Corelight Sensor AP 200, AP 520, AP 1001, AP 1100, AP 1200, AP 3000, AP 3100, AP 3200, AP 5000, AP 5002 & AP 5200 Common Criteria Guidance Document

December 20, 2024

0.2

Prepared by:
Corelight, Inc.
www.corelight.com

in collaboration with:
Gossamer Laboratories
https://gossamersec.com/

Contents

Revision History

Version	Date	Description
0.1	November 5, 2024	Initial version with response to comments
0.2	December 20, 2024	Response to comments

Overview

This document is intended to be a supplement to the Corelight Sensor documentation version 22.1. This Common Criteria guidance document contains configuration information needed to correctly configure and administer the Corelight Sensor in a way that conforms to the Common Criteria Certification requirements. The Corelight Sensor, properly configured, conforms to the Common Criteria **Network Device Profile Version 2.2e** [NDcPP v2.2e]. The information contained in this document is intended for administrators responsible for the configuration and management of the Corelight Sensor.

This document is meant to provide necessary guidance to the administrators of the Corelight appliance who require a configuration which leaves the system in compliance with Corelight's Common Criteria certification. There are several steps to be followed, and while some parameters allow for a level of flexibility, most parameters must be configured exactly as documented.

It is important to outline the various constraints that Corelight chose to impose on itself in order to better align the Common Criteria certification, and the safety requirements that it outlines, with the capabilities of the Corelight sensor.

1 Definitions

TOE - Target of Evaluation (the Corelight appliance)

2 TOE Overview

Simple to deploy and integrate with existing analysis tools, the Corelight Sensor appliances capture and transform high-volume network traffic into high-resolution data, which unlocks new capabilities for incident response, intrusion detection, forensics and more. The Sensor parses dozens of network protocols and generates rich, actionable data streams designed for security professionals.

3 TOE Product Information

Corelight makes multiple physical sensor models. The models AP 200, AP 520, AP 1001, AP 1100, AP 1200, AP 3000, AP 3100, AP 3200, AP 5000, AP 5002, and AP 5200 are part of the Common Criteria certification program, and thus in scope of this document. The Corelight Sensor, referred to as the TOE is a device which is composed of hardware and software that offers a scalable network analysis and insights solution to the end users. It satisfies all the criteria to meet the collaborative Protection Profile for Network Devices, Version 2.2e [NDcPP v2.2e]. Please refer to the User Guide, **Chapter Two - Quick Start Guides** for additional information about the available models as well as the model-specific details for initial rack and stack.

The TOE is comprised of the following models:

AP 200

Processor:	Intel Xeon Silver 4110 (Skylake)
Size and weight:	1U half depth rackmount (19 x 14.5 x 1.75 inches), 17 lbs.
Monitoring interface:	Four 1G SFP interfaces supporting copper and optical modules at 100M & 1G.
Management interface:	One 10/100/1000 copper ethernet port.
Power:	120/240 VAC 50/60 Hz single PSU. Approximately 83W when idle and 141W usage at load.

AP 520

Processor:	Intel Xeon Gold 5317 (Ice Lake)
Size and weight:	1U rackmount, (19 x 29 x 1.7 inches), 48 lbs
Monitoring interface:	Up to 2 x 1/10G or 10/25G SFP28 interfaces
Management interface:	2 x 1G ports, 2 x 10G SFP+ ports
Power:	Dual redundant 1100W MM 100-240 VAC Titanium PSUs. Approximately 200W usage when idle and 300W usage at load

AP 1001

Processor:	Intel Xeon Silver 4116 (Skylake)
Size and weight:	1U rackmount (19 x 25.6 x 1.75 inches), 40 lbs.
Monitoring interface:	Four 1G/10G SFP/SFP+ interfaces supporting copper and optical modules at 1G and 10G
Management interface:	One 10/100/1000 copper ethernet port and up to 2 10G ethernet ports
Power:	120/240 VAC 50/60 Hz redundant dual PSUs. 700W at 110V or 750W at 220V. Approximately 180W usage when idle and 290W usage at load.

