Hybrid Mesh Network Security

07 September 2026

LOG EXPORTER

Administration Guide

CHECK POINT™

Check Point Copyright Notice

Important Information

Latest Software

We recommend that you install the most recent software release to stay up-to-date with the latest functional improvements, stability fixes, security enhancements and protection against new and evolving attacks.

Certifications

For third party independent certification of Check Point products, see the Check Point Certifications page.

Check Point Log Exporter Administration Guide

Latest Version of this Document in English

Open the latest version of this document in a Web browser.
Download the latest version of this document in PDF format.

Feedback

Check Point is engaged in a continuous effort to improve its documentation.
Please help us by sending your comments.

Revision History

Date	Description
17 May 2026	Added *"Deployment of Log Exporter using API" on page 11*
19 February 2026	Updated *"Known Limitations" on page 10*
12 January 2026	Updated *"Filter Configuration" on page 62*
28 September 2025	Updated *"Filter Configuration" on page 62*
04 August 2025	Updated *"Introduction" on page 8*
15 July 2025	Added *"Deployment of Log Exporter in SmartConsole" on page 13* Updated *"Appendix A - Deployment of Log Exporter in CLI" on page 30*
28 February 2025	First release of this HTML and PDF document
15 January 2018	First release of the SK article sk122323

Table of Contents

Introduction

The main SK article for Log Exporter is sk122323.

Overview

Check Point's **Log Exporter** is an easy and secure method for exporting Check Point logs over the syslog protocol.

You can exporting using a few standard protocols and formats.

Log Exporter supports:

- **SIEM applications:**

  Splunk, LogRhythm, Arcsight, RSA, QRadar, McAfee, rsyslog, ng-syslog, and any other SIEM application that can run a Syslog agent.

- **Protocols:**

  Syslog over TCP, Syslog over UDP.

- **Formats:**

  Syslog, Splunk, CEF, LEEF, Generic, JSON, LogRhythm, RSA.

- **Security:**

  Mutual authentication TLS 1.2.

- **Log Types:**

  The ability to export Security logs, Audit logs, or both.

  Note - Audit logs exist on both the Management Server and the Log Server.

- **Filtering:**

  Choose what to export based on field values. Filter out (do not export) Security Gateway connection logs.

- **Links to Logs and Log Attachments:**

  Export links to the relevant log card in SmartView and to the log attachments.

How It Works

Log Exporter is a multi-threaded daemon service, running on a Management Server / Log Server. The Log Exporter daemon reads each log, transforms it into the desired format and mapping, and sends it to the configured target. Therefore, we recommend to deploy the Log Exporter on every server that contains logs to be exported.

On a Multi-Domain Security Management Server / Multi-Domain Log Server, if the Log Exporter is deployed on several Domains, each Domain Management Server has its Log Exporter daemon. If you export logs to several targets, each target have its Log Exporter daemon.

The Log Exporter is implemented as the ETL procedure:

- **Extract** - Reads incoming logs from the Security Gateway, stored in local files.
- **Transform** - Changes the logs according to configuration files (both exported format and field name/values, removing irrelevant fields).
- **Load** - Sends the logs to the configured target server over the TCP Syslog / UDP Syslog (takes into consideration the filter configuration, if it exists).
- **Data integrity** - Log Exporter stops exporting when disconnected from the 3rd party server and remembers the last position exported. After the connection is established again, the Log Exporter automatically starts exporting logs from the last known position.

The Log Exporter exports both online and offline (if any) logs in parallel. If the 3rd-party server is slow, the Log Exporter reduces the offline exporting rate to prioritize the online logs over the offline logs.

Note - The Log Exporter daemon is part of a self-updatable package. See sk182866.

Known Limitations

- On a Multi-Domain Security Management Server / Multi-Domain Log Server, when exporting logs from a certain Domain using the UDP protocol, the exported log's header contains the IP address of the Multi-Domain Server and not the IP address of the Domain Management Server / Domain Log Server.
- Log Exporter is not supported for Locally Managed Spark Firewall appliances. Use the Spark Firewall appliances WebUI to configure a destination syslog server. For more information, see the Locally Managed Spark Appliances Administration Guide.
- In a Multi-Domain Server environment, Log Exporter configuration in SmartConsole is not supported on the Multi-Domain Server level (Multi-Domain Server and Multi-Domain Log Server) or the Global SmartEvent Server.

Deployment of Log Exporter using API

Starting from R82.10, you can configure Log Exporter using the Check Point Management API.

Best Practice - Use API-based configuration for all R82.10 and later environments.

Example API command for creating a Log Exporter:

```
mgmt_cli add log-exporter name "newLogExporter" target-server
"1.2.3.4" target-port 1234 protocol "tcp" attachments.add-link-to-
log-attachment true --format json
```

Example API command for creating a Log Exporter with all filtering options:

This example shows how to create a Log Exporter with multiple filtering rules. The exporter forwards only logs that match the specified criteria.

```
mgmt_cli add log-exporter name "allFilterModesExporter" target-
server "10.20.30.40" target-port 514 protocol "udp"
filters.expressions-operator "and" filters.filter-out-connection-
logs true filters.expressions.1.field-name "action"
filters.expressions.1.operator "in" filters.expressions.1.values.1
"Accept" filters.expressions.1.values.2 "Drop"
filters.expressions.1.values.3 "Reject"
filters.expressions.2.field-name "src"
filters.expressions.2.operator "not-in"
filters.expressions.2.values.1 "10.0.0.1"
filters.expressions.2.values.2 "10.0.0.2"
filters.expressions.3.field-name "severity"
filters.expressions.3.operator "and"
filters.expressions.3.conditions.1.operator "neq"
filters.expressions.3.conditions.1.value "Low"
filters.expressions.3.conditions.2.operator "neq"
filters.expressions.3.conditions.2.value "Informational"
filters.expressions.4.field-name "bytes"
filters.expressions.4.operator "or"
filters.expressions.4.conditions.1.operator "eq"
filters.expressions.4.conditions.1.value "0"
filters.expressions.4.conditions.2.operator "gt"
filters.expressions.4.conditions.2.value "1000"
filters.expressions.4.conditions.3.operator "lt"
filters.expressions.4.conditions.3.value "100" --format json
```

Explanation:

- `expressions-operator` - Combines multiple field-based filters by using logical operators, such as "`and`" or "`or`".
- `filter-out-connection-logs` - Removes connection logs.
- Include only logs where the action is Accept, Drop, or Reject:

```
filters.expressions.1.field-name "action"
filters.expressions.1.operator "in"
filters.expressions.1.values.1 "Accept"
filters.expressions.1.values.2 "Drop"
filters.expressions.1.values.3 "Reject"
```

- Exclude logs where the source IP is 10.0.0.1 or 10.0.0.2:

```
filters.expressions.2.field-name "src"
filters.expressions.2.operator "not-in"
filters.expressions.2.values.1 "10.0.0.1"
filters.expressions.2.values.2 "10.0.0.2"
```

- Include only logs in which severity is neither "Low" nor "Informational":

```
filters.expressions.3.field-name "severity"
filters.expressions.3.operator "and"
filters.expressions.3.conditions.1.operator "neq"
filters.expressions.3.conditions.1.value "Low"
filters.expressions.3.conditions.2.operator "neq"
filters.expressions.3.conditions.2.value "Informational"
```

- Include logs in which bytes are 0, greater than 1000, or less than 100:

```
filters.expressions.4.field-name "bytes"
filters.expressions.4.operator "or"
filters.expressions.4.conditions.1.operator "eq"
filters.expressions.4.conditions.1.value "0"
filters.expressions.4.conditions.2.operator "gt"
filters.expressions.4.conditions.2.value "1000"
filters.expressions.4.conditions.3.operator "lt"
filters.expressions.4.conditions.3.value "100"
```

For complete API syntax and parameter description, see the Management API Reference Guide.

Deployment of Log Exporter in SmartConsole

You can configure Log Exporter directly from SmartConsole and link it to the relevant Log Servers.

Note - For advanced deployment, see *"Advanced Deployment of Log Exporter in CLI" on page 31*.

Procedure:

1. Create a new Log Exporter/SIEM object:
    a. From the top, click **Objects** > **More object types** > **Server** > **Log Exporter/SIEM**.
    b. In the **Object Name** field, enter the applicable name for the new Log Exporter.
    c. From the left, click the **General** page:
        i. In the **Export Configuration** section, select **Enabled**.
        ii. In the **Server Configuration** section:
            - In the **Target Server** field, enter the IPv4 address or the FQDN of the destination server
            - In the **Target Port** field, enter the number of the listening port on the destination server
            - In the **Protocol** field, select the applicable protocol - **UDP** (default) or **TCP**

              Note - For environments with high log volumes, we recommend using TCP. Unlike UDP, TCP provides delivery acknowledgment. With UDP, if the sender transmits logs faster than the receiver can process them, logs can be dropped without notification.

d. From the left, click the **Data Manipulation** page:

   i. In the **Format** field, select the applicable format for the exported logs:

      - **Syslog** (default)
      - **Common Event Format (CEF)**
      - **Log Event Extended Format (LEEF)**
      - **Generic**
      - **Splunk**
      - **LogRhythm**
      - **Json**

   ii. By default, update logs contain the data that was changed compared to the last log for the same event.

      Select **Aggregate log updates before export** to export all logs with the full data, and not just the changes.

      Starting from [[[Undefined variable Vars_Versions.r_R8220_TBD]]], you can select the **Aggregation Mode** for connection logs:

      - **Semi unified** - Export all logs with the full data. Each log entry includes all the previous updates, not only the changes.
      - **First and Last** - Sends the first log update when the connection is created and the last log update when the connection is closed.
      - **Last only** - Sends only the last log update when the connection is closed.

e. From the left, click the **Attachments** page:

   Log Exporter does not include attachments by default.

   Optional: Select the applicable options to configure the log attachments:

   - **Add link to Log Details in SmartView**
   - **Add link to Log Attachment in SmartView**
   - **Add Log Attachment ID**

f. Click **OK**.

2. Configure the Management Server or Dedicated Log Server / SmartEvent Server object:
    a. From the left navigation panel, click **Gateways & Servers**.
    b. Open the Management Server or Dedicated Log Server / SmartEvent Server object.
    c. From the left tree, click **Logs** > **Export**.
    d. Click [**+**] and select the **Log Exporter / SIEM** object you configured earlier.
    e. Click **OK**.
3. Install the database:
    a. From the top, click **Menu** > **Install database**.
    b. Select all objects.
    c. Click **Install**.

