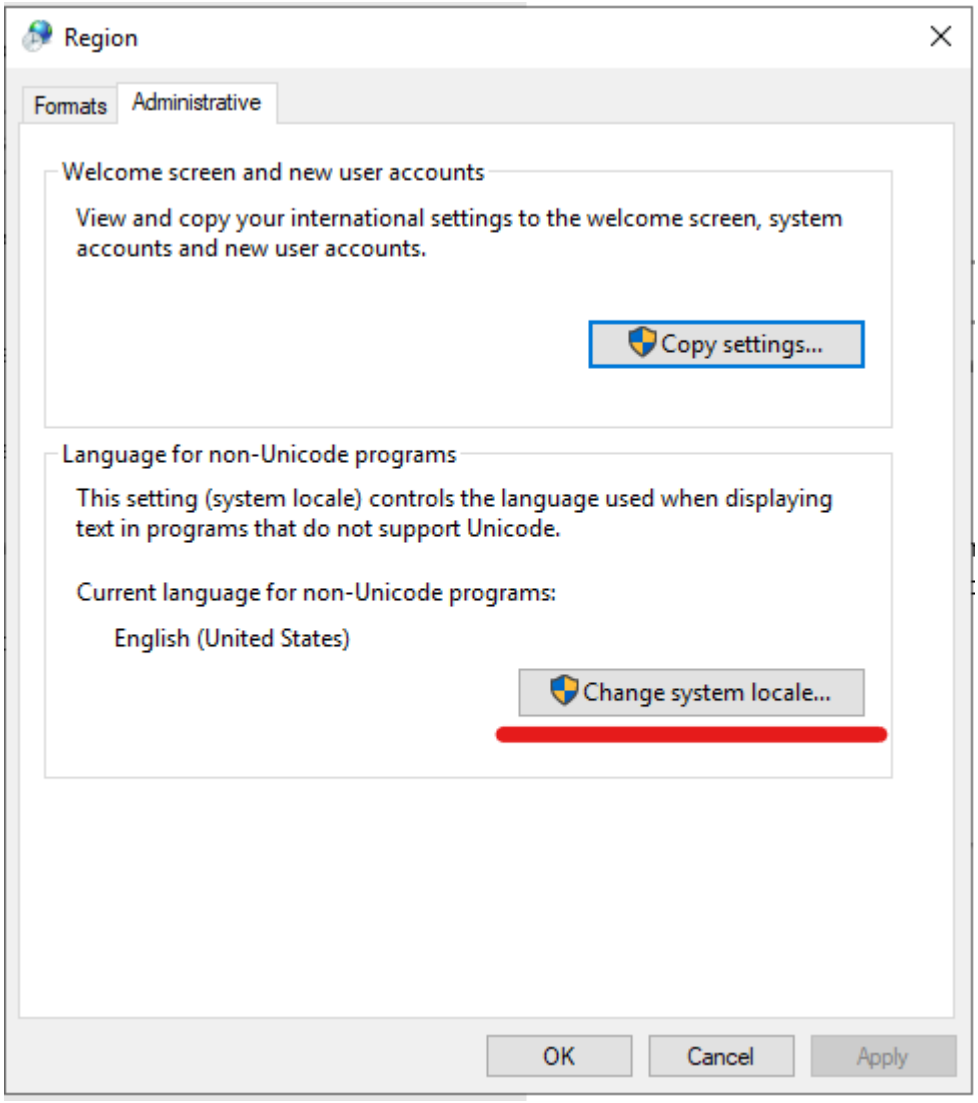
1. Go to Language settings



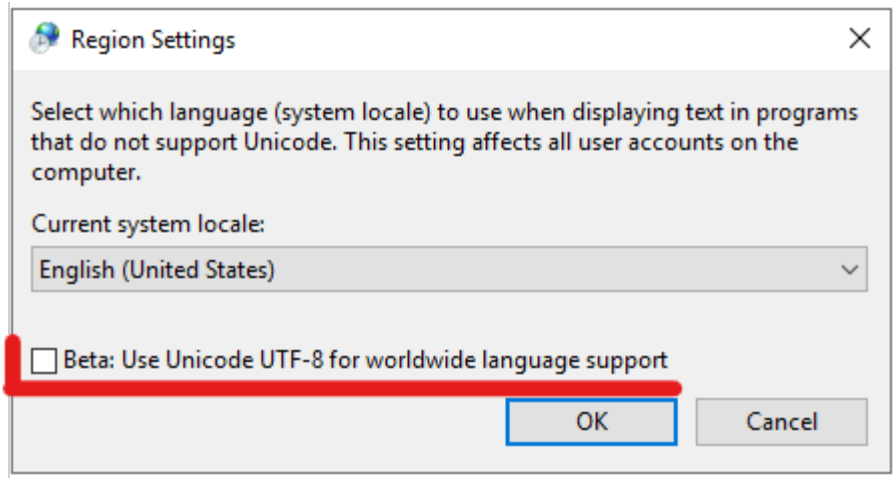
2. Open Administrative language settings



3. Click on `Change system locale...` button



4. Tick the checkbox `Use Unicode UTF-8.`.



5. To make this change active you have to reboot system.

My disk is full

The cluster has reached the flood stage watermark and switched to read-only mode. Non-administrative users cannot log in until the issue is resolved. For background on watermark levels see *Disk-based shard allocation*.

Step 1: free disk space

Remove old indices via **Index Management** in the GUI, or via the API:

```
curl -u $USER:$PASSWORD -XDELETE \
  '127.0.0.1:9200/<index-name>?pretty'
```

To trigger immediate log rotation (`/etc/logrotate.d/logtrail`, 14-day retention):

```
logrotate -f /etc/logrotate.d/logtrail
```

Step 2: unblock indices

When disk usage drops below the high watermark the block releases automatically. To release it immediately without waiting:

```
curl -u $USER:$PASSWORD -XPUT '127.0.0.1:9200/_all/_settings' \
  -H 'Content-Type: application/json' \
  -d '{"index.blocks.read_only_allow_delete": null}'
```

Step 3: verify

```
curl -u $USER:$PASSWORD '127.0.0.1:9200/_cluster/health?pretty'
curl -u $USER:$PASSWORD '127.0.0.1:9200/_cat/nodes?v'
```

Status should return `green` or `yellow`.

4.15 System Configuration

This chapter covers Energy Logserver system-level configuration and management. These settings control core functionality, security, integrations, and operational aspects of your SIEM deployment.

4.15.1 Core Configuration

- **Language Settings**: GUI language and translation management
- **Disk Management**: Shard allocation and storage optimization
- **TLS and Encryption**: Security protocols and certificate management
- **JVM Configuration**: Java Virtual Machine heap sizing and optimization
- **Plugins Management**: Installation and configuration of system plugins
- **Agents**: MasterAgent and Beats agent deployment and management
- **Event Collector**: Agentless collection of Windows events over WS-Man subscriptions
- **GUI Port and HTTPS Redirect**: Web GUI port and the installer's 443→5601 firewalld forward
- **MSSP Configuration**: Multi-tenant source management, licensing tracking, and service provider setup

4.15.2 Access and Security

- **Email and Notifications**: Email and alert notification setup
- **User Lifecycle and RBAC**: User management and role-based access control
- **Domain and 2FA Authentication**: Active Directory, LDAP, Radius, SSO, and two-factor authentication

Language Settings

- *Changing default language for GUI*
  - *Switching the GUI to Polish*
  - *Adding a translation for another language*
- *Preparing translation for GUI*
  - *Bullet points for translations*
  - *FAQ*
  - *Known issues*
- *Settings*
- *General Settings*
- *License (License Info)*
  - *Renew license*
- *Special accounts*

Changing default language for GUI

The GUI ships in English and Polish. The `i18n.locale` setting in `/etc/logserver-gui/logserver-gui.yml` selects the language for the whole GUI instance, so every user who logs in through that node sees the same language. Users cannot pick a language for their own account.

Switching the GUI to Polish

1. On every node that runs the GUI, edit `/etc/logserver-gui/logserver-gui.yml` and set:

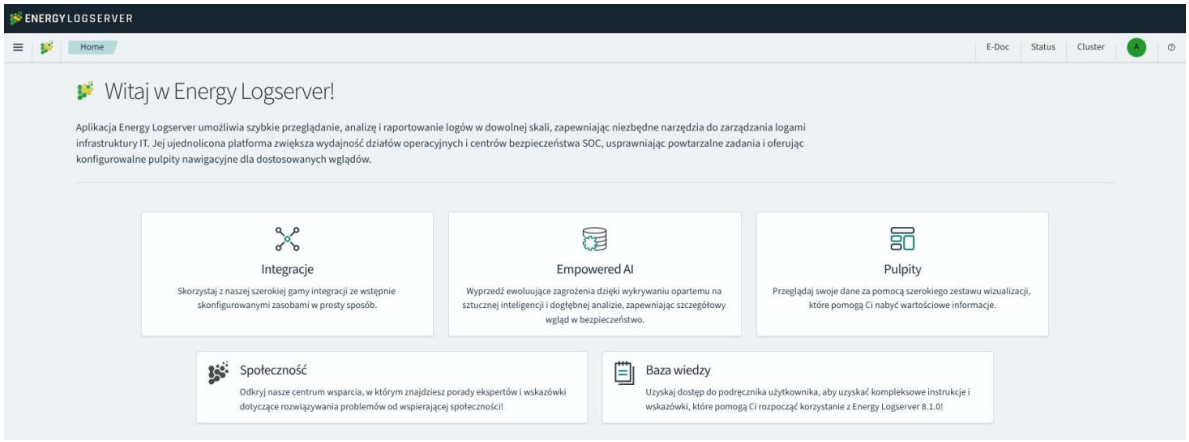
   ```
   i18n.locale: "pl-PL"
   ```

   The file ships with this line commented out as `#i18n.locale: "en-EN"`.

2. Restart the GUI:

   ```
   systemctl restart logserver-gui
   ```

3. Reload the page in the browser. The home page now shows the Polish texts:



To return to English, comment the `i18n.locale` line out again and restart `logserver-gui`.

The Polish translation covers the menus, the home page, Discover, Alerts and Config. A few elements stay in English:

- the login page,
- some values in drop-down lists, such as the rule types and alert methods in Alerts,
- the time picker labels, such as **Last 24 hours**, and the date format.

Adding a translation for another language

1. Prepare the translation file as described in *Preparing translation for GUI* and copy it to `/usr/share/logserver-gui/translations/`, for example as `ja-JP.json`.

2. Add the file name to the `translations` list in `/usr/share/logserver-gui/translations/.i18nrc.json`:

   ```
   {
     "paths": {},
     "exclude": [],
     "translations": [
       "pl-PL.json",
   ```

(continues on next page)

(continued from previous page)

```
    "ja-JP.json"
  ]
}
```

The GUI reads `.i18nrc.json` only from this directory. The package installs it as a configuration file, so an upgrade keeps your edited copy and saves a changed package version next to it as `.i18nrc.json.rpmnew`.

3. Set the owner of the new file:

```
chown logserver-gui:logserver-gui /usr/share/logserver-gui/translations/
↪ja-JP.json
```

4. Set the locale in `/etc/logserver-gui/logserver-gui.yml`:

```
i18n.locale: "ja-JP"
```

5. Restart the GUI with `systemctl restart logserver-gui`.

Preparing translation for GUI

Source file to use as a base for translations: `/usr/share/logserver-gui/translations/en-EN.example.json`

Bullet points for translations

For the translation to work you have to follow these steps.
Omitting some may result in missing translation in some parts of an application or an empty screen when entering a broken portion of an app.

The file with translation is JSON.

Translated values have the following structure:

```
{
    "message": {
      "key.for.the.value": "Translated value for the key"
    }
}
```

Every key is meant to be unique. There can be only one value for each key. In the `messages` object, each key has a "text" value, not a number and not an object.

But there are some structures in a source file that you will use as a base of your translation that have to be addressed during the process to achieve that.

1. Objects

```
{
  "messages": {
    "common.ui.aggTypes.buckets.filtersTitle": {
      "text": "Filters",
      "comment": "The name of an aggregation, that allows to specify␣
↪multiple individual filters to group data by."
    }
  }
}
```

This has to be transformed as described above - a key `common.ui.aggTypes.buckets.filtersTitle` has to have a text assigned to it. The value that needs to be translated is in the fields "text" and "comment" described to you how the value needs to be translated. The result of such will be:

```
{
  "messages":{
    "common.ui.aggTypes.buckets.filtersTitle": "Filtry"
  }
}
```

2. Template variables

```
{
  "messages": {
    "common.ui.aggTypes.buckets.dateHistogramLabel": "{filedName} per
↪{intervalDescription}"
  }
}
```

Any text that is encapsulated in `{}` has to be left as is. Those values are substituted by the application.

3. How to treat complicated structures, eg.: plurals, etc.

```
{
 "messages": {
   "kbn.discover.hitsPluralTitle": "{hits, plural, one {hit} other {hits}}
↪"
 }
}
```

As of now, there is a single example of the above. Contrary to the last point the value in `{}` has to be translated for that key. So `{hit}` and `{hits}`.

4. React compliant filenames

In the application codebase, some methods will take translated keys and substitute them. But many of those will work only if the name of the translation file is one of:

- `en`
- `en-US`
- `en-xa`
- `es`
- `es-LA`
- `fr`
- `fr-FR`
- `de`
- `de-DE`
- `ja`
- `ja-JP`
- `ko`
- `ko-KR`

- zh
- zh-CN
- pl
- ru
- ru-BY
- ru-KG
- ru-KZ
- ru-MD
- ru-UA

FAQ

1. Can I just paste everything into a basic(or advanced) translation software?
   ~No. There are some points to follow for the translation file to work at all.
2. I have the following error - is the application broken:
   - **Error formatting message: A message must be provided as a String or AST**
     ~It is possible you have not followed point 1 - you have left some object structures in your file.
   - **Blank page in GUI**
     ~It is usually caused by not following point 2 -some variable names were changed.
3. I have set "i18n.locale" in the configuration file but the app is not translated.
   ~You may have forgotten to put a reference for your file in `.i18nrc.json` file.

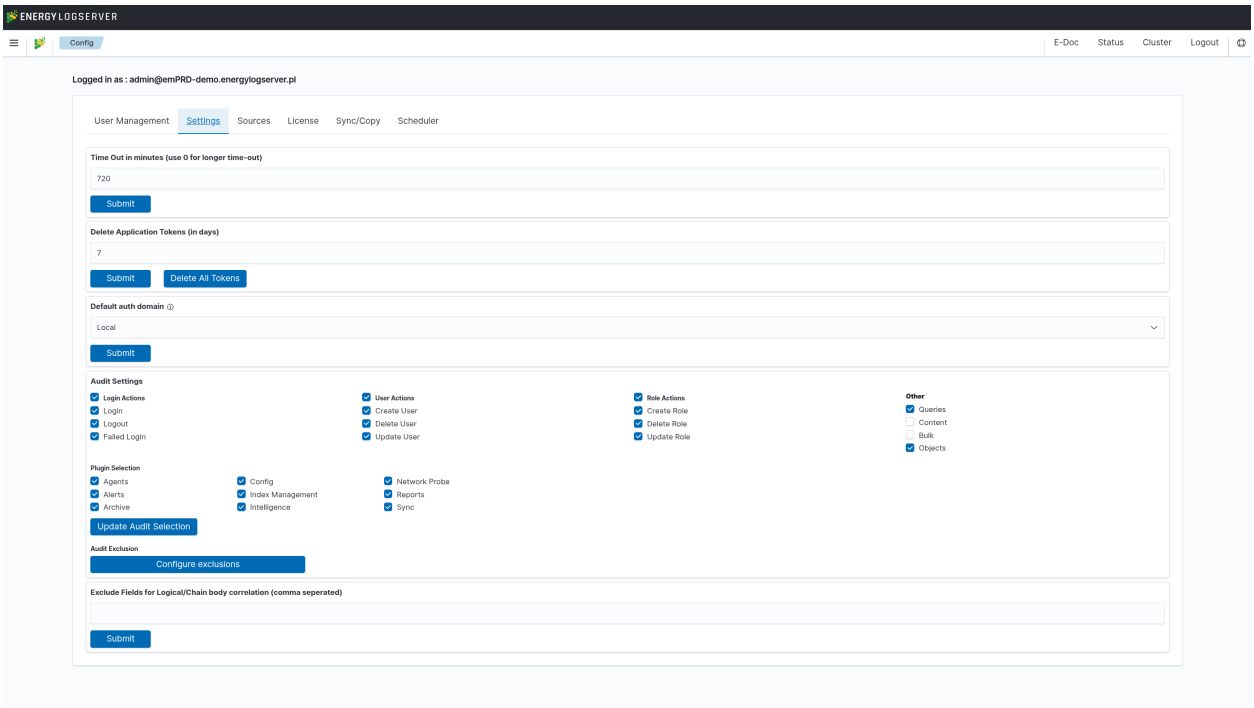
Known issues

1. Some text may not be translated in **Management -> Advanced settings** even though keys for them are present in translation files.
2. The same thing may happen in **Discover -> View surrounding documents**.

Settings

General Settings

The Settings tab is used to set the audit on different activities or events and consists of several fields:



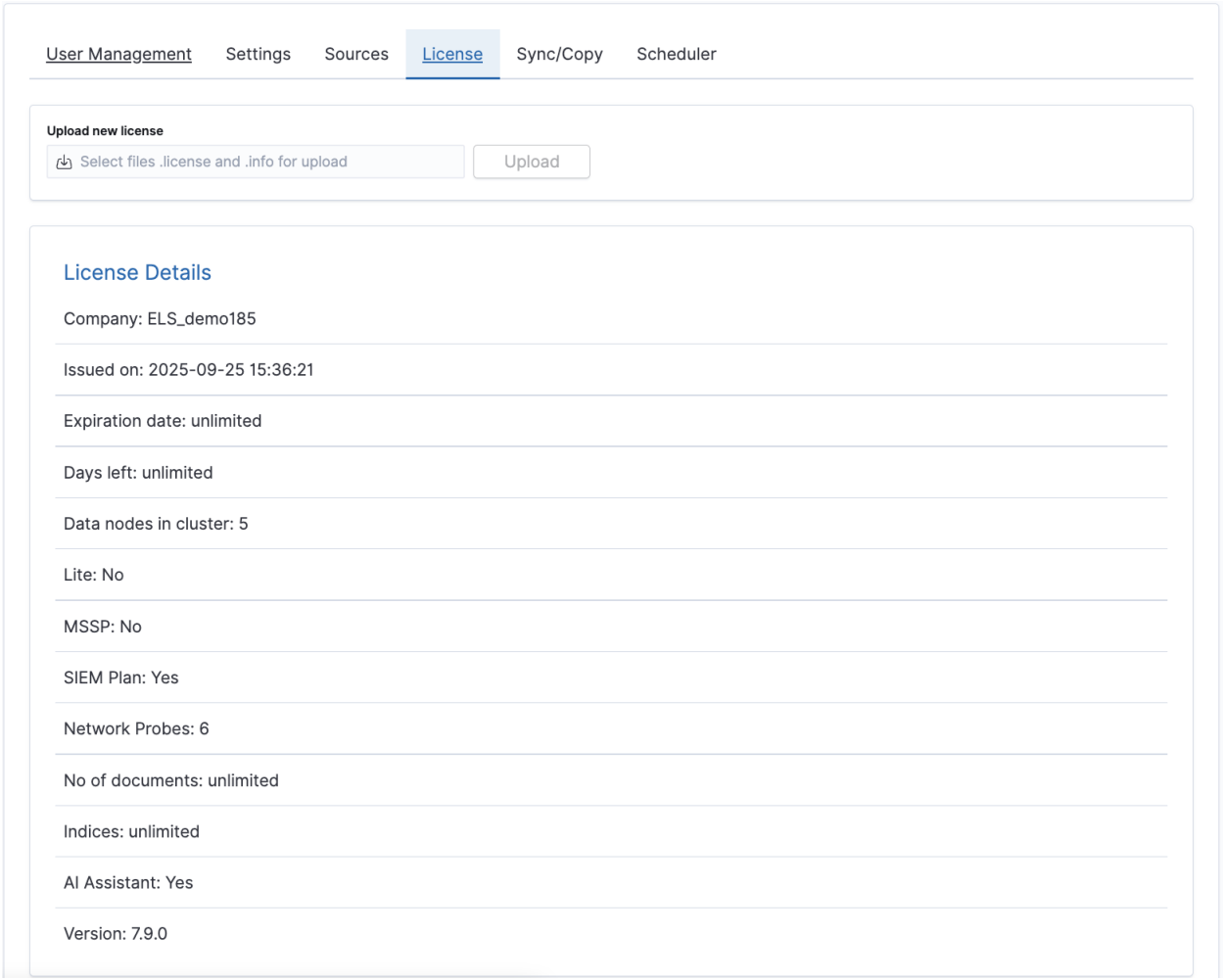
- **Time Out in minutes** field - this field defines the time after how many minutes the application will automatically log you off
- **Delete Application Tokens (in days)** - in this field, we specify after what time the data from the audit should be deleted
- **Delete Audit Data (in days)** field - in this field, we specify after what time the data from the audit should be deleted
- The next fields are checkboxes in which we specify what kind of events are to be logged (saved) in the audit index. The events that can be monitored are: logging (Login), logging out (Logout), creating a user (Create User), deleting a user (Delete User), updating user (Update User), creating a role (Create Role), deleting a role (Delete Role), update of the role (Update Role), start of export (Export Start), delete of export (Export Delete), queries (Queries), result of the query (Content), if attempt was made to perform a series of operation (Bulk)
- **Delete Exported CSVs (in days)** field - in this field, we specify after which time exported files with CSV extension have to be removed
- **Delete Exported PDFs (in days)** field - in this field, we specify after which time exported files with PDF extension have to be removed

Each field is assigned the "Submit" button thanks to which we can confirm the changes.

License (License Info)

The License Information tab consists of several non-editable information fields.

Also, if you are logged in as an administrator you will be able to upload new license from here.



These fields contain information:

- Company - who owns the license, in this case, Foo Bar.
- Issued on - license creation date
- Expiration date - when the license will expire
- Days left - how many days left before it will expire
- Data nodes in cluster - how many nodes we can put in one cluster - in this case, 10
- SIEM Plan enabled - is SIEM Plan covered by the license
- Network Probes - count of NetworkProbes covered by the license
- No of documents - license can be limited by docs count
- Indices - license can be limited by index pattern names
- Version - version of the product

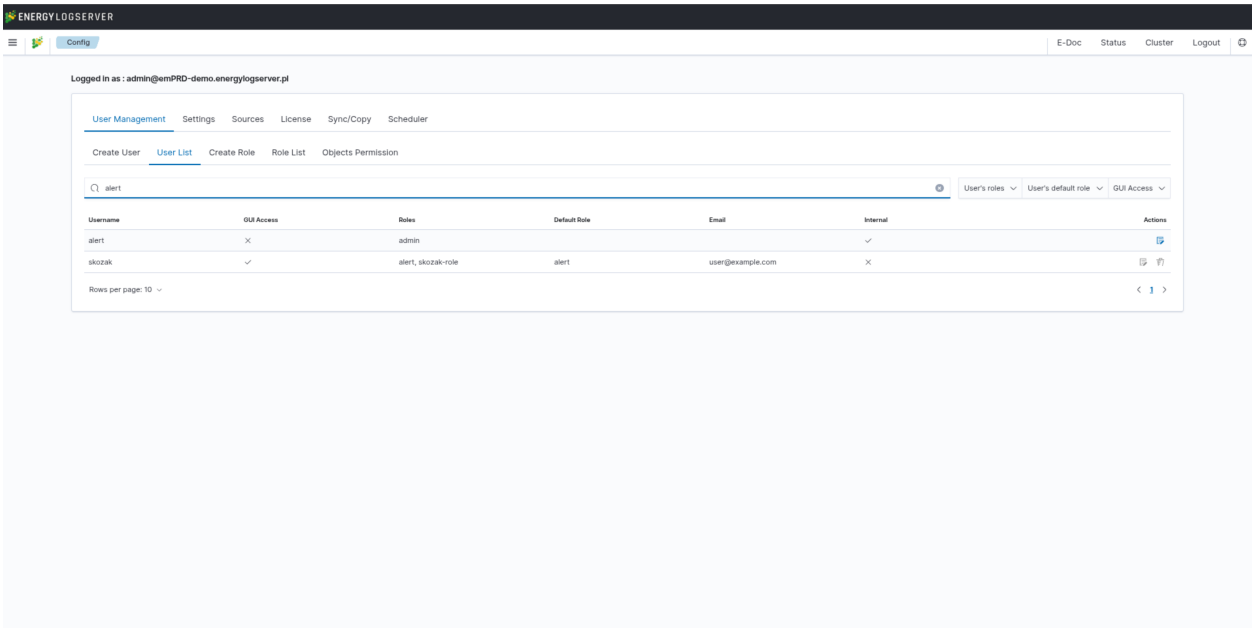
Renew license

To change the Energy Logserver license files on a running system, use the "Upload new license" option at the top of the "License" tab.

1. Select the provided license files, e.g.: es_123.info, es_123.license.
2. Click upload.
3. Files will be uploaded and verified. New license information will appear.
4. Verify new license information and click "Submit" when ready.
5. It will install itself in the cluster - no need to manually put it on all nodes.

Special accounts

Apart from the administrative account, the installation creates a service account for each component that talks to the database. They are marked with the Internal flag in the User List and cannot be deleted. Find them under **Management** → **Config** → **User Management** → **User List**, filtering by account name.



- **logserver** - used by the GUI, Cluster Console, Skimmer, and the Network Probe output
- **alert** - used by the Alert engine, which evaluates rules against the incoming data and fires the configured notification method, see *Alerting System*
- **intelligence** - used by the Empowered AI services, both `intelligence` and `intelligence-scheduler`, see *AI and Analytics*
- **e-doc** - used by the wiki portal for its SSO callbacks and search index
- **license** - used by the license service
- **probe** - used by the Network Probe

The passwords of these accounts live in a central keystore and are rotated with `logserver-password-util.sh`, described in *Change Default Credentials*.

Disk-based shard allocation

Since version **7.4.3** of Logserver a feature was added to keep you informed about your disk usage. Disk-based shard allocation settings are used to determine the GUI behavior.



These labels will appear when a certain watermark level is exceeded. Its function is to inform you to take action. Maybe tweak your retention settings or add disk space to your servers.

With the "FLOOD STAGE", non-administrative users will lose the ability to log into GUI, and those already logged in will be redirected to the status page. An administrator will still be able to browse and will have a chance to take action to fix the situation.

This is because with "FLOOD STAGE", the system switches into a read-only mode. No new documents can be created or updated.

What is a watermark

1. Low stage

   Controls the low watermark for disk usage. When exceeded Logserver will not allocate shards to that node. This setting does not affect the primary shards of newly-created indices but will prevent their replicas from being allocated.

2. High stage

   Controls the high watermark. When exceeded Logserver will attempt to relocate shards away from a node whose disk exceeds it. This setting affects the allocation of all shards, whether previously allocated or not.

3. Flood stage

   This setting is a last resort to prevent nodes from running out of disk space - which should be at all costs avoided.

   Controls the flood stage watermark. Logserver enforces a read-only index block (index.blocks.read_only_allow_delete) on every index that has one or more shards allocated on the node, and that has at least one disk exceeding the flood stage. The index block is automatically released when the disk utilization falls below the high watermark.

Even if you are running Logserver in a single node environment, configuring the above is highly recommended.

While the GUI will always inform you about your disk usage, to take advantage of the above benefits you will need to configure the thresholds yourself **especially if you are updating from an earlier Logserver version**.

Configuring watermark stages

1. On fresh **7.4.3** or newer installations of Logserver these are the default watermark values:
   - low: 95%
   - high: 97%
   - flood: 99%
2. To change them, you can use Logserver API like in the following example:

```
curl localhost:9200/_cluster/settings -XPUT -H 'content-type: application/
↪json' -d '{
  "persistent": {
    "cluster.routing.allocation.disk.threshold_enabled": "true",
    "cluster.routing.allocation.disk.watermark.flood_stage": "10gb",
    "cluster.routing.allocation.disk.watermark.high": "25gb",
    "cluster.routing.allocation.disk.watermark.low": "50gb"
  }
}'
```

With that the "low stage" starts when the disk space for Logserver indices falls below the 50 GB mark; "high" below 25 GB and "flood" below 10 GB.

3. You can also use percentages for that **but not both - you cannot mix byte values and percentages**.
4. You can also modify Logserver configuration file `/etc/logserver/logserver.yml`:

```
cluster.routing.allocation.disk.threshold_enabled: true
cluster.routing.allocation.disk.watermark.low: "50gb"
cluster.routing.allocation.disk.watermark.high: "25gb"
cluster.routing.allocation.disk.watermark.flood_stage: "10gb"
```

Take in mind you will have to change those values on all of your cluster nodes this way. Restart is not required as API was used earlier.

Important! Thanks to that, settings will be protected against a state failure.

Flood stage and read-only mode

When the cluster reaches the "flood stage", it is automatically put into a read-only mode. It prevents a cluster failure - **reaching 100% disk usage on a Logserver cluster is a critical issue!**

The best way to remove the read-only mode is to make more disk space available for the cluster - either by adding more space or removing data (indices) from it. When the cluster reaches the "low stage" or below, it will automatically switch from the read-only mode.

Adjusting the levels with the above API examples to suit your installation is another option.

Disabling the threshold checking is not recommended! Also, changing the "threshold_enabled" setting to "false" does not turn the read-only mode off automatically!

For step-by-step recovery when flood stage is already active, see *Troubleshooting, My disk is full*.

Transport layer encryption

Transport layer encryption secures communication between Energy Logserver components. This chapter covers certificate generation with the bundled TLS Tool, certificate deployment on the Data Node, and the TLS settings of the remaining components.

- *Offline TLS Tool*
- *General usage*
- *Command line options*
- *Examples*

- *Root CA*
- *Intermediate CA*
- *Node and Client certificates*
- *Admin and client certificates*
- *Documentation link*
- *Adding certificates after the first run*
- *Creating CSRs*
- *Configuring ciphers for TLS settings*

Offline TLS Tool

The TLS Tool is a program that can be used for:

- Generating Root and Intermediate CA's,
- Generating Node, Client, and Admin certificates,
- Generating CSRs,
- Validating certificates

Besides the actual certificates the tool also generated configuration snippets which you can directly copy and paste into your Data Node main .yml file.

General usage

The tls tool will read the node and certificate configuration settings from a yaml file, and outputs the generated files in a configurable directory.

You can choose to create the Root CA and (optional) intermediate CAs with your node certificates in one go. Or you can create the Root and intermediate CA first, and generate node certificates as you need them.

You will find the script in:

```
<installation directory>/tlstool.sh
```

Command line options

Table 8: TLS Command Line Options

Name	Description
-c,–config	Relative or absolute path to the configuration file. Required.
-t,–target	Relative or absolute path to the output directory. Optional, default: out
-v,–verbose	Enable detailed output, default: false
-f,–force	Force certificate generation despite of validation errors. default: false
-o,–overwrite	Overwrite existing node-, client and admin certificates if they are already present. default: false
-p,–pem	Change certificate file extension from ".crt" to ".pem". default: false
-ca,–create-ca	Create new Root and Intermediate CAs
-crt,–create-cert	Create certificates using an existing or newly created local certificate authority

Examples

```
<installation directory>/tlstool.sh -c config/logserver-cluster.yml -ca -crt
```

Reads the configuration from `config/logserver-cluster.yml` and generates the configured Root and intermediate CAs and the configured node, admin, and client certificates in one go. The generated files will be written to `out`.

```
<installation directory>/tlstool.sh -c config/logserver-cluster.yml -ca
```

Reads the configuration from `config/logserver-cluster.yml` and generates the configured Root and intermediate CAs only.

```
<installation directory>/tlstool.sh -c config/logserver-cluster.yml -crt
```

Reads the configuration from `config/logserver-cluster.yml` and generates node, admin, and client certificates only. The Root and (optional) intermediate CA certificates and keys need to be present in the output directory, and their filenames, keys and (optional) passwords have to be configured in `logserver-cluster.yml`.

Root CA

To configure the Root CA for all certificates, add the following lines to your configuration file:

```
ca:
   root:
      dn: CN=root.ca.example.com,OU=CA,O=Example Com, Inc.,DC=example,DC=com
      keysize: 2048
      pkPassword: root-ca-password
      validityDays: 3650
      file: root-ca.crt
```

Generated files:

- `root-ca.crt` - Root certificate
- `root-ca.key` - Private key of the Root CA
- `root-ca.readme` - Auto-generated passwords of the root and intermediate CAs

Options:

Table 9: Root CA Configuration Options

Name	Description
dn	The complete Distinguished Name of the Root CA. If you have special characters in the DN, you need to quote them correctly. Mandatory.
keysize	The size of the private key. Default: 2048
pkPassword	Password of the private key. One of: “none”, “auto” or a self-chosen password. Default: auto
file	File name of the certificate, optional. Default: “root-ca”

The pkPassword can be one of:

- **none**: The generated private key will be unencrypted
- **auto**: A random password is generated automatically. After the certificates have been generated, you can find the password in root-ca.readme file. To use these new passwords again, you must edit the tool config file and set the generated passwords there.
- **other value**: Values other than none or auto are used as password directly

Intermediate CA

In addition to the root CA you optionally also specify an intermediate CA. If an intermediate CA is configured, then the node, admin, and client certificates will be signed by the intermediate CA. If you do not want to use an intermediate CA, remove the following section from the configuration. The certificates are then signed by the root CA directly.

```
ca:
   intermediate:
      dn: CN=signing.ca.example.com,OU=CA,O=Example Com, Inc.,DC=example,
↪DC=com
      keysize: 2048
      validityDays: 3650
      pkPassword: intermediate-ca-password
      file: intermediate-ca.crt
```

Generated files:

- `intermediate-ca.crt` - Intermediate certificate
- `intermediate-ca.key` - Private key of the intermediate certificate
- `root-ca.readme` - Auto-generated passwords of the root and intermediate CAs

Node and Client certificates

Global and default settings

The default settings are applied to all generated certificates and configuration snippets. All values here are optional.

```
defaults:
  validityDays: 730
  pkPassword: auto
  generatedPasswordLength: 12
  nodesDn:
    - "CN=*.example.com,OU=Ops,O=Example Com, Inc.,DC=example,DC=com"
  nodeOid: "1.2.3.4.5.5"
  httpsEnabled: true
  reuseTransportCertificatesForHttp: false
```

Options:

Table 10: TLS Configuration Parameters

Parameter	Description
validityDays	Validity of the generated certificates, in days. Default: 730. Can be overwritten for each certificate separately.
pkPassword	Password of the private key. One of: "none", "auto" or a self-chosen password. Default: auto. Can be overwritten for each certificate separately.
generatedPasswordLength	Length of the auto-generated password for the private keys. Only takes effect when pkPassword is set to auto. Default: 12. Can be overwritten for each certificate separately.
httpsEnabled	Whether to enable TLS on the REST layer or not. Default: true
reuseTransportCertificates-ForHttp	If set to false, individual certificates for REST and Transport are generated. If set to true, the node certificates are also used on the REST layer. Default: false
verifyHostnames	Set this to true to enable hostname verification. Default: false
resolveHostnames	Set this to true to resolve hostnames against DNS. Default: false
signatureAlgorithm	Signature algorithm (for RSA default is SHA256withRSA, for EC its SHA256withECDSA)
useEllipticCurves	If set to true use elliptic curves instead of RSA. Default is false (since 1.7)
ellipticCurve	The curve to be used. Default is: NIST P-384 (since 1.7)

Node certificates

To generate node certificates, add the node name, the Distinguished Name, the hostname(s), and/or the IP address(es) in the `nodes` section:

```
nodes:
  - name: node1
    dn: CN=test.example.com,OU=Dev,O=Energy Software,C=Poland
    dns: test.example.com
    ip: 10.0.2.1
  - name: node2
    dn: CN=node2.example.com,OU=Dev,O=Energy Software,C=Poland
    dns:
      - node2.example.com
      - es2.example.com
    ip:
      - 10.0.2.1
      - 192.168.2.1
  - name: node3
    dn: CN=node3.example.com,OU=Dev,O=Energy Software,C=Poland
    dns: node3.example.com
```

Generated files:

- `[nodename].crt` - Node certificate
- `[nodename].key` - Private key of the node certificate
- `[nodename]_http.crt` - REST certificate, only generated if reuseTransportCertificatesForHttp is false
- `[nodename]_logserver_config_snippet.yml` - Logserver Guard configuration snippet for this node, add this to logserver.yml

Options:

Table 11: Node Certificate Options

Name	Description
name	Name of the node, will become part of the filenames. Mandatory
dn	The Distinguished Name of the certificate. If you have special characters in the DN, you need to quote them correctly. Mandatory
dns	The hostname(s) this certificate is valid for. Should match the hostname of the node. Optional, but recommended.
ip	The IP(s) this certificate is valid for. Optional. Prefer hostnames if possible.

Admin and client certificates

To generate admin and client certificates, add the following lines to the configuration file:

```
clients:
  - name: spock
    dn: CN=spock.example.com,OU=Dev,O=Energy Software,C=Poland
  - name: kirk
    dn: CN=kirk.example.com,OU=Dev,O=Energy Software,C=Poland
    admin: true
```

Generated files:

- `[name].crt` - Client certificate
- `[name].key` - Private key of the client certificate
- `client-certificates.readme` - Contains the auto-generated passwords for the certificates

Options:

Table 12: Client Certificate Options

Name	Description
name	Name of the certificate, will become part of the file name
dn	The complete Distinguished Name of the certificate. If you have special characters in the DN, you need to quote them correctly
admin	If set to true, this certificate will be marked as an admin certificate in the generated configuration snippet.

Note that you need to mark at least one client certificate as an admin certificate.

Documentation link

To update the link that is inserted into the generated readme files:

`documentationLink: "https://link-to-docs.com"`

Options:

Table 13: Documentation Link Option

Name	Description
documentationLink	Link to the site. Optional.

Adding certificates after the first run

You can always add more node- or admin certificates as you need them after the initial run of the tool. As a precondition

- the root CA and, if used, the intermediate certificates and keys must be present in the output folder
- the password of the root CA and, if used, the intermediate CA must be present in the config file

If you use auto-generated passwords, copy them from the generated `root-ca.readme` file to the configuration file.

Certificates that have already been generated in a previous run of the tool will be left untouched unless you run the tool with the `-o, --overwrite` switch. In this case, existing files are overwritten. If you have chosen to auto-generate passwords, new keys with auto-generated passwords are created.

Creating CSRs

If you just want to create CSRs to submit them to your local CA, you can omit the CA part of the config complete. Just define the `default`, `node`, and `client` section, and run the TLS tool with the `-csr` switch.

Configuring ciphers for TLS settings

How to enable, disable and/or limit available ciphers for Logserver services.

Logserver Data Node

Edit config file `/etc/logserver/logserver.yml`.

Example:

```
logserverguard.ssl.transport.enabled_ciphers:
- TLS_AES_256_GCM_SHA384
- TLS_AES_128_GCM_SHA256
- TLS_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

logserverguard.ssl.http.enabled_ciphers:
- TLS_AES_256_GCM_SHA384
- TLS_AES_128_GCM_SHA256
- TLS_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
```

After editing restart Data Node service. Note that this settings should be updated on all cluster nodes.

Logserver GUI

Edit config file `/etc/logserver-gui/logserver-gui.yml`.

Example:

```
server.ssl.cipherSuites:
- TLS_AES_256_GCM_SHA384
- TLS_AES_128_GCM_SHA256
- TLS_CHACHA20_POLY1305_SHA256
- ECDHE-ECDSA-AES256-GCM-SHA384
- ECDHE-RSA-AES256-GCM-SHA384
- ECDHE-ECDSA-CHACHA20-POLY1305
- ECDHE-RSA-CHACHA20-POLY1305
- ECDHE-ECDSA-AES128-GCM-SHA256
- ECDHE-RSA-AES128-GCM-SHA256
```

After editing restart Logserver GUI service

Cerebro

Since 7.6.1 there is a default configuration for enabled algorithms on path `/opt/cerebro/conf/java.config`.

You can edit it and after restarting cerebro.service, changes will be applied.

7.6.0 and before

1. To recreate current default behavior, create file `/opt/cerebro/conf/java.config` with content:

```
security.useSystemPropertiesFile=false
jdk.tls.ephemeralDHKeySize=2048
jdk.certpath.disabledAlgorithms=MD2, SHA1, MD5, DSA, RSA keySize < 2048
jdk.tls.disabledAlgorithms=DH keySize < 2048, TLSv1.1, TLSv1, SSLv3,␣
↪SSLv2, TLS_RSA_WITH_AES_256_CBC_SHA256, TLS_RSA_WITH_AES_256_CBC_SHA,␣
↪TLS_RSA_WITH_AES_128_CBC_SHA256, TLS_RSA_WITH_AES_128_CBC_SHA, TLS_RSA_
↪WITH_AES_256_GCM_SHA384, TLS_RSA_WITH_AES_128_GCM_SHA256, DHE_DSS, RSA_
↪EXPORT, DHE_DSS_EXPORT, DHE_RSA_EXPORT, DH_DSS_EXPORT, DH_RSA_EXPORT,␣
↪DH_anon, ECDH_anon, DH_RSA, DH_DSS, ECDH, 3DES_EDE_CBC, DES_CBC, RC4_40,
↪ RC4_128, DES40_CBC, RC2, HmacMD5, TLS_DHE_RSA_WITH_AES_128_CBC_SHA,␣
↪TLS_DHE_RSA_WITH_AES_256_CBC_SHA
jdk.tls.legacyAlgorithms=
```

2. Edit `ExecStart` definition of `/usr/lib/systemd/system/cerebro.service` like so:

```
ExecStart=/opt/cerebro/bin/cerebro "-Djava.security.properties=/opt/
↪cerebro/conf/java.config" "-Dconfig.file=/opt/cerebro/conf/application.
↪conf"
```

3. Reload systemd and restart cerebro.service

```
systemctl daemon-reload
systemctl reenable cerebro
systemctl restart cerebro
```

E-doc

Since 7.6.1 there is a default configuration for enabled ciphers on path `/opt/e-doc/env.config`.

You can edit it and after restarting e-doc.service, changes will be applied.

7.6.0 and before (E-doc)

1. To recreate current default behavior, create file `/opt/e-doc/env.config` with content:

```
NODE_OPTIONS=--tls-cipher-list='TLS_AES_256_GCM_SHA384:TLS_CHACHA20_
↪POLY1305_SHA256:TLS_AES_128_GCM_SHA256:ECDHE-RSA-AES128-GCM-
↪SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-
↪ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-
↪SHA256:DHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES256-
↪SHA384:ECDHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA256:!aNULL:!eNULL:!
↪EXPORT:!DES:!RC4:!MD5:!PSK:!SRP:!CAMELLIA'
```

2. Edit `/usr/lib/systemd/system/e-doc.service`. Add following under `[Service]` section:

```
EnvironmentFile=-/opt/e-doc/env.config
```

3. Reload systemd and restart e-doc.service

```
systemctl daemon-reload
systemctl reenable e-doc
systemctl restart e-doc
```

JVM Configuration

Data Node

The JVM configuration for the Data Node is stored in:

- `/etc/logserver/jvm.options` - main JVM options file (overwritten during upgrades, do not edit)
- `/etc/logserver/jvm.options.d/` - directory for user-defined JVM settings (persistent across upgrades)

By default, Data Node tells the JVM to use a heap with a minimum (Xms) and maximum (Xmx) size of 2 GB. When moving to production, it is important to configure heap size to ensure that Data Node has enough heap available.

Heap Sizing

Heap size settings are stored in `/etc/logserver/jvm.options.d/heap-size.options`. Edit this file to set the `-Xms` and `-Xmx` values:

```
-Xms16g
-Xmx16g
```

> **Warning**
>
> Do not place custom JVM settings in `/etc/logserver/jvm.options` directly. This file is overwritten during upgrades. All user-defined settings must go in `/etc/logserver/jvm.options.d/`.

Guidelines:

- Always set `-Xms` equal to `-Xmx` to prevent heap resizing at runtime.
- Do not exceed 31GB to maintain JVM compressed ordinary object pointers (compressed oops), which significantly improve memory efficiency.
- As a general rule, allocate no more than half of the available system RAM to the JVM heap, leaving the rest for the operating system and file system cache.

Warning

Setting heap larger than the available system RAM causes the Linux OOM killer to terminate the `logserver` process repeatedly. The service restarts automatically but the cycle continues until the heap is corrected. A typical symptom is a restart loop every 60-120 seconds with no application-level error logged.

To check for OOM kills:

```
dmesg | grep -i "killed process\|out of memory"
journalctl -u logserver | grep -i "oom\|killed"
```

After changing the heap size, restart the Data Node service:

```
systemctl restart logserver
```

Sizing Guidelines

Right-size the heap to the cluster workload, not just the host RAM:

- **Shards per heap:** keep at most ~25 active shards per 1 GB of JVM heap. A node with 16 GB heap can comfortably host ~400 active shards; beyond that, metadata overhead and GC pressure grow quickly.
- **Host RAM vs heap:** reserve at least 2× the heap size in system RAM. The OS page cache is what keeps search and aggregation latency low on cold data, so giving the JVM everything is counterproductive.
- **Ballpark tiers:**

Node role	Typical RAM	Heap
Small ingest / single	16 GB	8 GB
Standard data node	64 GB	31 GB
Hot node, heavy queries	128 GB+	31 GB

  The 31 GB cap stays in place regardless of host size, the only way to use more memory effectively above that point is more nodes, not bigger heap.