AP 1100

Processor:	Intel Xeon Silver 4314 (Sunny Cove/Ice Lake)
Size and weight:	1U rackmount, (19 x 31.85 x 1.7 inches), 48 lbs
Monitoring interface:	Four 1G/10G SFP/SFP+ modules. Support for copper and optical modules at 1G and/or 10G.
Management interface:	2 x 1G ports, 4 x 10G SFP+ ports
Power:	100-240 VAC 50/60 Hz redundant dual PSUs. Approximately 270W usage when idle and 439W usage at load

AP 1200

Processor:	AMD EPYC 9254 (Genoa/Zen 4)
Size and weight:	1U rackmount, (18.7 x 32.4 x 1.685 inches), 48 pounds
Monitoring interface:	Four 1G/10G SFP/SFP+ modules. Support for copper and optical modules at 1G and/or 10G.
Management interface:	2 x 1G ports, 4 x 10/25G SFP28 ports
Power:	Dual redundant 1400W MM 100-240 VAC Titanium PSUs. Approximately 196W usage when idle and 660W usage at load

AP 3000

Processor:	Intel Xeon Gold 6238 (Cascade Lake)
Size and weight:	1U rackmount (19 x 25.6 x 1.75 inches), 34 lbs.
Monitoring interface:	4 x 1/10G SFP/SFP+ modules OR 2 x QSFP28 modules capable of supporting 8x10G or 2x40G
Management interface:	One 10/100/1000 copper ethernet port and up to 2x10G ethernet ports
Power:	120/240 VAC 50/60 Hz redundant dual PSUs. Approximately 161W usage when idle and 445W usage at load.

AP 3100

Processor:	Intel Xeon Gold 5318Y (Sunny Cove/Icelake)
Size and weight:	1U rackmount, (19 x 31.85 x 1.7 inches), 48 lbs
Monitoring interface:	Up to 8 SFP/SFP+ or 2 QSFP+ modules. Support for copper and/or optical modules at 1G and 10G or 40G
Management interface:	2 x 1G ports, 4 x 10G SFP+ ports
Power:	100-240 VAC 50/60 Hz redundant dual PSUs. Approximately 501W usage when idle and 697W usage at load

AP 3200

Processor:	AMD APYC 9354 (Genoa/Zen 4)
Size and weight:	1U rackmount, (18.7 x 32.4 x 1.685 inches), 48 pounds
Monitoring interface:	4 x 1/10G SFP/SFP+ modules OR 2 x QSFP28 modules capable of supporting 8x10G, 2x40G or 2x100G interfaces.
Management interface:	2 x 1G ports, 4 x 10/25G SFP28 ports
Power:	Dual redundant 1400W MM 100-240 VAC Titanium PSUs. Approximately 227W usage when idle and 809 W usage at load

AP 5000

Processor:	AMD EPYC 7742 (Rome/Zen 2)
Size and weight:	1U rackmount (19 x 27 x 1.7 inches), 48 lbs.
Monitoring interface:	2 x QSFP28 modules capable of supporting 8x10G, 2 x40G or 2x100G interfaces.
Management interface:	One 10/100/1000 copper ethernet port and up to 4x10G ethernet ports
Power:	120/240 VAC 50/60 Hz redundant dual PSUs. Approximately 443W usage when idle and 852W usage at load.

AP 5002

Processor:	AMD EPYC 7713 (Milan/Zen 3)
Size and weight:	1U rackmount (17.1 x 29 x 1.75 inches), 47.4 lbs
Monitoring interface:	2 x QSFP28 modules capable of supporting 8x10G, 2 x40G or 2x100G interfaces.
Management interface:	2 x 1G ports, 4 x 10G SFP28 ports
Power:	100-240 VAC 50/60 Hz redundant dual PSUs. Approximately 443W usage when idle and 852W usage at load

AP 5200

Processor:	AMD EPYC 9754 (Bergamo/Zen 4c)
Size and weight:	1U rackmount, (18.7 x 32.4 x 1.685 inches), 48 pounds
Monitoring interface:	2 x QSFP56 modules capable of supporting 8x10G, 2x40G or 2x100G interfaces.
Management interface:	2 x 1G ports, 4 x 10/25G SFP28 ports
Power:	Dual redundant 1800W MM 200-240 VAC Titanium PSUs. Approximately 475W usage when idle and 1209W usage at load

4 TOE Delivery

The TOE is delivered via commercial carrier (i.e. DHL, FedEx, UPS, Expeditors etc). The shipment will contain a packing slip with the serial numbers of all shipped devices. The receiver must verify that the hardware serial numbers match the serial numbers listed in the packing slip. The Corelight appliance is shipped with all necessary software pre-installed. All software updates will be provided in the form of offline updates and will be made available by Corelight as part of the normal appliance release lifecycle.