    Important in a Multi-Domain Server environment - If you configured Log Exporter object(s) in the Global Domain and assigned Global Policy, you must install the database in SmartConsole connected to the applicable Domain Management Server.

After you upgrade a Management Server / Log Server / SmartEvent Server to a new version, you must:

1. In SmartConsole, from the top, click **Menu** > **Install database**.
2. Select all objects.
3. Click **Install**.

Note - During an upgrade to R81.20 and higher, the Log Exporter configuration is part of the upgrade.

Log Exporter Support for Amazon S3 Buckets

You can configure Log Exporter to send logs directly to Amazon S3 buckets. This allows you to store logs in AWS and integrate them with your existing monitoring, analytics, and archival workflows.

To securely access an Amazon S3 bucket, Log Exporter uses AWS IAM Roles Anywhere. IAM Roles Anywhere enables systems that run outside AWS, such as Check Point devices, to obtain temporary AWS credentials without storing long-term access keys on the device.

An IAM (Identity and Access Management) role is an AWS identity that defines a set of permissions. Unlike IAM users, which use long-term credentials, IAM roles provide temporary credentials that trusted entities can assume when they need access to AWS resources.

Before you configure Log Exporter to send logs to an Amazon S3 bucket, you must configure IAM Roles Anywhere in AWS and create the required trust relationships and permissions.

Prerequisites

- An active AWS account.
- An Amazon S3 bucket. See Creating a general purpose bucket - Amazon Simple Storage Service.

Configuration

1. Generate a certificate for the S3 Log Exporter on the applicable Check Point device that exports the logs.
    a. Make sure you have these items:
        - Bucket name
        - Region
        - Prefix

b. On the device which exports the logs, run this POST API request:

```
postman request POST 'https://{{machine_ip}}/web_api/add-
log-exporter' \
--header 'Content-Type: application/json' \
--header 'X-chkp-sid: {{sid}}' \
--body '{
"name": "{{exporter_name}}",
"target-cloud": {
"type": "s3-bucket",
"s3-bucket-auth": {
"operation": "create-certificate",
"create-certificate": {
"bucket": "{{bucket}}",
"region": "{{region}}",
"prefix": "{{prefix}}"
}
}
}
}'
```

Expected result:

The API returns a CA certificate.

c. Decode the CA certificate:

- If you are using Postman, use this script to decode the CA:

```
if (pm.response.code === 200) {
const json = pm.response.json();
const encodedCa = json["target-cloud"]["s3-bucket-auth"]
["create-certificate"]["ca-certificate"];
if (encodedCa) {
const decodedCa = atob(encodedCa);
pm.environment.set("decoded_ca", decodedCa);
console.log("Decoded CA:");
console.log(decodedCa);
} else {
console.error("ca-certificate not found in response:",
JSON.stringify(json));
}
} else {
console.error("Request failed with status:",
pm.response.code, pm.response.text());
}
```

- If you are using Linux, use this command to decode the CA:

```
echo "$response" | jq -r '.["target-cloud"]["s3-bucket-
auth"]["create-certificate"]["ca-certificate"]' | base64
-d
```

Expected results:

- The CA certificate and client certificates are generated successfully.
- Exporter is disabled until ARNs are provided.

d. Copy the CA Certificate in this format (including BEGIN CERTIFICATE and END CERTIFICATE).

```
-----BEGIN CERTIFICATE-----
...   …
-----END CERTIFICATE-----
```

You will use this certificate as a trust anchor in AWS. This action instructs AWS to trust any certificates that are signed by this Certificate Authority (CA).

2. Create an IAM role in the AWS Management Console:

a. Navigate to `https://console.aws.amazon.com/iam/`.

b. Sign in with your credentials.

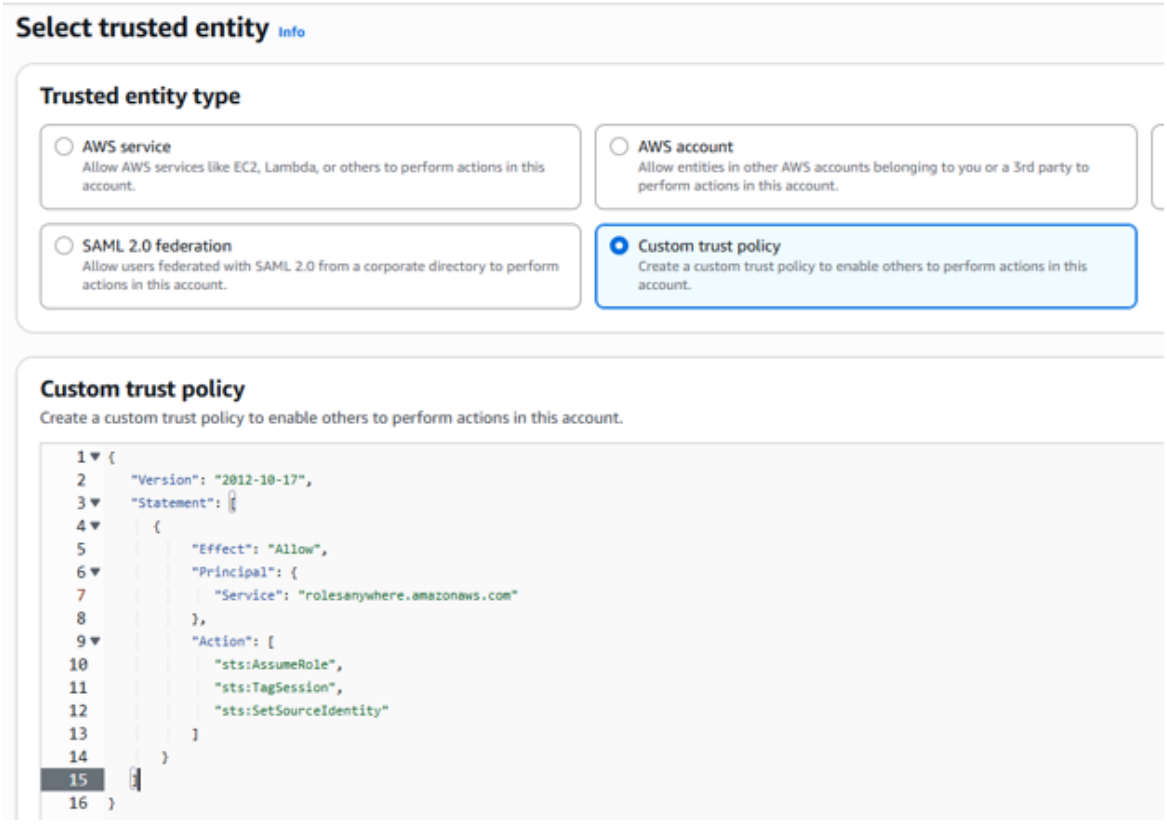
c. In the AWS Management Console, navigate to **Access Management** > **Roles**.

d. Click **Create role**.

e. In the **Select trusted entity** window, select **Custom trust policy**.

Paste this JSON:

```
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Principal": {
"Service": "rolesanywhere.amazonaws.com"
},
"Action": [
"sts:AssumeRole",
"sts:TagSession",
"sts:SetSourceIdentity"
]
}
]
}
```



f. Click **Next**.

The **Add Permissions** page opens.

g. From the list, select **AmazonS3FullAccess**.

h. Click **Next**.

i. Enter a name for the role.

Optional: Enter a description for the role.

The role is created.

j. From the top of the page, copy the role's ARN.

k. Save this ARN in a text file.

3. Create a trust anchor for IAM Roles Anywhere in the AWS Management Console:

a. In the top-right corner of AWS Management Console, make sure that you are using the correct **AWS Region**.

b. In the left navigation pane, go to **Roles** > **Roles Anywhere** > **Trust anchors**.

The **Create a trust anchor** page opens.

c. Enter a name for the trust anchor.

d. For **Certificate authority (CA) source**, select **External certificate bundle**.

e. In the **External certificate bundle** field, paste the CA certificate that you created earlier.

f. Click **Create a trust anchor**.

g. After the trust anchor is created, copy the trust anchor ARN from the **Trust anchor details** page and save it for use in a later step.

4. Create a profile in the AWS Management Console. A profile links the trust anchor to the IAM role. The trust anchor defines the trusted certificate authorities, and the IAM role defines the permissions granted to the workload:

a. Navigate to **Roles** > **Roles Anywhere** > **Profiles**.

b. Click **Create a profile**.

c. Enter a name for the profile.

d. In **Roles**, click **Add another role**, and then select the role you created in step 2.

e. In **Session duration**, enter **3600** seconds (1 hour).

f. Leave **Session policies** and **Session tags** empty.

g. Select **Create a profile**

h. After the profile is created, copy the Profile ARN and save it for later use.

Note - At this stage, you should have 3 ARNs:
- IAM Role Anywhere ARN
- Trust Anchor ARN
- Profile ARN

5. On the Check Point device, complete the S3 Log Exporter configuration by providing the required ARNs:

   a. Run this POST API request:

   ```
   postman request POST 'https://{{machine_ip}}/web_api/set-
   log-exporter' \
   --header 'Content-Type: application/json' \
   --header 'X-chkp-sid: {{sid}}' \
   --body '{
   "name": "{{exporter_name}}",
   "target-cloud": {
   "type": "s3-bucket",
   "s3-bucket-auth": {
   "operation": "complete-configuration",
   "complete-configuration": {
   "role-arn": "{{role_arn}}",
   "profile-arn": "{{profile_arn}}",
   "trust-anchor-arn": "{{trust_anchor_arn}}"
   }
   }
   }
   }'
   ```

   b. Verify that the response contains this message:

   ARNs configuration completed successfully. Log Exporter is now enabled.

6. In SmartConsole, connect the Log Exporter object to the applicable Check Point device that runs the Log Exporter.

   a. In the **Gateways & Servers** view, open the server object that stores and processes the logs.

   b. From the left navigation pane, select **Logs** > **Export**.

   c. Click the plus (**+**) sign.

   d. Select the applicable Log Exporter object.

   e. Click **OK**.

7. In the top-right corner of SmartConsole, click the menu button, and then click **Install Database**.

Log Exporter can now send logs to Amazon S3 buckets.

Integrating OCSF Logs with Amazon Security Lake

Log Exporter can export security logs in OCSF (Open Cybersecurity Schema Framework) format. OCSF is an open, vendor-neutral standard for security telemetry, developed by the Linux Foundation and supported by AWS, Splunk, IBM, and more than 145 other vendors.