For OpenSearch sizing references see the upstream Sizing Guidance.

Additional JVM Options

You can place additional JVM configuration files in `/etc/logserver/jvm.options.d/`. Each file should contain one JVM option per line. This is used, for example, for SSO configuration (see *Authentication and Directory Integration*).

Network Probe

The JVM configuration for the Network Probe is stored in:

- `/etc/logserver-probe/jvm.options`

To apply TLS-related Java security properties, make sure the following definition is present in `/etc/logserver-probe/jvm.options`:

```
-Djava.security.properties=/etc/logserver-probe/java.properties
```

After changing JVM options, restart the Network Probe service:

```
systemctl restart logserver-probe
```

Plugins Management

Plugin Management

Energy Logserver SIEM uses a comprehensive plugin architecture to provide extensible functionality. Plugins are categorized into GUI plugins for the ELS Console and Data Node plugins for the core engine.

- *GUI*
  - *Enabling/Disabling Plugins*
  - *Installing Plugins*
  - *Listing plugins*
  - *Removing plugins*
  - *Updating plugins*
- *Data Node*
  - *Enabling/Disabling Plugins (Data Node)*
  - *Installing Plugins (Data Node)*
  - *Listing Plugins (Data Node)*
  - *Removing Plugins (Data Node)*
  - *Updating Plugins (Data Node)*

GUI

Base installation of the Energy Logserver SIEM contains the Agents, Alerts, Archive, Automation, CMDB, Index Management, Intelligence, Network Probe, Reports, SQL plugins - These add-ons can be disabled or enabled via the configuration file without having to install or uninstall. You can extend the basic GUI functionality by installing custom plugins.

After installation, each node must be restarted before the plugin becomes visible.

The Logserver GUI provides three categories of plugins:

- Licenced Plugins - Energy Logserver SIEM
- Core Plugins - it is plugins that are part of the core project.

- Community-contributed - it is plugins that are external to the core project

Enabling/Disabling Plugins

Managing the `Agents` Plugin:

- **Disable**:
  - Add `agents.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `agents.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `Alerts` Plugin:

- **Disable**:
  - Change `alerts.enabled: true` to `alerts.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Change `alerts.enabled: false` to `alerts.enabled: true` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `Archive` Plugin:

- **Disable**:
  - Add `archive.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `archive.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `Automation` Plugin:

- **Disable**:
  - Change `automation.enabled: true` to `automation.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Change `automation.enabled: false` to `automation.enabled: true` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `CMDB` Plugin:

- **Disable**:
  - Add `cmdb.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `cmdb.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `Console` Plugin:

- **Disable**:
  - Change `console.enabled: true` to `console.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Change `console.enabled: false` to `console.enabled: true` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `Index Management` Plugin:

- **Disable**:
  - Add `index_management.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `index_management.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `Intelligence` Plugin:

- **Disable**:
  - Add `intelligence.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `intelligence.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `Network Probe` Plugin:

- **Disable**:
  - Add `network-probe.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

- **Enable**:
  - Remove or comment out the line `network-probe.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `Reports` Plugin:

- **Disable**:
  - Add `reports.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `reports.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `vis_type_timeline` Plugin:

- **Disable**:
  - Change `vis_type_timeline.enabled: true` to `vis_type_timeline.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Change `vis_type_timeline.enabled: false` to `vis_type_timeline.enabled: true` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `XLSX Import` Plugin:

- **Disable**:
  - Add `xlsx_import.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `xlsx_import.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Managing the `SQL` Plugin:

- **Disable**:
  - Add `sql.enabled: false` to the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `sql.enabled: false` in the file `/etc/logserver-gui/logserver-gui.yml`.
  - Run the command systemctl restart.

Installing Plugins

Additional GUI plugins can be installed as follows. Run the tool as the `logserver-gui` user, because it refuses to run as root:

```
cd /usr/share/logserver-gui/
sudo -u logserver-gui bin/logserver-gui-plugin install [plugin_name]
```

Examples:
Plugins from a custom link or filesystem can be installed as follows:

```
sudo -u logserver-gui bin/logserver-gui-plugin install file:///path/to/
↪plugin.zip
sudo -u logserver-gui bin/logserver-gui-plugin install file:///C:/path/to/
↪plugin.zip
sudo -u logserver-gui bin/logserver-gui-plugin install http://some.domain/
↪path/to/plugin.zip
```

Listing plugins

Listing currently loaded plugins:

```
sudo -u logserver-gui bin/logserver-gui-plugin list
```

Removing plugins

```
sudo -u logserver-gui bin/logserver-gui-plugin remove [pluginname]
```

Updating plugins

```
sudo -u logserver-gui bin/logserver-gui-plugin remove [pluginname]
sudo -u logserver-gui bin/logserver-gui-plugin install [pluginname]
```

Data Node

Base installation of the Energy Logserver SIEM contains the logserver_auth, join, logserver_guard plugin - These add-ons can be disabled or enabled via the configuration file without having to install or uninstall. You can extend the basic Data Node functionality by installing custom plugins.

Plugins contain JAR files, but may also contain scripts and config files, and must be installed on every node in the cluster.

After installation, each node must be restarted before the plugin becomes visible.

The Data Node provides three categories of plugins:

- Licenced Plugins - Energy Logserver SIEM
- Core Plugins - it is plugins that are part of the core project.
- Community-contributed - it is plugins that are external to the core project

Enabling/Disabling Plugins (Data Node)

Managing the `logserver_auth` Plugin:

- **Disable**:
  - Add `plugins.logserver_auth.enabled: false` to the file `/etc/logserver/logserver.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `plugins.logserver_auth.enabled: false` in the file `/etc/logserver/logserver.yml`.
  - Run the command systemctl restart.

Managing the `logserver_guard` Plugin:

- **Disable**:
  - Add `logserverguard.ssl.transport.enabled: false` to the file `/etc/logserver/logserver.yml`.
  - Add `logserverguard.ssl.http.enabled: false` to the file `/etc/logserver/logserver.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `logserverguard.ssl.transport.enabled: false` in the file `/etc/logserver/logserver.yml`.
  - Remove or comment out the line `logserverguard.ssl.http.enabled: false` in the file `/etc/logserver/logserver.yml`.
  - Run the command systemctl restart .

Managing the `sql` Plugin:

- **Disable**:
  - Add `plugins.sql.enabled: false` to the file `/etc/logserver/logserver.yml`.
  - Run the command systemctl restart.
- **Enable**:
  - Remove or comment out the line `plugins.sql.enabled: false` in the file `/etc/logserver/logserver.yml`.
  - Run the command systemctl restart.

Installing Plugins (Data Node)

Additional Data Node plugins can be installed as follows. Run the tool as the `logserver` user with the Java runtime and configuration directory that the `logserver` service uses:

```
cd /usr/share/logserver/
sudo -u logserver OPENSEARCH_JAVA_HOME=/etc/alternatives/jre OPENSEARCH_PATH_
↪CONF=/etc/logserver bin/logserver-plugin install [plugin_name]
```

Examples:
Plugins from a custom link or filesystem can be installed as follows:

```
sudo -u logserver OPENSEARCH_JAVA_HOME=/etc/alternatives/jre OPENSEARCH_PATH_
↪CONF=/etc/logserver bin/logserver-plugin install file:///path/to/plugin.zip
sudo -u logserver OPENSEARCH_JAVA_HOME=/etc/alternatives/jre OPENSEARCH_PATH_
↪CONF=/etc/logserver bin/logserver-plugin install file:///C:/path/to/plugin.
↪zip
sudo -u logserver OPENSEARCH_JAVA_HOME=/etc/alternatives/jre OPENSEARCH_PATH_
↪CONF=/etc/logserver bin/logserver-plugin install <http://some.domain/path/
↪to/plugin.zip>
```

Listing Plugins (Data Node)

Listing currently loaded plugins:

```
sudo -u logserver OPENSEARCH_JAVA_HOME=/etc/alternatives/jre OPENSEARCH_PATH_
↪CONF=/etc/logserver bin/logserver-plugin list
```

Removing Plugins (Data Node)

```
sudo -u logserver OPENSEARCH_JAVA_HOME=/etc/alternatives/jre OPENSEARCH_PATH_
↪CONF=/etc/logserver bin/logserver-plugin remove [pluginname]
```

Updating Plugins (Data Node)

```
sudo -u logserver OPENSEARCH_JAVA_HOME=/etc/alternatives/jre OPENSEARCH_PATH_
↪CONF=/etc/logserver bin/logserver-plugin remove [pluginname]
sudo -u logserver OPENSEARCH_JAVA_HOME=/etc/alternatives/jre OPENSEARCH_PATH_
↪CONF=/etc/logserver bin/logserver-plugin install [pluginname]
```

Configure email delivery

- *Configure email delivery for sending PDF reports in Scheduler*
  - *Configuration file for **postfix** mail client*
  - *Basic* postfix *configuration*
  - *Example of postfix configuration with SSL encryption enabled*
- *Custom notification on the workstation*

Configure email delivery for sending PDF reports in Scheduler

The default e-mail client that installs with the Linux system, which is used by Energy Logserver to send reports (*Reporting*), is ***postfix***.

Configuration file for postfix mail client

The *postfix* configuration directory is */etc/postfix*. It contains the following files:

main.cf - the main configuration file for the program specifying the basic parameters

Some of its directives:

Table 14: Postfix Configuration Directives

Directive	Description
queue_directory	The postfix queue location.
command_directory	The location of Postfix commands.
daemon_directory	Location of Postfix daemons.
mail_owner	The owner of Postfix domain name of the server
myhostname	The fully qualified domain name of the server.
mydomain	Server domain
myorigin	Host or domain to be displayed as origin on email leaving the server.
inet_interfaces	Network interface to be used for incoming email.
mydestination	Domains from which the server accepts mail.
mynetworks	The IP address of trusted networks.
relayhost	Host or other mail server through which mail will be sent. This server will act as an outbound gateway.
alias_maps	Database of aliases used by the local delivery agent.
alias_database	Alias database generated by the new aliases command.
mail_spool_directory	The location where user boxes will be stored.

master.cf - defines the configuration settings for the master daemon and the way it should work with other agents to deliver mail. For each service installed in the master.cf file seven columns define how the service should be used.

Table 15: Master.cf Column Definitions

Column	Description
service	The name of the service
type	The transport mechanism to be used.
private	Is the service only for user by Postfix.
unpriv	Can the service be run by ordinary users
chroot	Whether the service is to change the main directory (chroot) for the mail. Queue.
wakeup	Wake up interval for the service.
maxproc	The maximum number of processes on which the service can be forked (to divide into branches)
command + args	A command associated with the service plus any argument

access - can be used to control access based on an e-mail address, host address, domain, or network address.

Examples of entries in the file

Table 16: Access File Examples

Description	Example
To allow access for specific IP address:	192.168.122.20 OK
To allow access for a specific domain:	example.com OK
To deny access from the 192.168.3.0/24 network:	192.168.3 REJECT

After making changes to the access file, you must convert its contents to the access.db database with the postmap command:

```
postmap /etc/postfix/access
ll /etc/postfix/access*

  -rw-r--r--. 1 root root 20876 Jan 26 2014 /etc/postfix/access
  -rw-r--r--. 1 root root 12288 Feb 12 07:47 /etc/postfix/access.db
```

canonical - mapping incoming e-mails to local users.

Examples of entries in the file:

To forward emails to user1 to the [user1@yahoo.com] mailbox:

```
user1 user1@yahoo.com
```

To forward all emails for example.org to another example.com domain:

```
@example.org @example.com
```

After making changes to the canonical file, you must convert its contents to the canonical.db database with the postmap command:

```
postmap /etc/postfix/canonical
ll /etc/postfix/canonical*

  -rw-r--r--. 1 root root 11681 2014-06-10 /etc/postfix/canonical
  -rw-r--r--. 1 root root 12288 07-31 20:56 /etc/postfix/canonical.db
```

generic - mapping of outgoing e-mails to local users. The syntax is the same as a canonical file. After you make a change to this file, you must also run the postmap command.

```
postmap /etc/postfix/generic
ll /etc/postfix/generic*

  -rw-r--r--. 1 root root 9904 2014-06-10 /etc/postfix/generic
  -rw-r--r--. 1 root root 12288 07-31 21:15 /etc/postfix/generic.db
```

relocated – information about users who have been transferred. The syntax of the file is the same as canonical and generic files.

Assuming the user1 was moved from example.com to example.net, you can forward all emails received at the old address to the new address:

Example of an entry in the file:

```
<user1@example.com> <user1@example.net>
```

After you make a change to this file, you must also run the postmap command.

```
postmap /etc/postfix/relocated
ll /etc/postfix/relocated*

  -rw-r--r--. 1 root root 6816 2014-06-10 /etc/postfix/relocated
  -rw-r--r--. 1 root root 12288 07-31 21:26 /etc/postfix/relocated.db
```

transport – mapping between e-mail addresses and the server through which these e-mails are to be sent (next hops) in the transport format: nexthop.

Example of an entry in the file:

```
<user1@example.com> smtp:host1.example.com
```

After you make changes to this file, you must also run the postmap command.

```
postmap /etc/postfix/transport
ll /etc/postfix/transport*

  -rw-r--r--. 1 root root 12549 2014-06-10 /etc/postfix/transport
  -rw-r--r--. 1 root root 12288 07-31 21:32 /etc/postfix/transport.db
```

virtual - user to redirect e-mails intended for a certain user to the account of another user or multiple users. It can also be used to implement the domain alias mechanism.

Examples of the entries in the file:

Redirecting email for user1, to root users and user3:

```
user1 root,user3
```

Redirecting email for user 1 in the example.com domain to the root user:

```
<user1@example.com> root
```

After you make a change to this file, you must also run the postmap command:

```
postmap /etc/postfix/virtual
ll /etc/postfix/virtual

 -rw-r--r--. 1 root root 12494 2014-06-10 /etc/postfix/virtual
 -rw-r--r--. 1 root root 12288 07-31 21:58 /etc/postfix/virtual.db
```

Basic *postfix* configuration

Base configuration of *postfix* application you can make in `/etc/postfix/main.cf` configuration file, which must be completed with the following entry:

- section *# RECEIVING MAIL*

  ```
  inet_interfaces = all
  inet_protocols = ipv4
  ```

- section *# INTERNET OR INTRANET*

  ```
  relayhost = [IP mail server]:25 (port number)
  ```

In the next step, you must complete the canonical file of the *postfix*

At the end, you should restart the *postfix*:

```
 systemctl restart postfix
```

Example of postfix configuration with SSL encryption enabled

To configure email delivery with SSL encryption you need to make the following changes in the *postfix* configuration files:

- **/etc/postfix/main.cf** - file should contain the following entries in addition to standard (unchecked entries):

```
mydestination = $myhostname, localhost.$mydomain, localhost
myhostname = example.com
relayhost = [smtp.example.com]:587
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
smtp_tls_CAfile = /root/certs/cacert.cer
smtp_use_tls = yes
smtp_sasl_mechanism_filter = plain, login
smtp_sasl_tls_security_options = noanonymous
canonical_maps = hash:/etc/postfix/canonical
smtp_generic_maps = hash:/etc/postfix/generic
smtpd_recipient_restrictions = permit_sasl_authenticated
```

- **/etc/postfix/sasl_passwd** - file should define the data for authorized

```
[smtp.example.com]:587 [USER@example.com:PASS]
[smtp.example.com]:587 username:password
```

You need to give appropriate permissions:

```
chmod 400 /etc/postfix/sasl_passwd
```

and map configuration to the database:

```
postmap /etc/postfix/sasl_passwd
postmap /etc/postfix/canonical
postmap /etc/postfix/generic
```

next, you need to generate a CA cert file:

```
cat /etc/ssl/certs/Example_Server_CA.pem | tee -a /etc/postfix/cacert.pem
```

Finally, you need to restart the postfix

```
systemctl restart postfix
```

Custom notification on the workstation

The mechanism of *personalization of notification at the workstation* will be implemented by combining alerting mechanisms, triggering integrated commands, and triggering interaction scripts allowing for the transfer of a personalized notification to the workstation. The notifications will use a specific script, which can inform all logged-in users or the selected one about the detection of individual incidents.

Configuration steps

1. Create a new alert rule or edit an existing one according to the instruction: *Creating Alerts*,
2. In the `Alert Method` field select the `Command` method,

3. Add the following script name to `Path to script/command` field:

   `notifyworkstation.py`

Agents

- *Overview*
  - *Agents module*
  - *Preparations*
  - *Installation of MasterAgent - Server Side*
  - *Installation of Agent - Client Side*
  - *Beats - configuration templates*
  - *Agent module compatibility*
  - *Windows - Beats agents installation*
  - *Linux - Beats agents installation*

Overview

The Agents module lets you manage Beats agents on monitored hosts from the Logserver GUI. It uses the same `MasterBeatAgent.jar` executable in two roles:

- **MasterAgent** (server-side tool), installed at `/opt/agents/` on the Logserver host. The Agents GUI plugin invokes it on demand to send management commands to registered agents. It does not run as a persistent service.
- **Agent** (client service), installed as a systemd service on each monitored Linux or Windows host. It connects to Network Probe on TCP port 8080 to register itself and report status, and listens on TCP port 8081 for incoming management commands.

Agent state is stored in the `.agents` OpenSearch index, written by the Network Probe pipeline when agents connect.

From the **Agents** plugin in the Logserver GUI you can view registered agents with their detected Beats processes, start, stop, and restart individual agents or Beats, push configuration file templates to monitored hosts, and reindex the `.agents` index.

Both roles require Java 11 or later. Communication uses mutual TLS; certificates are generated with the bundled `tlstool.sh` script.

Agents module

Before use ensure that you have all the required files

- Tool for generating the necessary certificates: `/usr/share/logserver/utils/tlstool/tlstool.sh`;
- Network Probe utilities:

```
./integrations/masteragent/conf.d/masteragent {01-input-agents.conf,␣
↪050-filter-agents.conf, 100-output-agents.conf}
./integrations/masteragent/masteragent.yml.off.
```

- Linux Agent files, delivered as an archive at `./agents/masteragent/linux/masteragent-linux-<version>.zip`. Unpack it in place and the files appear in a `masteragent` directory beside it:

```
cd ./agents/masteragent/linux
unzip masteragent-linux-*.zip
```

  - Executable: `masteragent/MasterBeatAgent.jar`
  - Configuration File for MasterAgent (server): `masteragent/MasterBeatAgent.conf`
  - Configuration File for Agent (client): `masteragent/agent.conf`
  - Service file: `masteragent/masteragent.service`

  The Windows archive at `./agents/masteragent/windows/masteragent-windows-<version>.zip` has the same structure and contains `Agents.jar`, `agents.exe`, `agents.xml` and `agent.conf`.

  The same archives are available in the GUI under **Agents > Agents download**. Choose **Latest** or **8.1.0**. The list also offers **7.9.0**, which returns an error, and after an upgrade **8.0.0**, which downloads an empty archive.

Preparations

Note

All commands below must be executed from the `/INSTALL` directory.

1. Generate the certificates using `tlstool.sh` script from `/usr/share/logserver/utils/tlstool/`.
   - Update the IP of the node directive, by replacing `<Network Probe ip address>` with the Network Probe server ip in the provided masteragent.yml config (`/usr/share/logserver/utils/tlstool/config/masteragent.yml`).
   - Generate certs using masteragent configuration (by default it will be saved to the `out/` dir):

```
/usr/share/logserver/utils/tlstool/tlstool.sh -c /usr/share/logserver/
↪utils/tlstool/config/masteragent.yml -ca -crt -t agents/masteragent/
↪certificates
```

   - Create KeyStore and TrustStore. Set the KeyStore password of your choice that is utilized to securely store certificates:

```
cd agents/masteragent/certificates
keytool -import -file rootCA.crt -alias root -keystore root.jks -
↪storetype jks
openssl pkcs12 -export -in localhost.crt -inkey localhost.key -out␣
↪node_name.p12 -name localhost -certfile rootCA.crt
cd -
```

   - Set the KeyStore password of your choice that is utilized to securely store certificates.
   - Type 'yes' when "Trust this certificate?" prompt will be shown.
   - Set the TrustStore password of your choice that is used to secure CAs. Remember entered passwords - they'll be used later!
2. Configure firewall to enable communication on used ports (defaults: TCP 8080 -> Network Probe, TCP 8081 -> agent's server).

- These ports can be changed but must reflect "port" and "Network Probe" directives from an agent.conf file to ensure a connection with the agent.
- Commands for default ports:

```
firewall-cmd --permanent --zone public --add-port 8080/tcp
firewall-cmd --permanent --zone public --add-port 8081/tcp
```

3. Configure Network Probe:
   - Copy files:

```
cp -rf ./integrations/masteragent/conf.d/* /etc/logserver-probe/conf.
↪d/
```

   - Copy pipeline configuration:

```
cp -rf ./integrations/masteragent/*.yml.off /etc/logserver-probe/
↪pipelines.d/masteragent.yml
cat ./integrations/masteragent/masteragent.yml.off >> /etc/logserver-
↪probe/pipelines.yml
```

   - Configure SSL connection, by copying previously generated certificates:

```
mkdir -p /etc/logserver-probe/conf.d/masteragent/ssl
/bin/cp -rf ./agents/masteragent/certificates/localhost.* ./agents/
↪masteragent/certificates/rootCA.crt /etc/logserver-probe/conf.d/
↪masteragent/ssl/
```

   - Set permissions:

```
chown -R logserver-probe:logserver-probe /etc/logserver-probe/conf.d/
↪masteragent
```

   - Restart the Network Probe service:

```
systemctl restart logserver-probe
```

Installation of MasterAgent - Server Side

- Copy executable and config:

```
mkdir -p /opt/agents
/bin/cp -rf ./agents/masteragent/linux/masteragent/MasterBeatAgent.jar /
↪opt/agents
/bin/cp -rf ./agents/masteragent/linux/masteragent/MasterBeatAgent.conf /
↪opt/agents/agent.conf
```

- Copy certificates:

```
/bin/cp -rf ./agents/masteragent/certificates/node_name.p12 ./agents/
↪masteragent/certificates/root.jks /opt/agents/
```

- Set permissions:

```
chown -R logserver-probe:logserver-probe /opt/agents
```

- Update the configuration file with KeyStore/TrustStore paths and passwords. Use your preferred editor eg. vim:

```
vim /opt/agents/agent.conf
```

Installation of Agent - Client Side

Linux

Requires Java 11 or later.

Linux Agent - software installed on clients running on Linux OS:

1. Install net-tools package to use Agent on Linux RH / Centos:

```
yum install net-tools
```

2. Copy executable and config:

```
mkdir -p /opt/masteragent
/bin/cp -rf ./agents/masteragent/linux/masteragent/agent.conf ./agents/
↪masteragent/linux/masteragent/MasterBeatAgent.jar /opt/masteragent
/bin/cp -rf ./agents/masteragent/linux/masteragent/masteragent.service /
↪usr/lib/systemd/system/masteragent.service
```

3. Copy certificates:

```
/bin/cp -rf ./agents/masteragent/certificates/node_name.p12 ./agents/
↪masteragent/certificates/root.jks /opt/masteragent/
```

4. Update the configuration file with KeyStore/TrustStore paths and passwords. Also, set the `logstash` directive to the Network Probe URL and port. Use your preferred editor eg. vim:

```
vim /opt/masteragent/agent.conf
```

Key directives to set:

Directive	Description
`logstash`	Network Probe endpoint URL, e.g. `https://192.168.1.10:8080`
`port`	Port the agent listens on for management commands (default: 8081)
`https_keystore`	Path to the `.p12` KeyStore file
`https_keystore_pass`	KeyStore password
`https_truststore`	Path to the `.jks` TrustStore file
`https_truststore_pa`	TrustStore password
`http_enabled`	Enable the agent's embedded HTTP server for management commands (`true` by default)

5. Enable masteragent service:

```
systemctl daemon-reload
systemctl enable masteragent
systemctl start masteragent
```

6. Finally, verify in the Logserver GUI 'Agents' plugin if a newly added agent is present. Check masteragent logs executing:

```
journalctl -fu masteragent
```

Windows

Requires Java 11 or later.

1. Ensure that you have all required files (`./install/agents/masteragent/windows/masteragent`):
   - Installer and manifest: `agents.exe`, `agents.xml`
   - Client: `Agents.jar`
   - Configuration File: `agent.conf`
2. Configure firewall:

   Add an exception to the firewall to allow outgoing connection to TCP port 8080 (Network Probe).
   Add an exception to the firewall to listen on TCP port 8081, required only when `http_enabled = true` in `agent.conf` (the default). Disabling `http_enabled` prevents the agent from receiving management commands from the Logserver GUI.
3. Create `C:\Program Files\MasterAgent` directory.
4. Copy the contents of the `./install/agents/masteragent/windows/masteragent` directory to the `C:\Program Files\MasterAgent`.
5. Copy node_name.p12 and root.jks files from the `./install/agents/masteragent/certificates` to desired directory.
6. Update the `C:\Program Files\MasterAgent\agent.conf` file with KeyStore/TrustStore paths from the previous step and passwords. Also, set the `logstash` directive to the Network Probe URL and port. See the directive table in the Linux section above.
7. Start PowerShell as an administrator:

   To install an agent you can use interchangeably the following methods:
   - Method 1 - use installer:

```
cd "C:\Program Files\MasterAgent"
.\agents.exe install
.\agents.exe start
```

   - Method 2 - manually creating service:

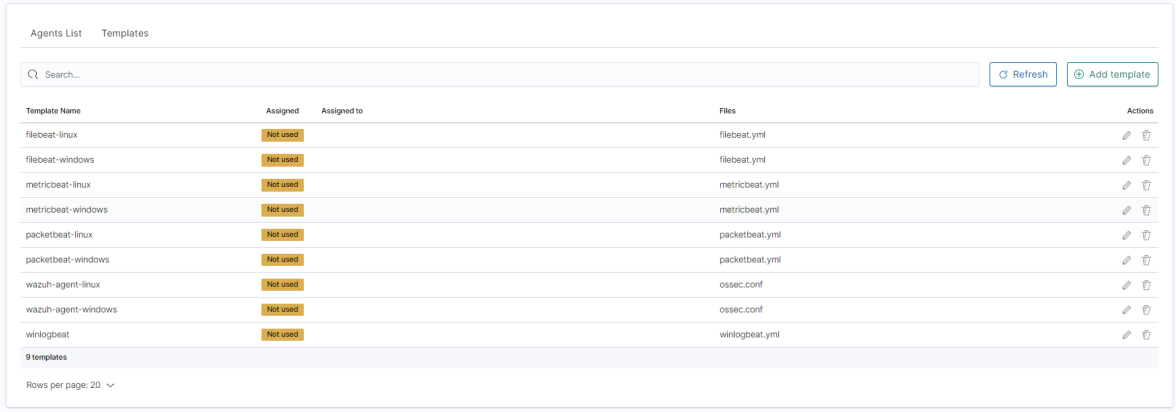
```
New-Service -name masteragent -displayName masteragent -
↪binaryPathName "C:\Program Files\MasterAgent\agents.exe"
```

8. Finally, verify in the Logserver GUI 'Agents' plugin if a newly added agent is present. To check out logs and errors, look for 'agents.out.log' and 'agents.err.log' files in the `C:\Program Files\MasterAgent` directory after the service starts. Also, check the service status:

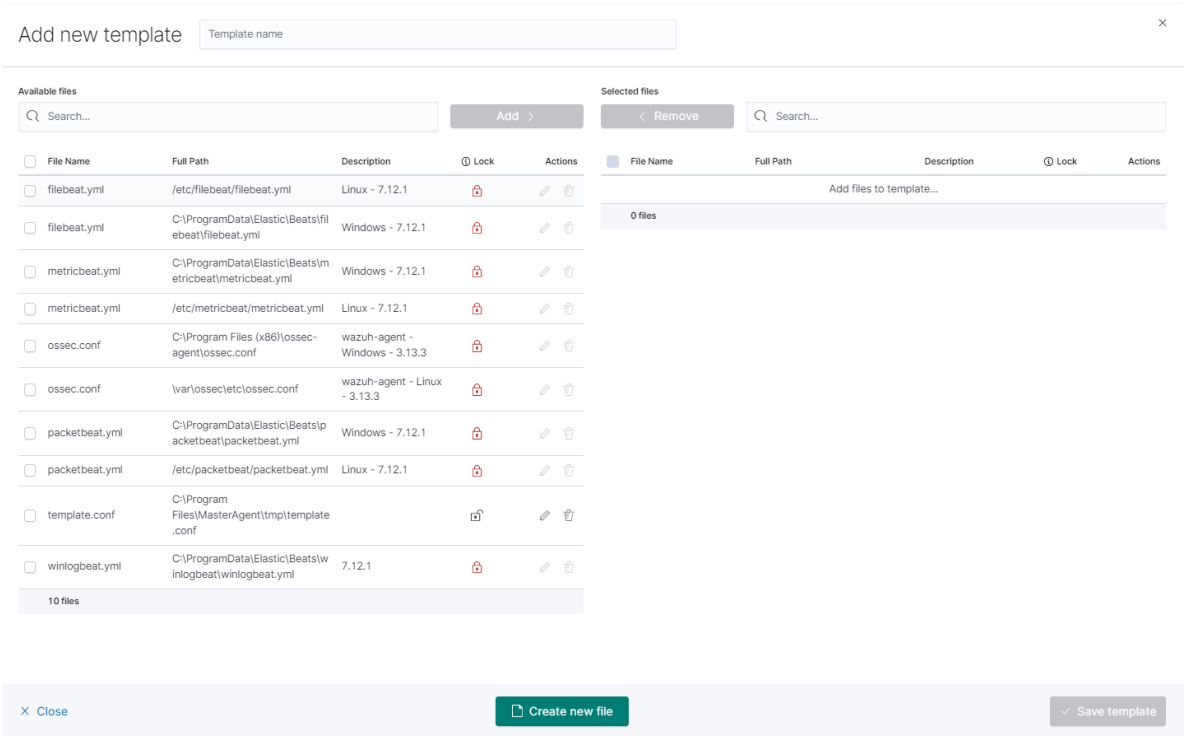
```
.\agents.exe status
```

Beats - configuration templates

1. Go to the `Agents` which is located in the main menu. Then go to `Templates` and click the `Add template` button.

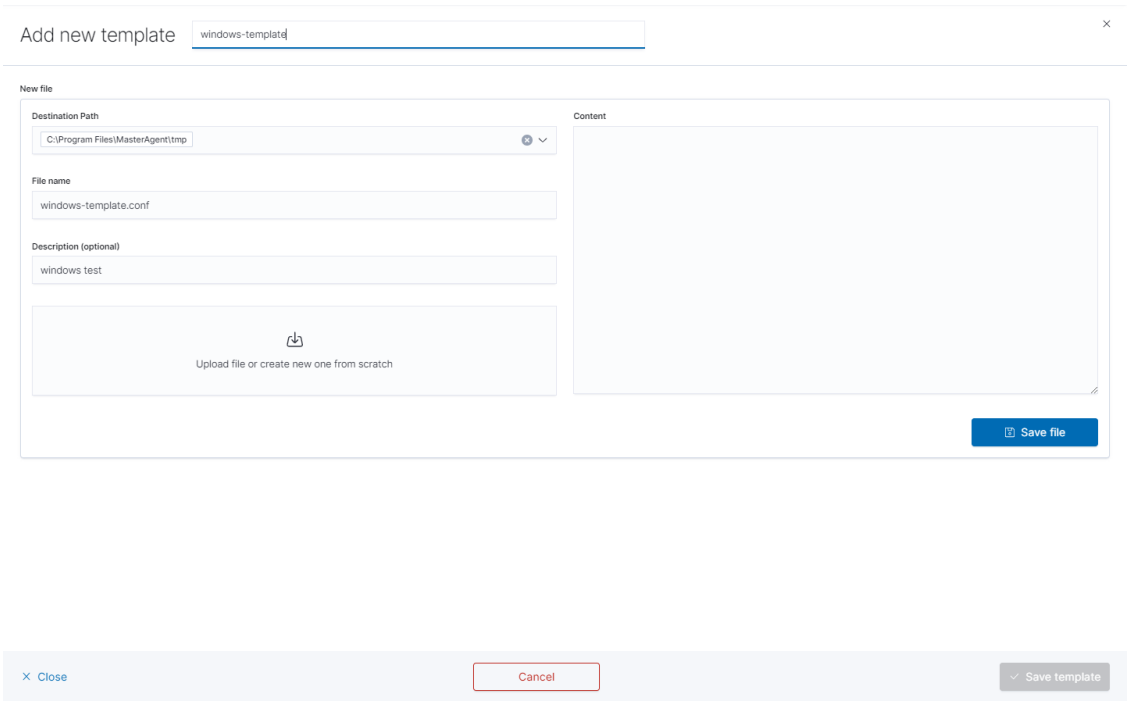


2. Click the `Create new` file button at the bottom.

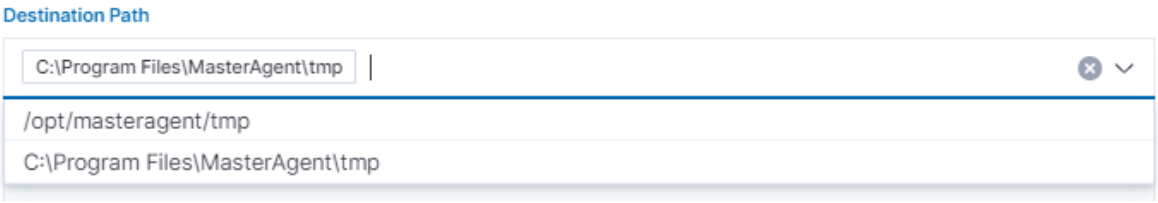


3. You will see the form to create a file that will be on a client system.
There are inputs such as:

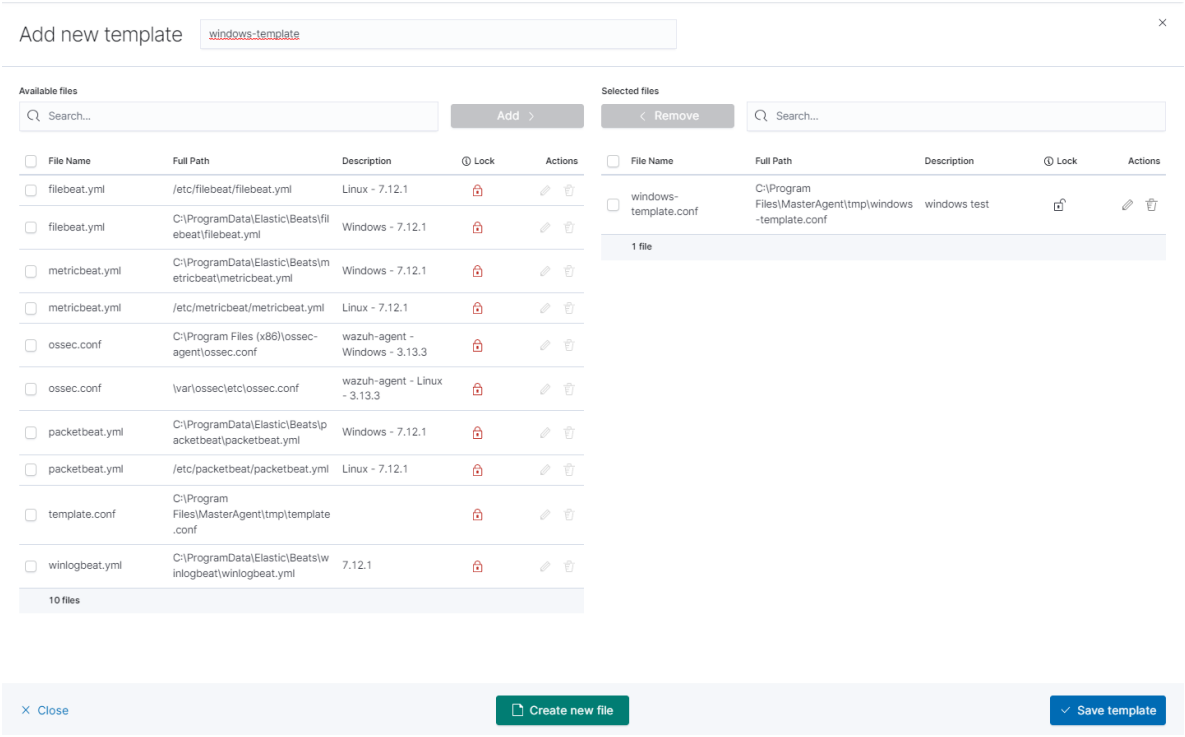
   - Destination Path,
   - File name,
   - Description,
   - Upload file,
   - Content.



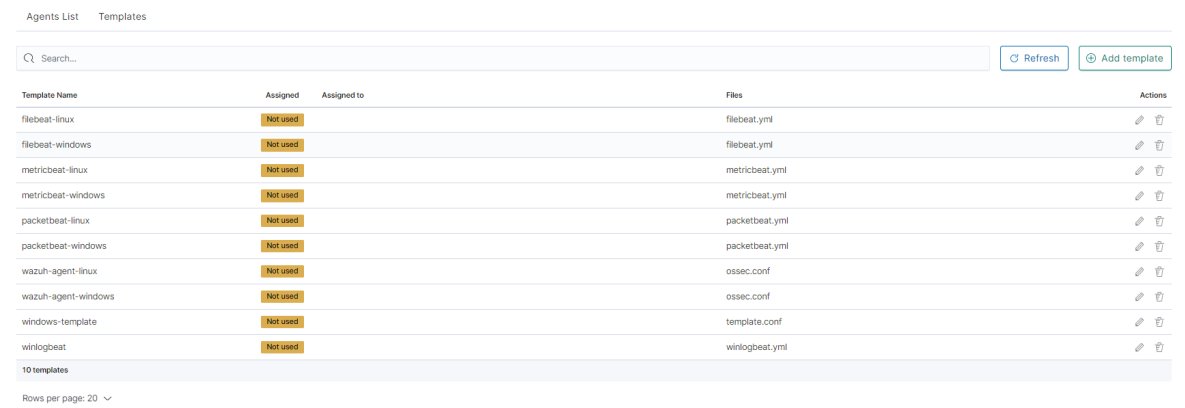
4. Remember that you must provide the exact path to your directory in the Destination Path field



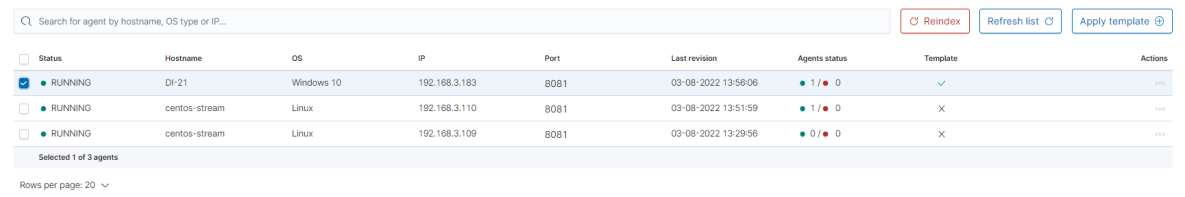
5. After that add your file to the template by checking it from the `Available files` list and clicking `Add` and then `Create new file`.



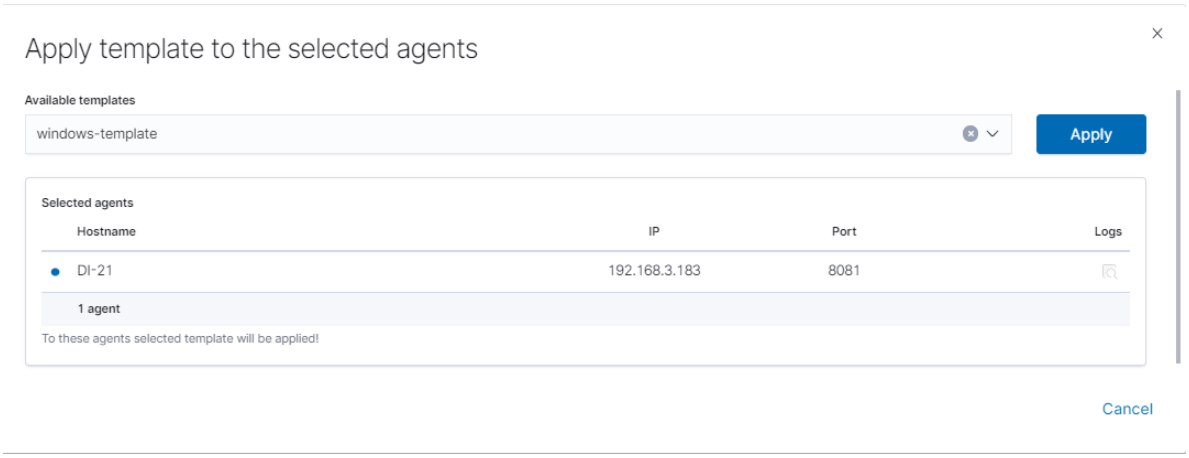
6. You can now see your template in the `Template` tab



7. The next step will be to add the template to the agent by checking the agent's form list and clicking `Apply Template`.



8. The last step is to apply the template by checking it from the list and clicking the `Apply` button.



You can also select multiple agents. Remember, if your file path is Windows type You can only select Windows agents. You can check the Logs by clicking the icon in the `logs` column.



Agent module compatibility

The Agents module among others is compatible with Beats agents in the following versions:

No	Agent Name	Beats Version
1	Filebeat	OSS 7.17.29
2	Packetbeat	OSS 7.17.29
3	Winlogbeat	OSS 7.17.29
4	Metricbeat	OSS 7.17.29
5	Heartbeat	OSS 7.17.29
6	Auditbeat	OSS 7.17.29

Windows - Beats agents installation

> **Note**
>
> Beats for Windows are shipped as ZIP archives containing the executable, a default `.yml`, and the `install-service-<beat>.ps1` script that registers the Windows service. There is no MSI, so rolling them out through Group Policy or a distribution tool means wrapping the unpack, the configuration file, and that script into a package of your own.
>
> MasterAgent does not change this. It distributes configuration and restarts Beats on hosts that already run them, so install the Beat first and let MasterAgent manage it afterwards. If you need an agent that installs unattended out of the box, the Endpoint Security agent ships as an MSI, see *SIEM Agents Configuration*.

Winlogbeat

Installation

1. Copy the Winlogbeat installer from the installation directory `install/agents/beats/windows/winlogbeat-oss-7.17.29-windows-x86_64.zip` and unpack
2. Copy the installation files to the `C:\Program Files\Winlogbeat` directory

Configuration

Editing the file: `C:\Program Files\Winlogbeat\winlogbeat.yml`:

1. In section:

```
winlogbeat.event_logs:
  - name: Application
    ignore_older: 72h
  - name: Security
  - name: System
```

change to:

```
winlogbeat.event_logs:
  - name: Application
    ignore_older: 72h
  - name: Security
    ignore_older: 72h
  - name: System
    ignore_older: 72h
```

2. In section:

```
setup.template.settings:
  index.number_of_shards: 1
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
```

3. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

4. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

5. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

6. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["winlogbeat"]
```

7. Run the `PowerShell` console as Administrator and execute the following commands:

```
cd 'C:\Program Files\Winlogbeat'
.\install-service-winlogbeat.ps1

Security warning
Run only scripts that you trust. While scripts from the internet can be␣
↪useful,
this script can potentially harm your computer. If you trust this script,␣
↪use
the Unblock-File cmdlet to allow the script to run without this warning␣
↪message.
Do you want to run C:\Program Files\Winlogbeat\install-service-winlogbeat.
↪ps1?
[D] Do not run  [R] Run once  [S] Suspend  [?] Help (default is "D"): R
```

Output:

```
Status   Name               DisplayName
------   ----               -----------
Stopped    Winlogbeat      Winlogbeat
```

8. Start Winlogbeat service:

```
sc start Winlogbeat
```

Test configuration:

```
cd 'C:\Program Files\Winlogbeat'
winlogbeat.exe test config
winlogbeat.exe test output
```

Drop event

We can also drop events on the agent side. To do this we need to use the `drop_event` processor

```
processors:
  - drop_event:
      when:
        condition
```

Each condition receives a field to compare. You can specify multiple fields under the same condition by using `AND` between the fields (for example, `field1 AND field2`).

For each field, you can specify a simple field name or a nested map, for example, `dns.question.name`.

See Exported Fields for a list of all the fields that are exported by Winlogbeat.

The supported conditions are:

- equals
- contains
- regexp
- range
- network
- has_fields
- or
- and
- not

equals

With the `equals` condition, you can compare if a field has a certain value. The condition accepts only an integer or a string value.

For example, the following condition checks if the response code of the HTTP transaction is 200:

```
equals:
  http.response.code: 200
```

contains

The `contains` condition checks if a value is part of a field. The field can be a string or an array of strings. The condition accepts only a string value.

For example, the following condition checks if an error is part of the transaction status:

```
contains:
  status: "Specific error"
```

regexp

The `regexp` condition checks the field against a regular expression. The condition accepts only strings.

For example, the following condition checks if the process name starts with `foo`:

```
regexp:
  system.process.name: "^foo.*"
```

range

The range condition checks if the field is in a certain `range` of values. The condition supports `lt, lte, gt, and gte`. The condition accepts only integer or float values.

For example, the following condition checks for failed HTTP transactions by comparing the `http.response.code` field with 400.