5 TOE Evaluated Configuration

The TOE in the evaluated configuration consists of the platform as stated in the previous section. The TOE supports secure connectivity with another IT environment device as stated in Table 2 below.

Table 2 – IT Components

Component	Required	Usage
Audit server (via SFTP server)	Yes	The TOE exports audit events to an external SFTP server via SSH v2 protocol.
Management workstation with SSH client	Yes	This includes any IT Environment Management workstation with an SSH client

6 Assumptions

This section describes the assumptions made in identification of the threats and security requirements for network devices. The network device is not expected to provide assurance in any of these areas, and as a result, requirements are not included to mitigate the threats associated.

Table 3 – Assumptions

ID	Assumption
A.PHYSICAL_PROTECTION	The Network Device is assumed to be physically protected in its operational environment and not subject to physical attacks that compromise the security or interfere with the device's physical interconnections and correct operation. This protection is assumed to be sufficient to protect the device and the data it contains. As a result, the cPP does not include any requirements on physical tamper protection or other physical attack mitigations. The cPP does not expect the product to defend against physical access to the device that allows unauthorized entities to extract data, bypass other controls, or otherwise manipulate the device. For vNDs, this assumption applies to the physical platform on which the VM runs.

A.LIMITED_FUNCTIONALITY	The device is assumed to provide networking functionality as its core function and not provide functionality/services that could be deemed as general purpose computing. For example, the device should not provide a computing platform for general purpose applications (unrelated to networking functionality). .
A.NO_THRU_TRAFFIC_PROTE CTION	A standard/generic Network Device does not provide any assurance regarding the protection of traffic that traverses it. The intent is for the Network Device to protect data that originates on or is destined to the device itself, to include administrative data and audit data. Traffic that is traversing the Network Device, destined for another network entity, is not covered by the ND cPP. It is assumed that this protection will be covered by cPPs and PP Modules for particular types of Network Devices (e.g., firewall).

ID	Assumption
A.TRUSTED_ADMINISTRATO R	The Security Administrator(s) for the Network Device are assumed to be trusted and to act in the best interest of security for the organization. This includes appropriately trained, following policy, and adhering to guidance documentation. Administrators are trusted to ensure passwords/credentials have sufficient strength and entropy and to lack malicious intent when administering the device. The Network Device is not expected to be capable of defending against a malicious Administrator that actively works to bypass or compromise the security of the device. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are expected to fully validate (e.g. offline verification) any CA certificate (root CA certificate or intermediate CA certificate) loaded into the TOE's trust store (aka 'root store', ' trusted CA Key Store', or similar) as a trust anchor prior to use (e.g. offline verification).
A.REGULAR_UPDATES	The Network Device firmware and software is assumed to be updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.

A.ADMIN_CREDENTIALS_SEC URE	The Administrator's credentials (private key) used to access the Network Device are protected by the platform on which they reside.
A.RESIDUAL_INFORMATION	The Administrator must ensure that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

7 Security Objectives for the Operational Environment

The following subsections describe objectives for the Operational Environment

Table 4 – Security Objectives for the Operational Environment

ID	Objective for the Operational Environment
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE. Note: For vNDs the TOE includes only the contents of the its own VM, and does not include other VMs or the VS.
OE.NO_THRU_TRAFFIC_PRO TE CTION	The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment.

OE.TRUSTED_ADMIN	Security Administrators are trusted to follow and apply all guidance documentation in a trusted manner. For vNDs, this includes the VS Administrator responsible for configuring the VMs that implement ND functionality. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are assumed to monitor the revocation status of all certificates in the TOE's trust store and to remove any certificate from the TOE's trust store in case such certificate can no longer be trusted.
OE.UPDATES	The TOE firmware and software is updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
OE.ADMIN_CREDENTIALS_SEC URE	The Administrator's credentials (private key) used to access the TOE must be protected on any other platform on which they reside.

ID	Objective for the Operational Environment
OE.RESIDUAL_INFORMATIO N	The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment. For vNDs, this applies when the physical platform on which the VM runs is removed from its operational environment.