Unlike traditional flat log formats such as Syslog, CEF, LEEF, and JSON, OCSF uses a structured, hierarchical JSON schema that follows the official OCSF specification. This standardized format allows OCSF-compatible platforms to ingest and process Check Point logs without requiring custom parsers or field mappings.

Log Exporter can send logs directly to Amazon S3 buckets. You can use this capability to export OCSF-formatted logs to S3 for integration with third-party platforms and services. This section describes the integration with Amazon Security Lake, which enables centralized storage, analysis, and querying of security data through Amazon Athena.

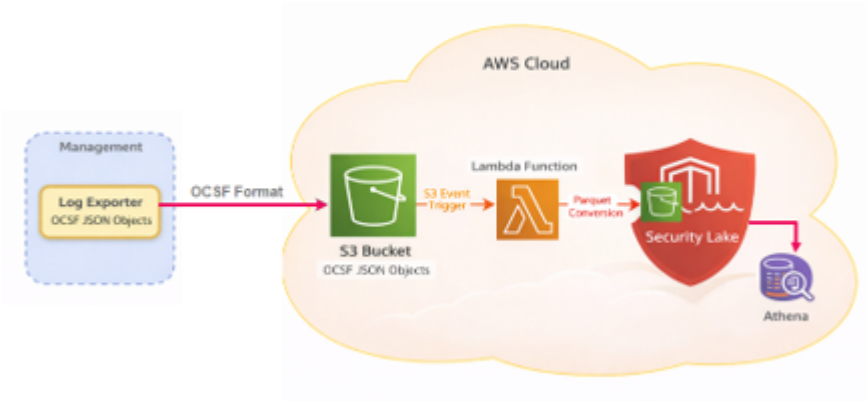
Supported Versions

- Management Server, Log Server, or dedicated SmartEvent Server running R82.20 or higher
- Earlier supported versions with a Log Exporter AutoUpdate package that includes OCSF format support.

Prerequisites

- For Security Lake integration: Amazon Security Lake enabled in the target AWS region.
- An AWS account with permissions to manage S3, Lambda, Amazon Security Lake, and Amazon Athena.

Security Lake Integration Workflow



1. Log Exporter writes security logs in OCSF NDJSON format to a designated Amazon S3 bucket.
2. When Log Exporter uploads a new OCSF log file to the bucket, Amazon S3 automatically triggers an AWS Lambda function.
3. The Lambda function converts the OCSF NDJSON data into Apache Parquet, which is the format required by Amazon Security Lake.
4. After the conversion completes, the Lambda function uploads the Parquet files to the Security Lake staging bucket.
5. Amazon Security Lake automatically detects and ingests the Parquet files from the staging bucket.
6. After ingestion, the data becomes available in Amazon Athena, where you can use SQL queries to search, analyze, and report on security events.

Notes:

- Log Exporter exports logs in standard OCSF NDJSON format. Although this section describes integration with Amazon Security Lake, OCSF logs exported to Amazon S3 can also be consumed by other OCSF-compatible platforms.
- The provided AWS CloudFormation template and Lambda-based conversion (Step 3) are supplied as-is. Configuration of the AWS services, including Amazon Security Lake, AWS Lake Formation, AWS Lambda, and Amazon Athena, are governed by the AWS documentation and are outside the scope of Check Point Support.

Configuration

Step 1: Configure Log Exporter to Export OCSF Logs to an Amazon S3 Bucket

Important - Use a dedicated Amazon S3 bucket for the Log Exporter output. Do not use the Security Lake staging bucket created in Step 2. Amazon Security Lake manages the staging bucket and expects only processed Parquet files. Do not store raw OCSF logs in this bucket or configure additional triggers for it.

1. Configure Log Exporter support for Amazon S3 buckets. See *"Log Exporter Support for Amazon S3 Buckets" on page 16*.
2. Verify the Configuration:
   a. Log in to the Management Server. Send a login request to obtain a session ID (sid):

```
POST 'https://{{device_ip}}:443/web_api/v2.2/login' \
--body '{
"user" : {{username}},
"password " : {{password }},
"session-timeout " : 3600
}'
```

   b. Save the "`sid`" value from the response.
   c. Run these commands:
      - In Web API:

```
POST: 'https://{{machine_ip}}:443/web_api/v2.2/show-log-exporter' \
--header 'Content-Type: application/json' \
--header 'X-chkp-sid: {{sid}}' \
--body '{ "name": "{{exporter_name}}" }'
```

      - In CLI:

```
‘mgmt_cli show log-exporter name "{{exporter_name}}" -r true’
```

   d. Verify that the output shows: `"format": "ocsf"`

Step 2: Create a Custom Source in Amazon Security Lake

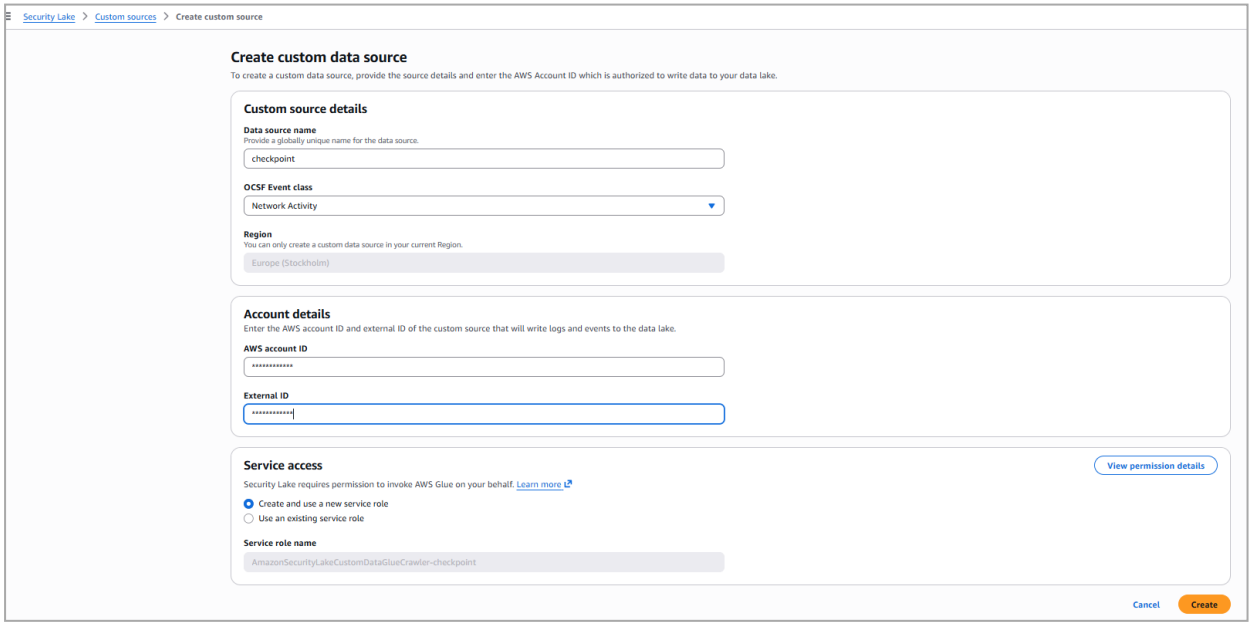
Important - Create the custom source from the Security Lake delegated administrator account.

To ingest third-party data into Amazon Security Lake, create a custom source. When you create a custom source, Security Lake automatically:

- Creates a dedicated staging location in the Security Lake S3 bucket: `s3://<security-lake-bucket>/ext/<source-name>/`
- Creates an AWS Glue crawler and table to catalog the ingested Parquet files.
- Creates an IAM role that allows the data provider to write data to the staging location.

To create the custom source:

1. Sign in to the AWS Management Console.
2. Open the Amazon Security Lake console in the target AWS region.
3. In the navigation pane, select **Custom sources**.
4. Select **Create custom source**.
5. Configure these parameters:
   - **Data source name** - The name of the custom source. For example: **checkpoint**.
   - **OCSF Event class** - Event class that the imported logs map to. Select **Network Activity**.
   - **AWS Account ID** - AWS account permitted to write data to the staging location. Enter the account that hosts the source bucket and the AWS Lambda conversion function (typically the current account).
   - **External ID** - Unique free-text identifier (for example, **checkpoint-ocsf**). Security Lake adds this value to the trust policy of the IAM role that it creates for the data provider.
   - **Service role name** - IAM role that AWS Glue uses to crawl converted Parquet files and maintain table partitions.



6. Select **Create**.
7. After the source is created, locate it in the **Custom sources** list and record its staging bucket path:

   `s3://<security-lake-bucket>/ext/<source-name>/`

   You will use this location when configuring the log conversion and ingestion workflow.

Step 3: Deploy the Converter with CloudFormation

Check Point provides an AWS CloudFormation template: `checkpoint-ocsf-template.yaml`, to deploy the components required to convert OCSF NDJSON log files to Apache Parquet format for Amazon Security Lake ingestion.

The template creates:

- An AWS Lambda function that converts OCSF NDJSON files to Apache Parquet.
- An IAM role for the AWS Lambda function.
- An Amazon S3 event notification that triggers the AWS Lambda function when new log files are uploaded to the source bucket.

When a new OCSF NDJSON file is created in the source bucket, the AWS Lambda function automatically:

- Reads the log file.
- Converts the data to Apache Parquet format using the pyarrow library.
- Writes the converted file to the Amazon Security Lake staging location using time-based partitions.