```
range:
  http.response.code:
    gte: 400
```

This can also be written as:

```
range:
  http.response.code.gte: 400
```

The following condition checks if the CPU usage in percentage has a value between 0.5 and 0.8.

```
range:
  system.cpu.user.pct.gte: 0.5
  system.cpu.user.pct.lt: 0.8
```

network

The `network` condition checks if the field is in a certain IP network range. Both IPv4 and IPv6 addresses are supported. The network range may be specified using CIDR notation, like "192.0.2.0/24" or "2001:db8::/32", or by using one of these named ranges:

- `loopback` - Matches loopback addresses in the range of 127.0.0.0/8 or ::1/128.
- `unicast` - Matches global unicast addresses defined in RFC 1122, RFC 4632, and RFC 4291 with the exception of the IPv4 broadcast address (`255.255.255.255`). This includes private address ranges.
- `multicast` - Matches multicast addresses.
- `interface_local_multicast` - Matches IPv6 interface-local multicast addresses.
- `link_local_unicast` - Matches link-local unicast addresses.

- `link_local_multicast` - Matches link-local multicast addresses.
- `private` - Matches private address ranges defined in RFC 1918 (IPv4) and RFC 4193 (IPv6).
- `public` - Matches addresses that are not loopback, unspecified, IPv4 broadcast, link-local unicast, link-local multicast, interface local multicast, or private.
- `unspecified` - Matches unspecified addresses (either the IPv4 address "0.0.0.0" or the IPv6 address "::").

The following condition returns true if the source.ip value is within the private address space.

```
network:
  source.ip: private
```

This condition returns true if the `destination.ip` value is within the IPv4 range of `192.168.1.0 - 192.168.1.255`.

```
network:
  destination.ip: '192.168.1.0/24'
```

This condition returns true when `destination.ip` is within any of the given subnets.

```
network:
  destination.ip: ['192.168.1.0/24', '10.0.0.0/8', loopback]
```

has_fields

The `has_fields` condition checks if all the given fields exist in the event. The condition accepts a list of string values denoting the field names.

For example, the following condition checks if the `http.response.code` field is present in the event.

```
has_fields: ['http.response.code']
```

or

The `or` operator receives a list of conditions.

```
or:
  - <condition1>
  - <condition2>
  - <condition3>
  ...
```

For example, to configure the condition `http.response.code = 304 OR http.response.code = 404`:

```
or:
  - equals:
      http.response.code: 304
  - equals:
      http.response.code: 404
```

and

The `and` operator receives a list of conditions.

```
and:
  - <condition1>
  - <condition2>
  - <condition3>
  ...
```

For example, to configure the condition `http.response.code = 200 AND status = OK`:

```
or:
  - <condition1>
  - and:
    - <condition2>
    - <condition3>
```

not

The `not` operator receives the condition to negate.

```
not:
  <condition>
```

For example, to configure the condition `NOT status = OK`:

```
not:
  equals:
    status: OK
```

Internal queue

Winlogbeat uses an internal queue to store events before publishing them. The queue is responsible for buffering and combining events into batches that can be consumed by the outputs. The outputs will use bulk operations to send a batch of events in one transaction.

You can configure the type and behavior of the internal queue by setting options in the `queue` section of the `winlogbeat.yml` config file. Only one queue type can be configured.

This sample configuration sets the memory queue to buffer up to 4096 events:

```
queue.mem:
  events: 4096
```

Configure the memory queue The memory queue keeps all events in memory.

If no flush interval and no number of events to flush is configured, all events published to this queue will be directly consumed by the outputs. To enforce spooling in the queue, set the `flush.min_events` and `flush.timeout` options.

By default `flush.min_events` is set to 2048 and `flush.timeout` is set to 1s.

The output's `bulk_max_size` setting limits the number of events being processed at once.

The memory queue waits for the output to acknowledge or drop events. If the queue is full, no new events can be inserted into the memory queue. Only after the signal from the output will the queue free up space for more events to be accepted.

This sample configuration forwards events to the output if 512 events are available or the oldest available event has been waiting for 5s in the queue:

```
queue.mem:
  events: 4096
  flush.min_events: 512
  flush.timeout: 5s
```

Configuration options

You can specify the following options in the `queue.mem` section of the `winlogbeat.yml` config file: `events` Number of events the queue can store. The default value is `4096` events.

`flush.min_events` Minimum number of events required for publishing. If this value is set to 0, the output can start publishing events without additional waiting times. Otherwise, the output has to wait for more events to become available.

The default value is `2048`.

`flush.timeout` Maximum wait time for flush.min_events to be fulfilled. If set to 0s, events will be immediately available for consumption. The default value is 1s.

Configure disk queue The disk queue stores pending events on the disk rather than the main memory. This allows Beats to queue a larger number of events than is possible with the memory queue, and to save events when a Beat or device is restarted. This increased reliability comes with a performance tradeoff, as every incoming event must be written and read from the device's disk. However, for setups where the disk is not the main bottleneck, the disk queue gives a simple and relatively low-overhead way to add a layer of robustness to incoming event data.

The disk queue is expected to replace the file spool in a future release.

To enable the disk queue with default settings, specify a maximum size:

```
queue.disk:
  max_size: 10GB
```

The queue will be used up to the specified maximum size on the disk. It will only use as much space as required. For example, if the queue is only storing 1GB of events, then it will only occupy 1GB on disk no matter how high the maximum is. Queue data is deleted from the disk after it has been successfully sent to the output.

Configuration options

You can specify the following options in the `queue.disk` section of the `winlogbeat.yml` config file:

`path` The path to the directory where the disk queue should store its data files. The directory is created on startup if it doesn't exist.

The default value is `"${path.data}/diskqueue"`.

`max_size (required)` The maximum size the queue should use on disk. Events that exceed this maximum will either pause their input or be discarded, depending on the input's configuration.

A value of 0 means that no maximum size is enforced, and the queue can grow up to the amount of free space on the disk. This value should be used with caution, as filling a system's main disk can make it inoperable. It is best to use this setting only with a dedicated data or backup partition that will not interfere with Winlogbeat or the rest of the host system.

The default value is `10GB`.

`segment_size` Data added to the queue is stored in segment files. Each segment contains some number of events waiting to be sent to the outputs and is deleted when all its events are sent. By default, segment size is limited to 1/10 of the maximum queue size. Using a smaller size means that the queue will use more data files, but they will be

deleted more quickly after use. Using a larger size means some data will take longer to delete, but the queue will use fewer auxiliary files. It is usually fine to leave this value unchanged.

The default value is `max_size / 10`.

`read_ahead` The number of events that should be read from disk into memory while waiting for an output to request them. If you find outputs are slowing down because they can't read as many events at a time, adjusting this setting upward may help, at the cost of higher memory usage.

The default value is `512`.

`write_ahead` The number of events the queue should accept and store in memory while waiting for them to be written to disk. If you find the queue's memory use is too high because events are waiting too long to be written to disk, adjusting this setting downward may help, at the cost of reduced event throughput. On the other hand, if inputs are waiting or discarding events because they are being produced faster than the disk can handle, adjusting this setting upward may help, at the cost of higher memory usage.

The default value is `2048`.

`retry_interval` Some disk errors may block the operation of the queue, for example, a permission error writing to the data directory, or a disk full error while writing an event. In this case, the queue reports the error and retries after pausing for the time specified in retry_interval.

The default value is `1s` (one second).

`max_retry_interval` When multiple consecutive errors are written to the disk, the queue increases the retry interval by factors of 2 up to a maximum of max_retry_interval. Increase this value if you are concerned about logging too many errors or overloading the host system if the target disk becomes unavailable for an extended time.

The default value is `30s` (thirty seconds).

Filebeat

Filebeat Installation

1. Copy the Filebeat installer from the installation directory `install/agents/beats/windows/filebeat-oss-7.17.29-windows-x86_64.zip` and unpack
2. Copy the installation files to the `C:\Program Files\Filebeat` directory

Configuration

Editing the file: `C:\Program Files\Filebeat\filebeat.yml`:

1. In section:

```
- type: log

  # Change to true to enable this input configuration.
  enabled: false
```

   change to:

```
- type: log

  # Change to true to enable this input configuration.
  enabled: true
```

2. In section:

```
    paths:
    - /var/log/*.log
    #- c:\programdata\logserver\logs\*
```

change to:

```
paths:
    #- /var/log/*.log
    #- c:\programdata\logserver\logs\*
    - "C:\Program Files\Microsoft SQL Server\*\MSSQL\Log\*"
    - "C:\inetpub\logs\*"
```

3. In section:

```
setup.template.settings:
  index.number_of_shards: 1
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
```

4. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

5. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

6. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

7. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["filebeat"]
```

8. Run the `PowerShell` console as Administrator and execute the following commands:

```
cd 'C:\Program Files\Filebeat'
.\install-service-filebeat.ps1

Security warning
Run only scripts that you trust. While scripts from the internet can be␣
↪useful,
this script can potentially harm your computer. If you trust this script,␣
↪use
the Unblock-File cmdlet to allow the script to run without this warning␣
↪message.
Do you want to run C:\Program Files\Filebeat\install-service-filebeat.ps1?
[D] Do not run  [R] Run once  [S] Suspend  [?] Help (default is "D"): R
```

Output:

```
Status   Name               DisplayName
------   ----               -----------
Stopped  Filebeat        Filebeat
```

9. Start Filebeat service:

```
sc start filebeat
```

You can enable, disable, and list Filebeat modules using the following command:

```
cd 'C:\Program Files\Filebeat'
filebeat.exe modules list
filebeat.exe modules apache enable
filebeat.exe modules apache disable
```

Test configuration:

```
cd 'C:\Program Files\Filebeat'
filebeat.exe test config
filebeat.exe test output
```

Metricbeat

Metricbeat Installation

1. Copy the Metricbeat installer from the installation directory `install/agents/beats/windows/metricbeat-oss-7.17.29-windows-x86_64.zip` and unpack
2. Copy the installation files to the `C:\Program Files\Metricbeat` directory

Configuration

Editing the file: `C:\Program Files\Metricbeat\metricbeat.yml`:

1. In section:

```
setup.template.settings:
  index.number_of_shards: 1
  index.codec: best_compression
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
  #index.codec: best_compression
```

2. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

3. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

4. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

5. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["metricbeat"]
```

6. Run the `PowerShell` console as Administrator and execute the following commands:

```
cd 'C:\Program Files\Metricbeat'
.\install-service-metricbeat.ps1

Security warning
Run only scripts that you trust. While scripts from the internet can be␣
↪useful,
this script can potentially harm your computer. If you trust this script,␣
↪use
the Unblock-File cmdlet to allow the script to run without this warning␣
↪message.
Do you want to run C:\Program Files\Metricbeat\install-service-metricbeat.
↪ps1?
[D] Do not run  [R] Run once  [S] Suspend  [?] Help (default is "D"): R
```

Output:

```
Status   Name               DisplayName
------   ----               -----------
Stopped  Metricbeat        Metricbeat
```

7. Start Metricbeat service:

```
sc start metricbeat
```

You can enable, disable, and list Metricbeat modules using the following command:

```
cd 'C:\Program Files\Metricbeat'
metricbeat.exe modules list
metricbeat.exe modules apache enable
metricbeat.exe modules apache disable
```

Test configuration:

```
cd 'C:\Program Files\Metricbeat'
metricbeat.exe test config
metricbeat.exe test output
```

Packetbeat

Packetbeat Installation

1. Copy the Packetbeat installer from the installation directory `install/agents/beats/windows/packetbeat-oss-7.17.29-windows-x86_64.zip` and unpack
2. Copy the installation files to the `C:\Program Files\Packetbeat` directory

Configuration

Editing the file: `C:\Program Files\Packetbeat\packetbeat.yml`:

1. In section:

```
setup.template.settings:
  index.number_of_shards: 3
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 3
```

2. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

3. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

4. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

5. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["packetbeat"]
```

6. Run the `PowerShell` console as Administrator and execute the following commands:

```
cd 'C:\Program Files\Packetbeat'
.\install-service-packetbeat.ps1

Security warning
Run only scripts that you trust. While scripts from the internet can be␣
↪useful,
this script can potentially harm your computer. If you trust this script,␣
↪use
```

(continues on next page)

(continued from previous page)

```
the Unblock-File cmdlet to allow the script to run without this warning␣
↪message.
Do you want to run C:\Program Files\Packetbeat\install-service-packetbeat.
↪ps1?
[D] Do not run  [R] Run once  [S] Suspend  [?] Help (default is "D"): R
```

Output:

```
Status   Name               DisplayName
------   ----               -----------
Stopped  Packetbeat         Packetbeat
```

7. Start Packetbeat service:

```
sc start packetbeat
```

Test configuration:

```
cd 'C:\Program Files\Packetbeat'
packetbeat.exe test config
packetbeat.exe test output
```

Auditbeat

Auditbeat Installation

1. Copy the Auditbeat installer from the installation directory `install/agents/beats/windows/auditbeat-oss-7.17.29-windows-x86_64.zip` and unpack
2. Copy the installation files to the `C:\Program Files\Auditbeat` directory

Configuration

Editing the file: `C:\Program Files\Auditbeat\auditbeat.yml`:

1. In section:

```
setup.template.settings:
  index.number_of_shards: 1
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
```

2. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

3. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

4. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

5. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["auditbeat"]
```

6. Run the `PowerShell` console as Administrator and execute the following commands:

```
cd 'C:\Program Files\Auditbeat'
.\install-service-auditbeat.ps1

Security warning
Run only scripts that you trust. While scripts from the internet can be␣
↪useful,
this script can potentially harm your computer. If you trust this script,␣
↪use
the Unblock-File cmdlet to allow the script to run without this warning␣
↪message.
Do you want to run C:\Program Files\Auditbeat\install-service-auditbeat.
↪ps1?
[D] Do not run  [R] Run once  [S] Suspend  [?] Help (default is "D"): R
```

Output:

```
Status   Name               DisplayName
------   ----               -----------
Stopped  Auditbeat        Auditbeat
```

7. Start Auditbeat service:

```
sc start auditbeat
```

Test configuration:

```
cd 'C:\Program Files\Auditbeat'
auditbeat.exe test config
auditbeat.exe test output
```

Heartbeat

Heartbeat Installation

1. Copy the Heartbeat installer from the installation directory `install/agents/beats/windows/heartbeat-oss-7.17.29-windows-x86_64.zip` and unpack
2. Copy the installation files to the `C:\Program Files\Heartbeat` directory

Configuration

Editing the file: `C:\Program Files\Heartbeat\heartbeat.yml`:

1. In section:

```
setup.template.settings:
  index.number_of_shards: 1
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
```

2. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

3. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

4. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

5. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["heartbeat"]
```

6. Run the `PowerShell` console as Administrator and execute the following commands:

```
cd 'C:\Program Files\Heartbeat'
.\install-service-heartbeat.ps1

Security warning
Run only scripts that you trust. While scripts from the internet can be␣
↪useful,
this script can potentially harm your computer. If you trust this script,␣
↪use
the Unblock-File cmdlet to allow the script to run without this warning␣
↪message.
Do you want to run C:\Program Files\Heartbeat\install-service-heartbeat.
↪ps1?
[D] Do not run  [R] Run once  [S] Suspend  [?] Help (default is "D"): R
```

Output:

```
Status   Name               DisplayName
------   ----               -----------
Stopped  Heartbeat        Heartbeat
```

7. Start Heartbeat service:

```
sc start heartbeat
```

Test configuration:

```
cd 'C:\Program Files\Heartbeat'
heartbeat.exe test config
heartbeat.exe test output
```

Linux - Beats agents installation

Filebeat (Linux)

Filebeat (Linux) Installation

1. Copy the Filebeat installer from the installation directory `install/agents/beats/linux/filebeat-oss-7.17.29-x86_64.rpm`
2. Install filebeat with the following command:

```
yum install -y filebeat-oss-7.17.29-x86_64.rpm
```

Configuration

Editing the file: `/etc/filebeat/filebeat.yml`:

1. In section:

```
- type: log

  # Change to true to enable this input configuration.
  enabled: false
```

change to:

```
- type: log

  # Change to true to enable this input configuration.
  enabled: true
```

2. In section:

```
setup.template.settings:
  index.number_of_shards: 1
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
```

3. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

4. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

5. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

6. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["filebeat"]
```

7. Enable and start Filebeat service:

```
systemctl enable filebeat
systemctl start filebeat
```

You can enable, disable, and list Filebeat modules using the following command:

```
filebeat modules list
filebeat modules enable apache
filebeat modules disable apache
```

Test configuration:

```
filebeat test config
filebeat test output
```

Metricbeat (Linux)

Metricbeat (Linux) Installation

1. Copy the Metricbeat installer from the installation directory `install/agents/beats/linux/metricbeat-oss-7.17.29-x86_64.rpm`
2. Install Metricbeat with the following command:

```
yum install -y metricbeat-oss-7.17.29-x86_64.rpm
```

Configuration

Editing the file: `/etc/metricbeat/metricbeat.yml`:

1. In section:

```
setup.template.settings:
  index.number_of_shards: 1
  index.codec: best_compression
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
  #index.codec: best_compression
```

2. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

3. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

4. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

5. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["metricbeat"]
```

Enable and start Metricbeat service:

```
systemctl enable metricbeat
systemctl start metricbeat
```

You can enable, disable, and list Metricbeat modules using the following command:

```
metricbeat modules list
metricbeat modules apache enable
metricbeat modules apache disable
```

Test configuration:

```
metricbeat test config
metricbeat test output
```

Packetbeat (Linux)

Packetbeat (Linux) Installation

1. Copy the Packetbeat installer from the installation directory `install/agents/beats/linux/packetbeat-oss-7.17.29-x86_64.rpm`
2. Install Packetbeat with the following command:

```
yum install -y packetbeat-oss-7.17.29-x86_64.rpm
```

Configuration

Editing the file: `/etc/packetbeat/packetbeat.yml`:

1. In section:

```
setup.template.settings:
  index.number_of_shards: 3
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 3
```

2. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

3. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

4. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

5. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["packetbeat"]
```

Enable and start Packetbeat service:

```
systemctl enable packetbeat
systemctl start packetbeat
```

Test configuration:

```
packetbeat test config
packetbeat test output
```

Auditbeat (Linux)

Auditbeat (Linux) Installation

1. Copy the Auditbeat installer from the installation directory `install/agents/beats/linux/auditbeat-oss-7.17.29-x86_64.rpm`
2. Install Auditbeat with the following command:

```
yum install -y auditbeat-oss-7.17.29-x86_64.rpm
```

Configuration

Editing the file: `/etc/auditbeat/auditbeat.yml`:

1. In section:

```
setup.template.settings:
  index.number_of_shards: 1
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
```

2. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

3. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

4. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

5. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["auditbeat"]
```

Enable and start Auditbeat service:

```
systemctl enable auditbeat
systemctl start auditbeat
```

Test configuration:

```
auditbeat test config
auditbeat test output
```

Heartbeat (Linux)

Heartbeat (Linux) Installation

1. Copy the Heartbeat installer from the installation directory `install/agents/beats/linux/heartbeat-oss-7.17.29-x86_64.rpm`
2. Install Heartbeat with the following command:

```
yum install -y heartbeat-oss-7.17.29-x86_64.rpm
```

Configuration

Editing the file: `/etc/heartbeat/heartbeat.yml`:

1. In section:

```
setup.template.settings:
  index.number_of_shards: 1
```

change to:

```
#setup.template.settings:
  #index.number_of_shards: 1
```

2. In section:

```
setup.kibana:
```

change to:

```
#setup.kibana:
```

3. In section:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
```

change to:

```
#output.elasticsearch:
  # Array of hosts to connect to.
  #hosts: ["localhost:9200"]
```

4. In section:

```
#output.logstash:
  # The Logserver-Probe hosts
  #hosts: ["localhost:5044"]
```

change to:

```
output.logstash:
  # The Logserver-Probe hosts
  hosts: ["logserver-probe-ip:5044"]
```

5. In section:

```
#tags: ["service-X", "web-tier"]
```

change to:

```
tags: ["heartbeat"]
```

Enable and start Heartbeat service:

```
systemctl enable heartbeat-elastic
systemctl start heartbeat-elastic
```

Test configuration:

```
heartbeat test config
heartbeat test output
```

User Management

- *Energy Logserver SIEM Platform*
- *Table of Contents*
- *User Lifecycle Management*
  - *User Creation*
  - *User Modification*
- *Role-Based Access Control*
  - *Predefined Roles*
  - *Custom Role Creation*
- *Field Level Security*
- *Users Limit for Role*

Energy Logserver SIEM Platform

Energy Logserver SIEM provides essential user management capabilities to support security operations teams with appropriate access controls and user lifecycle management.

> **Note**
>
> Advanced authentication methods (LDAP, SAML, RADIUS) are covered in *Authentication and Directory Integration*. This chapter focuses on practical user lifecycle and role management.

Table of Contents

1. *User Lifecycle Management*
2. *Role-Based Access Control*
3. *Field Level Security*
4. *Users Limit for Role*

User Lifecycle Management

User Creation

Standard User Creation:

Navigate to **ELS Console** → **Management** → **Config**

Required fields:

- **Username**: Unique identifier (3-50 characters)
- **Password**: User password
- **Email**: Valid email for notifications
- **Role**: Roles that user will be assigned to

User Modification

Profile Updates:

- Users can modify their password
- Admin can modify password, email, GUI Access, Roles, Default Role

Role-Based Access Control

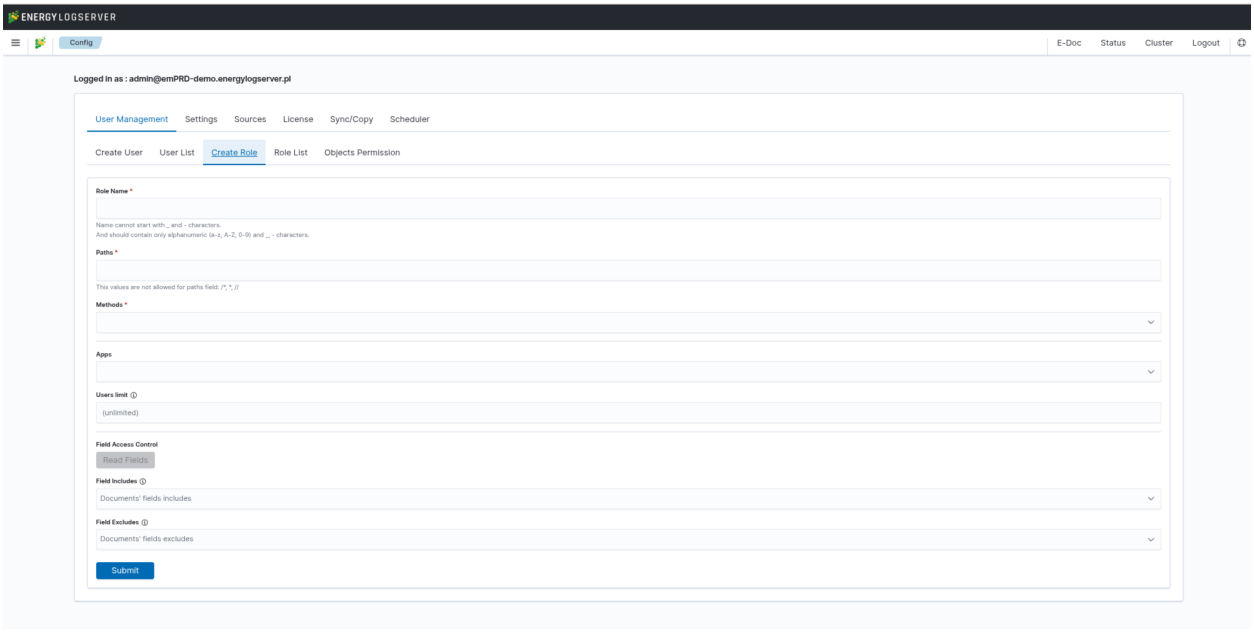
Predefined Roles

The system provides the following predefined roles:

- **admin**: Complete system administration with full access to all features and configurations
- **alert**: Access to alert management and configuration
- **e-doc**: Electronic document management capabilities
- **intelligence**: Threat intelligence and analytics features
- **license**: License management and monitoring
- **probe**: Network Probe and data pipeline configuration
- **report**: Report generation and management
- **security**: Security operations and monitoring
- **security-tenant**: Multi-tenancy security management

Custom Role Creation

Navigate to **ELS Console** → **Management** → **Config** → **User Management** → **Create Role**



Role Parameters:

- **Role Name***: unique identifier for the role (required). Name cannot start with _ or – characters and should contain only alphanumeric characters (a-z, A-Z, 0-9) and _ – characters.

- **Paths***: index patterns that the role can access (required). You can provide one or more index patterns separated by commas. Values `/*`, `*`, and `//` are not allowed.
- **Methods***: HTTP methods allowed for this role (required). Available methods: GET, POST, PUT, DELETE, HEAD.
- **Apps**: applications/modules accessible by this role.
- **Users limit**: maximum number of users that can be assigned to this role. Defaults to unlimited. Enter a number greater than 0 to set a specific limit.
- **Field Access Control**: click the **Read Fields** button to load available fields from the selected index pattern for use in Field Includes/Excludes.
- **Field Includes**: specific fields that will be visible to users with this role.
- **Field Excludes**: fields that will be hidden from users with this role.

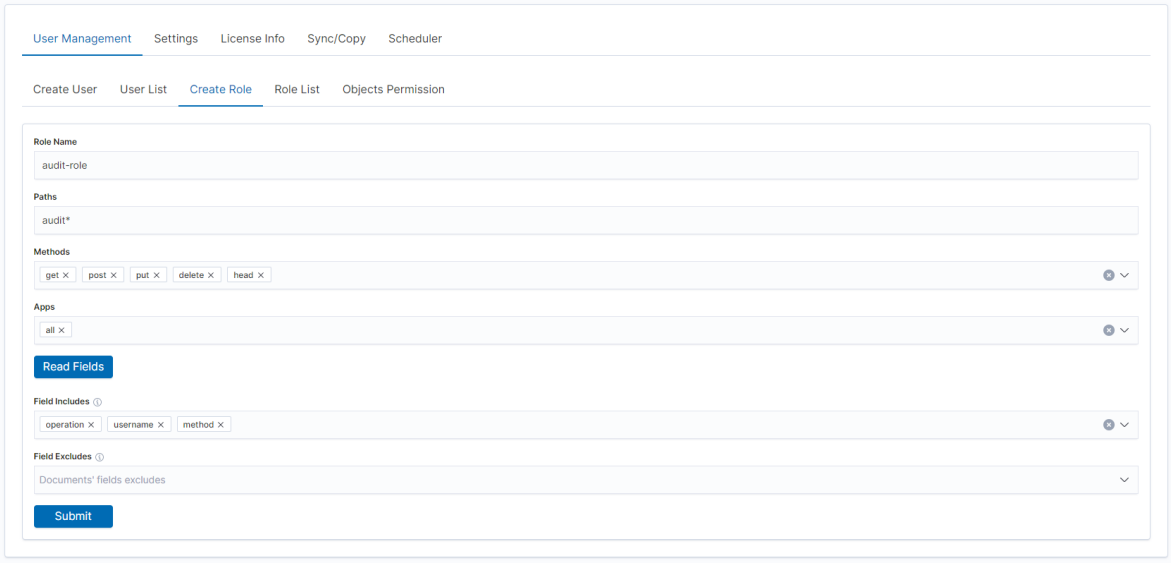
Click **Submit** to create the role. The new role will appear in the **Role List** tab.

Field Level Security

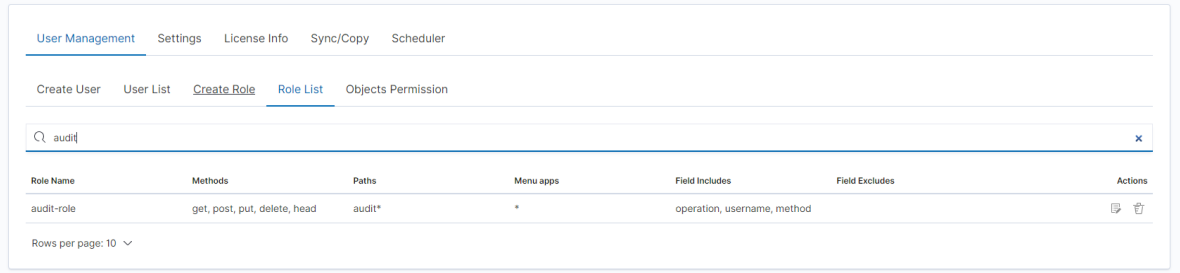
Field level security allows you to restrict access to specific fields in documents for a user role. For example, users can only view specific fields in the Discover module, while other fields will be inaccessible.

Configuration Steps:

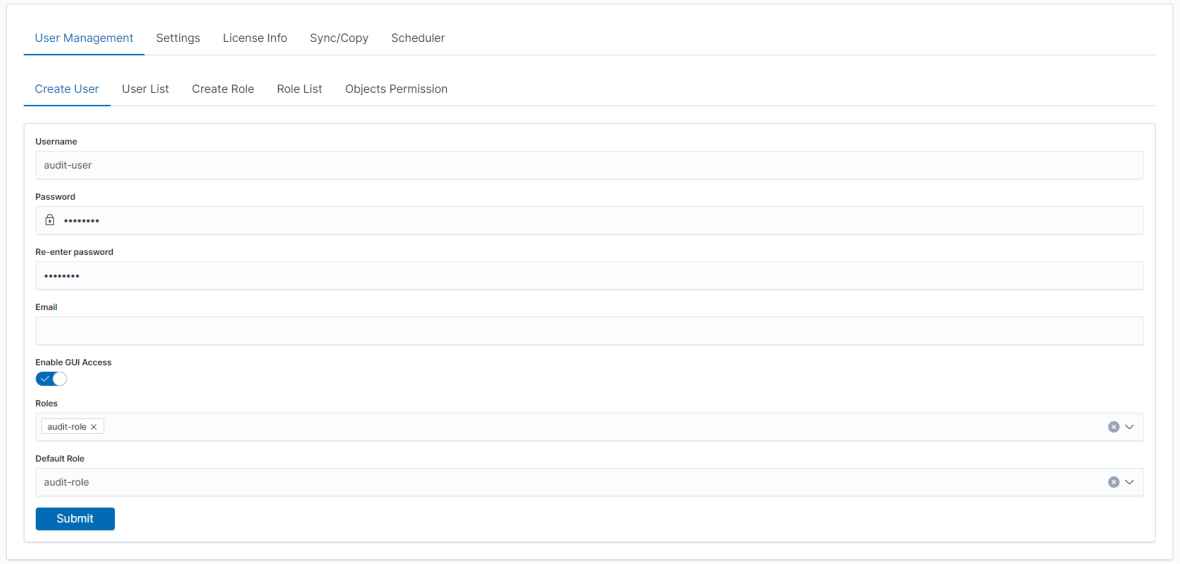
1. Add the index pattern to the `Field Includes` or `Field Excludes` parameters when creating or editing a role:
   - **Field Includes**: Only specified fields will be visible to the user
   - **Field Excludes**: Specified fields will be hidden from the user



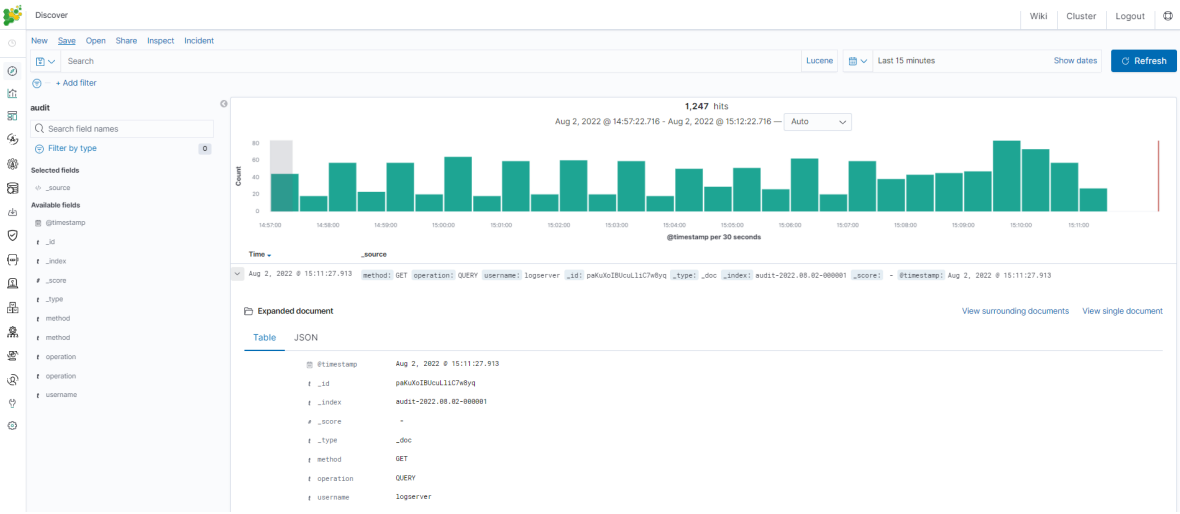
2. The new role will appear in the **Role List** tab:



3. Assign users to the new role:



4. When users log in with this role, they will only see the permitted fields in the Discover module:



Users Limit for Role

This feature allows administrators to define the maximum number of users that can be assigned to a specific role. Once the limit is reached, no additional users can be added to that role until existing assignments are removed or the limit is increased.

Purpose:

Role user limits help organizations maintain compliance with licensing requirements and enforce security policies by controlling role assignments based on organizational needs.

Configuration:

When creating or editing a role under **Management** → **Config** → **User Management**, you can specify the **Users limit** parameter:

- Leave the field empty for unlimited users
- Enter a number greater than 0 to set a specific limit

Behavior:

- When the user limit is reached, administrators cannot assign additional users to that role
- For non-local users (such as LDAP users) with roles assigned dynamically, if a login attempt would exceed the role's user limit, the login will fail
- A clear error message will be displayed on the login screen indicating that the role limit has been reached
- System users are not counted toward role limits to ensure system functionality is not affected by administrative limits

Domain and 2FA Authentication

- *Authentication with Active Directory*
  - *Configure SSL support for AD authentication*
  - *Role mapping*
  - *Password encryption*
- *Authentication with Radius*
  - *Configuration*
- *Authentication with LDAP*
  - *LDAP Configuration*
- *Configuring Single Sign On (SSO)*
  - *Configuration steps*
  - *Client (Browser) Configuration*
  - *KDC error codes*
- *Two-Factor Authentication*
  - *2FA with OAuth2 Proxy and Google (example)*
  - *SSO gateway with Keycloak and oauth2-proxy*
  - *2FA with Nginx and PKI client certificate*

Authentication with Active Directory

The AD configuration should be done in the `/etc/logserver/properties.yml` file.

Below is a list of settings to be made in the `properties.yml` file (the commented section in the file for the AD settings to start working, this fragment should be uncommented):

```
ldaps:
 - name: "example.com"  # domain that is configured
   host: "127.0.0.1,127.0.0.2"  # list of server for this domain
   #port: 389 # optional, default 389 for unencrypted session or 636 for␣
↪encrypted sessions
   ssl_enabled: false  # optional, default true
   #ssl_trust_all_certs: true # optional, default false
   #ssl.keystore.file: "path" # path to the truststore store
   #ssl.keystore.password: "path" # password to the trusted certificate store
   bind_dn: "admin@example.com" # account name administrator
   bind_password: "password" # password for the administrator account
   search_user_base_DN: "OU=lab,DC=example,DC=com" # search for the DN user␣
↪tree database
   #user_id_attribute: "uid # search for a user attribute optional, by␣
↪default "uid"
   #search_groups_base_DN: "OU=lab,DC=example,DC=com" # group database search.
↪ This is a catalog main, after which the groups will be sought.
   #unique_member_attribute: "uniqueMember" # optional, default "uniqueMember"
   #max_connections: 10 # optional, default 10
   connection_timeout_in_sec: 10 # optional, default 1
   request_timeout_in_sec: 10 # optional, default 1
   cache_ttl_in_sec: 60 # optional, default 0 - cache disabled
   #authentication_only: true    # optional ignore role-mapping settings
   #default_authentication_roles: [ "roleName1", "roleName2" ]      # roles␣
↪asigned to new users authenticating using this LDAP server, used when␣
↪authentication_only = true
```

If we want to configure multiple domains, then in this configuration file we copy the # LDAP section below and configure it for the next domain.

Below is an example of how an entry for 2 domains should look like. (It is important to take the interpreter to read these values correctly).