8 General understanding of requirements and constraints

Only the SFTP exporter has been included in the scope of the Common Criteria evaluation and it is therefore the only supported exporter which must be selected for the system to be compliant with Corelight's Common Criteria certification. This has a number of implications, one of which having to do with auditing and audit log export.

Audit logging is a requirement to comply with Common Criteria. To this end the sensor records audit events around configuration changes and any security-related matters, such as authentication attempts, successes, failures, cryptographic failures, attempted use of disallowed algorithms, failures in the customer-accessible API, etc. Auditing on the sensor is automatically enabled and those records are stored locally, however only last 7 days worth of messages are accessible. Audit log **export** however is not automatically enabled, and needs to be enabled explicitly.

Accurate timestamps are a prerequisite for usable audit records. You must make certain that the sensor's NTP configuration is working correctly.

Compliance with certification requires that all generated audit messages be securely transported to a remote logging server, periodically without any manual effort. Audit messages are sensitive in nature, may contain confidential information, and are critical to the detection of suspicious activity on the sensor. To comply with Common Criteria requirements, these messages must be protected from modification and from exposure in transport. Thus, a protocol with on-the-wire data encryption, and authenticatication must be used. SFTP was the protocol Corelight chose to use to provide for secure delivery of audit records. With SFTP message security and integrity are assured and the export mechanism used for audit records is the same as the export for Zeek logs.

SFTP exporter operates in batches, periodically transfering a batch of messages to the configured remote SFTP server. Audit records are batched-up hourly, meaning each batch will contain roughly 60 minutes worth of audit records. As of this writing this setting is not configurable. Thus, in every batch the oldest messages will be up to 60 minutes old. Timestamps within the actual messages are recorded at the time the messages are generated and thus as long as the clock on the sensor is accurate, the timestamps will be accurate and not affected in any way by the batching mechanism.

One of the more impacting choices which must be adhered to is a single administrative user (**admin** account) requirement for management of the sensor. This means that if multiple users are entrusted with the ability to configure the sensor, all users making configuration changes must be making those changes after logging in as the **admin** user. By extension this limits the visibility into who specifically made a particular configuration change. All audit events generated while the name

of the logged in user is known, are going to contain admin as the username. In some cases the name of the user is unknown, or irrelevant, such as in the instances where some system task leads to the generation of audit records. These events may not have any user associated with them, or may have the admin user referenced.
The sensor normally communicates with the Corelight Cloud Services infrastructure to enable remote support, share non-sensitive telemetry, enabling observability and monitoring by Corelight, and automatic updates. This remote connectivity must be disabled in order to comply with the Common Criteria certification. In this mode of operation remote access by Corelight's support personnel will not be possible and no telemetry will be sent back. Licensing and automatic upgrades are also disabled. It is still possible to correct entitlement and update the sensor via an alternate (offline) process.

Local and remote management of the sensor is assumed to be performed either via the keyboard directly at the console or via SSH if remote. To comply with the Common Criteria certification the sensor must be able to lock out users after some number of failed authentication attempts. This requirement applies only to remote management. Local authentications, those performed directly at the appliance's console are thus not in scope of this requirement. It will be necessary to enable account locking if remote management of the sensor (via SSH) will be permitted.

The account locking behaviour is only applicable to password-based authentication. It is possible to enable key-based authentication, in which case account locking is not a requirement. Key-based authentication is a mechanism where a previously generated key containing two parts, public and private, is used by the administrator to authenticate themselves to the sensor by having previously provided to the sensor the public part of the key, which the sensor "trusts". It does not make sense to lock an account when key-based authentication is used, because password authentication is at that point disabled, and the purpose behind the lockout mechanism is to defeat password-guessing brute-force attempts.