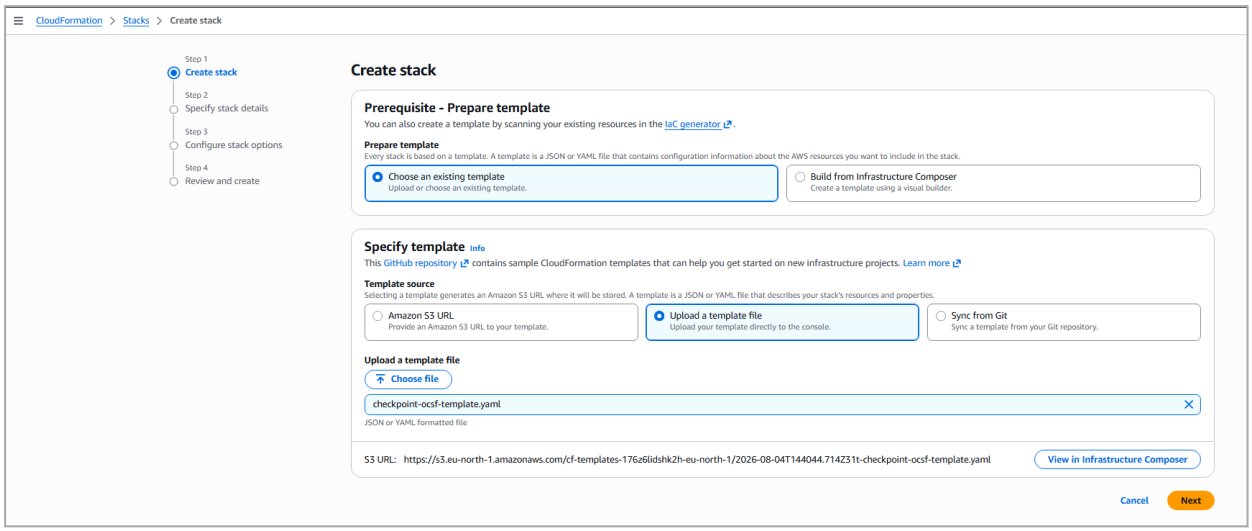
To deploy the CloudFormation template:

1. Sign in to the AWS Management Console.
2. Open the **AWS CloudFormation** console.
3. On the **Stacks** page, select **Create stack**.
4. In **Prepare template**, select **Choose an existing template**.
5. In **Specify template**:

a. Select **Upload a template file**.

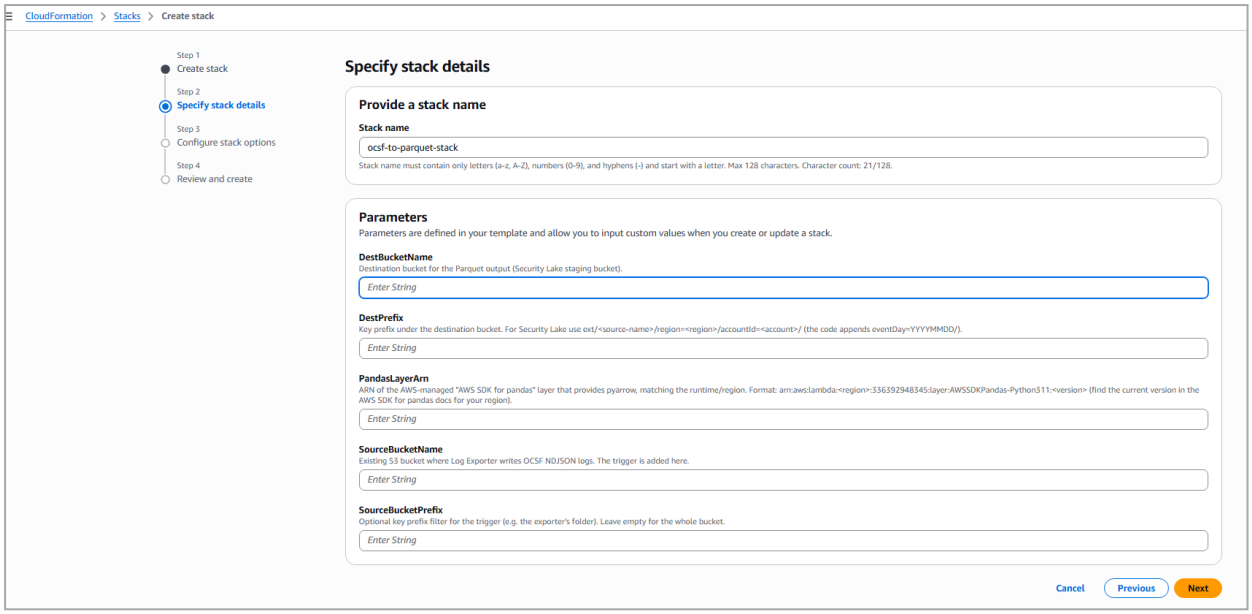
b. Upload the `checkpoint-ocsf-template.yaml` template file.

c. Click **Next**.



6. Enter a **Stack name** (for example: **ocsf-to-parquet**).
7. Configure the template parameters:
   - **DestBucketName** - The Security Lake staging bucket created when you configured the custom source.
   - **DestPrefix** - Destination prefix in the Security Lake staging bucket.

     Use the format:
     `ext/<source-name>/region=<region>/accountId=<account>/`
     The Lambda function automatically appends eventDay=YYYYMMDD/ when writing Parquet files.
   - **PandasLayerArn** - ARN of the AWS-managed AWS SDK for pandas Lambda layer that matches your AWS region and the Python 3.11 runtime. This layer provides the pyarrow library required for Parquet conversion. Example:
     `arn:aws:lambda:eu-north-1:336392948345:layer:AWSSDKPandas-Python311:33`
   - **SourceBucketName** - Name of the existing Amazon S3 bucket that receives OCSF logs from Log Exporter. The template creates an Amazon S3 event notification on this bucket.
   - **SourceBucketPrefix** - Optional prefix filter for exported log files. Leave blank to process all objects in the source bucket.
8. Click **Next**.



9. On the **Configure stack options** page, accept the default settings and select **Next**.
10. On the **Review** page, acknowledge that the stack creates IAM resources.
11. Click **Submit**.
12. Wait until the stack status changes to **CREATE_COMPLETE**.

Result:

Each new OCSF NDJSON file uploaded to the source bucket is automatically converted to Apache Parquet and written to the Security Lake staging location in this format:

```
ext/<source-
name>/region=<region>/accountId=<account>/eventDay=YYYYMMDD/
```

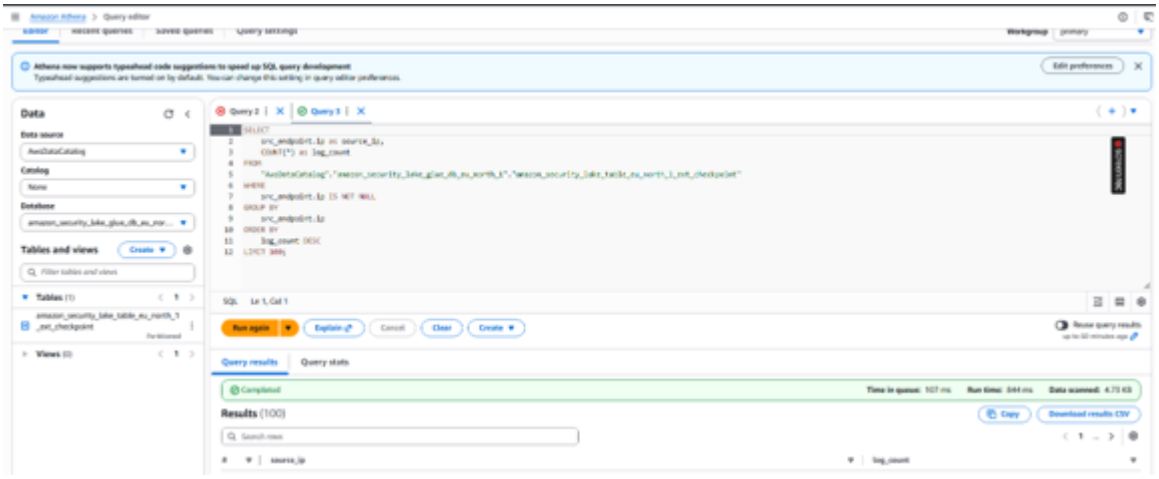
Note - Deleting the CloudFormation stack removes the AWS Lambda function, its associated resources, and the Amazon S3 event notification created by the template.

Important:

- The CloudFormation template overwrites any existing Amazon S3 event notification configuration on the source bucket. If the source bucket already has event notifications that must be preserved, configure the Lambda trigger manually from **Amazon S3** > **Buckets** > **<bucket name>** >**Properties** > **Event notifications** instead of using the template.
- Amazon S3 event notifications are generated only for newly created objects. Existing log files already stored in the source bucket are not processed automatically. To process existing files, copy or re-upload them so that Amazon S3 generates new object-created events.

Step 4: Query Logs with Amazon Athena

After you configure log streaming, Amazon Security Lake automatically ingests the converted Parquet files. The ingested data is typically available in Amazon Athena within a few minutes, where you can use standard SQL queries to search, analyze, and report on security events.



Appendix A - Deployment of Log Exporter in CLI

Basic Deployment of Log Exporter in CLI

Note - For basic deployment in SmartConsole, see *"Deployment of Log Exporter in SmartConsole" on page 13*.

Best Practice - For R82.10 environments and higher, configure the Log Exporter using the Check Point Management API. See *"Deployment of Log Exporter using API" on page 11*

Syntax

```
cp_log_export add name <Name> [domain-server {mds | all}] target-server <HostName or IP address of Target Server> target-port <Port on Target Server> protocol {udp | tcp} format {syslog | splunk | cef | leef | generic | json | logrhythm | rsa} [<Optional Arguments>]
```

Important:

- The `domain-server` argument is mandatory on a Multi-Domain Security Management Server / Multi-Domain Log Server.
  - `mds` (in small letters) - Exports logs only from the MDS level.
  - `all` (in small letters) - Exports logs from all Domains.
- The `target-server` argument can use either the target server IP address or its FQDN.
- The above command creates a new target directory with the unique name specified in the `name` parameter in the `$EXPORTERDIR/targets/` directory, and configures the target parameters with the connection details: IP Address, port, protocol, format, and read-mode.
- By default, logs are exported in clear text. To export logs using an encryption, see *"Advanced Deployment of Log Exporter in CLI" on the next page*.
- The Log Exporter daemon does **not** start automatically.
  To start it, run:

  ```
  cp_log_export restart
  ```

Advanced Deployment of Log Exporter in CLI

CLI Syntax

```
cp_log_export <CLI Command> [<CLI Parameters>]
```

To see the built-in help, run:

```
cp_log_export <CLI Command> help
```

CLI Commands

Important - After you change the configuration of a Log Exporter instance, you must restart it:

```
cp_log_export restart name <Name>
```

Name	Description
`add`	Deploys a new Log Exporter instance. **Syntax:** `cp_log_export add name <Name> [domain-server {mds | all}] target-server <HostName or IP address of Target Server> target-port <Port on Target Server> protocol {udp | tcp} format {syslog | splunk | cef | leef | generic | json | logrhythm | rsa} [<Optional Arguments>]`
`set`	Updates an existing Log Exporter instance configuration. **Syntax:** `cp_log_export set name <Name> [<Optional Arguments>]`
`delete`	Removes an existing Log Exporter instance. **Syntax:** `cp_log_export delete name <Name>`
`show`	Prints the current configurations of the existing Log Exporter instances. **Syntax:** `cp_log_export show [<Optional Arguments>]`