```
ldaps:
  - name: "example1.com" #DOMAIN 1
    host: "127.0.0.1,127.0.0.2"
    bind_dn: "esauth@example1.com"
    bind_password: "password"
    search_user_base_DN: "cn=Users,DC=example1,DC=com"
    ssl_enabled: false
  - name: "example2.com" #DOMAIN 2
    host: "127.0.0.1,127.0.0.2"
    bind_dn: "esauth@example2.com"
    bind_password: "password"
    search_user_base_DN: "cn=Users,DC=example2,DC=com"
    ssl_enabled: false
```

After completing the LDAP section entry in the `properties.yml` file, save the changes and reload the module with the below command:

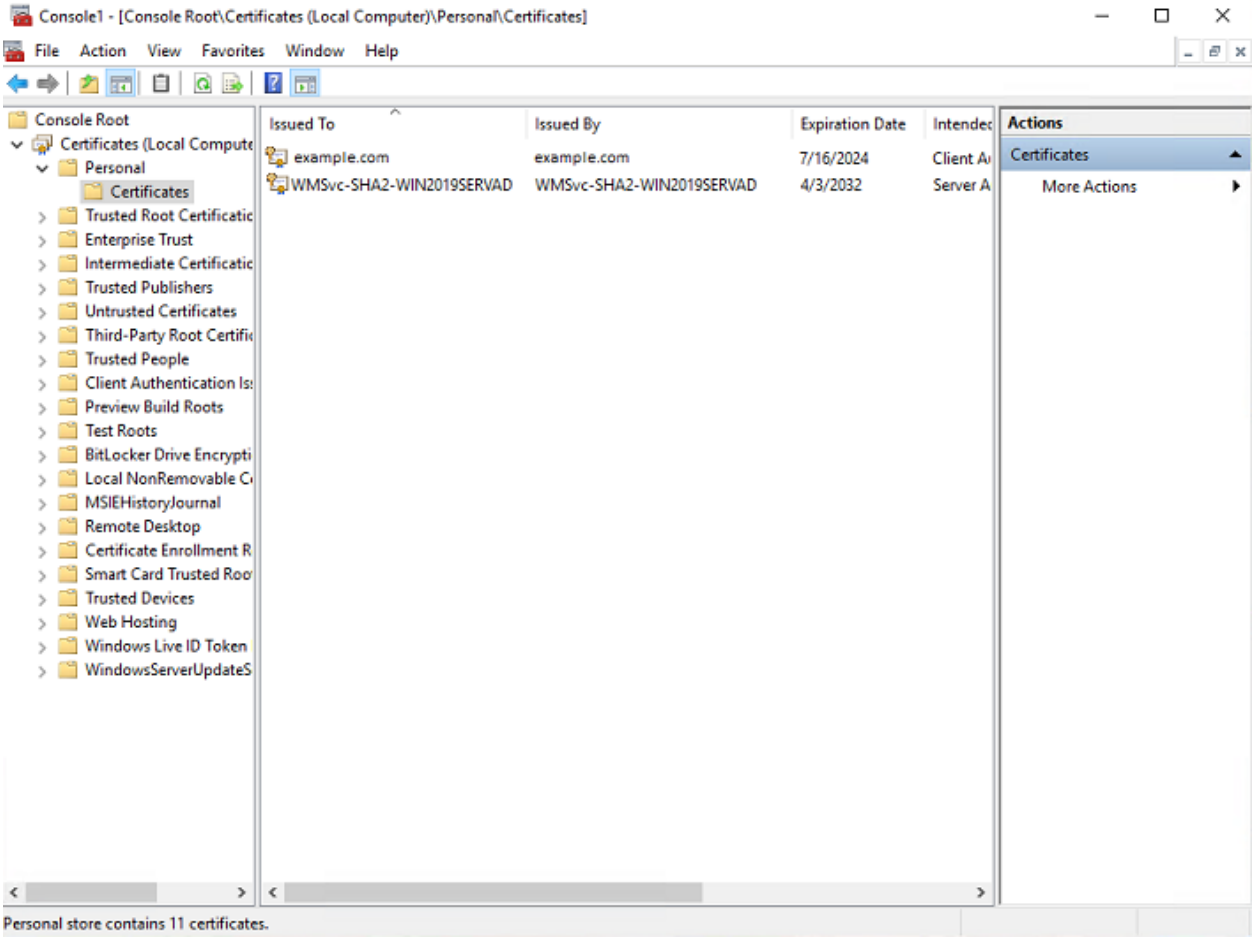
```
curl -sS -uUSER:PASSWORD localhost:9200/_logserver/auth/reload -XPOST
```

Example:

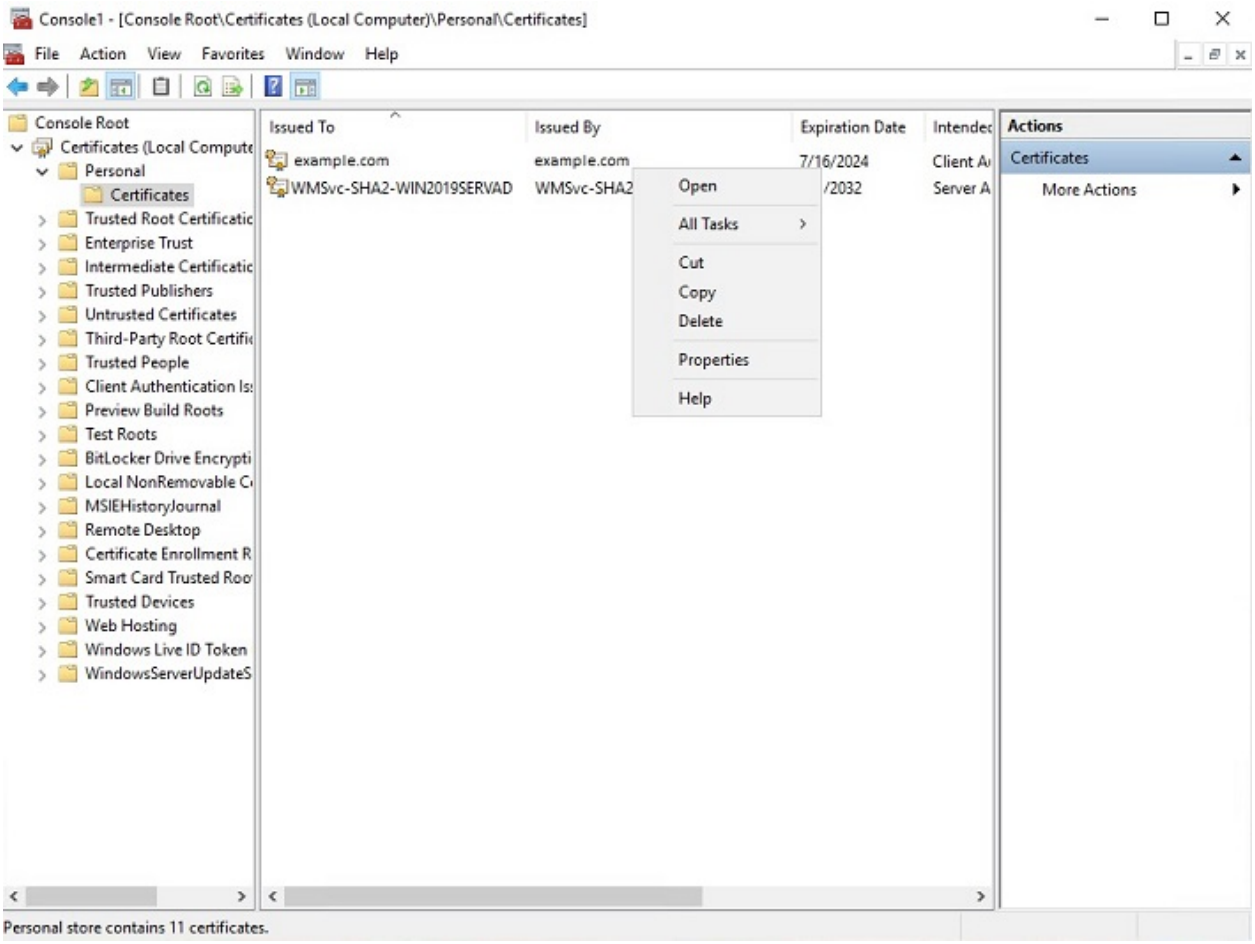
```
curl -sS -ulogserver:logserver localhost:9200/_logserver/auth/reload -XPOST
```

Configure SSL support for AD authentication

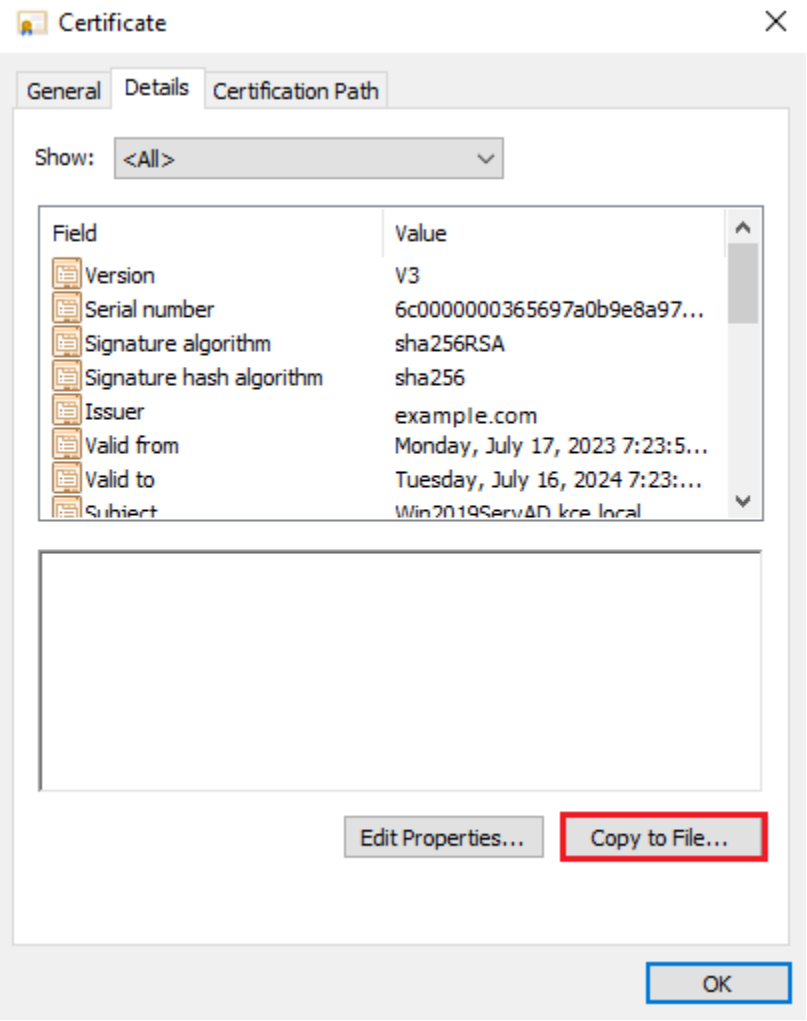
Open the certificate manager on the AD server.



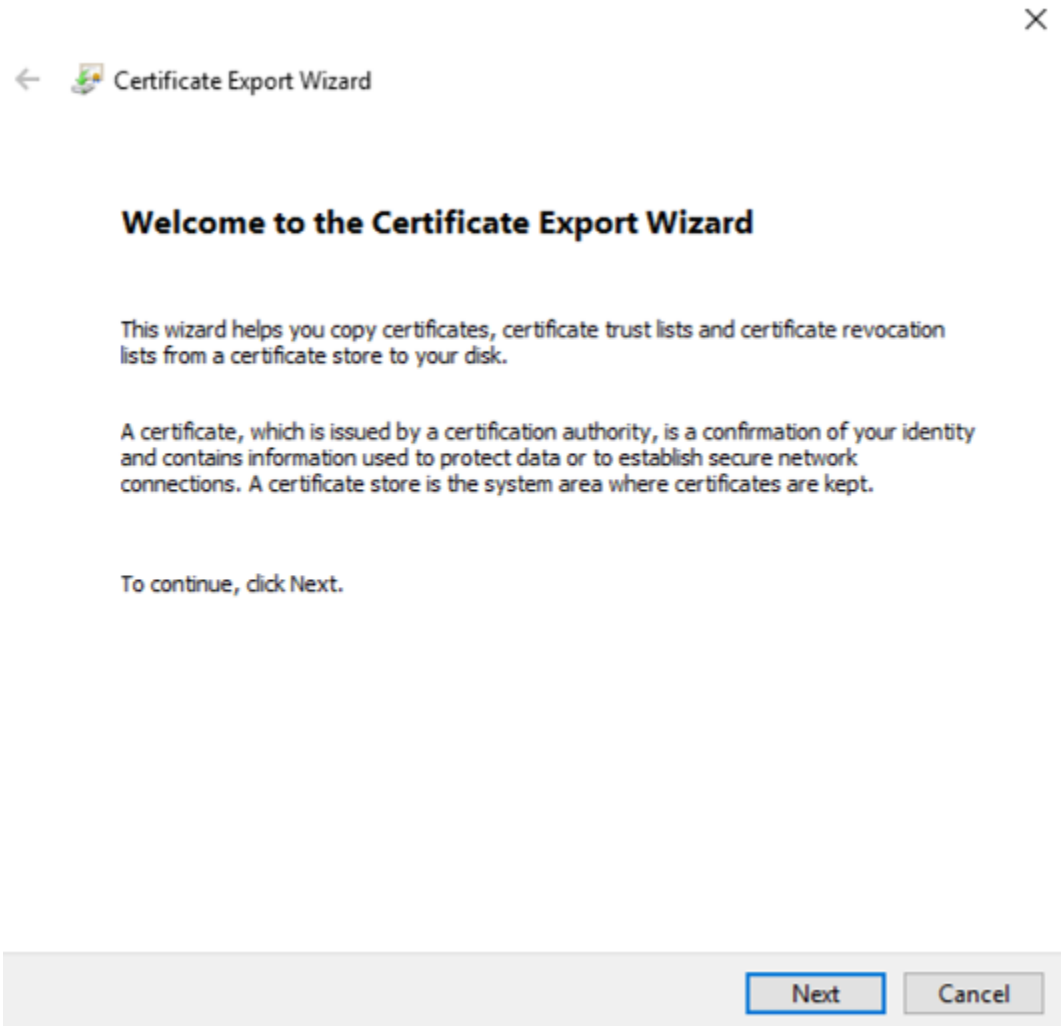
Select the certificate and open it



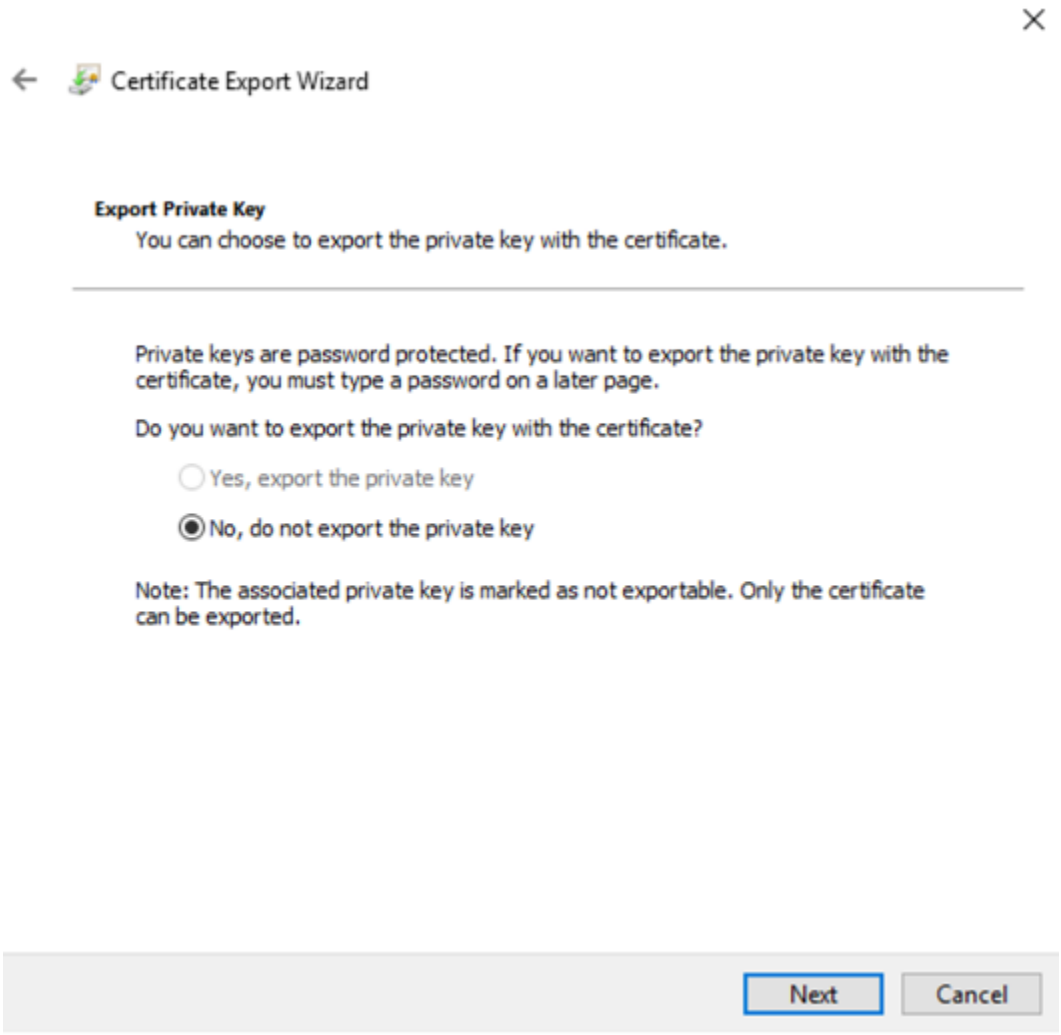
Select the option of copying to a file in the Details tab



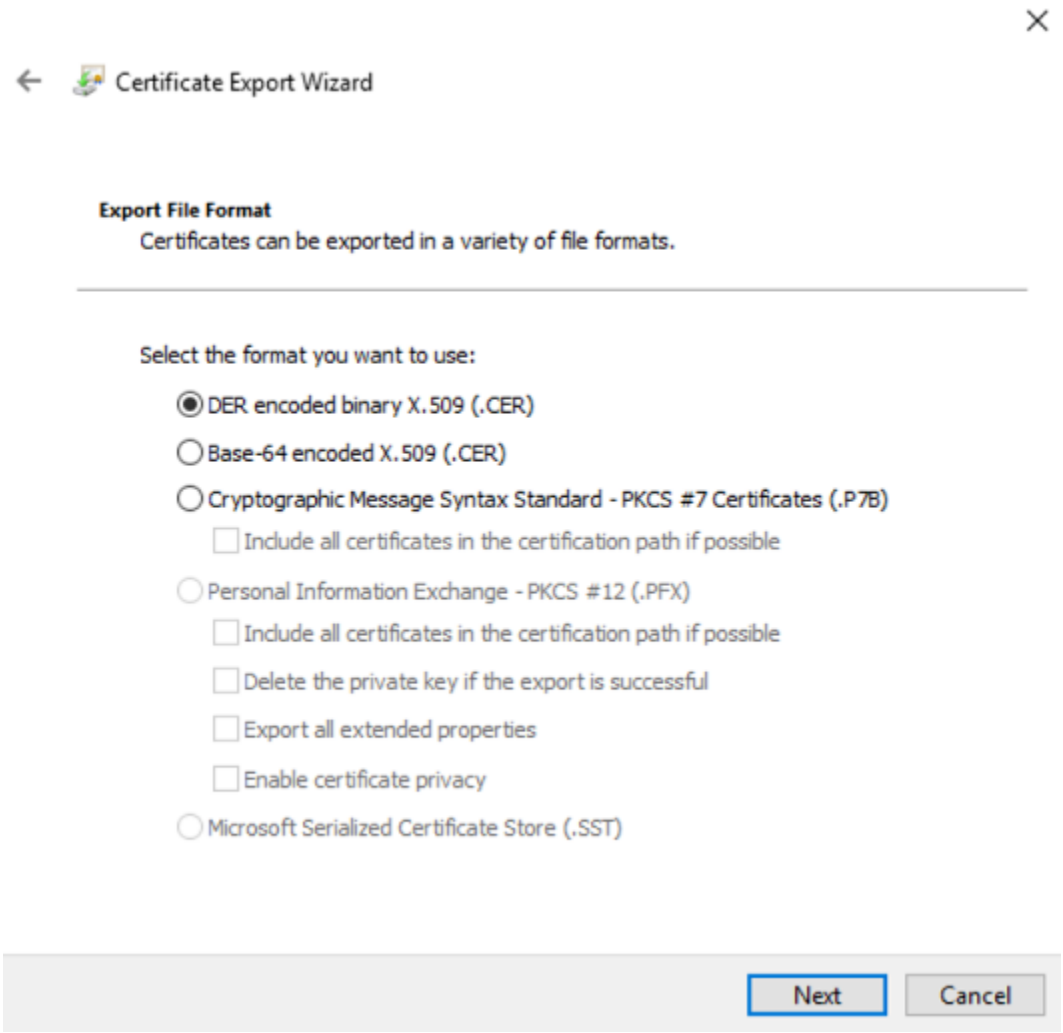
Click the Next button



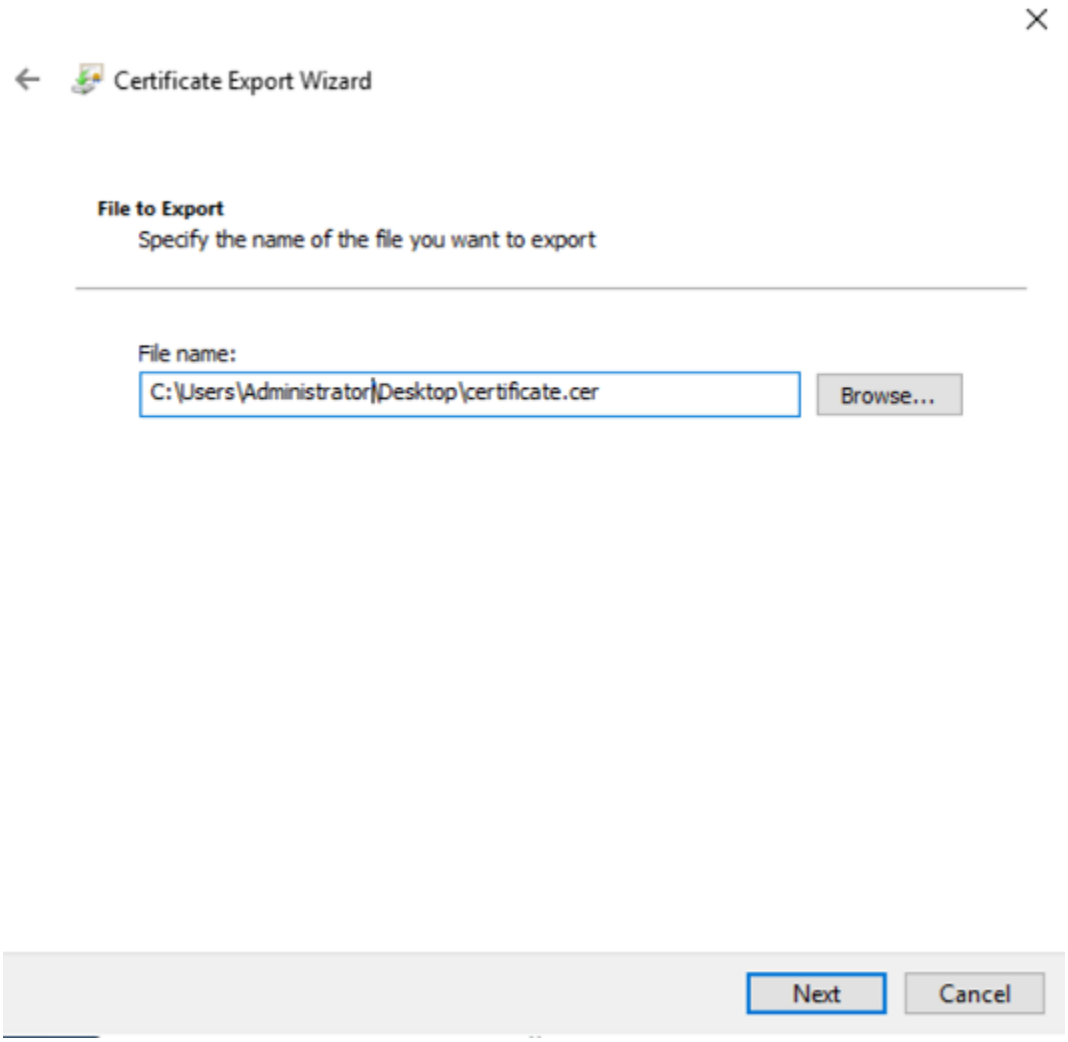
Keep the setting as shown below and click Next



Keep the setting as shown below and click Next.



Give the name a certificate



After the certificate is exported, this certificate should be imported into a trusted certificate file that will be used by the Data Node plugin.

To import a certificate into a trusted certificate file, a tool called „keytool.exe" is located in the JDK installation directory.

Use the following command to import a certificate file:

```
keytool -import -alias adding_certificate_keystore -file certificate.cer -
↪keystore certificatestore
```

The values `certificate.cer` and `certificatestore` should be changed accordingly.

By doing this, it will ask you to set a password for the trusted certificate store. Remember this password, because it must be set in the configuration of the Data Node plugin. The following settings must be set in the `properties.yml` configuration for SSL:

```
ssl.keystore.file: "<path to the trust certificate store>"
ssl.keystore.password: "<password to the trust certificate store>"
```

Role mapping

In the `/etc/logserver/properties.yml` configuration file you can find a section for configuring role mapping:

```
# LDAP ROLE MAPPING FILE
# rolemapping.file.path: /etc/logserver/role-mappings.yml
```

This variable points to the file `/etc/logserver/role-mappings.yml` Below is the sample content for this file:

```
# admin - superuser group
admin:
  - "CN=Admins,CN=Builtin,DC=example,DC=com"

gui-access:
  - "CN=Admins,CN=Builtin,DC=example,DC=com"
```

Attention. The role you define in the `role-mapping.yml` file must be created in the Energy Logserver.

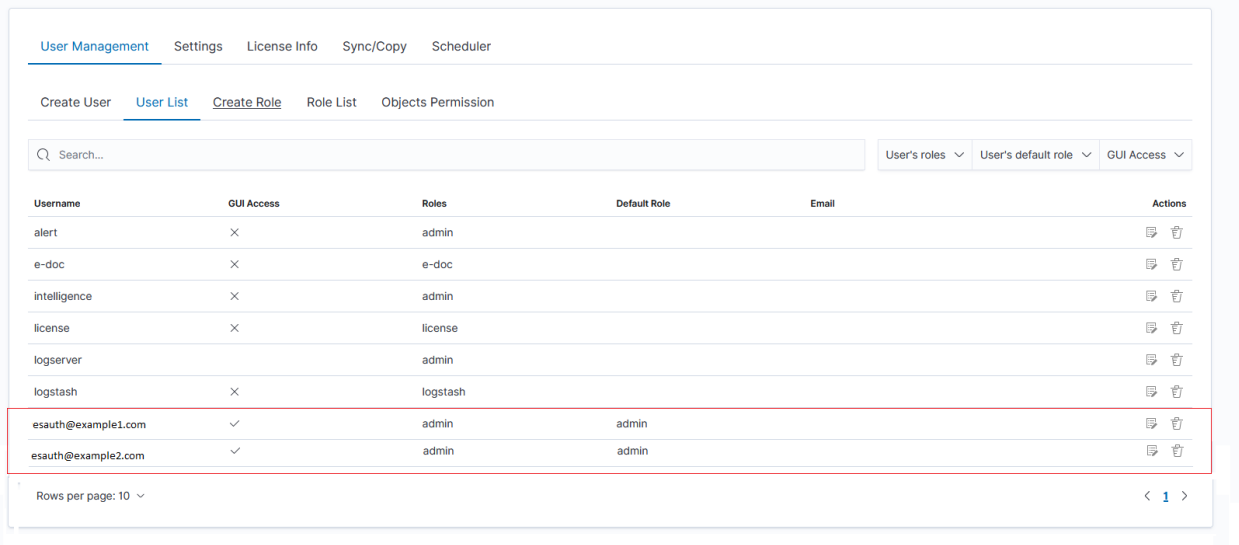
How does the mapping mechanism work?
An AD user logs in to Energy Logserver. In the application, there is a `gui-access` role, which through the file `role-mapping.yml` binds to the name of an AD group of which the user is a member. Additionally, this AD group binds to the Energy Logserver admin role, which points to permission granted to the user in the application.

Similarly, the mechanism will work for any other role in the application. Important in this configuration is to add every AD group to `gui-access` which grants permission to log in and at least one more role which grants permission to some data.

The `gui-access` role is not visible in GUI, it's only used to grant permission to log in.

If field `authentication_only` is `true`, user roles will not be mapped and they will be taken from the `default_authentication_roles` field. When the `default_authentication_roles` field is not added in properties.yml, a user without a role will be created.

Below is a screenshot of the console on which are marked accounts that were created by users logging in from AD



If you map roles from several domains, for example, dev.example1.com, and dev.example2.com then in the User List we will see which user from which domain with which role logged in Energy Logserver.

Password encryption

For security reasons, you can provide the encrypted password for Active Directory integration.
To do this use *pass-encrypter.sh* script that is located in the *Utils* directory in the installation folder.

1. Installation of *pass-encrypter*

```
cp -pr /installation_folder/logserver/pass-encrypter /usr/share/logserver/
```

2. Use *pass-encrypter*

```
/usr/share/logserver/utils/pass-encrypter/pass-encrypter.sh

Enter the string for encryption :
new_password
Encrypted string : MTU1MTEwMDcxMzQzMg==1GEG8KUOgyJko0PuT2C4uw==
```

Authentication with Radius

To use the Radius protocol, install the latest available version of Energy Logserver.

Configuration

The default configuration file is located at /etc/logserver/properties.yml:

```
# Radius opts
#radius.host: "10.4.3.184"
#radius.secret: "querty1q2ww2q1"
#radius.port: 1812
```

Use appropriate secret based on config file in Radius server. The secret is configured on `clients.conf` in the Radius server.

In this case, since the plugin will try to do Radius auth, the client IP address should be the IP address where the Data Node is deployed.

Every user by default at present gets the admin role

Authentication with LDAP

To use OpenLDAP authorization, install or update Energy Logserver too at least 7.0.2.

LDAP Configuration

The default configuration file is located at `/etc/logserver/properties.yml`:

- ldap_groups_search - Enable Open LDAP authorization. The `ldap_groups_search` switch with true/false values.
- search filter - you can define `search_filter` for each domain. When polling the LDAP / AD server, the placeholder is changed to the `userId` (everything before @domain) of the user who is trying to log in. Sample search_filter:

```
search_filter: "(&(objectClass=inetOrgPerson)(cn=%s))"
```

If no search_filter is given, the default will be used:

```
(&(&(objectCategory=Person)(objectClass=User))(samaccountname=%s))
```

- max_connections - for each domain (must be> = 1), this is the maximum number of connections that will be created with the LDAP / AD server for a given domain. Initially, one connection is created, if necessary another, up to the maximum number of connections set. If max_connections is not given, the default value = 10 will be used.
- ldap_groups_search - filter will be used to search groups on the AD / LDAP server to which the user is trying to log in. An example of groups_search_filter that works quite universally is:

```
groups_search_filter: "(|(uniqueMember=%s)(member=%s))"
```

Sample configuration:

```
licenseFilePath: /usr/share/logserver/

ldaps:

    - name: "dev.it.example.com"
      host: "192.168.0.1"
      port: 389                                                    #␣
↪optional, default 389
      #ssl_enabled: false                                         #␣
↪optional, default true
      #ssl_trust_all_certs: true                                  #␣
↪optional, default false
      bind_dn: "Administrator@dev2.it.example.com"
      bind_password: "password"
      search_user_base_DN: "OU=lab,DC=dev,DC=it,DC=example,DC=pl"
      search_filter: "(&(objectClass=inetOrgperson)(cn=%s))"       #␣
↪optional, default "(&(&
↪(objectCategory=Person)(objectClass=User))(samaccountname=%s))"
      user_id_attribute: "uid"                                     #␣
↪optional, default "uid"
      search_groups_base_DN: "OU=lab,DC=dev,DC=it,DC=example,DC=pl" #␣
↪base DN, which will be used for searching user's groups in LDAP tree
      groups_search_filter: "(member=%s)"                          #␣
↪optional, default (member=%s), if ldap_groups_search is set to true,␣
↪this filter will be used for searching user's membership of LDAP groups
      ldap_groups_search: false                                    #␣
↪optional, default false - user groups will be determined basing on user
↪'s memberOf attribute
      unique_member_attribute: "uniqueMember"                      #␣
↪optional, default "uniqueMember"
      max_connections: 10                                          #␣
↪optional, default 10
      connection_timeout_in_sec: 10                                #␣
↪optional, default 1
      request_timeout_in_sec: 10                                   #␣
↪optional, default 1
```

(continues on next page)

(continued from previous page)

```
    cache_ttl_in_sec: 60                                                    #␣
↪optional, default 0 - cache disabled
```

When the password is longer than 20 characters, we recommend using our pass-encrypter, otherwise, the backslash must be escaped with another backslash. Endpoint `role-mapping/_reload` has been changed to `_role-mapping/reload`. This is a unification of API conventions, following Data Node conventions.

Configuring Single Sign On (SSO)

To configure SSO, the system should be accessible by domain name URL, not IP address or localhost.

Correct: https://loggui.com:5601/login **Wrong:** https://localhost:5601/login, https://10.0.10.120:5601/login

The configuration description described below is based on:

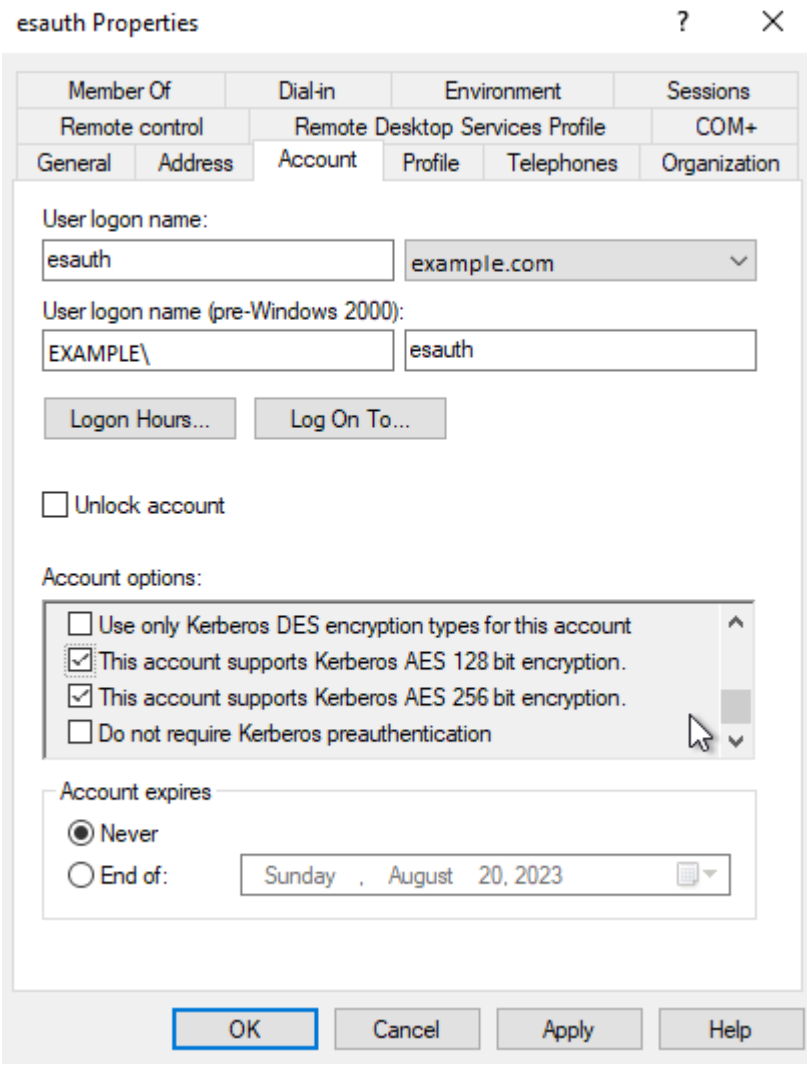
- AD Domain / Realm: `example.com`
- AD IP Address: `192.168.3.111`
- GUI Url: `loggui.com`

To enable SSO on your system, follow the steps below.

Configuration steps

1. Create a **User** Account for Data Node auth plugin In this step, a Kerberos Principal representing the Data Node auth plugin is created on the Active Directory. The principal name would be `name@EXAMPLE.COM`, while the `EXAMPLE.COM` is the administrative name of the realm.

   Create a User in AD. Set “Account never expires” and enable support for Kerberos encryption as shown below.



2. Define the Service Principal Name (SPN) and Create a keytab file for it

   Use the following command to create the keytab file identifying the SPN:

```
 C:> ktpass -out c:\Users\Administrator\esauth.keytab -princ HTTP/loggui.
↪com@EXAMPLE.COM -mapUser esauth -mapOp set -pass 'Sprint$123' -crypto␣
↪ALL -pType KRB5_NT_PRINCIPAL
```

   Details of the used switches:

   - `-out` - path to the keytab file
   - `-mapUser` - name of the previously created AD user. It might need to be preceded with pre-Windows 2000 logon if user cannot be found (e.g. `EXAMPLE\esauth` on the screenshot).
   - `-princ` - service principal name. Must start with uppercase HTTP/ and must end with uppercase domain after the @ sign. Will be used later to configure principal.
   - `-pass` - password that secures the keytab file itself (not connected user's password!). Will be used later to configure principal's password.

   For more details about the `ktpass tool`, please refer to the official documentation: ktpass details.

The `esauth.keytab` file should be placed on your Data Node, preferably `/etc/logserver/` with read permissions for Data Node user:

```
chmod 640 /etc/logserver/esauth.keytab
chown user: /etc/logserver/esauth.keytab
```

3. Create a file named *krb5Login.conf*:

```
  com.sun.security.jgss.initiate {
      com.sun.security.auth.module.Krb5LoginModule required
      principal="HTTP/loggui.com@EXAMPLE.COM"
      useKeyTab=true
      keyTab=/etc/logserver/esauth.keytab
      storeKey=true
      debug=true;
    };
  com.sun.security.jgss.krb5.accept {
    com.sun.security.auth.module.Krb5LoginModule required
    principal="HTTP/loggui.com@EXAMPLE.COM"
    useKeyTab=true
    keyTab=/etc/logserver/esauth.keytab
    storeKey=true
    debug=true;
  };
```

The principal user and keyTab location should be changed as per the values created in Step 2. Make sure the domain is in **UPPERCASE** as shown above.
The `krb5Login.conf` file should be placed on your Data Node, for instance, `/etc/logserver/` with read permissions for the Data Node user:

```
sudo chmod 640 /etc/logserver/krb5Login.conf
sudo chown user: /etc/logserver/krb5Login.conf
```

4. Uncomment and edit JVM arguments, in `/etc/logserver/jvm.options.d/single-sign-logon.options` as shown below:

```
-Dsun.security.krb5.debug=false \
-Djava.security.krb5.realm=**EXAMPLE.COM** \
-Djava.security.krb5.kdc=**192.168.3.111** \
-Djava.security.auth.login.config=/etc/logserver/krb5Login.conf \
-Djavax.security.auth.useSubjectCredsOnly=false
```

Change the `.krb5.realm` and `.krb5.kdc` to the appropriate values. `Realm` is defined as used domain (must be in UPPERCASE) realm and `.kdc` is AD's IP address. Those JVM arguments have to be set for the Logserver server.

5. Authentication options if `authentication_only: true` is set

   If a user does not exist, Logserver will create the user without a role. Role in `role-mapping.yml` would be ignored and role `gui-access` from `default_authentication_roles: ["gui-access"]` will be assigned.

6. Add the following additional (sso.domain, service_principal_name, service_principal_name_password) settings for LDAP in properties.yml file:

```
sso.domain: "example.com"
ldaps:
  - name: "example.com"
    host: "IP_address"
    port: 389                                    # optional, default 389
    ssl_enabled: false                           # optional, default␣
↪true
    ssl_trust_all_certs: false                   # optional, default␣
↪false
    bind_dn: "esauth@example.com"                # optional, skip for␣
↪anonymous bind
    bind_password: "password"                    # optional, skip for␣
↪anonymous bind (AD admin's password)
    search_user_base_DN: "cn=Users,DC=example,DC=com"
    user_id_attribute: "uid"                     # optional, default
↪"uid"
    unique_member_attribute: "uniqueMember"      # optional, default
↪"uniqueMember"
    service_principal_name: "HTTP/loggui.com@EXAMPLE.COM" # principal␣
↪name used while generating the keytab file
    service_principal_name_password: "Sprint$123"  # -pass used during␣
↪keytab creation
```

Note: At this moment, SSO works for only a single domain. So you have to mention for what domain SSO should work in the above property `sso.domain` - in this example it should be "example.com".

7. After completing the LDAP section entry in the properties.yml file, save the changes and send a request for reload authentication data with the command:

```
curl -sS -u username:password localhost:9200/_logserver/auth/reload -XPOST
```

8. Enable the SSO feature in the `logserver-gui.yml` file:

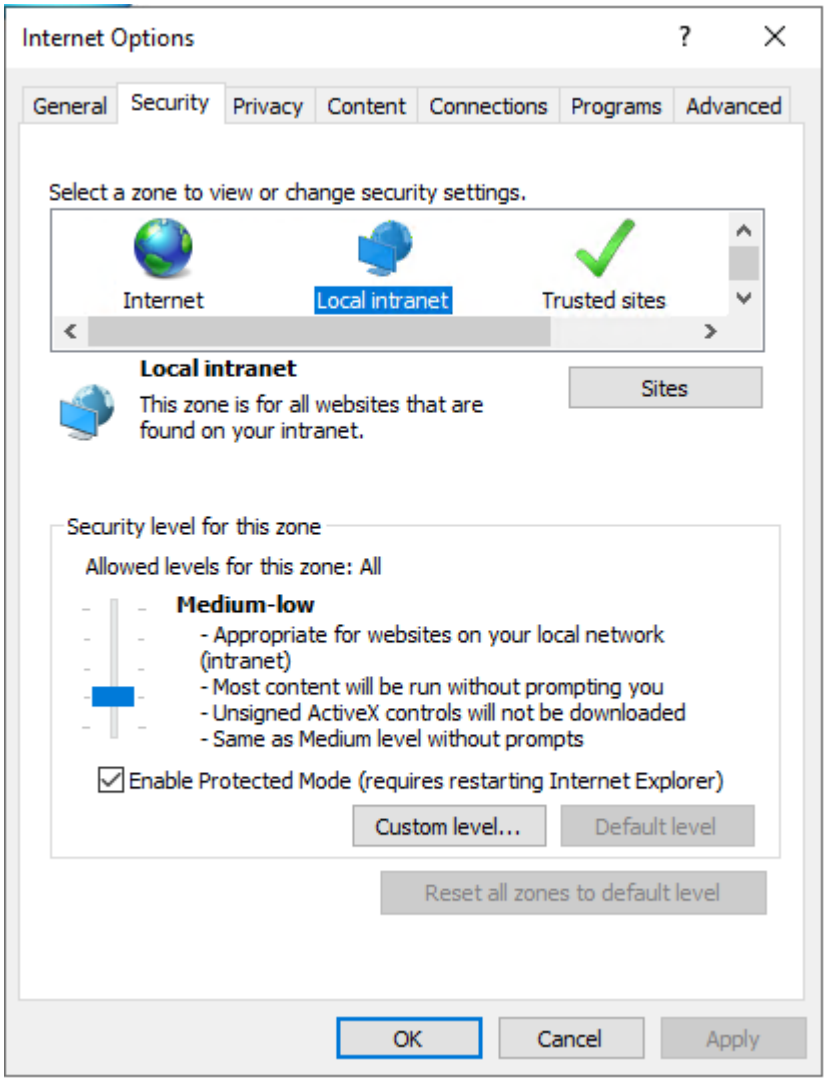
```
login.sso_enabled: true
```

9. After that Logserver GUI has to be restarted with systemctl restart command

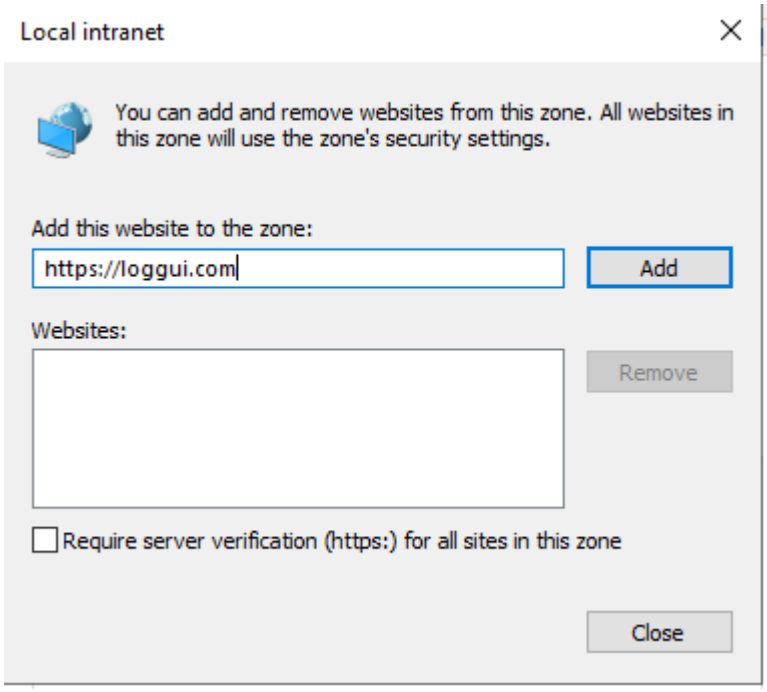
Client (Browser) Configuration

Internet Explorer configuration

1. Go to `Internet Options` from the `Tools` menu and click on `Security` Tab:

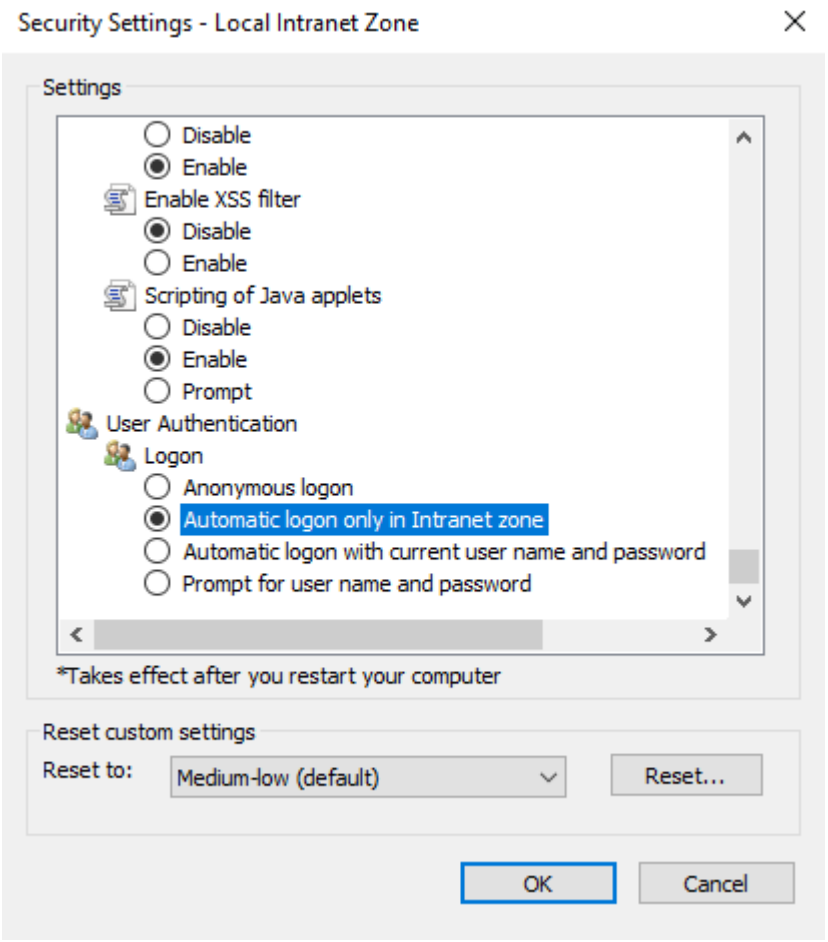


2. Select `Local intranet`, click on `Site` -> `Advanced` -> Provide correct URL -> Click `Add`:



After adding the site click close.

3. Click on the custom level and select the option as shown below:

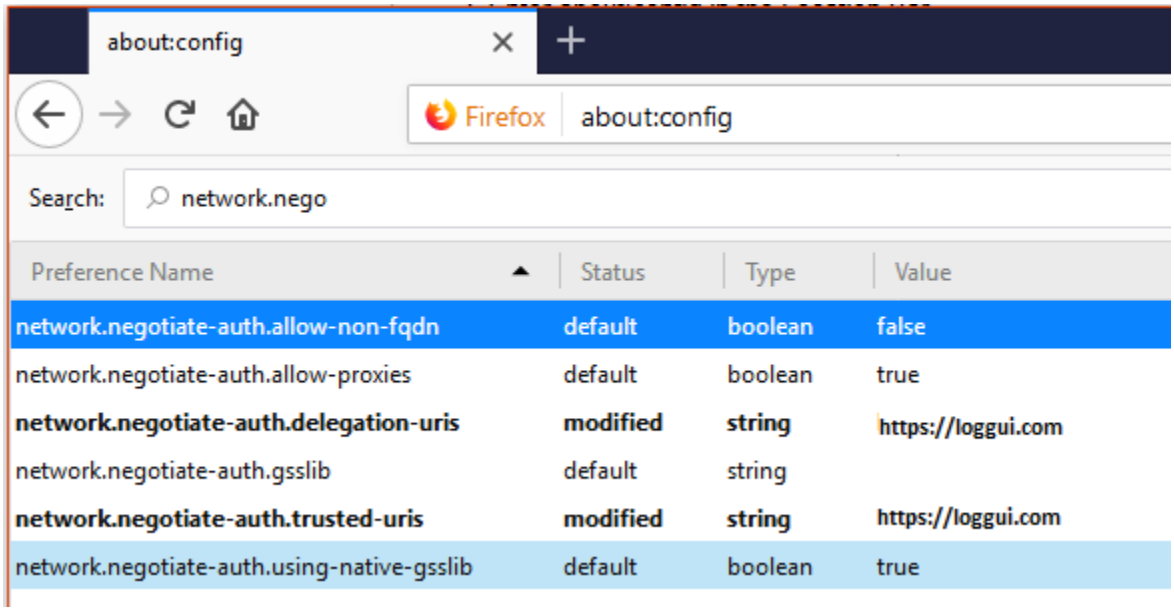


Chrome configuration

For Chrome, the settings are taken from the IE browser.

Firefox Configuration

Update the following config:



KDC error codes

KDC error codes	Description
0	No error
1	Client entry is expired
2	Server entry is expired
3	Protocol version is not supported
4	Client key is encrypted in an old master key
5	Server key is encrypted in an old master key
6	Client is not defined in the security registry
7	Server is not defined in the security registry
8	Principal is not unique in the security registry
9	No key is available for the principal
10	Ticket is not eligible for postdating
11	Ticket is never valid
12	Request rejected due to KDC policy
13	Request option is not supported
14	Encryption type is not supported
15	Checksum type is not supported
16	Preauthentication type is not supported
17	Transited data type is not supported
18	Client account is revoked
19	Server account is revoked
20	TGT is revoked
21	Client account is not valid yet
22	Server account is not valid yet
23	Password is expired
24	Preauthentication failed
25	Preauthentication required
26	Supplied authentication ticket is not for the requested server
27	Server requires user-to-user protocol

continues on next page

Table 17 – continued from previous page

KDC error codes	Description
31	Decryption integrity check failed
32	Ticket is expired
33	Ticket is not valid yet
34	Request is a replay of a previous request
35	Supplied authentication ticket is not for the current realm
36	Ticket and authenticator do not match
37	Clock skew is too great
38	Incorrect network address
39	Protocol version mismatch
40	Invalid message type
41	Message stream has been modified
42	Message is out of order
44	Key version is not available
45	Service key is not available
46	Mutual authentication failed
47	Incorrect message direction
48	Alternative authentication method required
49	Incorrect message sequence number
50	Inappropriate checksum type
60	Generic error detected
61	Field is too long
62	Client certificate is not acceptable
63	KDC certificate is not trusted or does not meet requirements
64	Certificate signature not valid
65	Client Diffie-Hellman key parameters not accepted
70	Client certificate could not be verified
71	Client certificate chain validation error occurred
72	Client certificate chain contains a revoked certificate
73	Revocation status for the certificate chain could not be determined
75	Kerberos client name does not match name bound to the client certificate
76	Kerberos KDC name does not match name bound to the KDC certificate
77	Key purpose restricts certificate usage
78	Certificate signature digest algorithm is not supported
79	PKAuthenticator is missing the required paChecksum
80	The signedData digest algorithm is not supported
81	The Public Key encryption delivery method is not supported

Two-Factor Authentication

Two-factor authentication (2FA) adds an additional verification layer on top of the standard Energy Logserver credentials. The two approaches below place NGINX in front of the Logserver-GUI (ELS Console, default port `5601`) and delegate the second factor to an external mechanism, either an OAuth2 identity provider or a client X.509 certificate.

2FA with OAuth2 Proxy and Google (example)

This integration uses oauth2-proxy as an authentication gateway between NGINX and Google as the identity provider.

Software used:

- NGINX, distribution package, or official NGINX repository (https://nginx.org/en/linux_packages.html)
- oauth2-proxy, https://github.com/oauth2-proxy/oauth2-proxy/releases

NGINX configuration:

1. Create `/etc/nginx/conf.d/ng_oauth2_proxy.conf` with the content below. The default server certificate and key generated by the Energy Logserver RPM are `/etc/logserver-gui/ssl/logserver-gui.crt` and `/etc/logserver-gui/ssl/logserver-gui.key`; replace `logserver.org.crt`/`logserver.org.key` accordingly or keep your organization's own certificate paths.