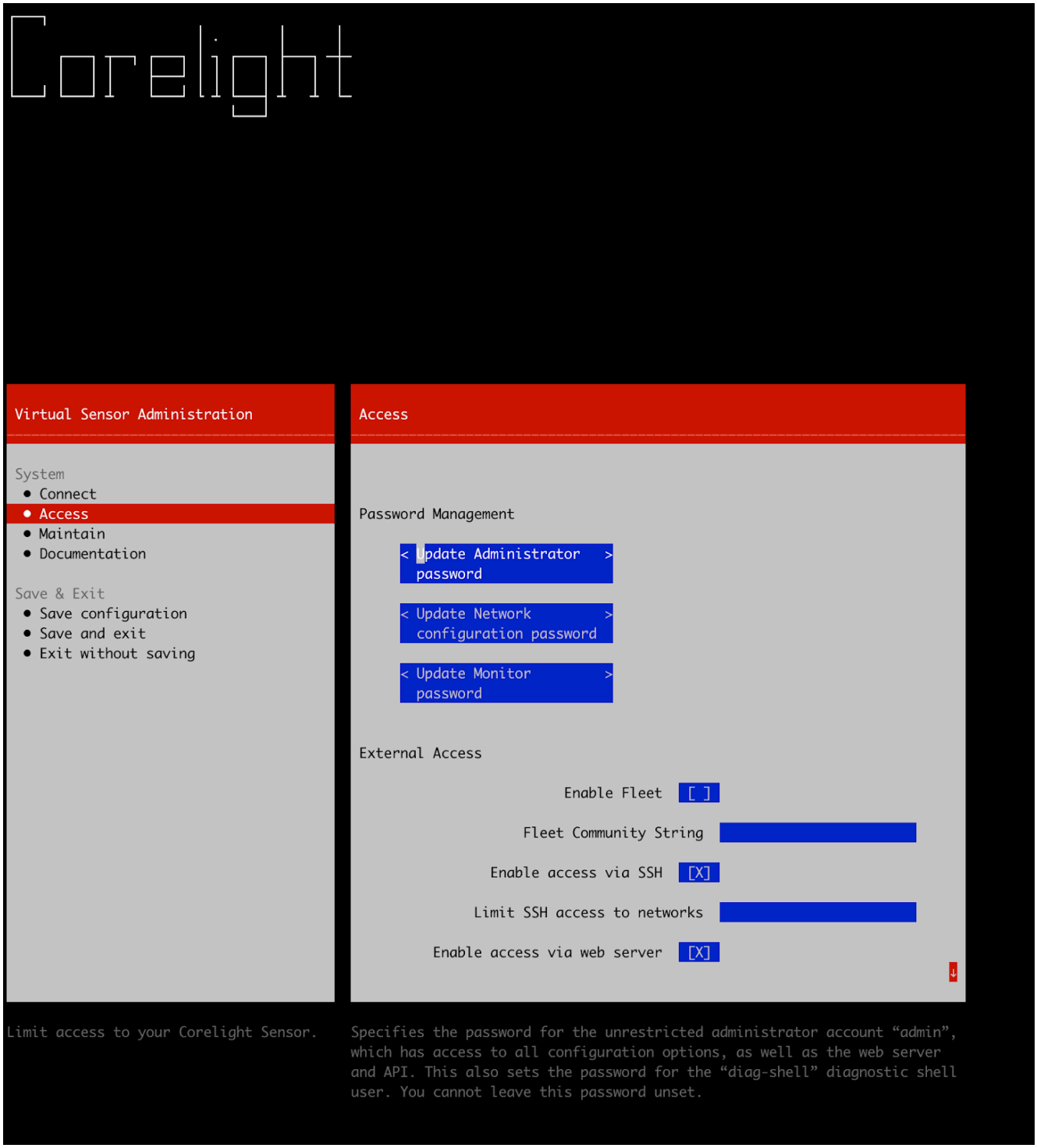
9 Initial onboarding of a brand-new appliance

If the system is brand new and thus never configured before, it will require some basic configuration before the remainder of this document could be followed. Out of the box the only usable configuration interface is the textual user interface, which is local only, requiring a keyboard and a monitor to be physically attached to the sensor being configured.

After the sensor boots there will be a login screen on the primary terminal which looks like the following screenshot.

Login as the admin user with the default password admin. Please be sure to change the password as soon as you authenticate successfully and are able to manage the device. To change the password after you have logged in, select **Access** from the left-hand side navigation and then the **right arrow key** on the keyboard to highlight the **Update Administrator password** and hit the **Enter** key. You will be prompted to enter the current password, which in this case will be the out-of-box default password and then once accepted you will be asked to enter and confirm the new password.

Repeat this process for the netconfig user. This is a default user, which ships with the system, but Corelight chose to exclude it from the Common Criteria certification. The account should have its password changed so as to not leave the default password set. This account must not be used otherwise. Changing its password at this point is strictly a security concern. Highlight the **Update Network configuration password** and hit the **Enter** key. You will be prompted to enter the current password, which in this case will be the out-of-box default password and then once accepted you will be asked to enter and confirm the new password.