Name	Description
`status`	Prints the overview statuses of the existing Log Exporter instances. **Syntax:** `cp_log_export status [<Optional Arguments>]`
`start`	Starts the Log Exporter instance. **Syntax:** `cp_log_export start name <Name>`
`stop`	Stops the Log Exporter instance. Syntax: `cp_log_export stop name <Name>`
`reconf`	Applies the Log Exporter configuration to all existing Log Exporter instances, or to a specified Log Exporter instance. **Syntax:** `cp_log_export reconf [name <Name>]`
`restart`	Restarts the Log Exporter instance. You must rung this command after you change the configuration of a Log Exporter instance. **Syntax:** `cp_log_export restart name <Name>`

Name	Description
`reexport`	Resets the current read position and re-exports all logs per the Log Exporter instance configuration. **Syntax to reexpot immediately:** `cp_log_export reexport name <Name> --apply-now` **Syntax to reexpot from a last exported position:** `cp_log_export reexport name <Name> start-position <Position of Last Exported Log> --apply-now` **Syntax to reexpot from a specific position and until a specific position:** `cp_log_export reexport name <Name> start-position <Position of Gap Start> end-position <Position of Gap End> --apply-now` For example: ■ On a Security Management Server. `cp_log_export reexport name <LogExporter Name> start-position 1715222525 end-position 1715243317 --apply-now` ■ On a Multi-Domain Security Management Server: `cp_log_export reexport name <LogExporter Name> domain-server <MyDomain> start-position 1715222525 end-position 1715243317 --apply-now`

CLI Parameters

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`name <Name>`	Specifies a unique name for the Log Exporter configuration. ■ Allowed characters are: Latin letters, digits ("0-9"), minus ("-"), underscore ("_"), and period ("."). ■ Must start with a letter. ■ The minimum length is two characters.	**Mandatory**	**Mandatory**	**Mandatory**	Optional Default is "`all`"	Optional Default is "`all`"	**Mandatory**
`domain-server {mds | all}`	■ On a Multi-Domain Server, specifies the applicable Domain Management Server context. ■ On a Multi-Domain Log Server, specifies the applicable Domain Log Server context.	**Mandatory**	**Mandatory**	**Mandatory**	N/A	Optional Default is "`all`"	**Mandatory**

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`target-server <Target-Server>`	Specifies the IP address or FQDN of the target server, to which you export the logs.	**Mandatory**	Optional	N / A	N / A	N / A	N / A
`target-port <Target-Server-Port>`	Specifies the listening port on the target server, to which you export the logs.	**Mandatory**	Optional	N / A	N / A	N / A	N / A
`protocol {tcp | udp}`	Specifies the transport protocol to use (TCP or UDP).	**Mandatory**	Optional	N / A	N / A	N / A	N / A
`format {generic | cef | json | leef | logrhythm | rsa | splunk | syslog}`	Specifies the format, in which the logs are exported. Default: `syslog`	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`read-mode {raw | semi-unified}`	Specifies the reading mode of the log files. ■ `semi-unified` - The default mode. In this mode, Log Exporter sends every log update with all the previous updates. ■ `last-only` - This mode is available for connection logs in [[[Undefined variable Vars_Versions.r_R8220_TBD]]] versions and higher. In this mode, Log Exporter sends only the last log update when the connection is closed.	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
	■ `first-and-last` - This mode is available for connection logs in [[[Undefined variable Vars_Versions.r_R8220_TBD]]] versions and higher. In this mode, Log Exporter sends the first log update when the connection is created and the last log update when the connection is closed.						
`enabled {true | false}`	Specifies whether to allow the Log Exporter to start when you run the `cpstart` or `mdsstart` command. Default: `true`	Optional	Optional	N / A	N / A	N / A	N / A
`encrypted {true | false}`	Specifies whether to use TLS (SSL) encryption to send the logs. Default: `false`	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`ca-cert <Path>`	Specifies the full path to the CA certificate file **.pem*. Applicable only when the value of '`encrypted`' is '`true`'.	Optional	Optional	N / A	N / A	N / A	N / A
`client-cert <Path>`	Specifies the full path to the client certificate **.p12*. Applicable only when the value of '`encrypted`' is '`true`'.	Optional	Optional	N / A	N / A	N / A	N / A
`client-secret <Phrase>`	Specifies the challenge phrase used to create the client certificate *.p12. Applicable only when the value of '`encrypted`' is '`true`'.	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`filter-action-in {"Action1","Action2",... | false}`	Specifies whether to export all logs that contain a specific value in the "**Action**" field. ■ Each value must be surrounded by double quotes (""). ■ Multiple values are supported and must be separated by a comma without spaces. To see all valid values: 1. In SmartConsole, go to the **Logs & Monitor** view and open the **Logs** tab. 2. In the top query field, enter **action:** and a letter.	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`filter-origin-in {"Origin1","Origin2",... | false}`	Specifies whether to export all logs that contain a specific value in the "**Origin**" field (the object name of the Security Gateway / Cluster Member that generated these logs). ■ Each origin value must be surrounded by double quotes (""). ■ Multiple values are supported and must be separated by a comma without spaces.	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`filter-blade-in {"Blade1","Blade2",... | false}`	Specifies whether to export all logs that contain a specific value in the "**Blade**" field (the object name of the Software Blade that generated these logs). ■ Each value must be surrounded by double quotes (""). ■ Multiple values are supported and must be separated by a comma without spaces. To see all valid values: 1. In SmartConsole, go to the **Logs & Monitor** view and open the **Logs** tab. 2. In the top query field, enter **blade:** and a letter. Valid Software Blade families: ■ **Access** ■ **TP** ■ **Endpoint** ■ **Mobile**	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`--apply-now`	Applies immediately any change that was done with the "`add`", "`set`", "`delete`", or "`reexport`" command.	Optional	Optional	**Mandatory**	N / A	N / A	**Mandatory**
`export-link {true | false}`	Specifies whether to add a field to the exported logs that represents a link to SmartView that shows the log card. Default: `false`	Optional	Optional	N / A	N / A	N / A	N / A
`export-attachment-link {true | false}`	Specifies whether to add a field to the exported logs that represents a link to SmartView that shows the log card and automatically opens the attachment. Default: `false`	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`export-link-ip {true | false}`	Specifies whether to make the links to SmartView use a custom IP address (for example, for a Log Server behind NAT). Applicable only when the value of the "`export-link`" argument is "`true`", or the value of the "`export-attachment-link`" argument is "`true`". Default: `false`	Optional	Optional	N / A	N / A	N / A	N / A
`export-attachment-ids {true | false}`	Specifies whether to add a field to the exported logs that represents the ID of log's attachment (if exists). Default: `false` Supported on Management Servers / Log Servers R81 and higher.	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`reconnect-interval {<Number> | default}`	Specifies the interval (in minutes) after which the Log Exporter must connect again to the target server after the connection is lost. ■ To disable, enter the value "`default`". ■ There is no default value. Supported on Management Servers / Log Servers R81.10 and higher.	Optional	Optional	N / A	N / A	N / A	N / A
`export-log-position {true | false}`	Specifies whether to export the log's position. Default: `false` Supported on Management Servers / Log Servers R81.10 and higher.	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
`time-in-milli {true | false}`	Specifies whether to export logs with the time resolution in milliseconds. Default: `false` Management Servers / Log Servers / Security Gateways support this feature in these versions: ■ R81 and higher ■ R80.40 Jumbo Hotfix Accumulator, Take 53 and higher Prerequisite on the Security Gateways: 1. Connect to the command line on the Security Gateway / each Cluster Member / Scalable Platform Security Group. 2. Log in. 3. If you default shell is Gaia Clish / Gaia gClish, then go to the Expert mode:	Optional	Optional	N / A	N / A	N / A	N / A

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
	``expert``` 4. To enable this feature, run this script with the value "1": Note - This command restarts the FWD process (for details about this process, see sk97638). ■ On the the Security Gateway / each Cluster Member: ```$FWDIR/scripts/enable_disable_time_in_milli.sh 1```						

Parameter Name	Description	add	set	delete	reconf	show, status, start, stop, restart	reexport
	▪ On the Scalable Platform Security Group: `g_all $FWDIR/scripts/enable_disable_time_in_milli.sh 1` To disable the feature, run the script with the value "0".						

Important - Using the "`filter-action-in`", "`filter-origin-in`", or "`filter-blade-in`" replaces any other filter configuration that was declared earlier on these fields directly in the filtering XML. Other field filters are not overridden.

Advanced Configuration

After deploying a new instance of Log Exporter, all related files to that deployment are located in this directory:

```
$EXPORTERDIR/targets/<Name of Log Exporter Configuration>
```

Notes:

- You must restart the Log Exporter instance for the new settings to take effect. Run the "`cp_log_export restart`" command.
- On a Multi-Domain Security Management Server / Multi-Domain Log Server, the environment variable "`$EXPORTERDIR`" exists in each Domain, and its value is changed automatically when you switch between context of Domains with the "`mdsenv`" command.

Target Configuration File

The Log Exporter configuration for the target server is saved in this file:

```
$EXPORTERDIR/targets/<Name of Log Exporter
Configuration>/targetConfiguration.xml
```

These are some configuration options:

Parameter	Description	Valid / Default Values
`<version></version>`	The current Log Exporter version - used for upgrades.	
`<is_enabled></is_enabled>`	Determines whether the process is monitored by the watch dog.	■ `true` ■ `false`

Destination Parameters

Parameter	Description	Valid / Default Values
`type`	Reserved for future use	
`<ip></ip>`	The IP address or FQDN of the target server	Any IPv4 address or FQDN
`<port></port>`	The port on which the target server is listening	Any valid port number
`<protocol></protocol>`	The Layer 4 protocol to use	TCP or UDP
`<reconnect_interval></reconnect_interval>`	Determines how frequently to start the connection to the target server after it is lost	Number of minutes

Security Parameters

For more information, see *"TLS Configuration" on page 71*.

Parameter	Description	Valid / Default Values
`<security></security>`	Determines whether the connection data is sent in clear text or encrypted.	▪ `clear` (clear text - this is the default) ▪ `tls` (encrypted)
`<pem_ca_file></pem_ca_file>`	The location of the root Certificate Authority PEM file.	
`<p12_certificate_file></p12_certificate_file>`	The location of the client key pair in the P12 format.	
`<client_certificate_challenge_phrase></client_certificate_challenge_phrase>`	The challenge phrase that was used to create the P12 certificate. The value is hashed after restarting the process.	