```
server {
    listen 443 default ssl;
    server_name logserver.local;
    ssl_certificate /etc/logserver-gui/ssl/logserver-gui.crt;
    ssl_certificate_key /etc/logserver-gui/ssl/logserver-gui.key;
    ssl_session_cache   builtin:1000  shared:SSL:10m;
    add_header Strict-Transport-Security max-age=2592000;

  location /oauth2/ {
    proxy_pass       http://127.0.0.1:4180;
    proxy_set_header Host                    $host;
    proxy_set_header X-Real-IP               $remote_addr;
    proxy_set_header X-Scheme                $scheme;
    proxy_set_header X-Auth-Request-Redirect $request_uri;
  }
  location = /oauth2/auth {
    proxy_pass       http://127.0.0.1:4180;
    proxy_set_header Host             $host;
    proxy_set_header X-Real-IP        $remote_addr;
    proxy_set_header X-Scheme         $scheme;
    proxy_set_header Content-Length   "";
    proxy_pass_request_body           off;
  }

  location / {
    auth_request /oauth2/auth;
    error_page 401 = /oauth2/sign_in;

    auth_request_set $user   $upstream_http_x_auth_request_user;
    auth_request_set $email  $upstream_http_x_auth_request_email;
    proxy_set_header X-User  $user;
    proxy_set_header X-Email $email;

    auth_request_set $token  $upstream_http_x_auth_request_access_token;
    proxy_set_header X-Access-Token $token;

    auth_request_set $auth_cookie $upstream_http_set_cookie;
    add_header Set-Cookie $auth_cookie;
```

(continues on next page)

(continued from previous page)

```
    auth_request_set $auth_cookie_name_upstream_1 $upstream_cookie_auth_
↪cookie_name_1;

    if ($auth_cookie ~* "(; .*)") {
        set $auth_cookie_name_0 $auth_cookie;
        set $auth_cookie_name_1 "auth_cookie__oauth2_proxy_1=$auth_cookie_
↪name_upstream_1$1";
    }

    if ($auth_cookie_name_upstream_1) {
        add_header Set-Cookie $auth_cookie_name_0;
        add_header Set-Cookie $auth_cookie_name_1;
    }

    proxy_pass https://127.0.0.1:5601;
  }
}
```

2. Set `ssl_certificate` and `ssl_certificate_key` to the certificate and key you intend to serve.

When SSL is terminated by NGINX, the Logserver-GUI can be started over HTTP. If encryption on the upstream is required, adjust `proxy_pass` accordingly.

oauth2-proxy configuration:

1. Prepare `oauth2_proxy.cfg`.
2. Create directories for the binary and configuration:

```
mkdir -p /usr/share/oauth2_proxy/
mkdir -p /etc/oauth2_proxy/
```

3. Copy files into directories:

```
cp oauth2_proxy /usr/share/oauth2_proxy/
cp oauth2_proxy.cfg /etc/oauth2_proxy/
```

4. Set directives according to the OAuth configuration created in the Google Cloud project:

```
client_id =
client_secret =
# the following limits domains for authorization (* - all)
email_domains = [
  "*"
]
```

5. Set the public hostname:

```
cookie_domain = "logserver-host.org"
```

6. Optional, restrict login to a specific Google group:
   - Create a service account: https://developers.google.com/identity/protocols/oauth2/service-account
   - Download the service account credentials as a JSON file and copy the Client ID.

- In the Google Cloud console go to **APIs & Services** → **APIs**, and enable the **Admin SDK API**.
- Follow https://developers.google.com/workspace/guides/create-credentials and grant the service account the following OAuth scopes:

```
https://www.googleapis.com/auth/admin.directory.group.readonly
https://www.googleapis.com/auth/admin.directory.user.readonly
```

- Follow https://support.google.com/a/answer/60757 to grant Admin API access.
- Select or create an administrative email in the Gmail domain, this is the value for `google-admin-email`.
- Select or create the group used for login, this is the value for `google-group`.
- Copy the downloaded JSON file to `/etc/oauth2_proxy/`.
- In `oauth2_proxy.cfg` set the path to the JSON credentials:

```
google_service_account_json =
```

Service start-up:

- Start the NGINX service.
- Start the oauth2-proxy service:

```
/usr/share/oauth2_proxy/oauth2_proxy -config="/etc/oauth2_proxy/oauth2_
↪proxy.cfg"
```

Open a browser and navigate to the Energy Logserver host address, the second factor is now enforced by Google before the request reaches the Console. The Console still presents its own login afterwards; the gateway adds a factor in front of it, it does not replace the Console login. See *SSO gateway with Keycloak and oauth2-proxy* for the same pattern with an OIDC provider and an explanation of that behavior.

SSO gateway with Keycloak and oauth2-proxy

This integration places a gateway built from NGINX and oauth2-proxy in front of the Logserver-GUI (ELS Console) and enforces a Keycloak login, including any MFA the realm requires, before a request reaches the Console. It was built and verified end to end on a test server (Oracle Linux 9.7, ELS 8.x). The provider is OIDC, so the same steps apply to Entra, Okta, or any modern IdP after adjusting the issuer and client values.

What this gives you, and what it does not

The Console authenticates users only against its own backends: local accounts, LDAP/Active Directory, Radius, or Kerberos/SPNEGO. Its login plugin does not read an upstream identity from request headers, and it has no native OIDC or SAML support. There is no configuration switch to make it trust a proxy.

A gateway built from NGINX and oauth2-proxy therefore gives you:

- **A Keycloak login enforced before anyone reaches the Console.** This is where MFA lives. Keycloak, or any IdP it federates, decides who passes.
- **One identity provider for the edge.** oauth2-proxy speaks OIDC/OAuth2, so it links to Keycloak, Entra, Okta, and almost any modern IdP.

It does not give you a single login that flows into the Console session. After Keycloak authenticates the user, the Console still shows its own login screen, and the user authenticates a second time against the Console (the same Active Directory, but a separate prompt).

If the requirement is to force MFA in front of the Console, this is the right design. If the requirement is one login with no second prompt, use native Kerberos/SPNEGO instead, see *Configuring Single Sign On (SSO)*. That path requires domain-joined workstations and an AD/KDC.

Architecture

```
Browser
  |
  |  https://els.example.com  (443)
  v
nginx  ----auth_request---->  oauth2-proxy (127.0.0.1:4180)
  |                                  |
  |                                  |  OIDC
  |                                  v
  |                            Keycloak  (login + MFA)
  |
  |  after the gateway authenticates, proxy_pass
  v
ELS Console (https://127.0.0.1:5601)  ->  still serves its own /login
```

NGINX owns port 443. Every request runs through an `auth_request` subrequest to oauth2-proxy. No valid session means a redirect to Keycloak; a valid session means NGINX forwards the request to the Console on 5601.

Prerequisites

- A running node with the Console on port 5601 (HTTPS, self-signed by default).
- Root access to that node.
- Network reachability from the node to Keycloak, and from the user's browser to both the node and Keycloak.
- A DNS name or stable IP for the node. This procedure uses `els.example.com`; substitute your own.
- From the Keycloak administrator: a realm and a confidential client (details in step 1).

1. Create the Keycloak client

Create a confidential client in the realm that holds the Console users.

- **Client ID:** `els-proxy` (any name, keep it consistent below)
- **Client type:** confidential (client authentication on)
- **Standard flow:** enabled
- **Valid redirect URI:** `https://els.example.com/oauth2/callback`
- **Web origin:** `https://els.example.com`

Collect three values:

- **Issuer URL**, for example `https://keycloak.example.com/realms/corp`. Confirm it by opening `<issuer>/.well-known/openid-configuration` and reading the `issuer` field; it must match exactly.
- **Client ID**
- **Client secret**

2. Free port 443 on the node

A default install publishes the Console on 443 with a firewalld forward that redirects 443 to 5601. That redirect runs in PREROUTING, so it captures external traffic before NGINX sees it. If you skip this step, the browser reaches the Console directly and the gateway never runs, while loopback tests still hit NGINX and look fine.

Check for the forward:

```
firewall-cmd --list-forward-ports
```

If you see `port=443:proto=tcp:toport=5601:toaddr=`, remove it:

```
firewall-cmd --permanent --remove-forward-port=port=443:proto=tcp:toport=5601
firewall-cmd --reload
```

The Console stays reachable directly on `https://<node>:5601`; port 443 now belongs to NGINX. To keep the Console on 443 untouched, run NGINX on another port (for example 8443) and adjust the redirect URI and `redirect_url` below accordingly. For background on this redirect, see *GUI Port and HTTPS Redirect*.

3. Install the packages

```
dnf install -y nginx
```

Install oauth2-proxy from its release page:

```
VER=v7.6.0
cd /tmp
curl -sL -o oauth2-proxy.tar.gz \
  https://github.com/oauth2-proxy/oauth2-proxy/releases/download/${VER}/
↪oauth2-proxy-${VER}.linux-amd64.tar.gz
tar xzf oauth2-proxy.tar.gz
install oauth2-proxy-${VER}.linux-amd64/oauth2-proxy /usr/local/bin/
oauth2-proxy --version
```

4. TLS certificate for the gateway

Use the customer certificate for `els.example.com` in production. For a lab, generate a self-signed one:

```
mkdir -p /etc/nginx/els-proxy
openssl req -x509 -newkey rsa:2048 -nodes -days 825 \
  -keyout /etc/nginx/els-proxy/server.key \
  -out /etc/nginx/els-proxy/server.crt \
  -subj "/CN=els.example.com" \
  -addext "subjectAltName=DNS:els.example.com"
```

5. Configure oauth2-proxy

Generate a cookie secret:

```
head -c32 /dev/urandom | base64 | tr '+/' '-_' | cut -c1-43
```

Write `/etc/oauth2-proxy/oauth2-proxy.cfg`:

```
http_address = "127.0.0.1:4180"
reverse_proxy = true

provider        = "oidc"
oidc_issuer_url = "https://keycloak.example.com/realms/corp"
client_id       = "els-proxy"
client_secret   = "PASTE_CLIENT_SECRET"

redirect_url = "https://els.example.com/oauth2/callback"

email_domains = ["*"]
scope         = "openid email profile"

# expose identity headers to nginx
set_xauthrequest = true
pass_access_token = true
pass_user_headers = true

cookie_secret = "PASTE_COOKIE_SECRET"
cookie_secure = true
cookie_name   = "_els_oauth2"

# send the user straight to Keycloak instead of an intermediate button
skip_provider_button = true
```

Create the service `/etc/systemd/system/oauth2-proxy.service`:

```
[Unit]
Description=oauth2-proxy for the ELS Console
After=network.target

[Service]
ExecStart=/usr/local/bin/oauth2-proxy --config=/etc/oauth2-proxy/oauth2-proxy.
↪cfg
Restart=always
RestartSec=3

[Install]
WantedBy=multi-user.target
```

Start it:

```
systemctl daemon-reload
systemctl enable --now oauth2-proxy
journalctl -u oauth2-proxy -n 20 --no-pager
```

The log should report `OAuthProxy configured for OpenID Connect Client ID: els-proxy`. An error here usually means a wrong issuer URL or an unreachable Keycloak.

6. Configure NGINX

Write `/etc/nginx/conf.d/els-proxy.conf`:

```
server {
    listen 443 ssl;
    server_name els.example.com;
    access_log /var/log/nginx/els_access.log;

    ssl_certificate     /etc/nginx/els-proxy/server.crt;
    ssl_certificate_key /etc/nginx/els-proxy/server.key;

    # oauth2-proxy endpoints
    location /oauth2/ {
        proxy_pass       http://127.0.0.1:4180;
        proxy_set_header Host                    $host;
        proxy_set_header X-Real-IP               $remote_addr;
        proxy_set_header X-Scheme                $scheme;
        proxy_set_header X-Auth-Request-Redirect $request_uri;
    }

    location = /oauth2/auth {
        proxy_pass       http://127.0.0.1:4180;
        proxy_set_header Host           $host;
        proxy_set_header X-Real-IP      $remote_addr;
        proxy_set_header X-Scheme       $scheme;
        proxy_set_header Content-Length "";
        proxy_pass_request_body         off;
    }

    # protect everything else, then forward to the Console
    location / {
        auth_request /oauth2/auth;
        error_page 401 = /oauth2/sign_in;

        auth_request_set $user  $upstream_http_x_auth_request_user;
        auth_request_set $email $upstream_http_x_auth_request_email;

        proxy_set_header X-User  $user;
        proxy_set_header X-Email $email;

        proxy_pass https://127.0.0.1:5601;
        proxy_ssl_verify off;
        proxy_set_header Host              $host;
        proxy_set_header X-Real-IP         $remote_addr;
        proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
    }
}
```

Allow NGINX to open outbound connections (SELinux blocks this by default), open the firewall, and start NGINX:

```
setsebool -P httpd_can_network_connect 1
firewall-cmd --add-port=443/tcp --permanent
```

(continues on next page)

(continued from previous page)

```
firewall-cmd --reload
nginx -t
systemctl enable --now nginx
```

The `X-User` and `X-Email` headers are passed for completeness. The Console ignores them; leaving them in place does no harm and documents the intent.

Verification

Run these from a machine that can reach the node.

Confirm NGINX, not the Console, answers on 443. The certificate subject must be your gateway certificate, not `CN=LOGSERVER`:

```
echo | openssl s_client -connect els.example.com:443 -servername els.example.
↪com 2>/dev/null \
  | openssl x509 -noout -subject
```

Confirm an unauthenticated request is sent to Keycloak:

```
curl -sk -D - -o /dev/null https://els.example.com/ | grep -i ^location
# expect: Location: https://keycloak.example.com/realms/corp/protocol/openid-
↪connect/auth?...
```

Then open `https://els.example.com/` in a browser. The expected sequence is the Keycloak login page (with MFA if the realm enforces it), and after login the Console login page. The second prompt is the documented limitation, not a fault. The NGINX access log shows the full chain:

```
GET /                          302   -> Keycloak
GET /oauth2/callback?code=     302   -> oauth2-proxy validated the token, set␣
↪the session
GET /                          302   -> forwarded to the Console
GET /login                     200   -> Console serves its own login
```

Troubleshooting

- **The browser lands on the Console login directly and never sees Keycloak.** The 443→5601 firewalld forward from step 2 is still active. Verify with `firewall-cmd --list-forward-ports` and remove it. A quick check: compare the TLS certificate from the browser side (`CN=LOGSERVER` means you are hitting the Console directly) against loopback on the node (`CN=els.example.com` means you reached NGINX).
- **oauth2-proxy fails to start with an issuer error.** The `oidc_issuer_url` must match the `issuer` field in `<issuer>/.well-known/openid-configuration` character for character, including http versus https and any trailing path.
- **NGINX returns 500 on every request.** The `auth_request` subrequest is failing. Check that oauth2-proxy is listening on 127.0.0.1:4180 and that `setsebool -P httpd_can_network_connect 1` was applied.
- **Redirect loop after Keycloak login.** The redirect URI in the Keycloak client does not match `redirect_url` in oauth2-proxy, or `cookie_secure = true` is set while the user reaches the site over plain HTTP. Align the URIs and keep the whole path on HTTPS.

2FA with Nginx and PKI client certificate

This integration places NGINX in front of the Logserver-GUI (ELS Console) and requires clients to present a valid X.509 certificate signed by a dedicated internal CA. On RHEL/Rocky/AlmaLinux the NGINX package from nginx.org runs under the `nginx` user and group.

1. Install NGINX

Follow the official NGINX installation guide: https://docs.nginx.com/nginx/admin-guide/installing-nginx/installing-nginx-open-source/

2. Create the client-certificate signing CA

Create a directory for the CA material:

```
cd /etc/nginx
mkdir CertificateAuthCA
cd CertificateAuthCA
chown root:nginx /etc/nginx/CertificateAuthCA/
chmod 770 /etc/nginx/CertificateAuthCA/
```

These permissions grant the user **root** (or the privileged user on your box) and the **nginx** group access. No one else should read this directory, it holds the root signing key.

Generate the CA private key. You will be prompted for a passphrase:

```
openssl genrsa -des3 -out myca.key 4096
```

Generate a CA certificate valid for 10 years (adjust as required; you will be prompted for CA attributes):

```
openssl req -new -x509 -days 3650 -key myca.key -out myca.crt
```

3. Create a client keypair

Repeat for **each user**: this can be scripted as part of user provisioning. You will be prompted for a passphrase, which will be distributed to the user together with the certificate.

> **Note**
>
> Never distribute the passphrase of the root CA private key from step 2.

```
openssl genrsa -des3 -out testuser.key 2048
openssl req -new -key testuser.key -out testuser.csr
```

Sign the client certificate with the CA. Valid for one year by default; increment the serial if you need to reissue:

```
openssl x509 -req -days 365 -in testuser.csr -CA myca.crt -CAkey myca.key -
↪set_serial 01 -out testuser.crt
```

For Windows clients, combine the key material into a single PFX file (you will be prompted for the passphrase set above):

```
openssl pkcs12 -export -out testuser.pfx -inkey testuser.key -in testuser.crt␣
↪-certfile myca.crt
```

Including the public portion of the CA in the PFX allows Windows to trust the internally signed CA.

4. Prepare the NGINX site certificate

Tighten directory permissions before generating site material:

```
chown -R root:nginx /etc/nginx/CertificateAuthCA
chmod 700 /etc/nginx/CertificateAuthCA
```

Generate the server key (you will be prompted for a passphrase and attributes):

```
openssl genrsa -out ./domain.com.key 2048
```

Create a CSR:

```
openssl req -new -sha256 -key ./domain.com.key -out ./domain.com.csr
```

Sign the server certificate:

```
openssl x509 -req -days 365 -in domain.com.csr -CA myca.crt -CAkey myca.key -
↪set_serial 01 -out domain.com.crt
```

Remove the passphrase from the server key (you will be prompted for the passphrase set above):

```
openssl rsa -in domain.com.key -out domain.com.key.nopass
```

Create the NGINX `sites-available` directory and an empty configuration file:

```
cd /etc/nginx
mkdir sites-available
cd sites-available
touch proxy.conf
```

5. NGINX site configuration

Make sure firewalld is installed and enabled. Paste the following into `proxy.conf`:

```
server {
    listen 443;  ## The listen port must match the firewall rule.
    ssl on;
    server_name 192.168.3.87;  ## Your IP as server_name
    proxy_ssl_server_name on;
    ssl_certificate         /etc/nginx/CertificateAuthCA/domain.com.crt;
↪ ## Server certificate
    ssl_certificate_key     /etc/nginx/CertificateAuthCA/domain.com.key.nopass;
↪ ## Server key (passphrase removed)
    ssl_client_certificate /etc/nginx/CertificateAuthCA/myca.crt;
↪ ## CA certificate used to verify client certs

    ssl_verify_client on;

    ## Optionally capture error codes and redirect to a custom page:
    ## error_page 495 496 497 https://someerrorpage.yourdomain.com;

    ssl_prefer_server_ciphers on;
    ssl_protocols TLSv1.1 TLSv1.2;
    ssl_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-
↪SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-
```

(continues on next page)

(continued from previous page)

```
↪SHA384:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-
↪RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-
↪AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-
↪SHA256:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA:AES128-
↪GCM-SHA256:AES256-GCM-SHA384:ECDHE-RSA-RC4-SHA:ECDHE-ECDSA-RC4-SHA:RC4-
↪SHA:HIGH:!aNULL:!eNULL:!EXPORT:!DES:!3DES:!MD5:!PSK';

    keepalive_timeout 10;
    ssl_session_timeout 5m;

    location / {
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $http_host;
        proxy_redirect off;
        proxy_set_header X-Forwarded-Proto https;
        proxy_pass http://localhost:5601/;    ## proxy_pass for Logserver-GUI␣
↪(ELS Console)
    }
}
```

6. Enable the site in NGINX

`nginx.conf` loads modular configs from `/etc/nginx/conf.d/*.conf`. Create a symlink:

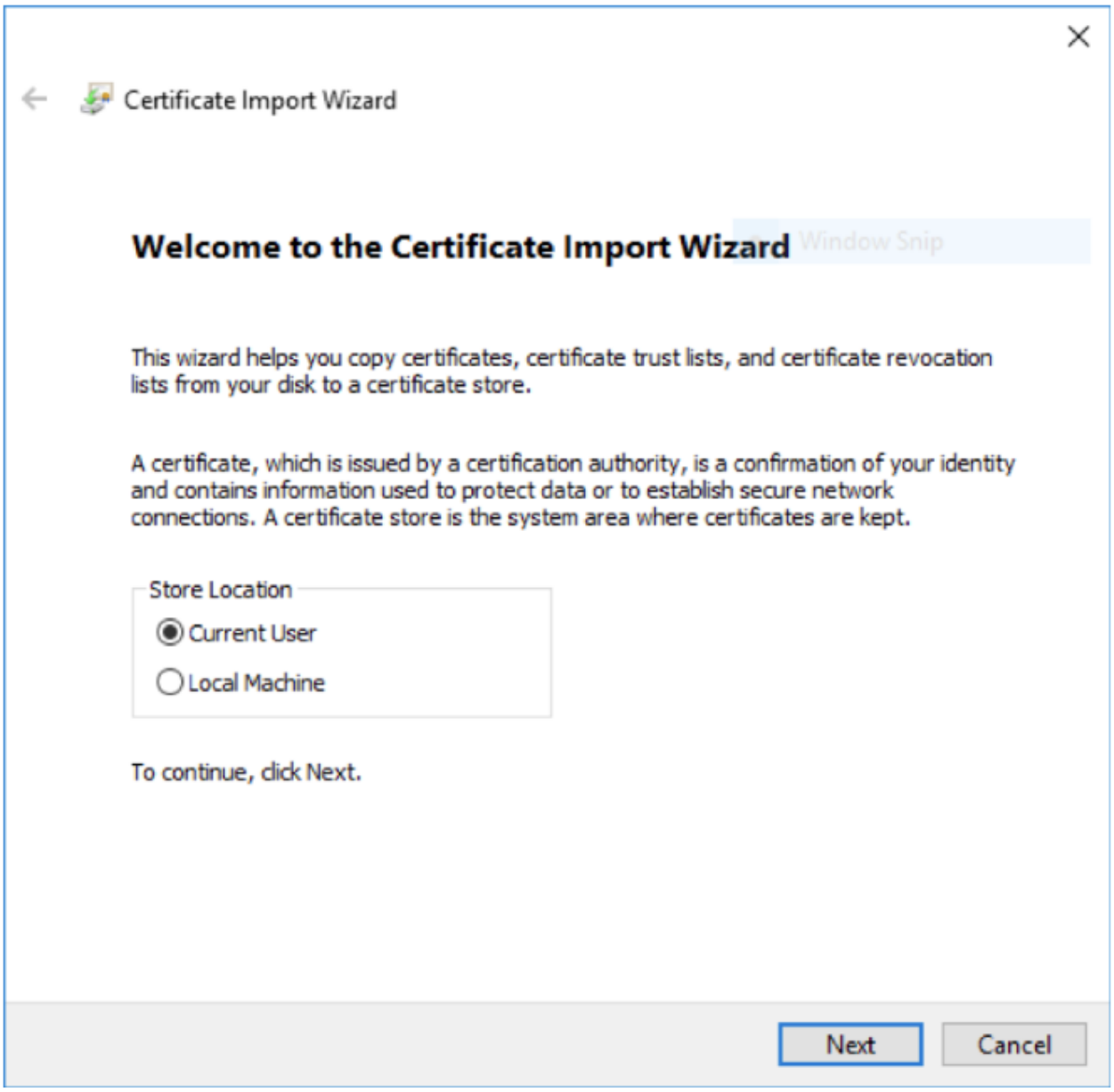
```
cd /etc/nginx
ln -s /etc/nginx/sites-available/proxy.conf /etc/nginx/conf.d/proxy.conf
```

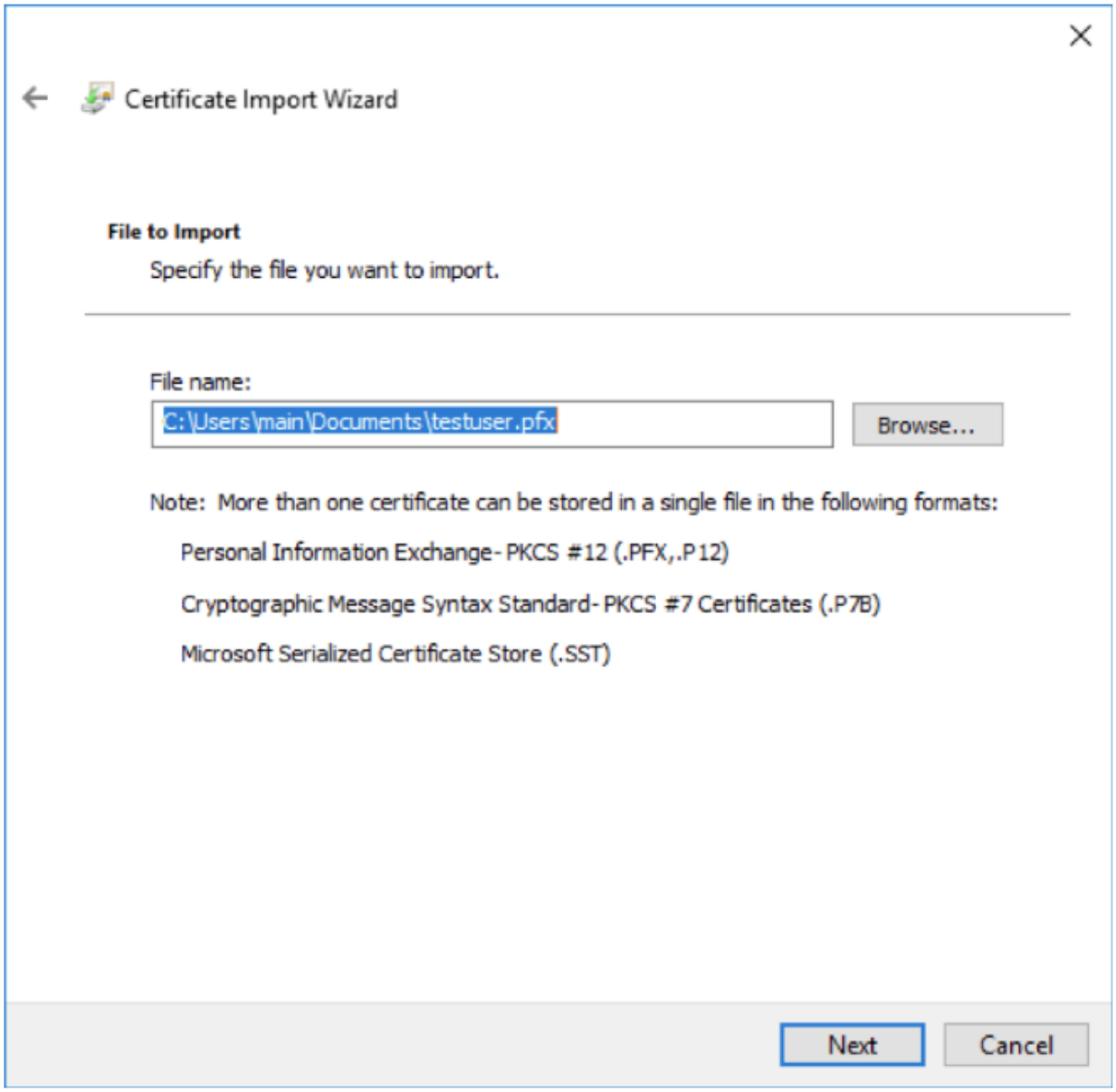
7. Restart NGINX

```
systemctl restart nginx
```

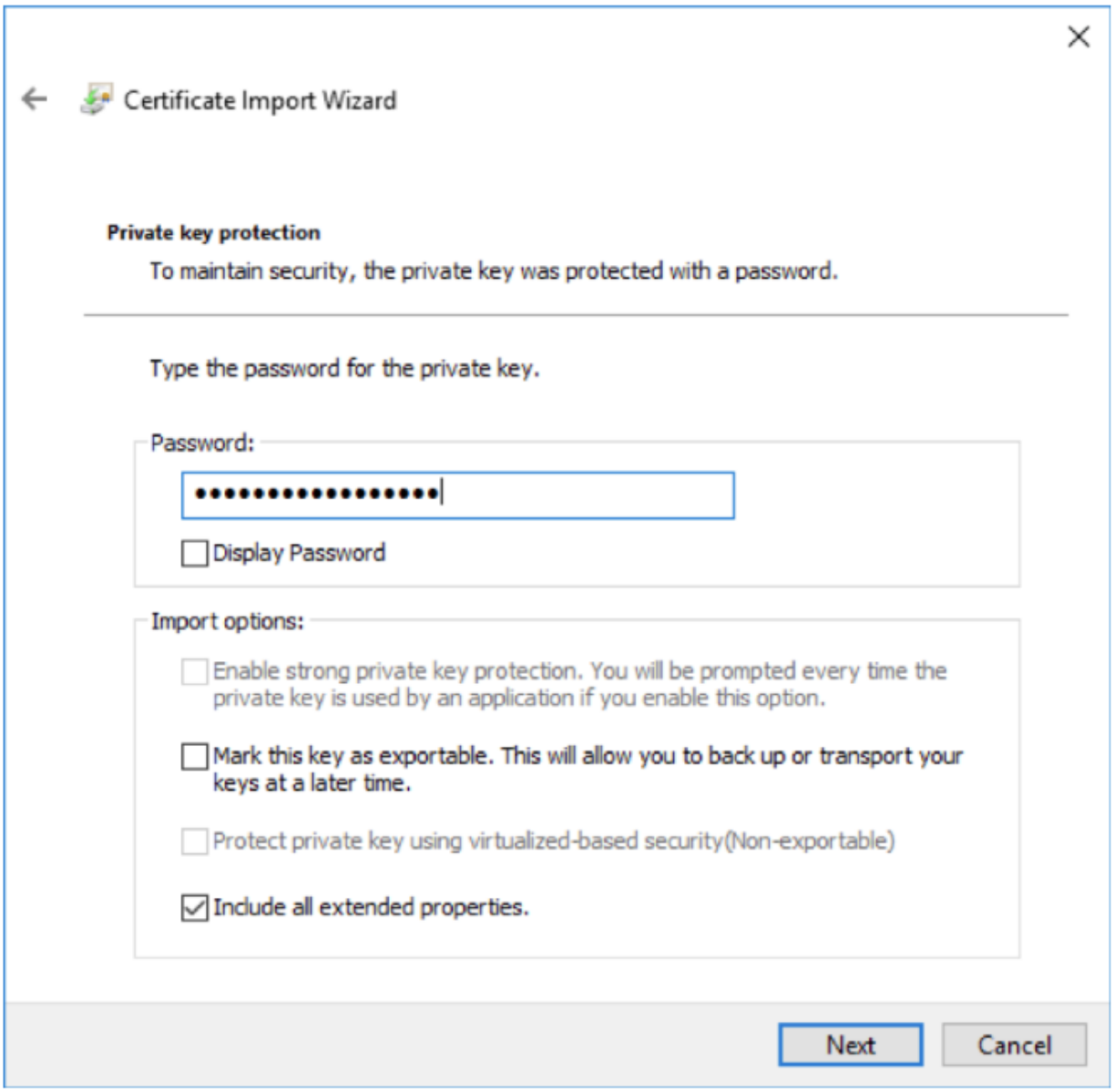
8. Import the client certificate on a Windows machine

Double click the `.pfx` file and select **Current User**.


Certificate Import Wizard
Welcome to the Certificate Import Wizard
This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.
A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.
Store Location
Current User
Local Machine
To continue, click Next.
Next
Cancel

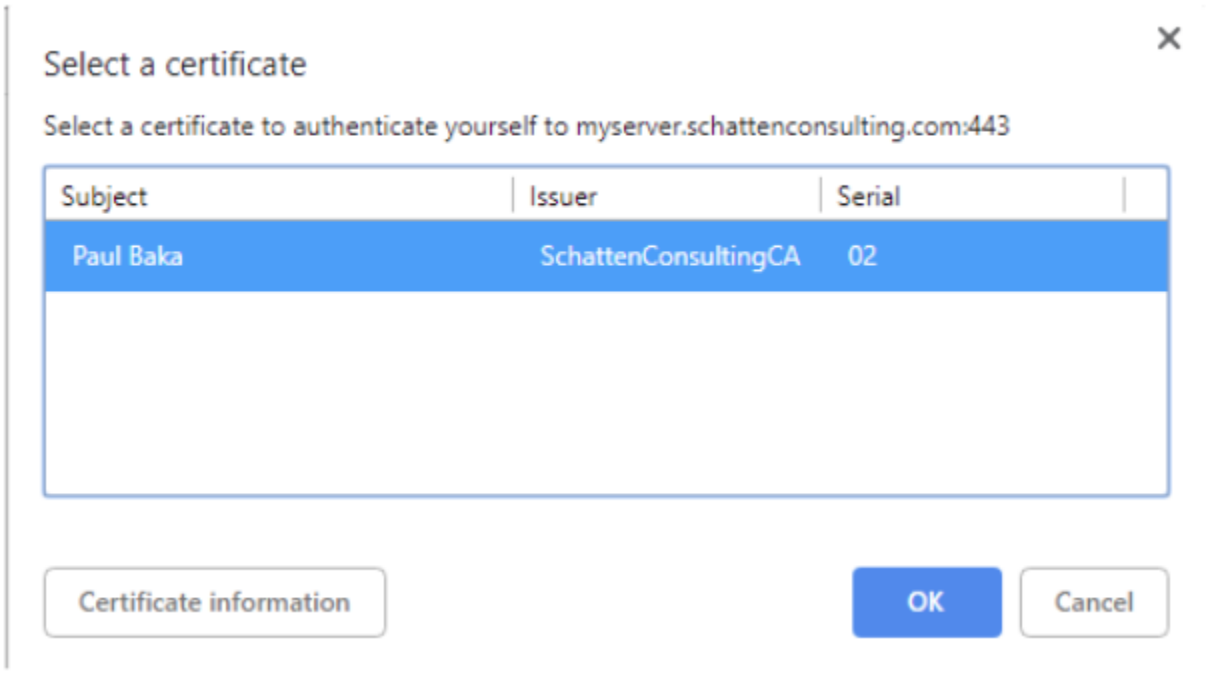


If the PFX has a passphrase, enter it here. Otherwise leave it blank and continue.



Add the target site to **Trusted Sites** in Internet Explorer. This allows the client certificate to be sent to the site (trusting it in Internet Explorer also trusts it in Chrome).

On the next visit the browser prompts to select a client certificate. Select the imported one and confirm.



Event Collector

The Event Collector allows you to get events from remote Windows computers and store them in the Energy Logserver indexes. The destination log path for the events is a property of the subscription. The Energy Logserver Event Collector allows to definition of an event subscription on an Energy Logserver collector without defining the event source computers. Multiple remote event source computers can then be set up (using for example a group policy setting) to forward events to the Energy Logserver. The Event Collector doesn't require installation of any additional applications/agents on Windows source hosts.

- *Configuration steps*
  - *Installation of Event Collector*
  - *Generate certificate*
  - *Event Collector Configuration*
  - *Install dependencies*
  - *Running Event Collector service*
  - *Windows host configuration*
  - *Network Probe pipeline configuration*
  - *Enabling Network Probe pipeline*
  - *Data Node template*
  - *Building the subscription filter*

Configuration steps

Installation of Event Collector

```
tar zxf wec_7x-master.tar.gz -C /opt/
mkdir /opt/wec
mv /opt/wec_7x-master/ /opt/wec/
mkdir /etc/wec
cp /opt/wec/sub_manager/config.yaml /etc/wec/config.yaml
```

Generate certificate

```
mkdir /opt/wec/certgen
cd /opt/wec/certgen
vim server-certopts.cnf
```

- Set `DNS.1` and `IP.1` for the WEC server:

```
[req]
default_bits = 4096
default_md = sha256
req_extensions = req_ext
keyUsage = keyEncipherment,dataEncipherment
basicConstraints = CA:FALSE
distinguished_name = dn

[ req_ext ]
subjectAltName = @alt_names
extendedKeyUsage = serverAuth,clientAuth

[ alt_names ]
DNS.1 = wec.local.domain
IP.1 = 192.168.13.163

[dn]
```

- Set `DNS.1` and `IP.1` for client certificate:

```
vim client-certopts.cnf
```

```
[req]
default_bits = 4096
default_md = sha256
req_extensions = req_ext
keyUsage = keyEncipherment,dataEncipherment
basicConstraints = CA:FALSE
distinguished_name = dn

[ req_ext ]
subjectAltName = @alt_names
extendedKeyUsage = serverAuth,clientAuth

[ alt_names ]
```

(continues on next page)

(continued from previous page)

```
DNS.1 = *local.domain

[dn]
```

- Generate the CA certificate and private key, next check fingerprint:

```
openssl genrsa -out ca.key 4096
openssl req -x509 -new -nodes -key ca.key -days 3650 -out ca.crt -subj '/
↪CN=wec.local.domain/O=example.com/C=CA/ST=QC/L=Montreal'
openssl x509 -in ca.crt -fingerprint -sha1 -noout | sed -e 's/\://g' > ca.
↪fingerprint
```

- Generate the client certificate and export it together with the CA in PFX format to be imported into the Windows certificate store:

```
openssl req -new -newkey rsa:4096 -nodes -out server.csr -keyout server.
↪key -subj '/CN=wec.local.domain/O=example.com/C=CA/ST=QC/L=Montreal'
openssl x509 -req -in server.csr -out server.crt -CA ca.crt -CAkey ca.key␣
↪-CAcreateserial -extfile server-certopts.cnf -extensions req_ext -days␣
↪365
```

- Generate the server certificate to be used by the WEC:

```
openssl req -new -newkey rsa:4096 -nodes -out client.csr -keyout client.
↪key -subj '/CN=wec.local.domain/O=example.com/C=CA/ST=QC/L=Montreal'
openssl x509 -req -in client.csr -out client.crt -CA ca.crt -CAkey ca.key␣
↪-CAcreateserial -extfile client-certopts.cnf -extensions req_ext -days␣
↪365
openssl pkcs12 -export  -inkey client.key -in client.crt -certfile ca.crt␣
↪-out client.p12
```

Event Collector Configuration

- Copy the server certificate and server key to the Event Collector installation directory:

```
cp server.crt server.key /opt/wec/sub_manager/certificates/
```

- Edit configuration file `config.yaml`

```
vim /etc/wec/config.yaml
```

  - set the following options:

```
external_host: wec.local.domain
#check ca.fingerprint file
ca_fingerprint: 97DDCD6F3AFA511EED5D3312BC50D194A9C9FA9A
certificate: /opt/wec/sub_manager/certificates/server.crt
key: /opt/wec/sub_manager/certificates/server.key
```

  - set the output for Event Collector to Network Probe forwarding:

```
remote_syslog:
    # forward events to remote syslog server
```

(continues on next page)

(continued from previous page)

```
    address: 192.168.13.170
    port: 6514
    type: TLS
```

Important

The port set here must match the port the Network Probe pipeline listens on. The pipeline shipped with the Event Collector (`001-input-wec.conf`, described below) listens on TCP 6514 with TLS enabled. If you change one side, change the other.

- set the output to saving events to a local file:

```
outputfile: /var/log/wec/events-{:%Y-%d-%m}.log
```

- disable local syslog output:

```
local_syslog: false
```

- set the filter section. `filters` belongs to the subscription, each entry pairs a Windows event source with an XPath filter:

```
subscriptions:
  - defaultLogSource:
      label: "Default Log Source"
      url: '/owinec/subscriptions/s2'
      filters:
        - source: 'Security'
          filter: '*[System[(Level=1 or Level=2 or Level=3 or Level=4 or␣
↪Level=0 or Level=5) and (EventID=4672 or EventID=4624 or EventID=4634)]]'
        - source: 'Application'
          filter: '*[System[(Level=1 or Level=2 or Level=3 or Level=4 or␣
↪Level=0 or Level=5)]]'
        - source: 'System'
          filter: '*[System[(Level=1 or Level=2 or Level=3 or Level=4 or␣
↪Level=0 or Level=5)]]'
```

Install dependencies

1. Python 3 installation:

```
sudo dnf -y install python3 python3-pip
python3 --version
```

2. Python requirements installation:

```
pip3 install PyYAML
pip3 install sslkeylog
```

Running Event Collector service

```
vim /etc/systemd/system/wec.service
```

```
[Unit]
Description=WEC Service
After=network.target

[Service]
Type=simple
ExecStart=/usr/bin/python3 /opt/wec/sub_manager/run.py -c /etc/wec/config.yaml
Restart=on-failure
RestartSec=42s
StandardOutput=syslog
StandardError=syslog
SyslogIdentifier=wecservice

[Install]
WantedBy=multi-user.target
```

```
systemctl daemon-reload
systemctl start wec
```

Windows host configuration

1. Open the `Microsoft Management Console (mmc.exe)`, select `File -> Add/Remove Snap-ins`, and add the `Certificates` snap-in.
2. Select `Computer Account`.
3. Right-click the `Personal` node, and select `All Tasks > Import`.
4. Find and select the client certificate (client.p12) and import this file.
5. The `PKCS #12` archive contains the CA certificate as well.
6. Move the CA certificate to the `Trusted Root Certification Authorities` node after the import.
7. Give `NetworkService` access to the private key file of the client authentication certificate:
8. To forward security logs:
   - In `CompMgmt.msc`, under `Local Users and Groups`, click `Groups > Event Log Readers` to open `Event Log Readers Properties`.
   - Add the "NETWORK SERVICE" account to the `Event Log Readers group`.

   8.1. For domain controller use "Group Policy Manager Editor" and edit: "Default Domain Controller Policy":
   - From `Computer Configuration > Policy`, expand `Windows Settings > Security Settings > Restricted Groups`;
   - From the context menu add: `Add Group`
   - Add the following configuration:
     - Group = `BUILTIN\Event Log Readers`
       - Members = `NT Authority\NETWORK SERVICE`
9. Make sure the collector server is reachable from the Windows machine

10. Run `winrm qc` and accept changes on the Windows machine
11. Run `winrm set winrm/config/client/auth @{Certificate="true"}` on windows machine to enable certificate authentication
12. Open `gpedit.msc`
13. Under the `Computer Configuration node`, expand the `Administrative Templates` node, then expand the `Windows Components` node, and then select the `Event Forwarding` node.
14. Select the **SubscriptionManagers** setting and enable it. Click the **Show** button to add a subscription (use the CA thumbprint you saved earlier):

    ```
    Server=https://<FQDN of the collector>:5986/wsman/SubscriptionManager/WEC,
    ↪Refresh=<Refresh interval in seconds>,IssuerCA=<Thumbprint of the root␣
    ↪CA>
    ```

    For example:

    ```
    Server=HTTPS://logserver.diplux.com:5986/wsman/SubscriptionManager/WEC,
    ↪Refresh=60,IssuerCA=549A72B56560A5CAA392078D9C38B52458616D25
    ```

    NOTE: If you wish to set up multiple subscriptions because you want to forward Windows events to multiple event collectors (such as WEC), then you can do that here.

15. Run the `cmd` console with administrative privileges and make the following command

    ```
    gpupdate /force
    ```

Network Probe pipeline configuration

Create a directory for Event Collector pipeline configuration files:

```
mkdir /etc/logserver-probe/conf.d/syslog_wec
```

Copy the following Network Probe configuration files to the pipeline directory:

```
cp 001-input-wec.conf /etc/logserver-probe/conf.d/syslog_wec/
cp 050-filter-wec.conf /etc/logserver-probe/conf.d/syslog_wec/
cp 060-filter-wec-siem.conf /etc/logserver-probe/conf.d/syslog_wec/
cp 100-output-wec.conf /etc/logserver-probe/conf.d/syslog_wec/
```

Warning

Two values in these files are placeholders and must be replaced before the pipeline is enabled:

- `001-input-wec.conf` points `ssl_cert`, `ssl_key` and `ssl_certificate_authorities` at test certificates. Replace them with the certificates generated above.
- `100-output-wec.conf` contains a hardcoded Data Node address and credentials. Replace them with the address and account used in your deployment.

Enabling Network Probe pipeline

To enable the `syslog_wec` Network Probe pipeline edit the `pipelines.yml` file:

```
vim /etc/logserver-probe/pipelines.yml
```

Add the following section:

```
- pipeline.id: syslog_wec
  path.config: "/etc/logserver-probe/conf.d/syslog_wec/*.conf"
```

And restart Network Probe with `systemctl restart logserver-probe` command.

Data Node template

Install the Data Node template for the Event Collector data index:

```
curl -ulogserver:logserver -X PUT "http://localhost:9200/_template/syslog_wec?
↪pretty" -H 'Content-Type: application/json' -d@template_wec.json
```

Building the subscription filter

1. Browse to Event Viewer
2. Right-click **Subscriptions** and **create subscription**
3. Click on Select Events and choose the type of logs that you want, for example, Event Level, Event Logs, Include Exclude Event ID, Keyword, etc.
4. Switch to XML view tab;
5. Copy the value of the `Select Path` key, for example:

```
<QueryList>
  <Query Id="0" Path="Security">
    <Select Path="Security">*[System[(Level=1  or Level=2 or Level=3) and␣
↪(EventID=4672 or EventID=4624 or EventID=4634)]]</Select>
  </Query>
</QueryList>
```

   string to copy:

```
*[System[(Level=1  or Level=2 or Level=3) and (EventID=4672 or␣
↪EventID=4624 or EventID=4634)]]
```

6. Paste the above definition into the Event Collector configuration file in the `filters` section:

```
vim /etc/wec/config.yaml
```

```
      filters:
        - source: 'Security'
          filter: '*[System[(Level=1  or Level=2 or Level=3) and␣
↪(EventID=4672 or EventID=4624 or EventID=4634)]]'
```

Restart the Event Collector service

```
systemctl restart wec
```

GUI Port and HTTPS Redirect

Overview

The Energy Logserver web GUI (`logserver-gui` service) listens on TCP port **5601** by default. To allow users to reach the GUI on the standard HTTPS port, the installer adds a **firewalld port-forwarding rule** that redirects incoming traffic from port **443** to port **5601** on the same host.

This redirect is created only when `firewall-cmd` is available on the system. On hosts without firewalld, no automatic redirect is configured, operators must set up their own mechanism (iptables, nginx, HAProxy, or a load balancer in front of the Console).

Changing the GUI port therefore requires updating **both** the GUI configuration file and the firewalld redirect; updating only the configuration file leaves the redirect pointing at the old port and the GUI appears unreachable on 443.

Default configuration

Item	Value
GUI configuration file	`/etc/logserver-gui/logserver-gui.yml`
Default listen port (`server.port`)	`5601`
Default bind address (`server.host`)	`"0.0.0.0"`
Installer-added redirect	firewalld forward `443/tcp` → `5601/tcp`
Service unit	`logserver-gui.service`

To inspect the active firewalld rules:

```
firewall-cmd --list-ports
firewall-cmd --list-forward-ports
```

A standard install reports `5601/tcp` (along with `5602/tcp` and `5603/tcp`) in the open ports list and `port=443:proto=tcp:toport=5601:toaddr=` in the forward-ports list.

Changing the GUI port

The example below changes the GUI port from `5601` to `5701`. Replace `5701` with your target port.

1. Update the GUI configuration file

Edit `/etc/logserver-gui/logserver-gui.yml` and set `server.port` to the new value:

```
server.port: 5701
server.host: "0.0.0.0"
```

`server.host` only needs to change if you also want to restrict the bind address (for example, to `127.0.0.1` when the GUI is fronted by a reverse proxy on the same host).

2. Update the firewalld redirect

Remove the old port-forward and the old port, then add the new ones:

```
firewall-cmd --permanent --zone=public --remove-forward-
↪port=port=443:proto=tcp:toport=5601
firewall-cmd --permanent --zone=public --remove-port=5601/tcp
firewall-cmd --permanent --zone=public --add-port=5701/tcp
firewall-cmd --permanent --zone=public --add-forward-
↪port=port=443:proto=tcp:toport=5701
firewall-cmd --reload
```

The `masquerade` setting on the `public` zone must remain enabled, the firewalld forward-port rule depends on it. The installer enables it by default; do not remove it.

3. Restart the GUI service

```
systemctl restart logserver-gui
```

4. Verify

```
ss -ltnp | grep 5701
curl -kI https://127.0.0.1:5701/
curl -kI https://<server-fqdn>/
```

The first command confirms the GUI is listening on the new port. The second confirms the GUI responds directly. The third confirms the 443 redirect now lands on the new port.

Hosts without firewalld

If `firewall-cmd` was not present at install time, the installer skipped both the open-port rules and the 443→5601 forward. In that case:

- Reaching the GUI on 443 requires an external mechanism that you control (iptables, nginx, HAProxy, an upstream load balancer, or a cloud-provider rule). The installer does not configure any of these.
- Changing the GUI port only requires steps 1, 3, and 4 above. The redirect update belongs to whichever tool you used to publish the GUI on 443.

For a documented NGINX reverse-proxy setup (used with OAuth2 / 2FA), see *Authentication and Directory Integration*. When you put your own proxy on port 443, remove the installer's `443 → 5601` forward first, otherwise the PREROUTING redirect intercepts external traffic before your proxy sees it, and the proxy never runs even though loopback tests succeed.

Caveats

- **Integrations with hard-coded GUI URLs.** Some bundled components (Alert, Reports, the Intelligence module, and parts of the installer's own post-install steps) call the GUI on `https://127.0.0.1:5601`. After changing `server.port`, search component configuration files for `5601` and update any references you find. Restart each affected service.

- **Re-running the installer.** If you re-run `install.sh` after changing the port, the installer may re-add the original `443 → 5601` forward-port rule. Re-apply your firewalld changes after any reinstall or upgrade that re-executes the installer's firewalld block.
- **Cluster deployments.** Apply the change on every Console node. The port must match across nodes that share a load balancer or DNS name.
- **TLS termination.** When TLS is terminated upstream (by NGINX or a load balancer), the GUI can be configured to listen over HTTP, disable `server.ssl.enabled` in `logserver-gui.yml` and bind `server.host` to `127.0.0.1` so the GUI is not reachable directly from the network.

Managed Security Service Provider (MSSP)

The **MSSP** mode allows the user to effectively manage sources collected throughout the system. The source can be defined as a host providing data to the system. When the **MSSP** mode is enabled, sources are continuously gathered and monitored so they can be displayed in an understandable way.

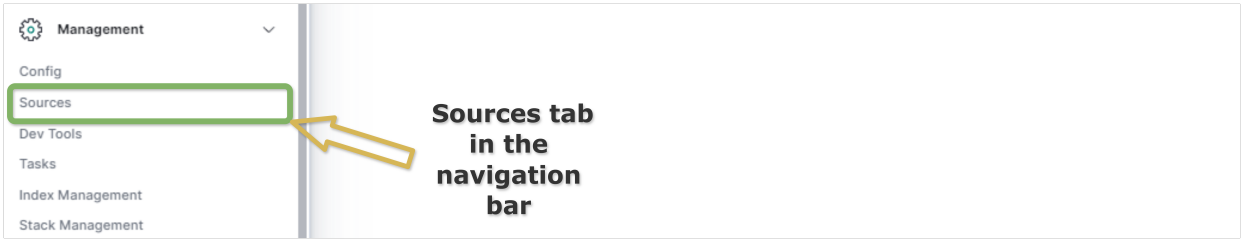
- *Collecting Sources*
- *Managing Sources*
- *Source Types*
- *Source Removal*
- *Manual Change of State from `Waiting` to `Allowed`*
- *Manual Change of State from `Allowed` to `Waiting`*
- *Fingerprint*
- *The Permissive Mode*
- *Applying a New License*
- *Check MSSP Status*
- *MSSP Disabled*

Collecting Sources

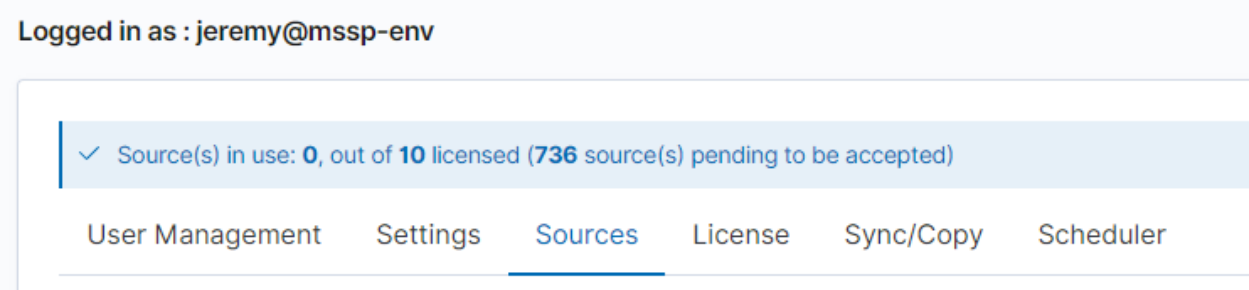
The sources are collected in two ways - by the `Network Probe` or the `Logserver` itself. The `Network Probe` utilizes a unique set of fields against documents of different technologies to generate an inimitable identifier. This makes it possible to determine from which source the data came and therefore also to manage the system's traffic.

Managing Sources

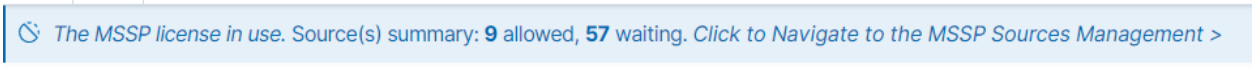
Only admin-level users can manage sources in the **Config** plugin. For such users, it displays the **Sources** tab in the side navigation bar, the **Sources** tab in the **Config** plugin, and a blue banner at the top of the screen.



Sources tab in the navigation bar



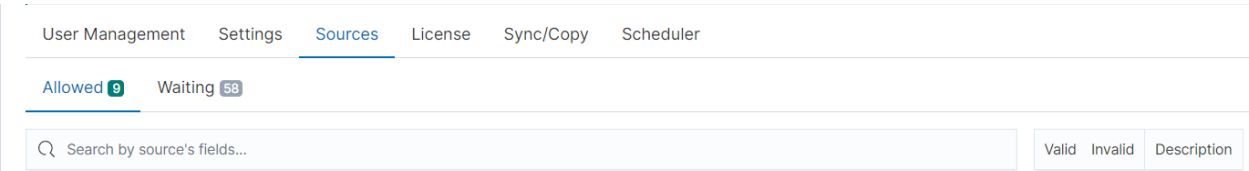
Sources tab in the Config plugin



Blue banner

Source Types

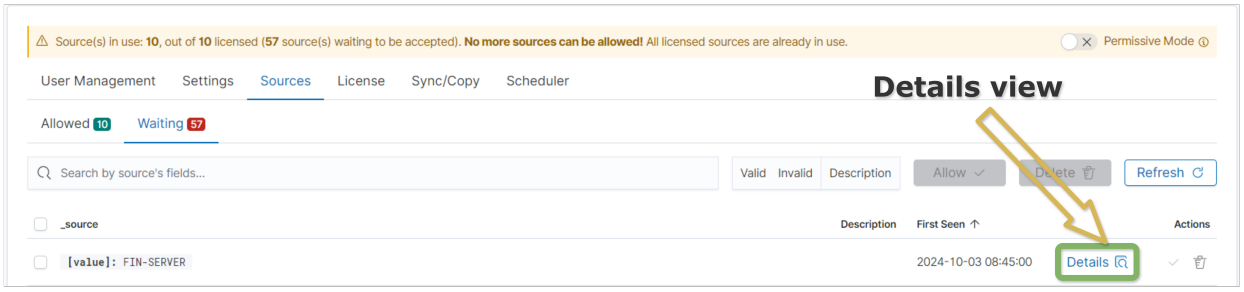
Two states of sources can be distinguished - `Waiting` and `Allowed`.



Sources types

1. Waiting

The `Waiting` source is a source pending approval. Data related to this source will not be ingested until it is accepted. Only the first document from a given **source** is collected and stored as an example. Its details include information such as when it was indexed, the source from where it was sent, and its contents. In addition, a note can be added to distinguish it from other **sources** and describe it in a memorable way.



2. Allowed

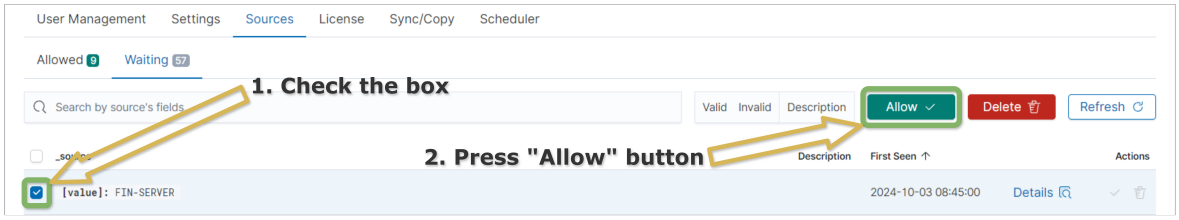
When a `Waiting` document is `Allowed`, the flow of its data is made possible, and the source consumes license allocation, enabling full processing and monitoring.

Source Removal

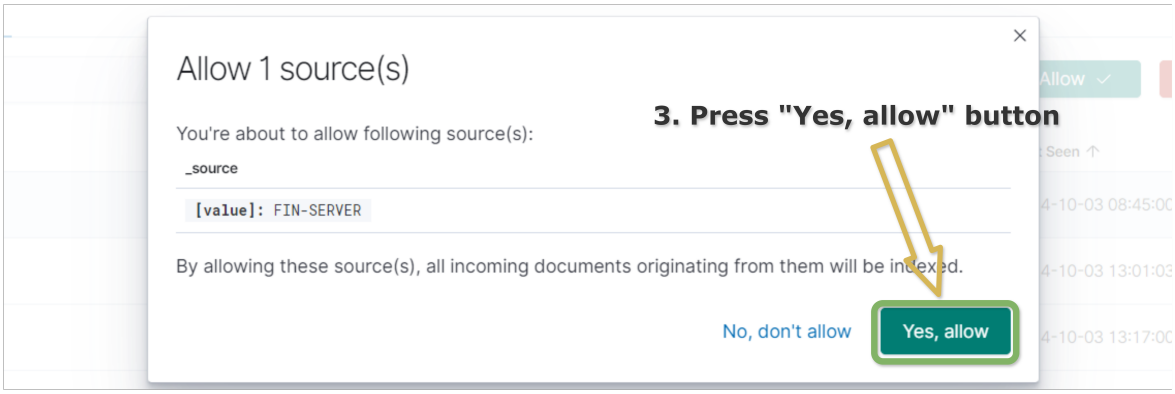
Both types of sources can be removed. After deleting the `Waiting` source, if the data is still coming in, after a brief moment the source will reappear in the table with the possibility of possible acceptance. On the other hand, if an `Allowed` source is deleted, the data will stop coming in, although the data collected up to the time of deletion will not be cleared.

Manual Change of State from `Waiting` to `Allowed`

1. Check the box of the sources which you want to move from `Waiting` to `Allowed` state.
2. Press on the button `Allow`.

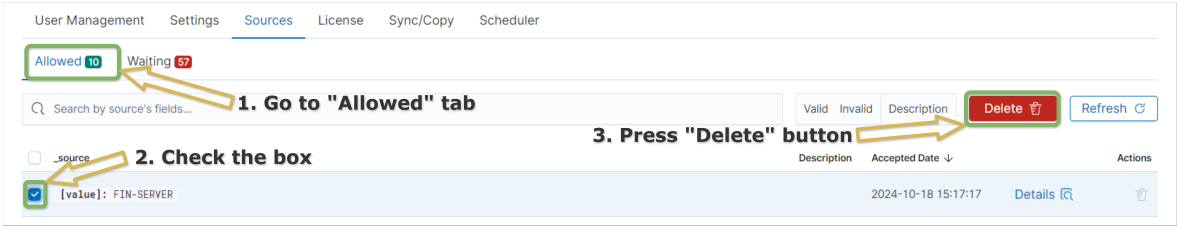


3. A details of the operation will be displayed. If you still want to perform the operation, press `Yes, allow` button.

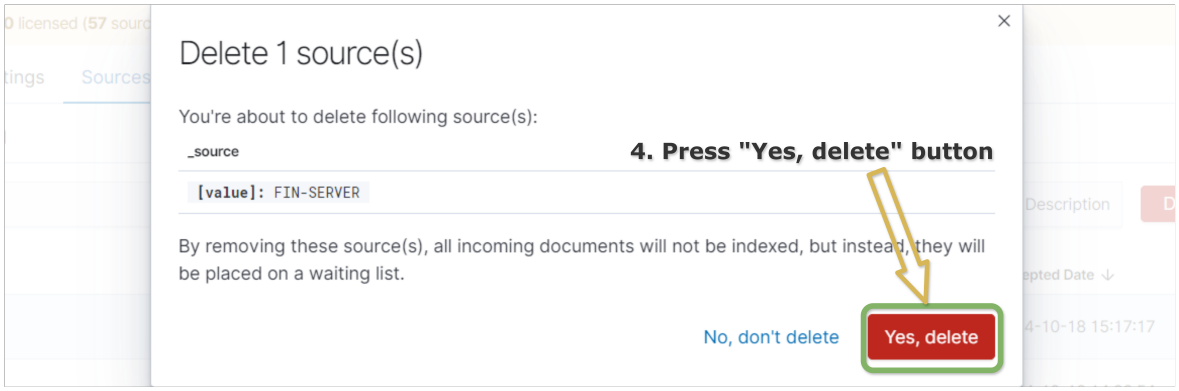


Manual Change of State from `Allowed` to `Waiting`

1. Go to the `Allowed` tab.
2. Check the box of the sources which you want to move from `Allowed` to `Waiting` list.
3. Press on the button `Delete`.



4. Details of the operation will appear. If you still want to perform the operation, press `Yes, delete` button.



Fingerprint

The `Network Probe` generates a unique identifier for each document called a **fingerprint** in the `_mssp_id` field. **Fingerprint** is encrypted data from the document. Sources without **fingerprints** are automatically dropped. Sources whose **fingerprints** cannot be decrypted are placed on a waiting list but without the possibility of being allowed. Documents with valid **fingerprints** are placed on a waiting list with the possibility of approval until the license limit is reached.

The Permissive Mode

The **Permissive Mode** is enabled out of the box. It automatically allows waiting **sources** until the license limit is completely used. If all available **sources** have been used, no others will be accepted, which is also indicated by the blue banner changing to orange with a warning. The purpose of this transition is to draw the user's attention to checking whether all allowed **sources** are still being used. If so, it means that a license covering more **sources** should be purchased.

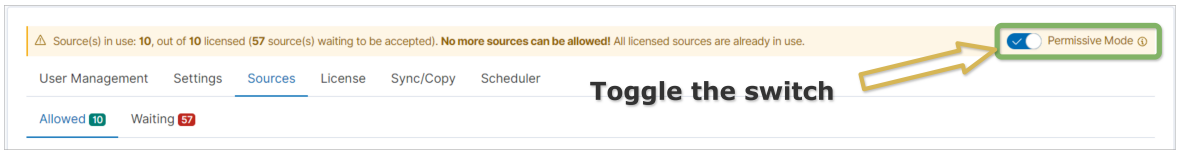


Orange banner

It was introduced to make it easier to use the system, without having to worry and remember about source management. On the other hand, if there is a need to allow or remove troublesome or erroneous **sources** manually, it is always possible to disable this mode and perform manual operations.

Turn On/Off the `Permissive Mode`

1. Go to the **Sources** tab.
2. Toggle the switch located on the blue bar captioned `Permissive Mode` to the state in which you want it to work.



Applying a New License

The `Network Probe` reads the MSSP state from the license once, when its pipelines start, and keeps that value until they are restarted. After you deploy a license that turns MSSP mode on or off, restart every `Network Probe` in the deployment.

> **Warning**
>
> Until you restart the probes, they keep processing traffic according to the previous license. A probe that started before MSSP mode was licensed does not route documents through the source validation pipeline, so no source is blocked or approved, even though the **Licence** tab already reports the new state.

Restart the probes in one of two ways:

- In the GUI, open the `Network Probe` plugin, go to the probe details, and restart the service in the **Services** section, as described in *Management Interface, Services Section*.
- On each probe host, from the console:

```
systemctl restart logserver-probe
```

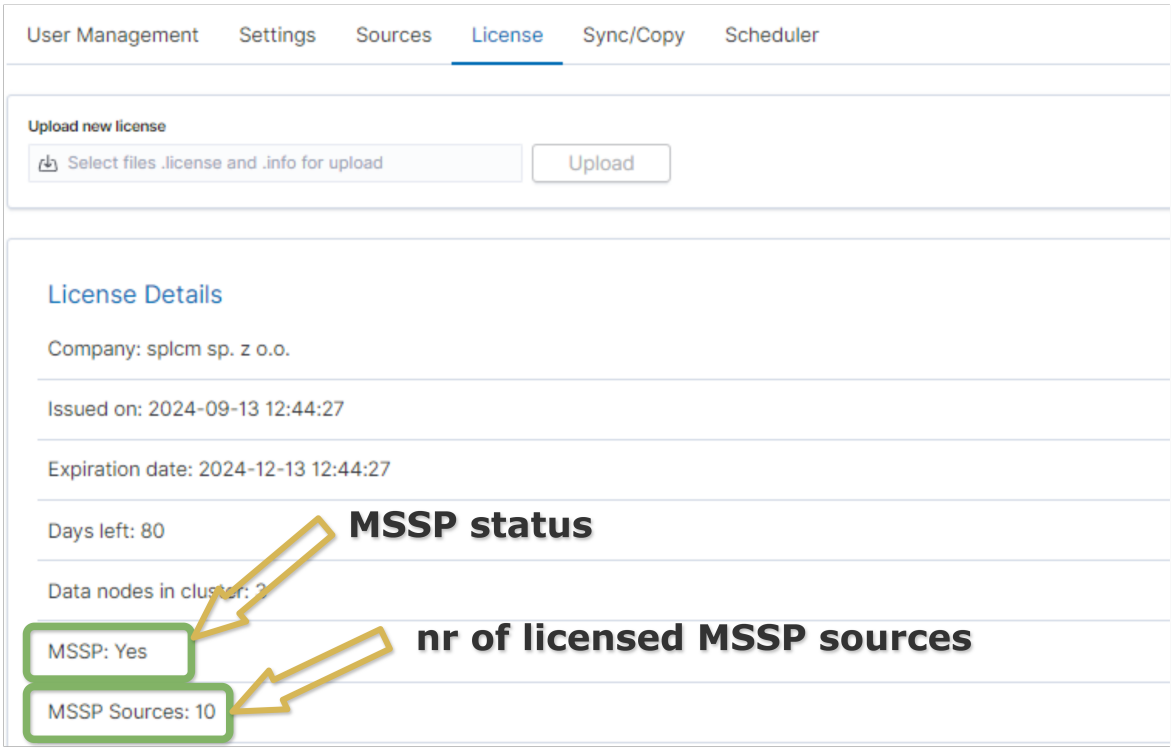
To confirm that a probe applied the new license, check the license entry it writes at startup:

```
grep "License check" /var/log/logserver-probe/logserver-probe-plain.log |␣
↪tail -1
```

The `is_mssp_enabled` value in that entry must match the `MSSP` field in **Config** → **License**. A mismatch means the probe still runs on the previous license and needs a restart.

Check MSSP Status

1. Go to the **Config** tab in the navigation bar.
2. Select **Licence** section in the top bar.
3. The value of the `MSSP` field in **Licence Details** indicates the status of the module.



MSSP Disabled

When `MSSP` mode is disabled, the system still gathers information about sources based on documents, but it does not block or reject any incoming documents. A list of sources can still be viewed in the `Config->Sources` tab.

Troubleshooting

Network Probe Troubleshooting

Network Probe registration and management is handled by the `license-service`. The commands below operate on this service.

Debugging

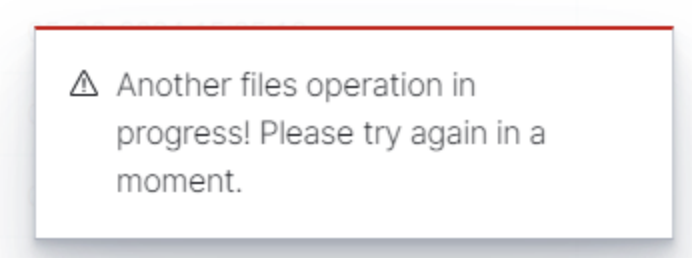
To check probe's logs:

```
journalctl -fu license-service
```

By default, only errors and warnings are logged. To change that, and enable for example `debug` level, set `log_level: DEBUG` in the `/opt/license-service/license-service.conf` file.

Another operation in progress

The message shown below may be encountered during actions performed on both files and pipelines.



It means that the system is already performing some actions, and in order to ensure the consistency of files, it is necessary to wait until previous actions are completed. After waiting a while, try again to perform the desired action and it should succeed without any problem.

"certificate signature failure" on One Click

Symptoms: During an integration installation, the pipeline step ("Loading pipelines") fails with a message such as:

```
Probe <name>: unable to execute operation! certificate signature failure
```

This is not specific to a single integration. Any component installed through **One Click** fails at the same step when the underlying condition is present, because they all go through the same channel.

Cause: The GUI talks to the probe agent over `https://<host>:9000` (the `license-service`, using mutual TLS). By default the GUI validates the agent's certificate signature against the certificate authority `/opt/license-service/ssl/license_service_CA.crt`. This is controlled by `network-probe.verificationMode` in `/etc/logserver-gui/logserver-gui.yml`, whose default value is `certificate`. The message means the CA the GUI trusts has drifted from the certificate the agent serves on `:9000`. This happens after the `license-service` certificates are regenerated without reissuing the server certificate, or in a distributed deployment where the probe has its own CA that was not added to the GUI trust store.

Verification (on the host where `license-service` runs):

```
echo | openssl s_client -connect <PROBE_IP>:9000 -CAfile /opt/license-service/
↪ssl/license_service_CA.crt 2>&1 | grep -i verify
openssl verify -CAfile /opt/license-service/ssl/license_service_CA.crt /opt/
↪license-service/ssl/license_service.crt
```

A result of `error 7` / `certificate signature failure` confirms the certificate mismatch.

Resolution: Bring the `license-service` certificates back into agreement, so that the CA the GUI trusts is the same CA that signed the agent certificate served on `:9000`. In a distributed deployment, add the probe's CA to the GUI trust store. Then restart the services:

```
systemctl restart license-service logserver-gui
```

Temporary workaround (not recommended in production): set `network-probe.verificationMode: none` in `/etc/logserver-gui/logserver-gui.yml` and restart `logserver-gui`. This disables validation of the probe agent certificate and should be reverted once the certificates are synchronized.

Restarting the probe

To restart the service enter:

```
systemctl restart license-service
```

Removing already registered probe

1. If the probe is running stop the service:

```
systemctl stop license-service
```

2. Open the "Network Probes" tab in the GUI and go into the details of the probe you are interested in.
3. After opening the details, copy the opened url and cut out the id located between the `/app/network_probe/probes/` and `/services` sections. In the url below the id is the long hexadecimal string between them.

```
https://127.0.0.1:5601/app/network_probe/probes/f7fdb48bf5252cb41ab4162d96
↪144a0d463b7ae5330bae6f37f501cb97fb272d199a59cc97bfdc1d2fd46e981ae42a8a59
↪c66a40932c4bce27d786efe1f2dcc3/services
```

4. Execute the following query, replacing `$ID` with the value obtained in the previous step, as well as `$USER` and `$PASSWORD` to Data Node:

```
curl -u$USER:$PASSWORD -XDELETE '127.0.0.1:9200/.networkprobes/_doc/$ID'
```

Re-registering probe

If you have a problem with the probe and would like to re-register it, you have to:

1. Follow the steps in the previous section - Removing already registered probe.
2. Remove the following file: `/opt/license-service/hashKeyKS.p12.network_probe`.
3. Start the probe:

```
systemctl start license-service
```

The probe will be registered as a brand new one, if the license allows it and free slots are available.

Authentication Troubleshooting

System accounts returning 401 / GUI authentication errors

Symptoms: The GUI shows "Error while fetching user details", API calls with valid credentials return HTTP 401, or `license-service` enters a crash loop.

The most common cause is a desynchronization between the `.auth` index and the keystore or service configuration files. This can happen after an upgrade, a partial restore, or manual intervention.

Step 1, Re-sync credentials

Run the password utility on the client node:

```
/usr/share/logserver/utils/logserver-password-util.sh update_services --yes
```

This propagates the keystore credentials to all service configuration files and restarts the affected services. If credentials in the keystore are correct but service configs drifted, this resolves the issue without any index changes.

Step 2, Verify cluster health

```
curl -u $USER:$PASSWORD http://127.0.0.1:9200/_cluster/health?pretty
```

Check that the cluster is green and the `.auth` index is present:

```
curl -u $USER:$PASSWORD http://127.0.0.1:9200/_cat/indices/.auth*?v
```

Step 3, If the above does not resolve the issue

Temporarily disable the `logserver_auth` plugin to regain API access, following the procedure in *Plugins Management*. After disabling, run the password utility again to ensure keystore and configs are consistent, then re-enable the plugin.

Warning

Disabling `logserver_auth` removes authentication enforcement on the Data Node API for the duration. Isolate the node from external network access before proceeding.

4.16 Integrations

The **Integrations** plugin provides a unified catalog of vendor and protocol integrations that can be deployed into Energy Logserver from the GUI. Each integration bundles the components required to ingest, visualize, and alert on data from a specific source: pipelines for the Network Probe, Discover dashboards, alert rules, and (for some integrations) downloadable collector binaries.

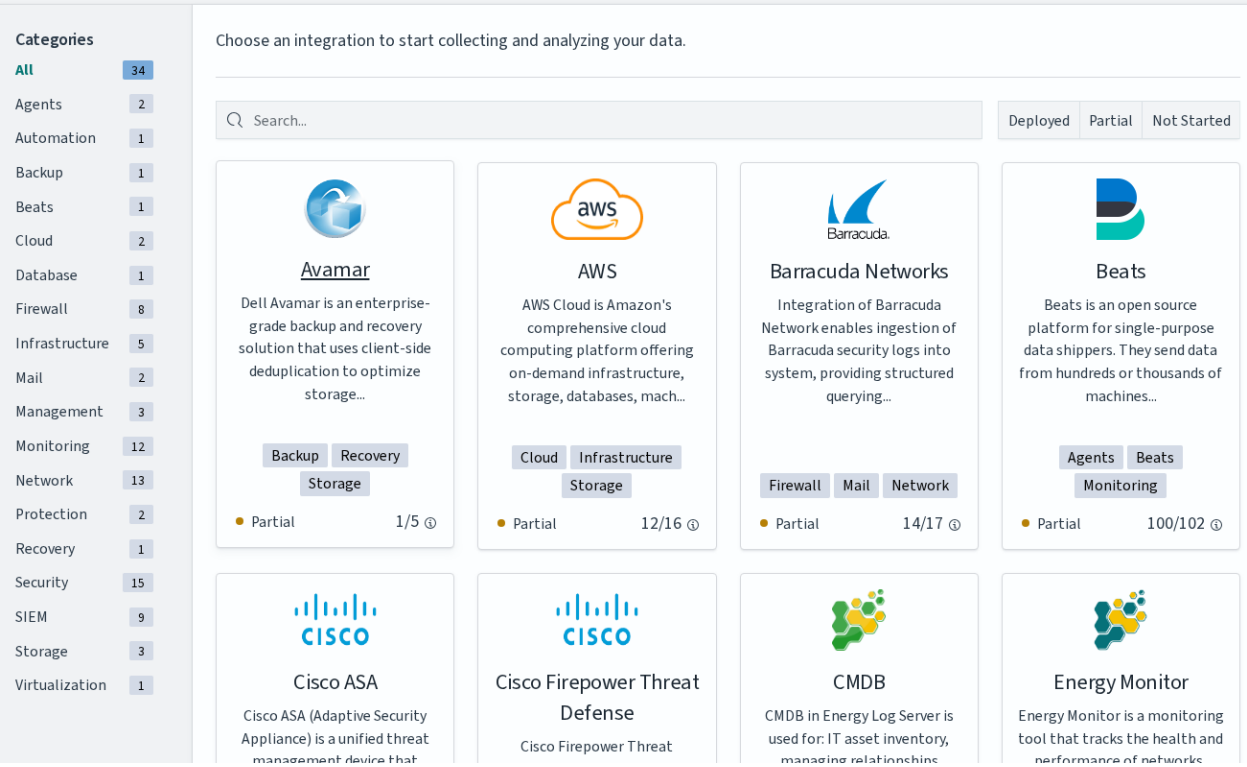
The plugin requires a license with the **siem-plan** add-on enabled.

Note

Some catalog entries are detection engines or flow collectors that ship bundled with the Network Probe rather than external data sources. Their One Click deployment installs the Energy Logserver side of the pipeline (and, where the integration ships them, dashboards and alerts), but the engine itself still needs host-side setup on the probe. These have dedicated guides: *Suricata IDS* and *Zeek*. To account for a host's own traffic and export it as NetFlow, see *pmacct Local Traffic Monitoring*.

4.16.1 Catalog Layout

Open the plugin from the sidebar under **Tools** → **Integrations**. The catalog page shows:



Area	Purpose
Categories (left panel)	Filter the catalog by integration type: `Agents`, `AI Assistant`, `Automation`, `Backup`, `Beats`, `Cloud`, `Database`, `Firewall`, `Infrastructure`, `Mail`, `Management`, `Monitoring`, `Network`, `Other`, `Protection`, `Recovery`, `SIEM`, `Security`, `Storage`, `Virtualization`. `All` above them clears the filter. The number next to each category reflects the count of integrations available for it.
Search	Filter the catalog by integration name.
Status filters	Restrict the visible cards to those that are **Deployed** (all components installed), **Partial** (some components installed or active), or **Not Started** (nothing installed).
Integration cards	One card per integration. Each card shows the icon, name, short description, the categories it belongs to, and the current deployment status. The status counter (for example, *Partial 14/17*) indicates how many of the integration's components are currently installed.

The available integrations are read directly from the product and refresh with each release. Treat the catalog itself as the authoritative list. This chapter does not duplicate the integration names.

For a log source the catalog does not cover, the **AI Assistant** tile builds a new integration from sample logs. See *Creating Integrations with the AI Assistant*.

4.16.2 Installation

Each integration can be installed in one of two modes, exposed as buttons on the integration's wizard page:

- **One Click** installs every component of the integration in a single action. Use this mode for the default deployment.

- **Advanced** opens a step-by-step wizard that lets you select pipelines, dashboards, alert rules, and (when applicable) agents to install or skip individually.

> **Note**
>
> The `Agents` step in the Integrations wizard refers to **collector binaries shipped with the selected integration**, such as Beats agents delivered with a pre-generated configuration. It is **not** related to the **Agents** module (Master-Agent) described in *Agents*.

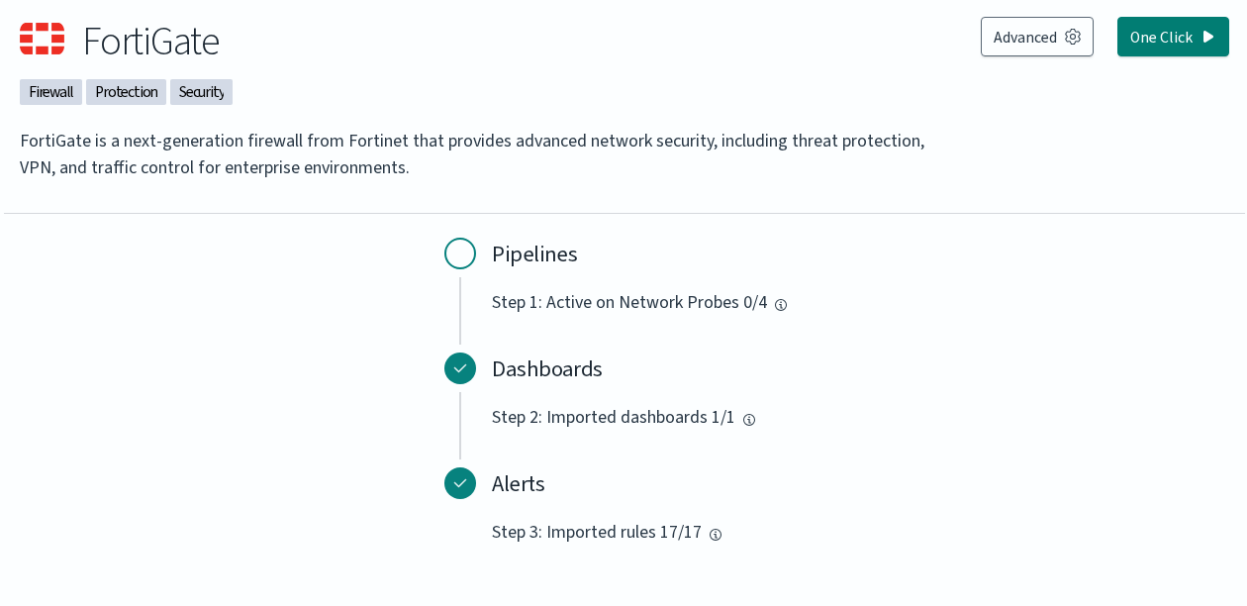
Choosing an Integration

1. Open **Tools** → **Integrations** in the sidebar.
2. Use the search box, the category filters, or the status filters (`Deployed`, `Partial`, `Not Started`) to locate the integration.
3. Click the integration card. The wizard page opens with two buttons in the top-right corner: **Advanced** and **One Click**.

One Click Installation

The One Click button installs every component of the integration in a single action: Network Probe pipelines, Discover dashboards, alert rules, and (for integrations that ship them) collector binaries.

The wizard page shows the current state of each component group as a checklist. After clicking **One Click**, every item that is not already installed is deployed, and the indicators update to show the new component counts. Each step title is a link that opens the same step in the Advanced wizard.

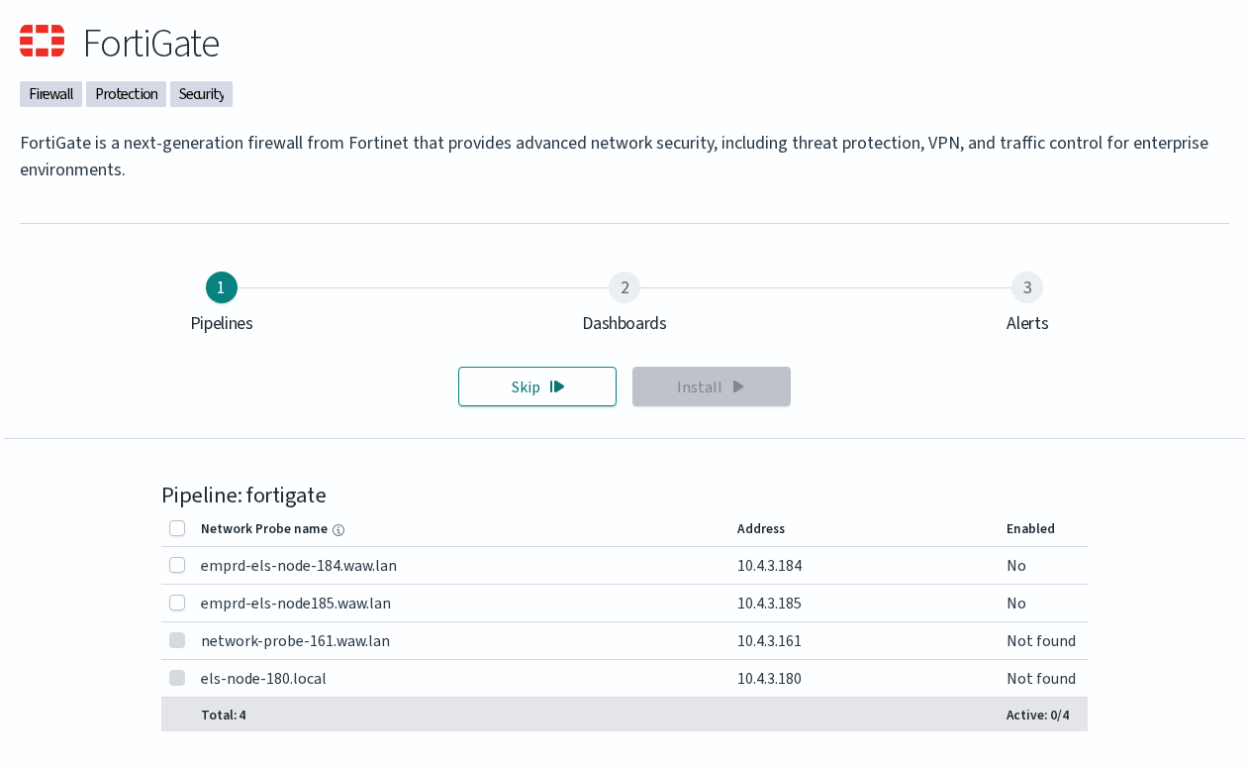


In the example above, the Dashboards (`1/1`) and Alerts (`17/17`) are already deployed, and only the Pipelines step (`0/4`) remains. Clicking **One Click** would activate the integration's pipeline on all four Network Probes registered in the cluster.

If One Click stops at the pipeline step with `certificate signature failure`, the cause is inconsistent `license-service` certificates rather than the integration itself, and it affects every integration installed this way. See *Network Probe Troubleshooting*.

Advanced Installation

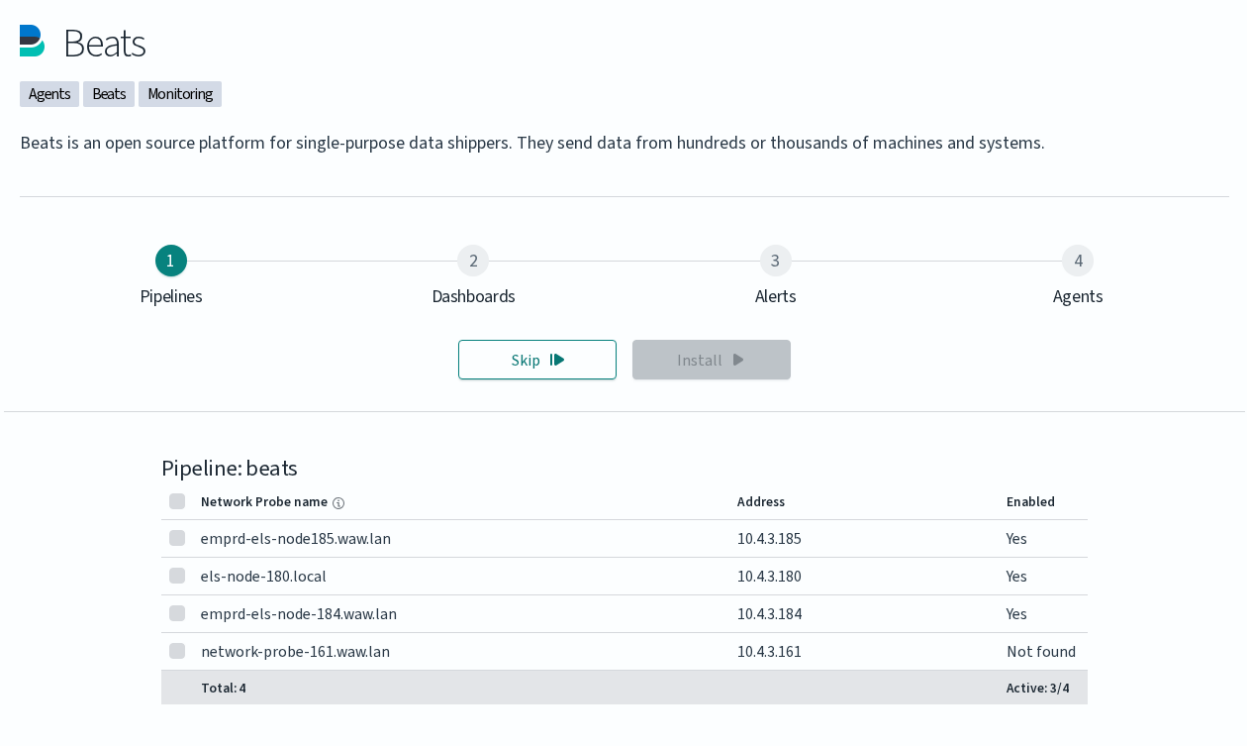
The Advanced button opens a numbered wizard. Each step lets you select the components to install for that group, then choose **Install** to apply or **Skip** to move on without installing. Steps that have nothing to install are marked complete automatically.



The wizard exposes three steps for most integrations:

Step	What it installs
1. Pipelin	Activates the integration's Network Probe pipeline on the probes you select. The table lists every Network Probe in the cluster with its address and current activation status (`Yes`, `No`, or `Not found` if the integration's pipeline files are not present on that probe).
2. Dashboards	Imports the integration's Discover dashboards.
3. Alerts	Imports the integration's alert rules.

Some integrations expose additional steps:



Step	What it installs
Agents	Lets you download the collector binaries shipped with the integration (for example, Beats agents) together with a configuration file generated for your cluster. The download is delivered to your local machine; the actual installation on the source host is performed manually by the administrator.
Index Management	Imports the integration's index management policies, such as a data retention action, which then run from the *Scheduler*.

The Agents step is shown only for integrations that bundle a collector, and the Index Management step only for integrations that ship policies. FortiGate and Tenable Security Center, for example, have only the three standard steps.

4.16.3 Deleting Integrations

Click **Delete** above the catalog, click **Select** on each integration to remove, click **Confirm**, and confirm in the dialog. Deleting an integration removes its Network Probe pipeline, dashboards and alert rules. A system integration can be re-enabled later from the Integrations page. An integration created with the AI Assistant is deleted permanently, with its configuration files.

4.16.4 Manual Configuration

A small set of integrations can be deployed without the GUI wizard by editing the Network Probe pipeline files directly. This route fits environments without GUI access and scripted deployments. Tenable.sc and QualysGuard are documented here because they require manual credentials and endpoint configuration that the wizard does not collect.

The procedure for both integrations follows the same shape:

1. Enable the pipeline in the Network Probe configuration. Edit `/etc/logserver-probe/pipelines.yml` and uncomment the relevant block:

For Tenable.sc:

```
- pipeline.id: tenable.sc
  path.config: "/etc/logserver-probe/conf.d/tenable.sc/*.conf"
```

For QualysGuard:

```
- pipeline.id: qualys
  path.config: "/etc/logserver-probe/conf.d/qualys/*.conf"
```

2. Configure the connection to the scanner manager by editing the integration's `main.py` file.

   For Tenable.sc, edit `/etc/logserver-probe/conf.d/tenable.sc/venv/main.py`:

   - `TENABLE_ADDR`: IP address and port of the Tenable.sc manager
   - `TENABLE_CRED`: username and password
   - `NETWORK_PROBE_ADDR`: IP address and port of the Network Probe

   Example:

```
TENABLE_ADDR = ('10.4.3.204', 443)
TENABLE_CRED = ('admin', 'password')
NETWORK_PROBE_ADDR = ('127.0.0.1', 10000)
```

   For QualysGuard, edit `/etc/logserver-probe/conf.d/qualys/venv/main.py`:

   - `NETWORK_PROBE_ADDR`: IP address and port of the Network Probe
   - `hostname`: address of the QualysGuard manager
   - `username` and `password`: credentials for the QualysGuard manager

   Example:

```
NETWORK_PROBE_ADDR = ('127.0.0.1', 10001)

conn = qualysapi.connect(
    username="admin",
    password="password",
    hostname="qualysguard.qg2.apps.qualys.eu"
)
```

3. Restart the Network Probe service to apply the changes:

```
systemctl restart logserver-probe
```

The collected data appears in the integration's dashboards (`[Vulnerability] Overview Tenable` for Tenable.sc, `[Vulnerability] Overview Qualys` for QualysGuard) once the pipeline runs and indexes its first results.

Creating Integrations with the AI Assistant

The **AI Assistant** tile in the Integrations catalog opens a creator that builds an integration for a log source the catalog does not cover. You give it sample logs, and it generates a Network Probe pipeline, a dashboard and a set of alerts, then installs them like any other integration.

Before You Start

- The license must include the AI Assistant add-on.
- You need the `admin` role. Other users can open the creator, but **Start** stops with *Not Authorized* under **Index name**.
- At least one Network Probe must be registered. The creator deploys the pipeline to every registered probe.
- The assistant uses the default connection from **AI Assistant > Providers**, see *AI Assistant*.
- Prepare a sample file with real logs from the source: `.json`, `.ndjson`, `.jsonl`, `.log` or `.txt`. In a text file each line is one sample.

The assistant sends the selected samples to the AI provider, at most 10 samples and at most 16,000 characters of each. See *Data Processing and Privacy*.

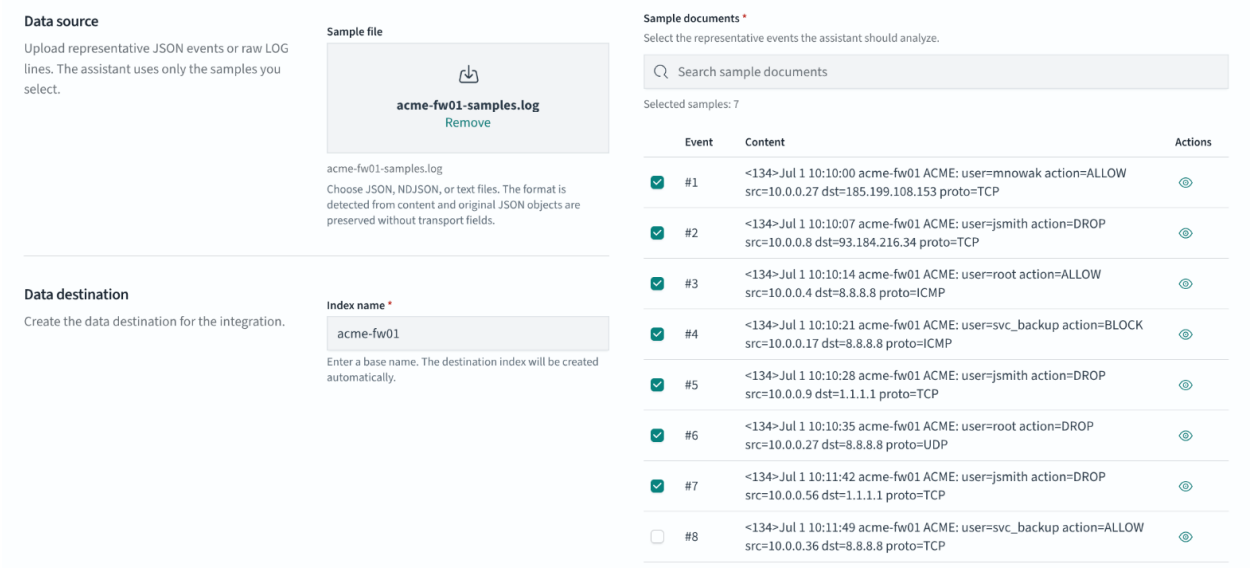
Creating an Integration

Open **Tools > Integrations** and click the **AI Assistant** tile. The creator has five steps. The example below uses a firewall that sends `key=value` syslog, the same source as in *Custom Integration*.

Step 1: Configuration

1. Enter a **Name**. It can contain letters, digits and spaces.
2. Optionally upload an **Icon file** (PNG, JPEG, SVG or WebP, up to 1 MB).
3. Upload the **Sample file** and select the samples in **Sample documents**. The assistant analyzes only the selected samples.
4. Enter the **Index name**, for example `acme-fw01`.
5. Click **Start**.

Start creates the index behind an alias with the index name (`acme-fw01` points to `acme-fw01-2026.09.29-000001`), a Data View for the alias, and a draft of the integration.



Step 2: Pipeline

1. Choose the input, **UDP** (default) or **TCP**, and a **Port** between 1024 and 65535 that is free on every Network Probe.
2. Click **Generate filter**. The assistant shows its progress. With 7 samples and the Energy Logserver provider it takes up to a minute; a slower provider takes longer.
3. Review the filter. Click **Edit** to change it, or **Add manually** to write your own.
4. Optionally enable **GeoIP** and **MISP blacklists**. The creator offers them when the filter produces compatible fields, such as `siem.src.ip` and `siem.dst.ip`.
5. Click **Next**.

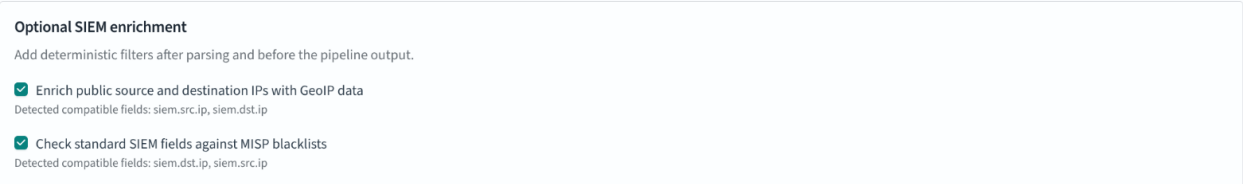
The assistant maps the fields it recognizes to `siem.*` (in the example `src` to `siem.src.ip` and `user` to `siem.subject.username`) and keeps the other fields under their own names. The pipeline always moves `message` to `siem.message` at the end.

Check the filter before you continue, because the assistant can return a different filter for the same samples. In the example the date pattern did not match the log, so every event got the `_np_parse_failure` tag and `@timestamp` holds the time the probe received the event.

Next opens the port in the firewall of each probe and deploys the pipeline to `/etc/logserver-probe/conf.d/<Name>_integration/`. If any probe cannot open the port, the creator shows *Input port is unavailable*, removes the pipeline and closes the port on every probe. Choose another port and click **Next** again.

The pipeline output writes to the alias with the probe user and the data node address from the Network Probe keystore, which the license service fills in. The generated output contains `ssl_certificate_verification => false`.

```
      mutate {
        convert => {
          "priority" => "integer"
        }
      }
      mutate {
        copy => {
          "priority" => "[siem][event][level]"
          "program" => "[siem][proc][name]"
          "user" => "[siem][subject][username]"
          "src" => "[siem][src][ip]"
          "dst" => "[siem][dst][ip]"
          "proto" => "[siem][net][protocol]"
        }
      }
    }
    else { mutate { add_tag => ["_np_unmatched_variant"] } }
    mutate { remove_field => ["[@metadata][np_selected_variant]", "[@metadata][np_variant_recognition]"] }
}
```



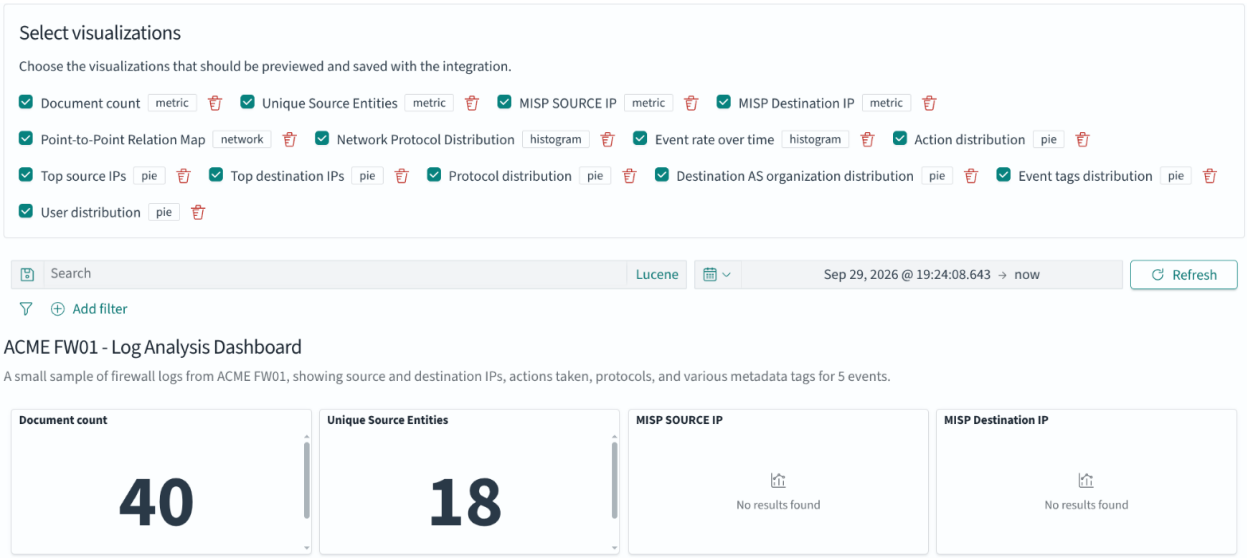
Step 3: Activation

Send an event from the source to the port you chose. The creator checks for new data every 5 seconds and moves on when the first event arrives; **Refresh** checks immediately. Click **Next**.

From this step on, Configuration and Pipeline are read-only.

Step 4: Dashboards

Click **Generate dashboard**. The assistant proposes panels with a live preview, up to 20. In the example it took about 20 seconds and proposed between 9 and 14 panels in four runs. Choose the panels under **Select visualizations** and click **Next**, or click **Skip dashboard**.



Step 5: Alerts

Select the alerts to install from the list of proposals, for example *No logs received*, *Failed logins* or *MISP blacklist match*. An alert marked **AI Alert** installs together with its AI use case (*Anomaly Detection - Text*). **Save integration** stays disabled until you select at least one alert; **Skip alerts** saves the integration without alerts.

Click **Save integration**. The creator saves the integration, removes the draft and runs the One Click installation: Pipelines, Dashboards, Alerts and AI Use Cases. The package is stored in `/usr/share/logserver-gui/data/integrations/creator_repository/<Name>/1.0.0/`.

Drafts

The creator saves your progress as a draft. The draft tile in the catalog has the **Draft** badge and the date of the last change; click it to continue from the saved step. Two users can open the same draft, and the one who saves second gets *Dashboard progress could not be saved.*

Deleting a draft removes the progress, the deployed pipeline and the Data View, and closes the port. The index and its data stay.

Deleting an Integration

Delete the integration from the catalog as described in *Deleting Integrations*. The pipeline, the package, the dashboard, the alert rules and the AI use case are removed. The open firewall port, the index with its data, the Data View and the dashboard visualizations stay; remove them by hand if you no longer need them.

Known Issues in 8.1.0

- **Add a visualization** in the Dashboards step always fails with *This visualization was not added*. To add a panel, save the integration, open its dashboard, click **Edit > Create new**, save the visualization with **Save and return**, and save the dashboard. Installing the dashboards again from **Advanced** with **Overwrite** restores the original panels.

- The assistant does not generate new alerts. The Alerts step offers only the listed proposals.

4.17 API and Integrations

Programmatic access to platform capabilities through REST APIs and data import tools for automation and integration workflows.

The API and Integrations module provides comprehensive programmatic interfaces for interacting with Energy Logserver. The REST API enables automation of dashboard management, data operations, index administration, document manipulation, alert configuration, and AI model interactions. Additionally, the XLSX import functionality facilitates bulk data ingestion from Excel and CSV files directly into indices.

4.17.1 Overview

Energy Logserver's API capabilities empower developers and administrators to automate platform operations and integrate security workflows with external systems. Key features include:

- **REST API**: Full-featured HTTP API with endpoints for dashboards, data queries, index management, document operations, alerts, and AI models
- **XLSX Import**: Bulk data import from Excel (xlsx) and CSV files directly into indices
- **Authentication and Security**: Secure API access with multiple authentication methods
- **Comprehensive Documentation**: Complete API reference with request/response examples
- **Best Practices**: Guidelines for efficient API usage and integration patterns
- **Automation Ready**: Enable scripting, CI/CD integration, and workflow automation

4.17.2 In This Section

This chapter covers the complete API reference and data import capabilities.

REST API

- *Overview*
- *Connecting to API*
- *Dashboards API*
  - *Dashboards Import API*
  - *Dashboards Export API*
- *Energy Logserver API*
- *Energy Logserver Index API*
  - *Adding Index*
  - *Delete Index*
  - *Split Index*
  - *API Useful Commands*
- *Energy Logserver Document API*
  - *Create Document*

- *Delete Document*
- *Useful Commands*

Overview

This chapter provides documentation for the Energy Logserver SIEM APIs, covering REST endpoints, authentication methods, and data operations.

Energy Logserver SIEM exposes multiple API layers for different operational needs:

- **ELS REST API**: Core data operations and cluster management.
- **ELS Console API**: Dashboard management, saved objects, and visualizations.
- **ELS-specific APIs**: License management, SIEM operations, and agent communication.

Connecting to API

To connect to APIs, you can use basic authorization or an authorization token.

To generate the authorization token, run the following command:

```
curl -XPUT http://localhost:9200/_logserver/login -H 'Content-type:␣
↪application/json' -d '
{
  "username": "$USER",
  "password": "$PASSWORD"
}'
```

The result of the command will return the value of the token, and you can use it in the API by passing it as a "token" header, for example:

```
curl -H 'token: 192783916598v51j928419b898v1m79821c2' -X GET "http://
↪localhost:9200/_cluster/health"
```

Dashboards API

The Dashboards import/export APIs allow users to import dashboards along with all corresponding saved objects, such as visualizations, saved searches, and index patterns.

Dashboards Import API

Request:

POST /api/opensearch-dashboards/dashboards/import

Query Parameters:

- `force` (optional, boolean) - Overwrite any existing objects on ID conflict.
- `exclude` (optional, array) - Saved object types that should not be imported.

Example:

```
curl -XPOST -ulogserver: -H "osd-xsrf: true" -H "Content-Type: application/
↪json" "https://127.0.0.1:5601/api/opensearch-dashboards/dashboards/import?
↪force=true" -d@"${DASHBOARD-FILE}"
```

Dashboards Export API

Request:

GET /api/opensearch-dashboards/dashboards/export

Query Parameters:

- `dashboard` (required, array|string) - The ID(s) of the dashboard(s) to export.

Example:

```
curl -XGET -ulogserver: -H "osd-xsrf: true" -H "Content-Type: application/json
↪" "https://127.0.0.1:5601/api/opensearch-dashboards/dashboards/export?
↪dashboard=${DASHBOARD-ID}" > ${DASHBOARD-FILE}
```

Energy Logserver API

The Energy Logserver API is a typical REST API, and data is received in JSON format over the HTTP protocol. By default, the tcp/9200 port is used to communicate with the Energy Logserver API. For the purposes of examples, communication with the Energy Logserver API will be carried out using the *curl* application.

Program syntax:

```
curl -XGET -u login:password '127.0.0.1:9200'
```

Available methods:

- **PUT** - sends data to the server.
- **POST** - sends a request to the server for a change.
- **DELETE** - deletes the index/document.
- **GET** - gets information about the index/document.
- **HEAD** - is used to check if the index/document exists.

Available APIs by roles:

- **Index API** - manages indexes.
- **Document API** - manages documents.
- **Cluster API** - manages the cluster.
- **Search API** - is used to search for data.

Energy Logserver Index API

The indices APIs are used to manage individual indices, index settings, aliases, mappings, and index templates.

Adding Index

Adding Index - automatic method:

```
curl -XPUT -u login:password '127.0.0.1:9200/syslog-2024.04.14/_doc/1?
↪pretty=true' -d'{
    "host" : "firewall-01",
    "@timestamp" : "2024-04-14T10:00:00",
    "message" : "Connection accepted from 10.0.0.5"
    }'
```

You should see the output:

```
{
  "_index" : "syslog-2024.04.14",
  "_id" : "1",
  "_version" : 1,
  "result" : "created",
  "_shards" : {
    "total" : 2,
    "successful" : 1,
    "failed" : 0
  },
  "_seq_no" : 0,
  "_primary_term" : 1
}
```

The parameter `action.auto_create_index` must be set to `true`.

Adding Index - manual method:

Setting the number of shards and replicas:

```
curl -XPUT -u login:password '127.0.0.1:9200/winlogbeat-2024.04.14?pretty=true
↪' -d'{
        "settings" : {
        "number_of_shards" : 1,
    "number_of_replicas" : 1
    }
 }'
```

You should see the output:

```
{
  "acknowledged" : true
}
```

Command for manual index generation:

```
curl -XPUT -u login:password '127.0.0.1:9200/winlogbeat-2024.04.14/_doc/1?
↪pretty=true' -d'{
    "host" : "dc-01",
    "@timestamp" : "2024-04-14T10:00:00",
    "message" : "User logon event"
}'
```

You should see the output:

```
{
  "_index" : "winlogbeat-2024.04.14",
  "_id" : "1",
  "_version" : 1,
  "result" : "created",
  "_shards" : {
    "total" : 2,
    "successful" : 1,
    "failed" : 0
  },
  "_seq_no" : 0,
  "_primary_term" : 1
}
```

Delete Index

Delete Index - to delete an index you need to use the following command:

```
curl -XDELETE -u login:password '127.0.0.1:9200/syslog-2024.04.14?pretty=true'
```

The delete index API can also be applied to more than one index, by either using a comma-separated list, or on all indices by using `_all` or `*` as index:

```
curl -XDELETE -u login:password '127.0.0.1:9200/syslog-2024*?pretty=true'
```

To allow deletion of indices via wildcards, set the `action.destructive_requires_name` setting in the configuration to `false`.

Split Index

The split index API splits an existing index into a new index with more primary shards. Use this when an index grows beyond the capacity of its current shard configuration and you need to redistribute the data, for example, after updating an index template to accommodate higher data volume.

Requirements:

- The target index must not exist.
- The cluster health status must be green.
- The source index must be set to read-only before splitting.
- The number of primary shards in the target must be a multiple of the number in the source.
- The node performing the split must have enough free disk space to hold a second copy of the index.

Step 1. Set the source index to read-only:

```
curl -XPUT -u login:password '127.0.0.1:9200/syslog-2024.04.14/_settings?
↪pretty=true' -d '{
  "settings": {
    "index.blocks.write": true
  }
}'
```

Step 2. Split the index:

```
curl -XPOST -u login:password '127.0.0.1:9200/syslog-2024.04.14/_split/syslog-
↪2024.04.14-split?pretty=true' -d '{
  "settings": {
    "index.number_of_shards": 2
  }
}'
```

The request returns immediately after the target index is added to the cluster state. The split operation runs in the background.

Monitor progress:

```
curl -XGET -u login:password '127.0.0.1:9200/_cat/recovery/syslog-2024.04.14-
↪split?v'
```

Once the split is complete, remove the write block from the source index if it is no longer needed:

```
curl -XPUT -u login:password '127.0.0.1:9200/syslog-2024.04.14/_settings?
↪pretty=true' -d '{
  "settings": {
    "index.blocks.write": null
  }
}'
```

API Useful Commands

Get information about replicas and shards:

```
curl -XGET -u login:password '127.0.0.1:9200/syslog-2024.04.14/_settings?
↪pretty=true'
```

```
curl -XGET -u login:password '127.0.0.1:9200/winlogbeat-2024.04.14/_settings?
↪pretty=true'
```

Get information about mapping and alias in the index:

```
curl -XGET -u login:password '127.0.0.1:9200/syslog-2024.04.14/_mappings?
↪pretty=true'
```

```
curl -XGET -u login:password '127.0.0.1:9200/syslog-2024.04.14/_aliases?
↪pretty=true'
```

Get all information about the index:

```
curl -XGET -u login:password '127.0.0.1:9200/syslog-2024.04.14?pretty=true'
```

Check if the index exists:

```
curl -XHEAD -u login:password '127.0.0.1:9200/syslog-2024.04.14?pretty=true'
```

Close the index:

```
curl -XPOST -u login:password '127.0.0.1:9200/syslog-2024.04.14/_close?
↪pretty=true'
```

Open the index:

```
curl -XPOST -u login:password '127.0.0.1:9200/syslog-2024.04.14/_open?
↪pretty=true'
```

Get the status of all indexes:

```
curl -XGET -u login:password '127.0.0.1:9200/_cat/indices?v'
```

Get the status of one specific index:

```
curl -XGET -u login:password '127.0.0.1:9200/_cat/indices/syslog-2024.04.14?v'
```

Display how much memory is used by the indexes:

```
curl -XGET -u login:password '127.0.0.1:9200/_cat/indices?v&h=i,tm&s=tm:desc'
```

Display details of the shards:

```
curl -XGET -u login:password '127.0.0.1:9200/_cat/shards?v'
```

Energy Logserver Document API

Create Document

Create a document with a specified ID:

```
curl -XPUT -u login:password '127.0.0.1:9200/syslog-2024.04.14/_doc/1?
↪pretty=true' -d'{
    "host" : "firewall-01",
    "@timestamp" : "2024-04-14T10:00:00",
    "message" : "Connection accepted from 10.0.0.5"
}'
```

You should see the output:

```
{
  "_index" : "syslog-2024.04.14",
  "_id" : "1",
  "_version" : 1,
  "result" : "created",
  "_shards" : {
    "total" : 2,
    "successful" : 1,
    "failed" : 0
  },
  "_seq_no" : 0,
  "_primary_term" : 1
}
```

Creating a document with an automatically generated ID (note: PUT -> POST):

```
curl -XPOST -u login:password '127.0.0.1:9200/syslog-2024.04.14/_doc?
↪pretty=true' -d'{
```

(continues on next page)

(continued from previous page)

```
    "host" : "firewall-01",
    "@timestamp" : "2024-04-14T10:10:00",
    "message" : "Connection denied from 192.168.1.100"
}'
```

You should see the output:

```
{
  "_index" : "syslog-2024.04.14",
  "_id" : "AV49sTlM8NzerkV9qJfh",
  "_version" : 1,
  "result" : "created",
  "_shards" : {
    "total" : 2,
    "successful" : 1,
    "failed" : 0
  },
  "_seq_no" : 1,
  "_primary_term" : 1
}
```

Delete Document

Delete a document by ID:

```
curl -XDELETE -u login:password '127.0.0.1:9200/syslog-2024.04.14/_doc/1?
↪pretty=true'
```

```
curl -XDELETE -u login:password '127.0.0.1:9200/syslog-2024.04.14/_doc/
↪AV49sTlM8NzerkV9qJfh?pretty=true'
```

Useful Commands

Get information about the document:

```
curl -XGET -u login:password '127.0.0.1:9200/syslog-2024.04.14/_doc/1?
↪pretty=true'
```

You should see the output:

```
{
    "_index" : "syslog-2024.04.14",
    "_id" : "1",
    "_version" : 1,
    "_seq_no" : 0,
    "_primary_term" : 1,
    "found" : true,
    "_source" : {
        "host" : "firewall-01",
        "@timestamp" : "2024-04-14T10:00:00",
        "message" : "Connection accepted from 10.0.0.5"
```

(continues on next page)

(continued from previous page)

```
    }
}
```

Get the source of the document:

```
curl -XGET -u login:password '127.0.0.1:9200/syslog-2024.04.14/_source/1?
↪pretty=true'
```

XLSX Import

Import Excel (xlsx) and CSV files into Energy Logserver indices.

The XLSX Import module allow to import your `xlsx` and `csv` file to indices.

Importing steps

1. Go to XLSX Import module and select your file and sheet:
2. After the data has been successfully loaded, you will see a preview of your data at the bottom of the window.
3. Press `Next` button.
4. In the next step, enter the index name in the `Index name` field, you can also change the pattern for the document ID and select the columns that the import will skip.
5. Select the `Configure your own mapping` for every field. You can choose the type and apply more options with the advanced JSON.
6. After the import configuration is complete, select the `Import` button to start the import process.
7. After the import process is completed, a summary will be displayed. Now you can create a new index pattern to view your data in the Discover module.

4.18 Changelog

- *v8.1.0*
- *v8.0.0*
- *v7.9.0*

4.18.1 v8.1.0

Breaking Changes

- **Alert:** The Microsoft Teams (`ms_teams`) alert type has been removed from the alert engine. Rules that use it must be moved to another notification method before the upgrade.

For detailed upgrade instructions, see *Upgrade to Version 8.1.0*.

Upgrades

- SIEM Engine upgrade to version 4.14.6
- Alerting upgrade to version 2.30.0
- Node.js upgrade to version 22.23.2

New Features

- Integrations/AI Assistant: Integrations can now be generated with the AI Assistant. This includes:
  - Field standardization for pipelines
  - Automatic parser generator
  - Standardized visualizations
  - Standardized alerts
  - Automatic dashboard generator
  - Port verification
  - Pipeline verification
  - Data-source selection
  - Alias support for the output index
  - Automatic index-pattern and alias creation
  - A “waiting for data” step in the Integration creator
  - Progress and state persistence of a created integration
  - Adding hosts to the keystore directly from the Integration creator
- SIEM Engine: Option to update all connected agents from the central GUI console

Improvements

- AI Assistant: Renamed the refresh button for clarity
- Alert: Added keyword_postfix support for the New Term rule type
- Alert: Added the ability to refresh IoC blacklists
- Alert: Support for the new summary_table_fields header
- Audit: Audit documents now record the real client source IP instead of the GUI (localhost) address
- Empowered AI: Cleaned up names and mappings of indices related to Intelligence
- Installation: install.sh now updates the keystore with the Probe user during Network Probe upgrades
- Integrations: Improved the integration uninstall/delete process
- License-Service: Added the ability to force the IP/hostname a probe registers under
- Network-Probe: Added a new input plugin for Google Pub/Sub
- Network-Probe: Fixes in create-nfacctd-instance.sh and create-sfacctd-instance.sh scripts for pmacct
- Reports: Report scheduling is now fully integrated with the main Scheduler

Bug Fixes

- Agents: MasterAgent 8.0.0 produced empty zip archives on Windows and Linux
- AI Assistant: Chapter names must now be unique
- AI Assistant: Chat box did not receive input focus
- AI Assistant: Incorrectly handled Discover data made the component disappear
- AI Assistant: Polish diacritics broke license handling
- Alert: Alert body composer did not validate that the generated body was valid YAML
- Alert: Alert definitions list order kept jumping around
- Alert: BlacklistIocRule used a custom, outdated generate-link
- Alert: Component did not correctly restore to a clean state after rule creation
- Alert: Enable/disable tooltip fixed
- Alert: Incorrect URL passed to discover_url while parsing YAML files
- Archive / Scheduler: Archive tried to update the audit document even when no scheduling (run now) was set
- Audit: ALERT RULE EDITED audit entry not generated correctly
- Audit: ARCHIVAL TASK CREATED audit content parsed incorrectly
- Audit: Audit exclusion configuration did not include newly created users
- Auth: Disabling MSSP incorrectly affected the ingest pipeline
- Auth: Empty values were cached when mssp_id decryption failed
- Auth: prepareindex for audit ignored the plugin's new mapping
- Auth: Username added via the Task API did not work correctly
- CMDB: Date filter did not work on first entering the plugin
- CMDB: Filtering issue
- Empowered AI: Removed an outdated log4j-core
- Index Management: Updated default sample tasks/examples to the new format
- Installation: install.sh Java update fix
- Installation: logserver.service now has proper permissions to write data to the /data path
- Integrations: Creating a new pipeline overwrote currently running pipelines
- License-Service: Hash key was written even after a failed license-read attempt
- License-Service: Removed or disabled pipelines remained stuck on the list as "unverified"
- Login: App Access for SIEM Engine stopped working after upgrade
- Login: Skip loading filter if navigating to a saved search without params on Discover page
- Login: Fixed log flooding caused by freeSocketKeepAliveTimeout
- Login: Global banner caused flicker/jumping when scrolling on some pages
- Network-Probe: Added a file move to the cron configuration when the system user was renamed to logserver-probe
- Network-Probe: File create/edit modal ignored file extensions configured on the License-Service side

- Network-Probe: File syntax verification error
- Network-Probe: Incorrect file filter for files linked to a pipeline
- Network-Probe: Logserver output plugin failed with an "uninitialized constant . . . OpenSSL" error
- Network-Probe: logstash-output-logserver failed to parse a value when validating an MSSP field
- Network-Probe: Managed service names displayed incorrectly
- Network-Probe: Removed a vulnerable log4j-core 2.9.1 bundled in the fat jar (CVE-2021-44228, "Log4Shell")
- Reports: Fixed deletion of HTML reports
- Reports: Manual report deletion failed due to incorrect user-context handling
- Scheduler: Improper handling of the audit index when used as an alias
- Scheduler: Plugins using Scheduler relied on outdated index mappings
- SIEM Engine: Vulnerability detection did not correctly handle permissions required by manager
- Utils: GeoIP database update corrupted database fix
- Utils: GeoIP database-update cron job ran under the wrong user

4.18.2 v8.0.0

Breaking Changes

- **Full cluster restart required:** Rolling upgrades from 7.x to 8.x are not supported. All nodes must be stopped and upgraded together.
- **JVM options location:** The file `/etc/logserver/jvm.options` is now overwritten during upgrades. All custom JVM settings must be placed in `/etc/logserver/jvm.options.d/`. Heap size moved to `/etc/logserver/jvm.options.d/heap-size.options`, temporary directory to `/etc/logserver/jvm.options.d/tmp-dir.options`.
- **Service names:** the services run as `logserver`, `logserver-gui`, and `logserver-probe`. The previous names remain systemd aliases, so `systemctl` still accepts them, but `journalctl -u` requires the new names.
- **Index compatibility:** Indices created on versions older than 7.4.0 must be reindexed or deleted before upgrading. The installer performs an automatic compatibility check.
- **Empowered AI:** Forecast and Relationship Mining have been removed.

For detailed upgrade instructions, see *Upgrade to Version 8.0.0*.

Upgrades

- Energy Logserver database upgrade to 3.3.2 - faster engine (newer Lucene + query optimizations), better data compression, optimised for vectors
- User interface upgrade to version 3.3.0
- Complete SIEM Engine upgrade to version 4.14.2
- Complete Network Probe upgrade to version 7.17.29
- Complete Alerting upgrade to version 2.29.0
- Support for Beats OSS Agents => 7.17.29

New Features

- Empowered AI: AI Agent capable of operating on large datasets
- Empowered AI: Introducing Chapters - datasets for AI Agents
- Empowered AI: A new Knowledge tab has been added, allowing users to manage knowledge as chapters for AI
- Empowered AI: Knowledge Chapters can now be created based on data from Discover
- Empowered AI: Knowledge can now be uploaded as PDF files, creating additional chapters
- Empowered AI: It is now possible to select multiple knowledge sources called chapters during conversations with the agent
- Empowered AI: Support for vector storage and semantic search has been implemented
- Empowered AI: The vectorization job now includes execution time estimation and document throughput tracking
- Installation: Better support for offline / security zone installs, the installation process supports the –offline switch
- Scheduler: Scheduled tasks are now audited and have their own history - for Archive, Sync, Index Management modules

Improvements

- Alert: Ability to include alert fields directly from Discovery for Quick Alert feature
- Empowered AI: Added an AI On-Prem as provider to the providers list
- Empowered AI: Added a Run button in the Performance tab
- Empowered AI: Added cluster statistics for the clustering use case
- Empowered AI: Added the ability to dismiss notifications about issues with Empowered rule verification
- Empowered AI: Tabs order updated to: Prompts, Knowledge, Providers
- AI Assistant: AI will create Alert based on semantic meaning of logs
- AI Assistant: Improved assistant access control: anyone can execute prompts, but only users with admin/intelligence roles can modify them
- AI Assistant: Improved error handling and user-facing error messages
- AI Assistant: Preloading prompts for logs
- AI Assistant: Improved prompts: Explain log and Detect threat
- AI Assistant: Prompt list sorting improved: editing a prompt no longer changes its position
- Installation: License verification supports local files and REST API
- Installation: License will be automatically loaded into product database state
- Integrations: Improved error handling when activating integration
- Integrations: Mikrotik integration updated
- License-Service: Implemented round-robin for accessing many configured hosts (for license verification)
- Network-Probe: Information about syntax errors directly while editing files
- Network-Probe: Stopping the service from the GUI prevents the service from auto-starting after the machine reboot
- Reports: Ability to filter export fields directly from Discovery for Quick Export feature

- Reports: Graphical reports are now faster and lighter
- Scheduler: Allows to configure the history period of running job
- SIEM Engine: New outlook of generated reports
- Sources Management: Improved presentation of data-sources in the Sources tab

Bug Fixes

- Alert: Quick Alert - creating an alert incorrectly assigns it to the "Other" group and it is not visible on the Alert list
- Archive: Restore now recreates metadata if missing
- Audit: Auditing includes all actions from the Alert and Index-Management modules
- Empowered AI: Fixed issues with agent performance for MSSP license
- Index Management: proper handling of too short index_age warning
- Installation: Improved user initialization process during installation
- Installation: Support for different line breaks in the license .info file
- License: In case of expired license added possibility to upload a new one using GUI
- License: Corrected handling of .info files causes compatibility issues
- License-Service: Corrected error about a busy port
- Login: Ldap authenticator - no escaping of special characters in ldapSearch
- Network-Probe: Corrected file permissions when editing from the GUI
- Network-Probe: Corrected verification for creating intelligence group in %post
- Network-Probe: Restored the ability to use the pipeline directive for non-MSSP modes
- Scheduler: Corrected handling of keystore in logserver-gui

4.18.3 v7.9.0

New Features

- Empowered AI: AI Assistant - Meet "Prem", our language model assistant
- Empowered AI: AI Assistant - work with Energy AI Provider or choose local LLM as AI on Prem
- Installation: Support for distributions based on Red Hat Enterprise Linux (RHEL) version 9
- Integrations: Barracuda integration with dedicated dashboard and SIEM alerts
- Login: The ability to limit the number of users in roles/groups(RBAC) has been introduced
- new plugin: CMDB! Get a list of all sources connected to SIEM
- new plugin: CRUD! CRUD indexes can work as internal database with edit and update option. Easy way to enrich SIEM with custom records.

Improvements

- Agents: Introduced version checking of connected agents
- Agents: Support for additional configuration files for Integrations plugin
- Alert: IoC blacklists - support for End of Life (How long an IoC remains useful varies and is dependent on factors including initial confidence level, fragility, and precision)

- Alert: Visibility of UI elements related to the Alerts/Quick Alert based on user role
- Empowered AI: Create own AI prompts using buttons available from Discover mode
- Empowered AI: Simplified installation of Use Cases from the Empowered AI Store - added Advanced mode as an option
- Index Management: Validation/User Notification: If a user sets the automatic data deletion period to less than 90 days, a warning will be displayed
- Installation: The base repository is now sufficient for installation, the epel repository is no longer required
- Integrations: Improved error handling when activating integration
- Integrations: Unification of categories and names in Integrations and AI Store
- Reports: Visibility of UI elements related to the Reports/Quick Report based on user role
- Skimmer: Unification of field names in cluster monitoring metrics
- Utils: configuration-backup.sh - added missing files: logserver.keystore & license files & empowered-ai configuration
- Utils: logserver-password-util.sh - default passstore renamed to logserver.keystore
- Utils: logserver-password-util.sh - improved error handling on missing files
- Utils: logserver-password-util.sh - support for passwords with special character

Bug Fixes

- Agents: Cleaning up unnecessary headers accumulating in the .agents index
- Alert: Problem with discover_url in incidents tab
- Empowered AI: AI Store - the option to select value for Rareness Threshold in advanced mode has been removed
- Empowered AI: Anomaly multivariate results - the table does not reset after clicking reset/refresh
- Empowered AI: API parameters escaping errors
- Empowered AI: Build time frame and/or start date are set incorrectly by default
- Empowered AI: Creating alert rules for text anomaly - a flat yaml file is not created with the rule
- Empowered AI: Creating incidents from anomalies fix
- Empowered AI: Deleting a rule forgets the page layout and search also sorting and filtering are cleared
- Empowered AI: Logging of empowered-ai-realtime plugin in Network-Probe
- Empowered AI: Logrotate configuration for the service (intelligence.log)
- Empowered AI: Type error for limit_to_values field
- Index Management: Default schedule cron pattern is incorrect in Rollover policy
- Index Management: Incorrect handling of verify errors in Rollover policy and custom rules
- Installation: Errors during installation due to the old cache version in sssd service
- Integrations: Integration title instead of name in manifest files
- Integrations: Prepareindex does not work in advanced mode
- Login: Mapping for .reports and .agents has been added by default
- Login: Role update for alerts and reports plugins

- Network-Probe: Security fixes for dependencies: CVE-2025-6442 (webrick), CVE-2025-46727 (rack), CVE-2024-49761 (rexml)
- Network-Probe: The API for verifying the status of all probes crashes when the registration documents in .networkprobes are malformed
- Reports: Missing logo in .docx reports
- Reports: Quick report does not take response from "pending" but from "running" state
- Reports: The "origin" field in "schedulingData" blocks the creation of scheduled reports for dashboard export
- SIEM Engine: GUI plugin overwrites the defaultIndex in the user config document

Older versions (v7.8.0 and below) - Click to expand

v7.8.0

New Features

- Archive: support for retention policy
- Empowered AI: Online AI Store - get new models from https://energylogserver.com/ai-store/
- Empowered AI: Online AI Store - integrated into Empowered AI Model Library directly from the system
- Empowered AI: Online AI Store - AI model is delivered along with Alert definition
- Empowered AI: Online AI Store - first Use Cases available for Netflow traffic
- Network-Probe: Ability to create new,custom pipeline from GUI
- new Plugin Integrations! Ability to manage integrations from GUI
- new SIEM visualization! Live Threat Globe Map
- Reports: Data Export Wizard from Discover tab

Improvements

- Alert: GUI - Default alert method set to NONE [create alert form]
- Alert: GUI - Deleting multiple alerts at once
- Alert: GUI - Enabled/disabled popup removed [create alert form]
- Alert: GUI - Group/Category selection [create alert form]
- Alert: GUI - No need to save alerts after creating/editing [create/edit alert form]
- Alert: Performance - fix for Blacklist-IOC Alert Rule
- Alert: Performance - size in query for aggregation in new_term rule
- Alert: Playbooks - added pagination & collapse list
- Empowered AI: Added a rule_name field to the results
- Empowered AI: Allow aggregations only if they are smaller than the prediction time
- Empowered AI: Time field taken from index pattern with the possibility of change + validation
- Network-Probe: Improved misleading message when stopping/starting pipeline
- Network-Probe: More filter plugins: added cybertrace, ldap, syslog filters
- Network-Probe: status_refreshing_interval_seconds = 60s as default
- Network-Probe: translog performance change - threshold_size:1mb, retention.age: 60s

- Reports: better support for heavy overloaded systems
- Reports: fully redesigned page for Data Export
- Reports: saved_search support for Data Export
- Reports: support for creating report for data without timestamp fields, like list of alerts
- SIEM Engine: enriched application logs with username person who actively work in user interface
- SIEM Engine: Documentation available on main screen as link to "Knowledge Base"
- SIEM Engine: logserver_auth: performance boost - synchronous API replaced by asynchronous API

Bug Fixes

- Alert: Manual Incident from Discover incorrectly saves discover_url field
- Alert: no access to the discover_url from the alert by groups other than admin
- Empowered AI: improvements for Anomaly Alerts in text
- Empowered AI: Dark theme fix
- Empowered AI: better doc_count support
- Empowered AI: Error after opening a rule in a new window when it contains special characters
- Empowered AI: Improved form for field mapping
- Empowered AI: No mapping for last_modified field
- Empowered AI: Sorting ai.anomaly_score and rules table
- Index Management: Illegal_argument_exception fix
- Index Management: Prepare_index exception fix
- Network-Probe: Added support for invalid registration documents
- Network-Probe: Fix installation of local .gem plugins
- Network-Probe: Keystore create exception fix
- Network-Probe: Registration ID should only be returned after the registration document has been successfully created
- Reports: Index list download for Data Export
- Reports: Timeout exception when generating Report export
- Reports: URL in the data export contains an incorrect address/hostname in the sent email
- SIEM Engine: Data table vis applies wrong filter when sorted
- SIEM Engine: logserver_auth: hanging of http response when using synchronous client
- SIEM Engine: patch for CVE-2025-2401 & CVE-2025-24016
- Task Management: support for universal date format

v7.7.0

New Features

- Alert: Wizard from Discover tab adds possibility to create alert rules
- Empowered AI: import and export mechanism of rules

- Empowered AI: rules as Use Cases with Categories!
- Integrations: Stormshield integration with dedicated dashboard and SIEM alerts
- OVA Appliance: base image is now Oracle Linux 8
- Security: force safe cipher/SSL settings for all Energy Logserver components

Improvements

- Login: system gain extra stability when running under a data flood state. SIEM will stop indexing before running out of free space. Administrator will be still able to log in and clear unnecessary data.
- Alert: added warning that changing a rule name affects chain and logical rules
- Archive: added debug logs that will be saved by default if a task has failed with any error
- Archive: added verification of the number of restored documents
- Archive: changed the default archives location to `/usr/share/logserver-gui/data/archive/archives/` - breaking change
- Archive: changed the way the files are recreated - added 'Recreate missing files' option [default: false] that verifies if all archived files exist in the archive folder path
- Archive: extended the functionality of archive verification on demand - integrity is verified, as well as checksum
- Archive: introduced `archive.integrityCheck` option [default: true] to verify the integrity of .zstd archive files at the end of the archivization task
- Archive: optimized the preparation of the archivisation process by checking its document counts and .zstd file size instead of always calculating checksum
- Empowered AI: ability to immediately stop building/scoring
- Empowered AI: ability to view AI rule while building/scoring
- Empowered AI: improvements in anomaly spread graph interactions
- Empowered AI: information that Univariate works on aggregations
- Installation: better support for multi-node environments
- Integrations: updated translate{} syntax to new format
- Integrations: redesigned welcome screen
- Network-Probe: clear info about actions that network probe is performing such as enabling pipelines or editing files
- Network-Probe: option to create label describing probe
- Network-Probe: option to navigate to pipeline's files directly from the details section
- Network-Probe: option to remove installed probe directly from the GUI - probe will be stopped as well as its services
- Network-Probe: template for `.networkprobes` index to ensure correct mapping in case of any issues
- Network-Probe: warning if probe's local time may be misconfigured
- Network-Probe: warning when probe version does not match the Energy Logserver SIEM version
- UBA/UEBA: new, sophisticated dashboards & documenation update
- Skimmer: updated to v1.0.26

Bug Fixes

- Alert: does not refresh aliases if roles for selected alerts are updated
- Alert: fixed discover_url feature
- Archive: fixed restoration of potentially corrupted archives - if any part of the file has been decompressed then an attempt will be made to upload its parts to the index
- Archive: issue with not using `last_archive_date` from archives metadata
- Empowered AI: progress bar bugfixes
- Empowered AI: fixed rules get stuck in Scoring status and never end [intelligence-scheduler]
- Network-Probe: fixed initialization process
- Network-Probe: fixed not closing managed services when shutting down [SIEM mode]
- Network-Probe: fixed time displayed in the pipeline details
- Network-Probe: logging - redirected some messages to the debug, while emphasizing only the important ones
- Network-Probe: management - in rare cases old documents from unregistered probes were corrupting the output of some APIs
- Network-Probe: not checking pipelines statuses when logserver-probe service is not reachable
- Network-Probe: suricata upgraded to v6.0.20
- Network-Probe: verification if probe is currently active and available
- Reports: issues with enabling/disabling scheduled reports

v7.6.0

New Features

- Energy Logserver SIEM LITE - Initial step into Cyber Security
- MSSP license provides transparent view to licensed sources
- Sources Management: identify each source activity
- UBA: SIEM module for tracking the nature of the user's behaviour and its changes
- UBA AI: predefinied AI rules to detect anomalies in user behavior
- Login: Introducing a new user to manage the system's graphical interface: admin (disables logserver account)

Improvements

- Alert: cross-field-correlation in Logical alert method
- Alert: improved time tolerance for run_once option
- Alert: mapping change to be able to search by rule name in alert_error indices
- Audit: new audit selection - now includes all plugins by default
- Archive: improved error handling and task resuming
- Cerebro: HSTS header support
- Config: configuration-backup.sh includes agent keys and siem-engine config file
- Config: logserver-password-util.sh: new password management tool changes system passwords easily
- Empowered AI: create rule with Saved Search or index pattern with query dsl

- Empowered AI: aggregation preview for univariate rule
- Empowered AI: improved model encryption
- Integrations: Dell Avamar added to default integrations
- Login: status-page link added to top menu
- Skimmer: output addresses support lists of nodes for fault tolerance
- Skimmer: support for URI schemes in config file, i.e. http, https
- Skimmer: migrated to libcurl
- Skimmer: updated to v1.0.25

Bug Fixes

- Alert: adding a note to an incident doesn't work
- Alert: long_term method does not work for objects
- Cerebro: fix for permissions verification
- Cerebro: limit on the used HEAP memory
- Empowered AI: anomalies increase by zooming in on the graph
- Empowered AI: cleared placeholder in text anomaly
- Empowered AI: incorrect handling of state:storeInSessionStorage
- Empowered AI: name change of the axis in the charts of univariate
- Empowered AI: performance tab display error for large number of fields in multivariate
- Empowered AI: rareness threshold does not fill in the scheduled model
- SIEM Engine: incorrect verification when diskspace thresholds are not enabled
- SIEM Engine: redirect loop after session expiration
- SIEM Engine: the list of domains available for logging does not refresh after changing the authorization plugin settings
- SIEM Engine: cluster does not start after disabling authorization plugin
- SIEM Engine: filter error messages in AuthService
- SIEM Engine: task of deleting old tokens now starts automatically
- Network-Probe: pipeline details are now displayed correctly
- Reports: banner with information about the expiring license or the diskspce visible on docx reports
- Reports: fix for executing scheduled reports
- Reports: visualization selection does not display all of them if there are more than 1000 of them
- SIEM Engine: on_start inventories disabled by default
- Task Management: show query duration
- Task Management: show user matching similar query

v7.5.0

New Features

- Empowered AI - anomaly detection in text message - rare words probability
- Empowered AI - anomaly detection in numbers
- Empowered AI - anomaly detection in multi dimention numbers
- Empowered AI - Root Cause tracing based on knowledge model
- Empowered AI - Relations Mining builds knowledge model
- Empowered AI - unsupervised data clustering
- Empowered AI - forecasting alerting method
- Empowered AI - AI input for network probe
- Empowered AI - realtime processing for AI rules
- Empowered AI - Model Library - save, store and upload AI models
- Empowered AI - Model Library - reuse and retraint saved models
- Empowered AI - Text Anomaly default alerts
- Empowered AI - create manual incident based on AI results
- Empowered AI - easy create alerts tab from AI rules config
- Empowered AI - progress bar for started rules

Bug Fixes

- Alert: added custom arguments to Energy SOAR integration
- Alert: added support for external_link to Energy SOAR integration
- Alert: groups management
- Alert: missing url in alert_text arguments of the Energy SOAR method
- Archive: clearing issue with empty with non existing file metadata
- Archive: scrolling in case of visible warnings
- CMDB: data fetching at the plugin startup
- Integrations: built-in templates now use wildcards
- SIEM Engine: improved alias refresh synchronization
- Network Probe: deleting config files and handling deleted files from disk
- Network Probe: filtering of probe's statuses has been fixed
- Network Probe: fixed when probes' services statuses were unavailable
- Network Probe: layout improvements and readability refinements
- Network Probe: updated log messages to be more comprehensible
- Network Probe: fixed permissions problem with external services
- Reports: improved handling of time fields
- SIEM Engine: improved RBAC mapping existence verification for non-admin users
- SIEM Engine: updated to v4.7.4 due cve [CVE-2023-42463, CVE-2024-32038]

- Task Management: improved filtering tasks by their duration

v7.4.3

New Features

- Query management: identify and stop long running query
- Introducing Network-Probe as mandatory Input Layer
- Archive: checksum verification on demand
- Empowered-AI: default AI forecasting rules
- License: GUI license upload with automatic distribution in cluster environment
- Introducing "Status page": showing health check in case of system problem without ability to log in
- Free space warning on status and login page
- Free space protection: Enabling Watermarks to keep system running in case of free space issue
- Audit: enchancements to audit more GUI actions

Improvements

- Alerts - Blacklist: wrong file name support
- Alerts: Risk key can be set on non default field - SOAR integration
- Alerts: secure und insecure webhook support
- Archive: Date format change to epoch in milisec
- Input layer uses Logserver-Probe-OSS 7.17.18
- license-service: dedicated API
- Skimmer: self monitoring of free space on cluster nodes
- Skimmer: self monitoring of license API status
- Support for Beats OSS Agents 7.17.18

Bug Fixes

- Alerts: cannot select more than one index-pattern when creating/editing a rule
- Alerts: empty role list when creating a rule without the admin role
- Alerts: Energy SOAR method wrong WYSIWYG behavior
- Alerts: errors when creating risks if any already exist
- Alerts: Manual Incident: user without admin role cannot create an incident
- Alerts: Manual Incident: user without admin role cannot see his incident
- Alerts: notifications are not sent as a valid HTML email
- Alerts: rule name change did not remove the old rule
- Archive: partial restore
- Archive: preparing data for archiving
- Audit: exclusions on _nodes and _stats do not work
- Audit: missing information about operations on users and roles

- Audit: missing query content - if selected
- Intelligence - view in discover: application not found
- license-service: memory limitations
- Login: AD login exception for users without mapped roles
- Login: SSO login duplicate users
- Reports: short link when creating docx report
- SIEM Engine: Agent/Client updated to v4.5.4
- SIEM Engine: permission denied after upgrade
- status_page: missing branding
- xlsx-import: fixing bug when writing more than 500 documents

v7.4.2

New Features

- Introducing Empowered-AI - Your data science module
- Empowered-AI: Forecasting usecase !
- Alerts: NEW rule type for Forecasting : Difference Multi Pattern - matches the difference between two index patterns calculated in a unit of time.
- Archive: repository validation (automatic scan of archive files and indices)
- SQL query support: query Your data with SQL query with dedicated GUI console
- Integrations: NEW Labyrinth - Deception-based threat detection

Improvements

- Archive: cataloging for better retention: $archivefolderpath/$year/$month
- Archive: sorting, pagination and filtering on task lists
- Archive: support for huge repositories
- Disaster Recovery: improvements during cluster initialization and recovery
- Disaster Recovery: logs for damaged indexes have been enriched with index_id
- Disaster Recovery: possibility of disabling the authorization plugin
- GUI: improvements in updating the client (browser) cache after Update
- license-service: possibility to change log_level & default log_level changed to WARN
- Reports: accept only the unix cron format in recurring reports
- Reports: clear descriptions for settings which deletes obsolete files
- Reports: dedicated MIME type for docx reports
- Reports: filenames created by recurring reports now based on creation date
- Sync: improved logging and error handling

Bug Fixes

- Archive: delete the results file when deleting a search task
- Archive: missing .zstd files and .dec files are not deleted after decryption
- Archive: unable to prepare data for selected indices fix
- Audit: user and role actions were filtered from audit queue due to missing username
- configuration-backup & support-tool: now supports all logserver versions
- E-doc: e-doc user requires gui-access to query the GUI authorization for a token
- GUI: wait until refreshAliases finishes at user login
- install.sh: problem with symlink when installing only the data-node
- Login: deprecated route to the default home plugin
- Reports: enable/disable for recurring report was not shown in GUI
- Reports: impossible to delete a recurring report without assigned file
- Reports: incorrect capture of “data table” and “tag cloud” visualization
- Reports: incorrect formatting of email messages and the “mail” command
- Reports: selected time field was not saved in the “data export” report
- Reports: temporary jpeg file not deleted after creating pdf report
- Reports: tsvb-based visualizations are incorrectly captured in docx reports
- Scheduler: “Archive task updated, but error occured when updating scheduler object. Please retry” fix
- Sync: tasks cannot be deleted
- Sync: unable to create/update profile
- xlsx-import: invalid file extension validation

SIEM Plan

- Alerts: NEW rule type: Difference Multi Pattern - matches the difference between two index patterns calculated in a unit of time.
- Alerts: bugfix: alert index rollover causes service errors
- Alerts: bugfix: sorting alert risk on incident tab did not work properly
- Alerts: bugfix: problem with updating alert rules
- Alerts: bugfix: Energy SOAR + metric_aggregation does not create artifacts
- Alerts: bugfix: Run Once old history after updating alert rule
- SIEM Engine: bugfix: duplicate index-pattern siem*

v7.4.1

New Features

- Reports: DOCX support!

Improvements

- Alert: multi-language support for alert rules
- API: gui-access role is required to interact with the API
- tlstool.sh: new ssl certificate management tool

Bug Fixes

- Archive: support for "secure" and "insecure" mode (without valid certificates)
- GUI: better-handled exceptions for custom plugins
- GUI: defaultAppId directive has been restored
- GUI: invalid directory for keystore
- GUI: Module Access Control permission fix
- GUI: users have aliases for different indexes after migration
- Index Management: missing verification for "on save" action
- Index Management: errors during rollover
- Index Management: filtering using the "Enabled" column
- Index Management: unable to update job after changing cron
- Integrations: improved command for importing dashboards
- Reports: custom logo moves the visualization on the dashboard
- Reports: deleting reports (multi, single) does not refresh the list
- Reports: enabling and disabling periodic reports by users
- Reports: incorrect visualization titles are inserted when creating a Data Table report
- Reports: long comment goes off the page when creating a PDF report
- Reports: long title goes off the page when creating a PDF report
- Reports: not translated statuses in the task list
- Reports: problem with Tag Cloud visualization when creating PDF report
- Reports: reports role paths to update, now require `.reports`
- Scheduler: status table sorted by "start date" instead of "name"
- Timeline/Timelion: regex not working due to an incorrectly built package

SIEM Plan

- Alerts: bugfix: incorrect _id of the edited alert causes duplicates
- Alerts: bugfix: unable to retrieve a list of risk key fields when updating a rule
- SIEM Engine: better-handled exceptions in RBAC integration

Security related fixes

- CVE-2023-32002
- CVE-2023-32006
- CVE-2023-32559
- CVE-2021-32014
- CVE-2021-32012
- CVE-2021-32013
- CVE-2023-30533
- CVE-2022-24785
- CVE-2022-31129
- CVE-2022-24785
- CVE-2022-31129
- CVE-2023-22467
- CVE-2023-30533
- CVE-2023-26115

v7.4.0

Upgrades

- Complete database redefinition:
  - Segment replication
  - Searchable snapshots
  - Search backpressure feature can now cancel queries at the coordinator level
- Complete user interface redefinition
- Complete SIEM Engine redefinition:
  - New manager
  - New App
  - New Agent
- Input layer uses Logserver-Probe-OSS 7.17.11
- Support for Beats OSS Agents => 7.17.11

New Features

- Logserver: RBAC integration with Wazuh Engine (users can map roles between systems)

Improvements

- CMDB: Browser-based Time Zone
- Improved error handling when reloading a license (*logserver/license/reload*)
- Archive: deleting tasks with multiselect option
- Unification and organization of Energy Logserver system APIs

- Alert: WebHook: added support for nested fields in http post payload
- Agents: built-in agents templates updated to 7.17.11

Bug Fixes

- CMDB: incorrect parsing of values in the date filter
- Archive: blank line in index list on restore

v7.3.0

New Features

- Multi-Language Support

Improvements

- Improved security by using response security headers
- Network Probe: version lock prevents accidental updates
- configuration-backup.sh activated by default

Bug Fixes

- Reports: usage of "Include unmapped fields" cause "No data" when exporting csv
- Agents: corrected manifest file for downloading agents
- Archive: error while restoring encrypted archives
- Cerebro: corrected auto-login after redirect

Integrations

- VMware: Integration with dedicated dashboard and alerts
- AWS: Integration with dedicated dashboard and alerts
- Ruckus Networks: Integration with dedicated dashboard and alerts
- Added Beats templates to beats integration

SIEM Plan

- WatchGuard: Integration with dedicated dashboard and alerts
- IDS Suricata: Integration with dedicated dashboard and alerts
- Alerts: updated rule database with 90 new alert rules including new Windows Security Group
- Alerts: bugfix: Jira integration
- Alerts: bugfix: duplication of alarms in specific cases
- Alerts: bugfix: top_count_keys doesn't work properly with multiple query_keys
- Alerts: bugfix: Broken Chain method TypeError
- Alerts: bugfix: Exclude Fields for Logical/Chain body correlation
- Alerts: NoLog rule for each alarm group

Network-Probe

- Added support for sFlow - sfacctd service
- Added IDS Suricata integration with dedicated dashboard and alerts

Security related

- log4j - logserver-probe-input-tcp

Required post upgrade

- Recreate bundles/cache: `rm -rf /usr/share/logserver-gui/optimize/bundles/* && systemctl restart logserver-gui`

v7.2.0

Breaking changes

- Login: changed how gui access is granted for administrative users - access for any administrator has to be explicitly granted
- Wiki portal renamed to E-Doc

New Features

- CMDB: Infrastructure - create an inventory of all sources sending data to SIEM
- CMDB: Relations - ability to create relation topology map based on sources inventory
- Extended auditing support - each plugin can be enabled in GUI config to save its actions in the audit index
- Syntax Assistant for Alerts, Agents, Index Management, Network Probe. Check YAML definition and structure

Improvements

- Update process will not override /etc/sysconfig/logserver config
- Clear GUI message for expired license
- Agents: improved services information display for not running agents
- Archive: optimization and improvements; added multi threaded processing and Task Retry support
- Login: redesigned audit selection and exclusion settings GUI
- Reports: tasks edit is now more robust and allows modification of advanced parameters
- Reports: moved settings into new Config tab in the plugin from Config -> Settings
- Alerts: loading new alarm Rule Set during update process [install.sh]
- Beats: updated to v7.17.8
- Skimmer: negotiate highest TLS1.3 version if possible
- Skimmer: fixes regarding ssl connection
- Skimmer: added logserver_ssl config option
- Skimmer: added new metric: node_stats_fs_total_free_in_pct
- Skimmer: updated to v1.0.22
- Elasticdump updated to v6.79.4

Bug Fixes

- Refreshing audit exclusions caused ELS node to freeze in rare cases
- Update process on RedHat 7.9 could not be run caused by missing package
- LDAP login: improved validation on username input
- Table visualization: fix for “Count percenteges”, which was inacurate in some cases
- Skimmer: sometimes did not start after installation
- Agents: small GUI improvements
- Alerts: long alert names presented outside the frame
- Alerts: sorting alert risk on incident tab did not work properly
- Intelligence: malware scanners would rise a false positive on one of the plugin dependencies
- Reports: data export (csv) improvements on file integrity
- Reports: a rare case of a race condition when removing temporary directories
- E-Doc: improvements to https handling when using Logserver as a search engine
- install.sh: installation process always uses LC_ALL=C

Integrations

- Added new integrations: FireEye, Infoblox, ArcSight Common Event Format

SIEM Plan

- Agents: SIEM agents updated to 3.13.6
- Alerts: new notification methods: ServiceNow, WebHook, TheHive, Jira
- Alerts: risk values on incident tab formated for clarity
- Alerts: example description supplied with new values regarding escalate and recovery
- Alerts: all alerts in a goup can be seen with a proper row selection
- Alerts: creating risks is now supported on no time based indices
- Alerts: long alert names presented outside of message frame
- Alerts: on incident tab sorting by risk did not work properly
- Alerts: added Ransomware Detection rules

Network-Probe

- Increased tolerance for status/verification calls

Security related

- axios - CVE-2021-3749
- qs - CVE-2022-24999
- express - CVE-2022-24999
- moment - CVE-2022-24785
- moment - CVE-2022-31129

- minimist - CVE-2021-44906
- char.js - CVE-2020-7746
- async - CVE-2021-43138
- minimist - CVE-2021-44906
- requestretry - CVE-2022-0654
- xmldom - CVE-2022-39353
- underscore - CVE-2021-23358
- flask-cors - CVE-2020-25032
- logserver-gui - CVE-2022-23707

Required post upgrade

- Recreate bundles/cache: `rm -rf /usr/share/logserver-gui/optimize/bundles/* && systemctl restart logserver-gui`
- Wiki portal renamed to E-Doc: if data migration is required follow the steps of UPGRADE.md

v7.1.3

Security related

- log4j updated to 2.19.0
- kafka updated to 2.13-3.3.1 (log4j dependency removed)
- logserver-probe: removed obsolete bundled jdk

v7.1.2

New Features

- Energy SOAR: Redesigned and improved integration (Security Orchestration, Automation And Response)
- Intelligence: Redesigned and improved Forecasting [experimental]
- Masteragent: New feature: Configuration Templates
- New plugin: CMDB - simple implementation of Configuration Management Database

Improvements

- es2csv - Performance boost and Memory optimization
- Reports: Support for large report files
- Redirection of HTTPS connection to GUI enabled by default - 443 => 5601
- Login: Home Page moved to Integrations Page
- diagnostic-tool.sh - Added logserver-probe logs
- Logserver: Global timeouts changed to 60s
- Updated LICENSE in all components
- Index Management: Prepare index has been moved from Config to Index-Management tab
- Masteragent: Setting authorization with a client certificate by default
- Masteragent: Possibility to fully disable the HTTP server on masteragent clients

Bug Fixes

- Login: Fixed problems with sharing Short Links
- Discover: Fixed problem with index-patterns name overlapping
- Index Management: Fixed execution time for builin logtrail policies
- Masteragent: Fixed error when getting installed services

Integrations

- windows-ad: Fixed error in Ad Accounts dashboard
- beats - Fixes in waf ruby filter

SIEM Plan

- Vectra.AI: Integration with dedicated dashboard and alerts
- MITRE added to SIEM Dashboard
- Agents: SIEM agents updated to 3.13.4
- Agents: Vulnerability detection & feeds enabled by default
- Alert: Simplified discover_url feature
- Alert: theHive project - Improved integration
- Alert: Fixed exception for risk query
- Alert: SIEM alert group changed to “Correlated”
- Alert: Fixed problem with TypeError: deprecated_search()
- Alert: Fixed logs problem after rotating the file
- Alert: Fixed permission problem in Run Once mode
- Alert: Fixed indentation in query_string
- [bugfix] Added missing library to Qualys Quard venv
- [bugfix] Added missing ports 1514udp-tcp/1515tcp to install.sh

Required post upgrade

- Recreate bundles/cache: `rm -rf /usr/share/logserver-gui/optimize/bundles/* && systemctl restart logserver-gui`
- (SIEM only) Update/ReImport SIEM Dashboard for MITRE

v7.1.1

New Features

- Logserver Join support - API level query

Improvements

- es2csv - Breakthrough (50%) performance boost
- es2csv - Renamed to els2csv
- diagnostic-tool.sh - Added logs encryption

- diagnostic-tool.sh - Renamed to `support-tool.sh`
- Skimmer: Indices_stats: run only on master node
- Skimmer: Added two metrics: indices_stats_patterns and indices_stats_regex
- Skimmer: Added cached info about nodes when poll errors out
- Logtrail: Disabled ratelimit in rsyslog for logtrail source files
- Logtrail: Parsing in pipeline for alert,logserver-gui,elasticearch,logserver-probe [added standardized log_level field]
- Logtrail: Added default filter showing only errors ["NOT log_level: INFO"]
- Index Management: Added built-in index policies for common actions
- Discover: Default QueryLanguage changed to Lucene
- Cerebro updated to v0.9.4
- Curator updated to v5.8.4
- Elasticdump updated to v6.79.4
- Wiki.js updated to v2.5.274

Bug Fixes

- Login: In case of unsuccessful login information about "redirection" is lost when using link sharing
- Login: When logging using SSO auth, it doesn't redirect when using link sharing
- Login: Fixed "unable to parse url" when using link sharing
- Login: Corrected Session expired message
- Login: gui-access role added to role-mappings.yml
- Login: When logging using SSO auth, sending the entered password as a default action
- Skimmer: Index store value of _cat/shards in bytes
- Skimmer: Disabled ssl handshake on logserver-probe api
- Logtrail: Corrected syntax highlighting
- Logtrail: Fixed filter selector on columns
- Discover: Fixed timeout handling
- Wiki: Removed gui-access group
- Index Management: Wait for updates before refreshing the list
- Index Management: Fixed id problem during custom update

Integrations

- windows-ad/beats: fixed error in ruby{} filter
- netflow - Fixes from 7.1.0
- netflow - network_vis - Fixed incorrect filtering
- netflow - network_vis - Added new option "skip null values"
- syslog-mail - Fixes from 7.1.0

SIEM Plan

- Added Log4j RCE attacks to Detection Rules [“Wazuh alert [HIGH] - rule group: custom - Log4j RCE”]
- Alert: Fixed problem with modifying alertrulemethod
- Alert: Fixed malfunction of Test Rule in case of “verify_certs: false” setting
- Alert: Simplified Discover URL
- Alert: Logtrail - Cluster Services Error Logs added to Cluster-Health group

Security related

- http-proxy - CVE-2022-0155
- xlsx - CVE-2021-32013
- json-schema - CVE-2021-3918
- lodash - CVE-2021-23337
- json-schema - CVE-2021-3918
- pdf-image - CVE-2020-8132
- angular-chart.js - CVE-2020-7746
- pyyaml - CVE-2020-14343
- cryptography - CVE-2020-25659
- aws-sdk - CVE-2020-28472
- pyyaml - CVE-2020-14343
- nodemailer - CVE-2020-7769
- objection - CVE-2021-3766
- socket.io - CVE-2020-28481
- nodejs - CVE-2021-44531

v7.1.0

New Features

- Added support for AlmaLinux and RockyLinux
- Agents: Added local repository with GUI download links for agents installs
- Archive: Added ‘Run now’ for scheduled archive tasks
- Archive: Added option to enable/disable archive task
- Archive: Added option to encrypt archived data
- Audit: Added report of non-admin user actions in GUI
- Logserver: Added field level security access control for documents
- Logserver-GUI: Added support for Saved Query object in access management
- Logserver-GUI: Added support for TLS v1.3
- Logserver-GUI: Added new plugin Index Management - automate index retention and maintanance

- Reports: Added new report type created from data table visualizations - allows creating a raport like table visualization including all records (pagination splitted into pages)
- Reports: Added option to specify report task name which sets destination file name

Improvements

- Security: log4j updated to address vulnerabilities: CVE-2021-44228, CVE-2021-45046, CVE-2021-45105, CVE-2021-44832, CVE-2021-4104
- Added new directives for LDAP authenctication
- Agents: Changed agent's action name from drop to delete
- Archive: Improvement and optimization of "resume" feature
- Archive: Optimised archivization proces by saving data directly to zstd file
- Archive: Multiple 'Upload' GUI improvements
- Archive: Improved logs verbosity
- Audit: Added template for audit index
- Beats: Updated to v7.12.1
- Curator: Added curator logs for rotation
- Logserver: Extended timeout for starting service
- Logserver: Updated engine to v7.5.2
- install.sh: Improved update section for better handling of services restart
- Logserver-GUI: Updated engine to v7.5.2
- Logserver-GUI: Clean SSL info in logs
- Logserver-GUI: Improved built-in roles
- Logserver-GUI: Disabled telemetry
- Logserver-GUI: Set Discover as a default app
- Logserver-GUI: Optimized RPM
- Logserver-GUI: Improved handling of unauthorized access in Discover
- Logserver-GUI: small changes in UI - Improved Application RBAC, product version
- Logserver-GUI: Added new logos
- Logserver-GUI: Improved login screen, unauthorized access info
- Logserver-GUI: Restricted access to specific apps
- Logserver-GUI: Added option to configure default app
- Logrotate: Added Skimmer
- Logserver-Probe: Updated to v7.12.1
- Network visualization: UI improvements
- Object permission: Index pattern optimizations
- Plugins: Moved Cluster Management inoto the right top menu, Scheduler and Sync moved to the Config
- Reports: Added report's time range info to raport details description

- small_backup.sh: Added cerebro and alert configuration
- Skimmer: Updated to v1.0.20
- Skimmer: Added new metrics, pgpgin, pgpgout
- Skimmer: Optimised duration_in_milis statistics
- Skimmer: Added option to specify types
- Skimmer: Added option to monitor disk usage
- Wiki: Added support for nonstandard logserver-gui port
- Wiki: Several optimizations for roles
- Wiki: Changed default search engine to logserver
- Wiki: Added support for own CAs
- Wiki: Default authenticator improvements
- XLSX Import: UI improvements

Bug Fixes

- Archive: Fixed problems with task statuses
- Archive: Fixed application crash when index name included special characters
- Archive: Fixed 'checksum mismatch' bug
- Archive: Fixed bug for showing unencrypted files as encrypted in upload section
- Logserver: Fixed bug when changing role caused client crash
- Elastfilter: Fixed "_msearch" and "_mget" requests
- Elastfilter: Fixed bug when index pattern creation as an admin caused logserver-gui failure
- Logserver-GUI: Fixed timeout handling
- Logserver-GUI: Fixed a bug causing application crash when attempting to delete data without permission to it
- Logserver-Probe: Fixed breaking geoip db when connection error occurred
- Object permission: Fixed adding dashboard when all its related objects are already assigned
- Reports: Added clearing .tmp files from corrupted csv exports
- Reports: Fixed sending PDF instead of JPEG in scheduled reports
- Reports: Fixed not working scheduled reports with domain selector enabled
- Skimmer: Fixed expected cluster nodes calculation
- Wiki: Added missing home page
- Wiki: Added auto start of wiki service after installation
- Wiki: Fixed logout behaviour

Integrations

- Fixed labels in Skimmer dashboard
- Fixed Audit dashboard fields
- Updated Windows + AD dashboard and pipeline

- Added Linux Mail dashboard and pipeline
- Added Cisco ASA dashboard and pipeline
- Added FortiGate dashboard and pipeline
- Added Paloalto dashboard and pipeline
- Added Oracle dashboard and pipeline
- Added Waystream dashboard and pipeline
- Added CEF dashboard and pipeline (CheckPoint, FireEye, Air-Watch, Infoblox, Flowmon, TrendMicro, CyberX, Juniper Networks)
- Added monitoring of the alert module on Alert Dashboard

SIEM Plan

- Updated SIEM dashboard
- Updated QualysGuard integration
- Updated Tenable.SC integration
- Alert: Updated detection rules (370+)
- Alert: Added Cluster-Health alert rules
- Wazuh: Updated to v3.13.3
- Wazuh: UI improvements
- Alert: Improved groups management
- Alert: Multiple UI/UX tweaks
- Alert: Revised alerts' descriptions and examples
- Alert: Adding included fields when invert:true
- Alert: Changed startup behaviour
- Alert: Added field from 'include' to match_body
- Alert: Optimised loading files with misp lists
- Alert: Added option to set sourceRef in alert definition
- Alert: Include & Exlcude in blacklist-ioc lists
- Alert: Fixed several issue in chain and logical alerts
- Alert: Fixed error when user tried to update alert from newly added group
- Alert: Fixed top_count_keys not working with multiple query_key
- Alert: Fixed bug when match in blacklist-ioc is breaking other rules
- Alert: Fixed empty risk_key breaking alert rule
- Alert: Fixed endless loop during scroll

Network-Probe

- Added integration with license service
- Changed plugin icon
- Changed default settings
- Changed logs mapping in logserver-probe
- Optimised netflow template to be more efficient
- Updated .service files
- Updated Network-Probe dashboard

API Changes

- Logserver: Updated API endpoints.
  - Following endpoints deprecated and update with:
    * `/_auth/account` -> `/_logserver/accounts`
    * `/_license/reload` -> `/_logserver/license/reload`
    * `/_role-mapping/reload` -> `/_logserver/auth/reload`
    * `/user/updatePassword` -> `/_logserver/user/password`
  - Following endpoint was removed and replaced with:
    * `/_license` -> `/_logserver/license`

Breaking changes

- During the update, the "logserver-gui" role will be removed and replaced by "gui-access", "gui-objects", "report". The three will automatically be assigned to all users that prior had the "logserver-gui" role. If you had a custom role that allowed users to log in to the GUI this WILL STOP WORKING and you will have to manually enable the access for users.
- The above is also true for LDAP users. If role mapping has been set for role logserver-gui this will have to be manually updated to "gui-access" and if required "gui-objects" and "report" roles.
- If any changes have been made to the "logserver-gui" role paths, those will be moved to "gui-objects". GUI objects permissions also will be moved to "gui-objects" for "gui-access" cannot be used as a default role.
- The "gui-access" is a read-only role and cannot be modified. By default, it will allow users to access all GUI apps; to constrain user access, assign user a role with limited apps permissions.
- "small_backup.sh" script changed name to "configuration-backup.sh" - this might break existing cron jobs
- SIEM plan is now a separate add-on package (requires an additional license)
- Network-Probe is now a separate add-on package (requires an additional license)
- (SIEM) Verify rpmsave files for alert and restore them if needed for following:
  - /opt/alert/config.yaml
  - /opt/alert/op5_auth_file.yml
  - /opt/alert/smtp_auth_file.yml

Required post upgrade

- Role “wiki” has to be modified to contain only path: “.wiki” and all methods

v7.0.6

New Features

- Alert: Added 5 alerts to detect SUNBURST attack
- Incidents: Added the ability of transferring the calculated risk_value to be sent in any alarm method
- Indidents: Added visibility of unassigned incidents based on user role - security-tenant role
- install.sh: Added the ability to update with ./install.sh -u

Improvements

- Object permission: Object filtering optimization
- Reports: Date verification with scheduler enabled tasks
- Reports: UI optimization

Bug Fixes

- Agents: CVE-2020-28168
- Alert: Fixes problem with Syslog notifications
- Alert: Fixes problem with Test Rule functionality
- Alert: CVE-2020-28168
- Archive: CVE-2020-28168
- Cerebro: CVE-2019-12384
- Logserver-GUI-xlsx-import: CVE-2020-28168
- Login: CVE-2020-28168
- Reports: CVE-2020-28168
- Reports: Fixes errors related to background tasks
- Sync: CVE-2020-28168

v7.0.5

New Features

- New plugin: Wiki - integration with wiki.js
- Agents: Added index rotation using rollover function
- Alert: Added counter with information about how many rules there are in a given group
- Alert: Added index rotation using rollover function
- Alert: First group will be expanded by default
- Alert: New Alert method for Syslog added to GUI
- Archive: Added compression level support - archive.compressionOptions [logserver-gui.yml]
- Archive: Added mapping/template import support

- Archive: Added number of matches in files
- Archive: Added regexp and extended regexp support
- Archive: Added size information of created archive on list of files for selection
- Archive: Added support for archiving a selected field from the index
- Archive: Added timestamp field for custom timeframe fields
- Audit: Added index rotation using rollover function
- Config: Added configuration possibility for Rollover (audit/alert/.agents indexes) in Settings tab
- Object Permission: When deleting an object to a role in "object permission" now is possible to delete related objects at the same time
- Reports: Ability to delete multiple tasks at once
- Reports: Added details field for each task that includes information about: user, time range, query
- Reports: Added Scheduler for "Data Export" tab
- Reports: Fields to export are now alphabetical, searchable list
- Reports: Scheduled tasks supports: enable, disable, delete
- Reports: Scheduled tasks supports: Logo, Title, Comments, PDF/JPEG, CSV/HTML
- Installation support for Centos7/8, RedHat7/8, Oracle Linux7/8, Scientific Linux 7, Centos Stream
- iFrame embedding support: new directive login.isSameSite in logserver-gui.yml ["Strict" or "None"]

Improvements

- Access management: Plugin Login for app management will show itself as Config
- Alert: Added support for nested fields in blacklist-ioc alert type
- Alert: Alert Dashboard rewritten to alert_status pattern - allows you to filter visible alarms per user
- Alert: Cardinality - fix for _thread._local' object has no attribute 'alerts_sent'
- Alert: Chain/Logical - few improvements for output content
- Alert: Rule type example is hidden by default
- Alert: RunOnce - improved results output
- Alert: RunOnce - information that the process has finished
- Alert: TestRule - improved error output
- Archive: Added document sorting, which speeds up logserver response
- Archive: API security -> only admin can use (previously only visual information)
- Archive: Archiving process uses a direct connection, bypassing the elastfilter - proxy
- Archive: Changed UTC time to local time
- Archive: Information about problems with reading/writing to the archive directory
- Archive: Optimized function for loading large files - improved loading time
- Archive: Optimized saving method to a temporary flat file
- Archive: Optimized scroll time which speeds up logserver response
- Audit: Converted SEARCH _id: auditselection to GET _id: auditselection

- Audit: Removed background task used for refresh audit settings
- Beats: Updated to v6.8.14
- Blacklist-IOC: Added Duplicates removal mechanism
- Blacklist-IOC: Automatic configuration of repository access during installation [install.sh]
- Cerebro: Updated to v0.9.3
- Config: Character validation for usernames and roles - can consist only of letters a-z, A-Z, numbers 0-9 and characters _,-
- Config: Deleting a user deletes his tokens/cookies immediately and causes logging out
- Config: Securing the default administrator account against deletion
- Config: Session timeout redirect into login screen from all modules
- Config: Workaround for automatic filling of fields with passwords in modern browsers
- Curator: Updated to v5.8.3 and added support for Python3 as default
- ElasticDump: Updated to v6.65.3 and added support for backup all templates at once
- Logserver: Removed default user "scheduler" with the admin role - is a thing of history
- Logserver: Removed indices.query.bool.max_clause_count from default configuration - causes performance issues
- Logserver: Role caching improvements
- GEOIP: Automatic configuration of repository access during installation [install.sh]
- Incidents: Switching to the Incidents tab creates pattern alert* if not exist
- install.sh: Added workaround for cluster.max_shards_per_node=1000 bug
- Logserver-GUI: Removed logserver-gui.autocomplete from default configuration - causes performance issues
- License: Revision and update of license files in all system modules
- Logserver-Probe: Updated logserver-probe-codec-sflow to v2.1.3
- Logserver-Probe: Updated logserver-probe-input-beats to v6.1.0
- Logserver-Probe: Updated to v6.8.14
- Logtrail: Added default actionfile for curator - to clean logtrail indexes after 2 days
- Network visualization: corrected legend and better colors
- Reports: Added Switch button for filtering only scheduled tasks
- Reports: Admin users should see all scheduled reports from every other user
- Reports: Changed "Export Dashboard" to "Report Export"
- Reports: Changed "Export Task Management" to "Data Export"
- Reports: Crontab format validated before Submit in Scheduler
- Reports: Default task list sorted by "start time"
- Reports: Improved security by using kernel namespaces - dropped suid permissions for chrome_sandbox
- Reports: Moved "Schedule Export Dashboard" to "Report Export" tab
- Reports: Try catch for async getScheduler function

- Skimmer: Added alerts: High_lag_on_Kafka_topic, High_node_CPU_usage, High_node_HEAP_usage, High_Flush_duration, High_Indexing_time
- Skimmer: New metric - _cat/shards
- Skimmer: New metric - _cat/tasks
- Skimmer: Updated to v1.0.17
- small_backup.sh: Added sync, archive, wiki support
- small_backup.sh: Information about the completed operation is logged
- Wazuh: Searching in the rule.description field

Bug Fixes

- Access Management: Cosmetic issue in apps select box for default roles (like admin, alert, intelligence, logserver-gui etc.)
- Alert: Category name did not appear on the "Risk" list
- Alert: Description update for find_match alert type
- Alert: Fixes bug where after renaming the alert it is not immediately visible on the list of alerts
- Alert: Fixes bug where editing of alert, causes it returns to the Other group
- Alert: Fixes incorrect function alertMethodData - problem with TestRule operation [itrs op5 alert-method]
- Alert: Fixes problem with '[]' in rule name
- Alert: Fixes process status in Alert Status tab
- Alert: In groups, if there is pagination, it is not possible to change the page - does not occur with the default group "Others"
- Alert: Missing op5_url directive in /opt/alert/config.yaml [itrs op5 alert-method]
- Alert: Missing smtp_auth_file directive in /opt/alert/config.yaml [itrs op5 alert-method]
- Alert: Missing username directive in /opt/alert/config.yaml [itrs op5 alert-method]
- Alert: Overwrite config files after updating, now it should create /opt/alert/config.yml.rpmnew
- Archive: Fixes exception during connection problems to logserver
- Archive: Missing symlink to runTask.js
- Cerebro: Fixes problems with PID file after cerebro crash
- Cerebro: Overwrite config files after updating, now it should create /opt/cerebro/conf/application.conf.rpmnew
- Config: SSO login misreads application names entered in Access Management
- Logserver: Fixes "No value present" message log when not using a radius auth [properties.yml]
- Logserver: Fixes "nullPointerException" by adding default value for licenseFilePath [properties.yml]
- Incidents: Fixes problem with vanishing status
- install.sh: Opens the ports required by logserver-probe via firewall-cmd
- install.sh: Set openjdk11 as the default JAVA for the operating system
- Logserver-GUI: Fixes exception during connection problems to logserver - will stop restarting
- Logserver-GUI: Fixes URL shortening when using Store URLs in session storage
- Logtrail: Fixes missing logrotate definitions for Logtrail logfiles

- Logtrail: Overwrite config files after updating, now it should create /usr/share/logserver-gui/plugins/logtrail/logtrail.json.rpmnew
- Object Permission: Fixes permission verification error if the overwritten object's title changes
- Reports: Fixes Image Creation failed exception
- Reports: Fixes permission problem for checkpass Reports API
- Reports: Fixes problems with AD/Radius/LDAP users
- Reports: Fixes problem with choosing the date for export
- Reports: Fixes setting default index pattern for technical users when using https
- Skimmer: Changed kafka.consumer_id to number in default mapping
- Skimmer: Fixes in indices stats monitoring
- Skimmer: Overwrite config files after updating, now it should create /opt/skimmer/skimmer.conf.rpmnew

v7.0.4

New Features

- New plugin: Archive specified indices
- Applications Access management based on roles
- Dashboards: Possibility to play a sound on the dashboard
- Tenable.SC: Integration with dedicated dashboard
- QualysGuard: Integration with dedicated dashboard
- Wazuh: added installation package
- Beats: added to installation package
- Central Agents Management (masteragent): Stop & start & restart for each registered agent
- Central Agents Management (masteragent): Status of detected beats and master agent in each registered agent
- Central Agents Management (masteragent): Tab with the list of agents can be grouped
- Central Agents Management (masteragent): Autorolling documents from .agents index based on a Settings in Config tab
- Alert: New Alert method for op5 Monitor added to GUI.
- Alert: New Alert method for Slack added to GUI.
- Alert: Name-change - the ability to rename an already created rule
- Alert: Groups for different alert types
- Alert: Possibility to modify all alarms in selected group
- Alert: Calendar - calendar for managing notifications
- Alert: Escalate - escalate alarm after specified time
- Alert: TheHive integration

Improvements

- Object Permission: When adding an object to a role in "object permission" now is possible to add related objects at the same time
- Skimmer: New metric - increase of documents in a specific index
- Skimmer: New metric - size of a specific index
- Skimmer: New metric - expected datanodes
- Skimmer: New metric - kafka offset in Kafka cluster
- Installation script: The setup script validates the license
- Installation script: Support for Centos 8
- AD integration: Domain selector on login page
- Incidents: New fieldsToSkipForVerify option for skipping false-positives
- Alert: Added sorting of labels in comboxes
- User Roles: Alphabetical, searchable list of roles
- User Roles: List of users assigned to a given role
- Audit: Cache for audit settings (performance)
- Diagnostic-tool.sh: Added cerebro to audit files
- Alert Chain/Logical: Few improvements

Bug Fixes

- Role caching fix for working in multiple node setup.
- Alert: Aggregation schedule time
- Alert: Loading new_term fields
- Alert: RecursionError: maximum recursion depth exceeded in comparison
- Alert: Match_body.logserver-gui_discover_url malfunction in aggregation
- Alert: Dashboard Recovery from Alert Status tab
- Reports: Black bars after JPEG dashboard export
- Reports: Problems with Scheduled reports
- Logserver-auth: Forbidden - not authorized when querying an alias with a wildcard
- Dashboards: Logserver_table is not present in 7.X, it has been replaced with basic table
- Logserver-Probe: Mikrotik pipeline - failed to start pipeline

v7.0.3

New Features

- Alert: new type - Chain - create alert from underlying rules triggered in defined order
- Alert: new type - Logical - create alert from underlying rules triggered with defined logic (OR,AND,NOR)
- Alert: correlate alerts for Chain and Logical types - alert is triggered only if each rule return same value (ip, username, process etc)
- Alert: each triggered alert is indexed with uniqe alert_id - field added to default field schema

- Alert: Processing Time visualization on Alert dashboard - easy to identify badly designed alerts
- Alert: support for automatic search link generation
- Input: added mikrotik parsing rules
- Auditing : added IP address field for each action
- Auditing : possibility to exclude values from auditing
- Skimmer: indexing rate visualization
- Skimmer: new metric: offset in Kafka topics
- SKimmer: new metric: expected-datanodes
- MasterAgent: added possibility for beats agents restart and the master agent itself (GUI)

Improvements

- Search and sort support for User List in Config section
- Copy/Sync: now supports "insecure" mode (operations without certificates)
- Fix for "add sample data & web sample dashboard" from Home Page -> changes in default-base-template
- Skimmer: service status check rewriteen to dbus api
- Masteragent: possibility to exclude older SSL protocols
- Masteragent: now supports Centos 8 and related distros
- XLSX import: updated to 7.6.1
- Logserver-Probe: masteragent pipeline shipped by default
- Blacklist: Name field and Field names in the Fields column & Default field exclusions
- Blacklist: runOnce is only killed on a fatal Alert failure
- Blacklist: IOC excludes threats marked as false-positive
- Incidents: new design for Preview
- Incidents: Note - new feature, ability to add notes to incidents
- Risks: possibility to add new custom value for risk, without the need to index that value
- Alert: much better performance with multithread support - now default
- Alert: Validation of email addresses in the Alerts plugin
- Alert: "Difference" rule description include examples for alert recovery function
- Logtrail: improved the beauty and readability of the plugin
- Security: jquery updated to 3.5.1
- Security: bootstrap updated to 4.5.0
- The HELP button (in logserver-gui) now leads to the official product documentation
- Centralization of previous alert code changes to single module

Bug Fixes

- Individual special characters caused problems in user passwords
- Bad permissions for scheduler of Copy/Sync module has been corrected
- Wrong Alert status in the alert status tab
- Skimmer: forcemerge caused under 0 values for cluster_stats_indices_docs_per_sec metric
- diagnostic-tool.sh: wrong name for the archive in output
- Reports: export to csv support STOP action
- Reports: scroll errors in csv exports
- Alert: .alertrules is not a required index for proper system operation
- Alert: /opt/alert/testrules is not a required directory for proper system operation
- Alert: .riskcategories is not a required index for proper system operation
- Malfunction in Session Timeout
- Missing directives service_principal_name in bundled properties.yml
- Blacklist: Removal of the *doc* type in blacklist template
- Blacklist: Problem with "generate_logserver-gui_discover_url: true" directive
- Alert: Overwriting an alert when trying to create a new alert with the same name
- Reports: When exporting dashboards, PDF generates only one page or cuts the page
- Wrong product logo when viewing dashboards in full screen mode

v7.0.2

New Features

- Manual incident - creating manual incidents from the Discover section
- New logserver-gui plugin - Sync/Copy between clusters
- Alert: Analyze historical data with defined alert
- Indicators of compromise (IoC) - providing blacklists based on Malware Information Sharing Platform (MISP)
- Automatic update of MaxMind GeoIP Databases [asn, city, country]
- Extended LDAP support
- Cross cluster search
- Diagnostic script to collect information about the environment, log files, configuration files - utils/diagnostic-tool.sh
- New beat: op5beat - dedicated data shipper from op5 Monitor

Improvements

- Added `_license` API for logserver (it replaces `license` path which is now deprecated and will stop working in future releases)
- `_license` API now shows **expiration_date** and **days_left**
- Visual indicator on **Config** tab for expiring license (for 30 days and less)
- Creating a new user now requires reentering the passoword

- Complexity check for password fields
- Incidents can be supplemented with notes
- Alert Spike: more detailed description of usage
- ElasticDump added to base installation - /usr/share/logserver-gui/elasticdump
- Alert plugin updated - frontend
- Reimplemented session timeout for user activity
- Skimmer: new metrics and dashboard for Cluster Monitoring
- Wazuh config/keys added to small_backup.sh script
- Logrotate definitions for Logtrail logfiles
- Incidents can be sorted by Risk value
- UTF-8 support for credentials
- Wazuh: wrong document_type and timestamp field

Bug Fixes

- Audit: Missing Audit entry for succesfull **SSO** login
- Report: "stderr maxBuffer length exceeded" - export to csv
- Report: "Too many scroll contexts" - export to csv
- Intelligence: incorrect work in updated environments
- Agents: fixed wrong document type
- Logserver-GUI: "Add Data to Logserver-GUI" from Home Page
- Incidents: the preview button uses the wrong index-pattern
- Audit: Missing information about login errors of ad/ldap users
- Netflow: fix for netflow v9
- MasterAgent: none/certificade verification mode should work as intended
- Incorrect CSS injections for dark theme
- The role could not be removed in specific scenarios

v7.0.1

- init
- migrated features from branch 6 [latest:6.1.8]
- XLSX import [logserver-gui]
- curator added to /usr/share/logserver-gui/curator
- node_modules updated! [logserver-gui]
- logserver upgraded to 7.3.2
- logserver-gui upgraded to 7.3.2
- dedicated icons for all logserver-gui modules
- eui as default framework for login,raports

- bugfix: alerts type description fix