If you plan on enabling remote access to manage the appliance over SSH, now is a good time to enable this functionality. Simply hit the down arrow key a number of times, until you reach the **Enable access via SSH** checkbox. Check the box by hitting the **spacebar key** and hit the **left arrow key** to return to the primary left-hand side navigation. We will cover key-based authentication in

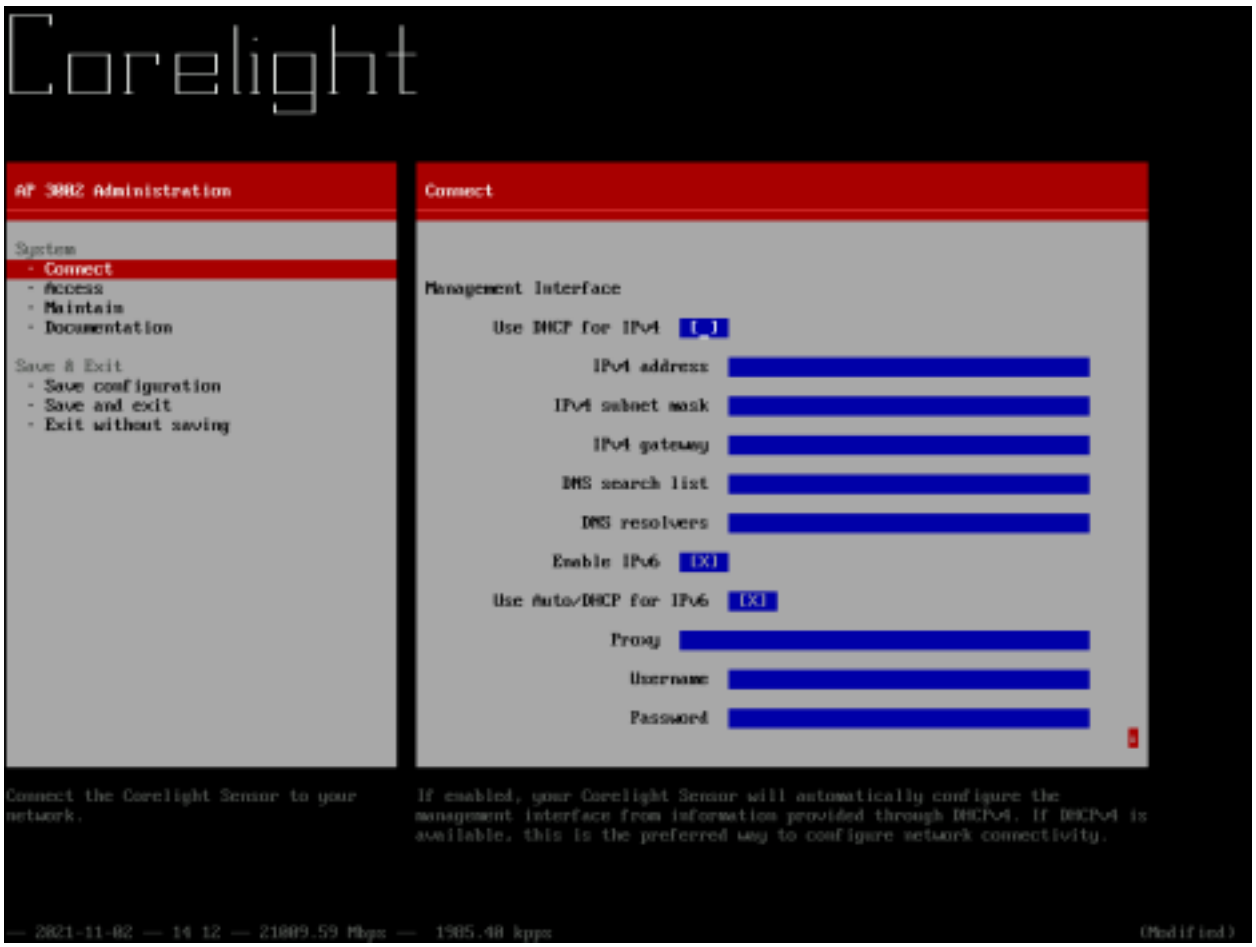
the later part of this document. After saving these changes it will become possible to connect to the sensor remotely via SSH as soon as the management network interface is configured, and make further configuration changes from the convenience of your workstation.