Source Parameters

Parameter	Description	Valid / Default Values
`<folder></folder>`	The path where the log files are located	The default location is `$FWDIR/log/`
`<log_files></log_files>`	Determines which log records to export or how far back to read the log records from the `$FWDIR/log/fw.log` file	■ *<Number>* Reads logs from the specific number (default=1) of days back (recommended) ■ *<Specific File Name>* Reads logs from the specified file ■ `on-line` ■ If no value is specified, uses '`on-line`'
`<log_types></log_types>`	Determines which logs to export based on their type	■ `all` (default) ■ `log` ■ `audit`
`<read_mode></read_mode>`	Determines whether to export complete logs or only their delta.	■ `semi-unified` (default in R81 and higher) ■ `raw`

Resolver Parameters

Parameter	Description	Valid / Default Values
`<mappingConfiguration></mappingConfiguration>`	Configures the XML file that contains the log field mapping scheme. If left empty, uses the default settings.	Default values are based on the 'format'.
`<exportAllFields>true</exportAllFields>`	When this field is set to '`true`', all log **fields** are sent regardless of whether they appear in the mapping scheme, except for specifically black-listed fields in the relevant log format mapping file (`<exported>false</exported>`). When this field is set to '`false`', only those fields which appear in the relevant log format mapping file are sent (with exported flag set to '`true`': `<exported>`**`true`**`</exported>`)	■ `true` ■ `false`

Format Parameters

Parameter	Description	Valid / Default Values
`<formatHeaderFile></formatHeaderFile>`	Configures the XML file that contains the log header format scheme. If left empty, uses the default settings.	Default values are based on the 'format'.

General Filter Configuration Path

Parameter	Description	Valid / Default Values
`<dynamicFilter></dynamicFilter>`	Configures the XML file that contains the filtering configuration. If left empty, uses the default settings.	The default path is: `conf/FilterConfiguration.xml`

SmartView Link Parameters

Parameter	Description	Valid / Default Values
`export_log_link`	Adds a field to the exported log that represents a link to SmartView that shows the log card.	■ `true` ■ `false` (default)
`export_attachment_link`	Adds a field to the exported log that represents a link to SmartView that shows the log card and automatically opens the attachment.	■ `true` ■ `false` (default)
`export_link_ip`	Makes the '`export_log_link`' and the '`export_attachment_link`' use a customized IP address (for example, for a Log Server behind NAT).	■ empty (default) ■ IPv4 Address

Parameters to filter out the Security Gateway connections

This configuration allows Log Exporter instance to filter out the Security Gateway traffic logs for several Software Blades ('Firewall-1 & VPN-1', 'HTTPS Inspection' and 'Security Gateway/Management').

Note - Security Gateway session logs are still exported (generated by tracking a Security Gateway rule per session).

Parameter	Description	Valid / Default Values
`<filter filter_out_by_ connection="false">`	Determines whether to filter out the access logs. **Note** - No other Software Blade filters are currently supported. This is planned in future releases.	▪ `true` filters out the connection logs ▪ `false`

Important - HTTPS Inspection logs, Security Gateway logs generated not from rules, and a few NAT update logs are still exported.

Configuration After an Upgrade

If you customized your configuration files in the Log Exporter instance, then after upgrade, you will not get the updated configuration of the latest version.

To get the latest configuration files, do these steps:

1. Edit the `targetConfiguration.xml` file.
2. Delete the path of new configuration from the file.
3. Restart the Log Exporter instance.

Format Configuration

Overview of Format Configuration

Every format has its own predefined format configuration file that configures the format of the exported logs, the delimiters, fields that are part of the header, and so on.

These format configuration files are configured in this file:

```
$EXPORTERDIR/targets/<Name of Log Exporter
Configuration>/conf/<Format>FormatDefinition.xml
```

Important - Do **not** edit the original `<Format>FormatDefinition.xml` files. **Doing so causes a data loss after an upgrade**.
Instead, create a copy of the file and modify the copied file, while leaving the original intact.
After modifying the copied file, refer to it (using a full path) in the `<formatHeaderFile>` element in the relevant `targetConfiguration.xml` file (see *"Advanced Configuration" on page 48*).

Example list of files:

- `$EXPORTERDIR/conf/CefFormatDefinition.xml`
- `$EXPORTERDIR/conf/GenericFormatDefinition.xml`
- `$EXPORTERDIR/conf/JsonFormatDefinition.xml`
- `$EXPORTERDIR/conf/LeefFormatDefinition.xml`
- `$EXPORTERDIR/conf/LogRhythmFormatDefinition.xml`
- `$EXPORTERDIR/conf/RsaFormatDefinition.xml`
- `$EXPORTERDIR/conf/SplunkFormatDefinition.xml`
- `$EXPORTERDIR/conf/SyslogFormatDefinition.xml`

Body

Parameter	Description	Syslog	Splunk	RSA	CEF	LEEF	LogRhythm	Generic
<start_message_body></start_message_body>	The character preceding the log data payload.	[						
<end_message_body></end_message_body>	The character following the log data payload.	]						
<message_separator></message_separator>	The delimiter that separates logs.
(means '\n')
(means '\n')
(means '\n')
(means '\n')
(means '\n')
(means '\n')	('\n')
<fields_separatator></fields_separatator>	The delimiter that separates log fields.	'; ' (semicolon + space)	| (pipe)	' ' (space)	' ' (space)	&#09 (<TAB>)	| (pipe)	' ' (space)

Parameter	Description	Syslog	Splunk	RSA	CEF	LEEF	LogRhythm	Generic
<field_value_separator></field_value_separator>	The assignment operator.	:	=	=	=	=	=	=
<value_encapsulation_start>"</value_encapsulation_start>	The value encapsulation operator (start)	"				"		"
<value_encapsulation_end>"</value_encapsulation_end>	The value encapsulation operator (end)	"				"		"

Parameter	Description	Syslog	Splunk	RSA	CEF	LEEF	LogRhythm	Generic
<escape_chars><char><orig></orig><escaped></escaped></char></escape_chars>	Escaping unwanted characters. The escape functionality replaces the string that's encapsulated by the 'orig' tags with the one encapsulated by the 'escaped' tags	;\ --> \\ " --> \"
--> ' '] --> \]	| --> ; = --> \=
--> ' '	= --> \=
--> ' '	;\ --> \\ = --> \=
--> ' ' | --> |	= --> \=
--> ' '	| --> ; = --> \=
--> ' '	\ --> \\ " --> '
--> ' '

Header

Parameter	Description	Default values for Syslog	Default values for Splunk	Default values for RSA	Default values for CEF	Default values for LogRhythm
`<header_format></header_format>`	The delimiter between the header values and the number of values. Every `{}` is replaced with one value.	`' '` (space)	`time={}|hostname={}|`	`<134>`	`| ;`	`LOGV2 {}|`

Notes:

- To add a constant string to the header, add the string to the `<header_format>` tag value.
- To add a new field to the header, add a new header format replacement string (for example: `{}`) to the '`<header_format>`' and add the relevant information in the '`<headers>`' tag.

Fields Configuration

Every format has its own predefined fields configuration file that allow to change the name / value of the exported field, filter out irrelevant fields, and so on.

These files are located in each Log Exporter instance directory:

```
$EXPORTERDIR/targets/<Name of Log Exporter
Configuration>/conf/<Format>FieldsMapping.xml
```

Example list of files:

- `$EXPORTERDIR/conf/CefFieldsMapping.xml`
- `$EXPORTERDIR/conf/GenericFieldsMapping.xml`
- `$EXPORTERDIR/conf/JsonFieldsMapping.xml`
- `$EXPORTERDIR/conf/LaaSFieldsMapping.xml`
- `$EXPORTERDIR/conf/LeefFieldsMapping.xml`
- `$EXPORTERDIR/conf/LogRhythmFieldsMapping.xml`
- `$EXPORTERDIR/conf/RsaFieldsMapping.xml`
- `$EXPORTERDIR/conf/SplunkFieldsMapping.xml`

Important - Do **not** edit the original "`<Format>FieldsMapping.xml`" files.
Doing so causes a data loss after an upgrade.
Instead, create a copy of the file and modify the copied file.
After modifying the copied file, refer to it (using a full path) in the `<mappingConfiguration>` element in the relevant `targetConfiguration.xml` file (see *"Advanced Configuration" on page 48*).

Parameter	Description	Values
`<table>`	Some fields appear in tables depending on the log format. This information can be found in the ELG log - one entry for every new field. A field can appear in multiple tables, each distinct instance is considered as a new field.	

Parameter	Description	Values
`<exported></exported>`	Optional. You can filter out specific fields by using the '`exported`' tag (value: '`true`', or '`false`') in the mapping configuration file. Alternatively, if the '`exportAllFields`' tag in the '`targetConfiguration.xml`' file is set to '`false`', only those fields which are listed in the mapping file are exported. **Note** - You can also configure this parameter on a table field level to allow / prevent its export when the field is part of a table.	■ `true` ■ `false`
`<origName></origName>`	The name of the field to be mapped to '`<dstName>`'. See *"Mapping of Log Fields for Advanced Fields Configuration" on page 75*.	
`<dstName></dstName>`	The new mapping scheme name for the desired field.	
`<required></required>`	Optional. When set to '`true`', only logs which contain this field are exported.	■ `true` ■ `false`

Note - If the field you want to add in the `<Format>FieldsMapping.xml` file is in the '`table`' section of the log, then add it in the '`table`' section of the `<Format>FieldsMapping.xml` file. Make sure the '`<exported>`' value for the table is set to '`true`'.

Below is an example log entry. In this log entry, the '`src`' field is a regular field, while the fields '`layer_name`', '`rule_uid`', and so on, are inside the table called '`match_table`':

```
11420 ==>Version: 5 Sequence Number: 218 LUUID:
{0x630e5a89,0x8f,0x9078664e,0xfa671e8c} ... ['src': 5.5.5.5] ['s_
port': 443] ['UP_match_table': TABLE_START] ['ROW_START': 0]
['match_id': 25] ['layer_uuid': "xxxxxxx"] ['layer_name': "xxxxx"]
['rule_uid': "xxxxx"] ['rule_name': "test"] ['ROW_END': 0] ['UP_
match_table': TABLE_END]
```

Example for a `<Format>FieldsMapping.xml` file:

```
<?xml version="1.0" encoding="utf-8"?>
 <fields>
   <field>
     <exported>true</exported>
     <origName>time</origName>
     <dstName>start</dstName>
     <required>false</required>
   </field>
   <field>
     <origName>src</origName>
     <dstName>cef_src</dstName>
   </field>
   <table>
     <tableName>match_table</tableName>
     <tableFormat></tableFormat>
     <exported>true</exported>
     <required>false</required>
     <fields>
       <field>
         <origName>layer_name</origName>
         <exported>true</exported>
       </field>
     </fields>
   </table>
 </fields>
```

Filter Configuration

Overview of the Filtering Feature

The Log Exporter can filter logs based on the field values.

Because field mapping operation is done **before** the actual filtering, make sure to use the **dst** name / value of the fields, if configured specifically in the `<Format>FieldsMapping.xml` file (based on the format). Otherwise, use the name / value as written in the raw log file.

It is possible to configure what to export or what not to export.

The filter configuration file is located in this file:

```
$EXPORTERDIR/targets/<Name of Log Exporter Configuration>/conf/FilterConfiguration.xml
```

Parameter	Description	Valid / Default Values
`<filterGroup operator=""></filterGroup>`	A group of fields that determine what to export. There can be only one such parameter in the `FilterConfiguration.xml` file. The relation between the fields is determined by the operator value. See *"Mapping of Log Fields for Advanced Fields Configuration" on page 75*.	`operator [and | or]`

Parameter	Description	Valid / Default Values
`<field name="" operator="">` `  <value operation=""></value>` `</field>`	Declare a single field filter that participates in the filter group. ■ `name` The name of the field to filter on. ■ `operator` Declares the operator (and / or) between the various declared operations. ■ `operation` Declares the matching logic regarding the declared value. ■ `value` The specific value to filter on. Multiple values for a single operation is supported and should be added as a separate row.	`operator [and | or]` `operation[eq - equal | neq - not equal | gt - greater than | lt - less than ]`

Configuring the Filtering Feature

These are the ways to configure the filtering feature:

- **Use the "cp_log_export" command**

  This command configures filtering only for the fields `Action`, `Blade`, and `Origin`.

  The syntax must **not** contain spaces between the values:

  ```
  cp_log_export set name <name> filter-action-in
  "value1,value2"
  ```

  ```
  cp_log_export set name <name> filter-origin-in
  "value1,value2"
  ```

```
cp_log_export set name <name> filter-blade-in
"value1,value2"
```

In addition, it is possible to use predefined families for "`filter-blade-in`" value:

- Action
    - Accept
    - Allow
    - Ask User
    - Authorize
    - Block
    - Bypass
    - Deauthorize
    - Decrypt
    - Detect
    - Do not send
    - Drop
    - Encrypt
    - Extract
    - Failed Log In
    - Forgot Passcode
    - HTTPS Bypass
    - HTTPS Inspect
    - Inform User
    - Inline
    - Inspect
    - IP Changed
    - Key Install
    - Log In
    - Log Out
    - Open Shell
    - Packet Tagging
    - Prevent

- Quarantine
- Redirect
- Reject
- Remote Wipe
- Reset Passcode
- Run Script
- Send
- System Backup
- System Restore
- Update
- VPN Routing

- **Access**

  For exporting Access logs only:

  - Application Control
  - Compliance
  - Connectra
  - Content Awareness
  - Core
  - DDoS Protector
  - Firewall
  - Identity Awareness
  - Identity Logging
  - Mobile Access
  - Security Gateway/Management
  - UA WebAccess
  - URL Filtering
  - VPN-1 & FireWall-1

- **TP**

  For exporting Threat Prevention logs only:

  - `Anti-Bot`
  - `Anti-Malware`
  - `Threat Emulation`
  - `IPS`
  - `IPS-1`
  - `SmartDefense`
  - `New Anti Virus`
  - `Anti-Spam and Email Security`
  - `Threat Extraction`
  - `MTA`

- **EndPoint**

  For exporting Endpoint logs only:

  - `Anti-Bot`
  - `Anti-Malware`
  - `Anti-Exploit`
  - `Anti-Ransomware`
  - `Capsule Docs`
  - `Endpoint Compliance`
  - `Forensics`
  - `Full Disk Encryption`
  - `Media Encryption & Port Protection`
  - `SecureClient`
  - `Threat Emulation`
  - `Threat Extraction`
  - `Zero Phishing`

- **Mobile**

  For exporting Mobile logs only:

  - `WIFI Network`
  - `Mobile App`
  - `OS Exploits`
  - `Device`
  - `Network Security`
  - `Cellular Network`
  - `Network Access`
  - `iOS Profiles`
  - `Text Message`
  - `On-device Network Protection`

  Example:

```
cp_log_export set name <name> filter-blade-in
Access,TP,EndPoint
```

■ **Modify the "FilterConfiguration.xml" file manually**

You can add new fields to the `FilterConfiguration.xml` file.

For example:

```
<filters>
  <filterGroup operator="and">
    <field name="action" operator="and">
    </field>
    <field name="origin" operator="and">
    </field>
    <field name="product" operator="or">
    </field>
    <field name="severity" operator="or">
      <value operation="eq">3</value>
      <value operation="eq">4</value>
    </field>
    <field name="service_id" operator="or">
        <value operation="eq">https</value>
        <value operation="eq">http</value>
    </field>
    <field name="src" operator="or">
        <value operation="eq">1.2.1.2</value>
        <value operation="eq">3.2.3.2</value>
    </field>
    <field name="dst" operator="or">
        <value operation="eq">1.2.1.2</value>
        <value operation="eq">3.2.3.2</value>
    </field>
    <field name="rule_uid" operator="or">
        <value operation="eq">a24163d2-f2f2-1426-52d1-
fa42133d04bd</value>
        <value operation="eq">b54152f1-a3c2-7242-52c1-
dc86434a28ac</value>
    </field>
    <field name="NAT_rule_uid" operator="and">
        <value operation="eq">a24163d2-f2f2-1426-52d1-
fa42133d04ac</value>
    </field>
    </filterGroup>
</filters>
```

Limitations of the Filtering Feature

- The relation between the values of the same operation is only logical "OR".

  Example:

  ```
  cp_log_export set name <target-name> filter-action-in
  "accept,drop"
  ```

  Only logs with '`action = "accept"`' OR '`action = "drop"`' are exported.

- Filtering is not supported for any of these fields:
  - `app_category`
  - `app_desc`
  - `app_properties`
  - `app_risk`
  - `app_rule_name`
  - `appi_name`
  - `category`
  - `cvpn_category`
  - `cvpn_resource`
  - `desc`
  - `HTTPS_inspection_rule_name`
  - `matched_category`
  - `name`
  - `properties`
  - `time`
  - `UUID`
- Filtering for a certain field with the a double "NOT" condition "`not equal(value1) OR not equal(value2)`" is not supported.

  When editing the filtering XML, make sure to have a maximum of one line of "`neq`" operation in each field.
- Filter configuration of the LAAS exporter is supported only through the SmartConsole user interface. File configuration for LAAS Exporter is not supported.

TLS Configuration

You can configure a secured connection between the Log Exporter and the Syslog server.

Log Exporter can export logs over an encrypted connection using the TLS protocol.

When using TLS, it is important to know that only **mutual authentication** is allowed.

For mutual authentication, the **Log Exporter** needs these certificates:

- **CA certificate** (in the **PEM** format) that signed both the client (Log Exporter side) and server (syslog server side) certificates
- **Client certificate** (in the **P12** format) on the Management Server / Log Server with Log Exporter

Notes:

- For Log Exporter TLS Configuration on Smart-1 Cloud, refer to sk181142.
- The Management Server / Log Server with Log Exporter must be able to connect to CA.
- In addition to these two certificates, a third certificate should be installed on the Syslog server (based on the server requirement).
  It is also possible to use self-signed certificates.
- Log Exporter supports only keys encrypted and signed with the **RSA** for the TLS communication (ECC keys are not supported).

Important:

- The procedure below uses the **openssl** commands on a non-Check Point server.
- For testing purposes you can use the `Cert_Generator.sh` script to automate these steps.
  To download the script, click here.

Part 1 of 4 - Creating a CA Certificate:

1. Create a CA Key file:

```
openssl genrsa -out ca.key 2048
```

2. Create a CA Certificate file:

```
openssl req -x509 -new -nodes -key ca.key -days 2048 -out
ca.pem
```

You are prompted to provide information regarding the certificate.

Apart from the Common Name (it is recommended to use the device IP address as the Common Name), all other fields are optional and can be skipped.

If you are purchasing an SSL certificate from a certificate authority, it often requires these additional fields.

Part 2 of 4 - Creating a Log Exporter Certificate:

Warning - Do **not** share the client key files with anyone.

1. Create a Log Exporter Key file:

```
openssl genrsa -out cp_client.key 2048
```

2. Create a Log Exporter CSR file:

```
openssl req -new -key cp_client.key -out cp_client.csr
```

3. Create a Log Exporter CRT file:

```
openssl x509 -req -in cp_client.csr -CA ca.pem -CAkey ca.key -
CAcreateserial -out cp_client.crt -days 2048 -sha256
```

4. Create Log Exporter P12 file:

```
openssl pkcs12 -inkey cp_client.key -in cp_client.crt -export
-out cp_client.p12
```

Part 3 of 4 - Creating a Syslog Server Certificate:

Warning - Do **not** share the server key files with anyone.

1. Create a Target Server key file:

```
openssl genrsa -out server.key 2048
```

2. Create a Target Server CSR file:

```
openssl req -new -key server.key -out server.csr
```

3. Create a Target Server CRT file:

```
openssl x509 -req -in server.csr -CA ca.pem -CAkey ca.key -
CAcreateserial -out server.crt -days 2048 -sha256
```

Note - Several SIEM applications require the Server certification to be in a specific format.
For more information, see *Instructions for Specific SIEM Applications*.

Part 4 of 4 - Put the certificate files on the Check Point Management Server / Log Server

1. Connect to the command line on the Check Point Management Server / Log Server.
2. Log in to the Expert mode.
3. Create the sub-directory "`certs`" in the applicable Log Exporter directory:

   ```
   mkdir -p $EXPORTERDIR/targets/<Name of Log Exporter Configuration>/certs
   ```

4. Copy these files to the Management Servers / Log Server to the sub-directory "`certs`":
   - `ca.pem`
   - `cp_client.p12`
5. Assign the "read" permission the certificate files:

   ```
   chmod -v +r $EXPORTERDIR/targets/<Name of Log Exporter Configuration>/certs/ca.pem
   ```

   ```
   chmod -v +r $EXPORTERDIR/targets/<Name of Log Exporter Configuration>/certs/cp_client.p12
   ```

6. Configure the Log Exporter:

   ```
   cp_log_export add name <Name> [domain-server {mds | all}] target-server <HostName or IP address of Target Server> target-port <Port on Target Server> protocol tcp format {syslog | splunk | cef | leef | generic | json | logrhythm | rsa} encrypted true ca-cert $EXPORTERDIR/targets/<Name of Log Exporter Configuration>/certs/ca.pem client-cert $EXPORTERDIR/targets/<Name of Log Exporter Configuration>/certs/cp_client.p12 client-secret <Password for cp_client.p12>
   ```

Mapping of Log Fields for Advanced Fields Configuration

Check Point's solution includes multiple log fields, representing the diversity of Check Point's products.

The log fields mapping helps you understand security threats, logs language, complex queries and SIEM.

For information on Check Point's Log Fields Mapping, refer to sk144192.

Troubleshooting

Troubleshooting Scenario 1

Symptoms

Logs are not exported after adding a filter to the `FilterConfiguration.xml` file, or by using the `cp_log_export` command.

Cause

"cp_log_export" adds the default values to the `FilterConfiguration.xml` file, while the field names should be the same as the exported name.

This causes the filter mechanism to not match any log.

Suggested Solution

1. In the relevant `<Format>FieldsMapping.xml` file, look for the relevant mapped field.
2. Find the element named `<dstName>` and copy it.
3. Edit the `$EXPORTERDIR/targets/<Name of Log Exporter Configuration>/conf/FilterConfiguration.xml` file.
4. Replace the field name to the previously copied one.
5. Restart the Log Exporter to load the new settings:

```
cp_log_export restart name <Name>
```

Troubleshooting Scenario 2

Symptoms

Configured a field in the `<Format>FieldsMapping.xml` file as '`export =false`', but Log Exporter keeps exporting this field.

For example, you configured the field "`layer_uuid export=false`", but you keep seeing this field as part of the log on your Log Server.

Cause

The field is part of a '`table`' in the log, and the standard configuration to filter out a field is not effective on a '`table`' field.

Suggested Solution

To prevent these fields from been exported, you need to:

1. Go to the `$EXPORTERDIR/targets/<exporter_name>/conf/` directory.
2. Edit the "Fields Mapping" file you use (that corresponds to the format you export).
3. Look for the tag '`<tableName>match_table</tableName>`'.
4. Add the required lines:
   - If the '`<tableName>match_table</tableName>`' tag does not exist, add these lines inside the '`<fields`' tag:

```
<table>
   <tableName>match_table</tableName>
      <fields>
         <field>
            <origName>field_name</origName>
            <exported>false</exported>
         </field>
      </fields>
</table>
```

   - If the '`<tableName>match_table</tableName>`' tag exists, add this line inside the '`<fields>`' tag:

```
<table>
   <tableName>match_table</tableName>
      <fields>
         <field>
            <origName>field_name</origName>
            <exported>false</exported>
         </field>
   </fields>
</table>
```

The file should look like this:

```
<fields>
<!-- Filter out fields -->
  <field><origName>field_
name1</origName><exported>false</exported></field>
  <field><origName>field_
name2</origName><exported>false</exported></field>
  <field><origName>field_
name3</origName><exported>false</exported></field>
    ... ...
    <table>
       <tableName>match_table</tableName>
         <fields>
           <field> ... </field>
           ... ...
           <field>
              <origName>field_name</origName>
              <exported>false</exported>
           </field>
         </fields>
    </table>

<!-- End of filter out -->
</fields>
```

5. Save the change in the file and close it.
6. Restart the Log Exporter to load the new settings:

```
cp_log_export restart name <Name>
```

Troubleshooting Scenario 3

Symptoms

A SIEM application cannot parse logs correctly, in which the "`Blade`" field contains several Software Blades

Cause

The `<`*`Format`*`>FieldsMapping.xml` file does not contain the required configuration (Issue PMTR-113944)

SmartConsole / SmartView > **Logs & Monitor** view > **Logs** tab, may show logs from a Security Gateway that contain data from several Software Blades.

For such logs:

1. On the **Logs** tab, the column "**Blade**" shows "**Multiple Blades**".
2. The log card, in the "**Log Info**" section, field "**Blade**" shows the list of the relevant Software Blades.

Suggested Solution

To make sure the SIEM application that receives such logs can parse them correctly:

1. Create a copy of the relevant `<Format>FieldsMapping.xml` file for your SIEM:
   - For the "`CEF`" format:

     `$EXPORTERDIR/conf/CefFieldsMapping.xml`
   - For the "`LaaS`" format:

     `$EXPORTERDIR/conf/LaaSFieldsMapping.xml`
   - For the "`RSA`" format:

     `$EXPORTERDIR/conf/RsaFieldsMapping.xml`
   - For the "`Splunk`" format:

     `$EXPORTERDIR/conf/SplunkFieldsMapping.xml`
2. Edit the relevant `<Format>FieldsMapping.xml` file.
3. Go to this section:

```
... ...
<field>
   <origName>product</origName>
   <callback>
      <name>replace_value</name>
      <args>
         <args ... />
      </args>
   </callback>
</field>
... ...
```

4. For each list of relevant Software Blades, add the new "`<arg ... />`" row and configure the value of the key called "`key`" to contain the names of the relevant Software Blades that are separated with the entity for the carriage return: `
`

   Important - You must add as many "`<arg ... />`" rows as the number of possible combinations of Software Blade names.
5. Restart the Log Exporter to load the new settings:

```
cp_log_export restart name <Name>
```

Example:

In a log card, in the "`Log Info`" section, the field "`Blade`" shows two rows: `QoS` and `Firewall`.

Because there are two Software Blades, it is necessary to add these two rows in the relevant `<Format>FieldsMapping.xml` file:

```
... ...
<arg key="VPN-1 & FireWall-1" value="Firewall"/>
<arg key="VPN-1 & FireWall-1&#10;FG" value="QoS & FireWall"/>
<arg key="FG&#10;VPN-1 & FireWall-1" value="QoS & FireWall"/>
... ...
```

Troubleshooting Scenario 4 (R81.20 only)

Symptoms

When creating new Log Exporter on the R81.20 Management Server, `CefFieldsMapping.xml` is not in the Log Exporter's "`conf`" directory.

Cause

In R81.20, the existing behavior aligns with the intended outcome. The XML files are exclusively located in the primary exporter directory, specifically within the `$EXPORTERDIR/conf` path.

Suggested Solution

To edit a file, create a copy from the `$EXPORTERDIR/conf/` directory, make the necessary changes, and then update the Log Exporter's `targetConfiguration.xml`.

Troubleshooting Scenario 5 (R81.20 only)

Symptoms

After creating Log Exporter target in SmartConsole, the output of the `cp_log_export show` command does not show the created Log Exporter target.

Cause

There is an issue with this file:

`/var/opt/CPsuite-R81.20/fw1/conf/logExporters.json`

Suggested Solution

1. Connect to the command line on the Management Server / Log Server.
2. Log in to the Expert mode.
3. Navigate to the configuration directory:

   ```
   cd /var/opt/CPsuite-R81.20/fw1/conf/
   ```

4. Back up the existing `logExporters.json` file:

   ```
   mv -v logExporters.json{,_BKP}
   ```

5. Go to SmartConsole and create a new Log Exporter.
6. Publish the session.
7. Install database (click **Menu** > **Install database** > select all objects > click **Install**).
8. Verify if the new Log Exporter is visible:

   ```
   cp_log_export show
   ```

Appendix B - Additional Information

Special Log Fields

Field	Description
`loguid`	Some Check Point logs are updated over time. Update logs have the same `loguid` value. Check Point SmartEvent correlates those updates into a single unified log. When sending update logs to 3rd party (SIEM) servers, in raw read-mode, they arrive as distinct logs. Best use the semi-unified read-mode, which sends few instances of the log, but each instance contains the entire event chain until this update. Administrators can alternatively use the `loguid` field to correlate update logs and get the full event chain themselves. **Note** - All related logs & log-updates share the same initial time as the 1st log (in semi-unified mode). Example of update logs includes the total number of bytes sent and received over time or the severity field which is updated over time as more information becomes available.
`hll_key`	Stands for High-Level Log key. This concept was introduced in the R80.10 release, where multiple connection logs can comprise one session with one shared "`hll_key`". For example, when browsing a webpage, you might have multiple connection logs which are related to the same session. Connection logs which are part of the same session share the same "`hll_key`" value.

Configuration for Syslog-NG Listener

When configuring a source on a Syslog NG server it is recommended to use the syslog-protocol flag.

For example:

```
source s_network { network(transport("tcp") port(514) flags
(syslog-protocol) ); };
```

Configuration for Splunk Listener

It is recommended to add these time settings to your source type:

```
TIME_FORMAT = %s
TIME_PREFIX = time=
MAX_TIMESTAMP_LOOKAHEAD = 15
```

Configuration for ArcSight Listener

The Log Exporter solution does not work with the OPSEC LEA connector.

Instead, you must install the ArcSight Syslog-NG connector.

Mapping for the ArcSight Common Event Format (CEF)

CEF is an extensible, text-based format designed to support multiple device types by offering the most relevant information. Message syntaxes are reduced to work with ESM normalization. Specifically, CEF defines a syntax for log records comprised of a standard header and a variable extension, formatted as key-value pairs. The CEF format can be used with on-premise devices by implementing the ArcSight Syslog SmartConnector. CEF can also be used by cloud-based service providers by implementing the SmartConnector for ArcSight Common Event Format REST.

CEF Header format

	Version	Device Vendor	Device Product	Device Version	Device Event Class ID	Name	Severity
Default	CEF:0	Check Point	Log Update	Check Point	Log	Log	0
Values	-	-	Product Name (Blade)	-	Attack Name Protection Type Verdict Matched Category DLP Data Type Application Category Application Properties	Protection Name Application Name Message Info Service ID Service	Application Risk Risk Severity

Mapping for the QRadar Log Event Extended Format (LEEF)

The Log Event Extended Format (LEEF) is a customized event format for IBM Security QRadar.

LEEF Header format

	LEEF:Version	Vendor	Product	Version	EventID
Default	LEEF:2.0	Check Point	Log Update	1.0	Check Point Log
Values	-	-	Product Name (Blade)	-	Protection Name Application Name Action

Note - The time format is not compliant with the official LEEF format.
Until the time IBM adds support for Epoch time format, Log Exporter with the LEEF format is supported only partially.