Corelight chose to exclude support for Fleet from the Common Criteria certification, thus we do not cover its enablement in this document.

The Corelight appliance defaults to DHCP to automatically manage network settings on the management interface. By default, the management network is also the network used for export of telemetry and additionally audit log data. However, if DHCP is not enabled on the network where the sensor is connected, or it is not actually desired, it will be necessary to configure the network settings manually. To configure the management interface manually, select **Connect** from the left hand side navigation and then hit the **right arrow key**. The screen will look similar to one in the next screenshot. This is where you can set your preferences for IPv4 and IPv6 networking, DHCP vs. static, etc.

In order to configure a static address you must first uncheck the **Use DHCP for IPv4** checkbox. Once that's unchecked additional fields will appear, allowing you to specify the desired interface IP address, subnet mask, and DNS configuration. Use the arrow keys to navigate these settings and when done return to the left-hand side navigation by hitting the **left arrow key**.


Corelight
AP 3002 Administration
System
- Connect
- Access
- Maintain
- Documentation
Save & Exit
- Save configuration
- Save and exit
- Exit without saving
Connect
Management Interface
Use DHCP for IPv4 [X]
Enable IPv6 [X]
Use Auto/DHCP for IPv6 [X]
Proxy
Username
Password
NTP server 0.pool.ntp.org,1.pool.ntp.org
Timezone < UTC Offset > < UTC+0 >
Syslog server
Connect the Corelight Sensor to your network.
If enabled, your Corelight Sensor will automatically configure the management interface from information provided through DHCPv4. If DHCPv4 is available, this is the preferred way to configure network connectivity.
— 2021-11-02 — 14:06 — 10320.50 Mbps — 1729.31 kpps
(Modified)



In order to have reliable timestamps, which is necessary to support audit logging, be sure to also configure the NTP server(s) and timezone. These settings are at the bottom of the same screen. Simply navigate to the bottom with the **down arrow key**. Without setting any timezone, the system will assume Universal Time Coordinated (UTC). It is recommended to configure at least two NTP servers in order to make clock synchronization more resilient to transient outages of NTP servers. Multiple servers may be specified by separating the DNS names or IP addresses with a comma. By default, two public NTP servers are already configured. They are 0.pool.ntp.org and 1.pool.ntp.org. DNS is for all intents and purposes required, and assumed to be working correctly, whether configured automatically via DHCP or manually. These names will not be resolvable otherwise and this will lead to a broken NTP configuration and by extension drifting clock.

10 Accessing product documentation

Corelight appliance documentation is not openly distributed. Each appliance will contain necessary information to access this documentation from a secure site. Username, password and the URL for the docs site will be provided. This information will be accessible from the diagnostic shell, via the following command.

corelight-client information get | grep doc-

11 Audit server requirements for audit log export

One of the core requirements of Common Criteria is ability to audit the system and broadly any configuration changes, cryptographic failures, etc. The Corelight appliance is designed to have capacity for 100,000 audit records. This capacity should be sufficient to store multiple days' worth of auditing data, even with a large number of events such as those generated during the appliance reconfigurations. Once this limit is reached, oldest records are removed in order to permit new records to be added. Administrators can access the audit events locally, however while there is room to store months of data on a typical system, API queries cannot back further than 7 days. In order to remain compliant with the certification all audit data must be exported to a secure external audit server in order to protect from loss of this information. The mechanism for export exists and its configuration is covered later in this document.

11.1 Audit Log Export

The TOE supports secure communication with an external audit server. This communication is secured with the SSH protocol. The Corelight appliance is able to buffer a large number of records and this is meant to protect from loss of information in the instances where the external audit server is not available for some period of time. Audit records are automatically exported in batches on a regular schedule. A batch will contain zero or more JSON-encoded records, with a few examples presented in the *Sample Audit Events* below.

The Corelight appliance extended its Zeek log batch exporter infrastructure in order to support secure export of audit logs via an already established mechanism. Currently, due to constraints we imposed with Common Criteria, the only supported method is via the SFTP log exporter. The only supported exporter under Common Criteria is the SFTP exporter, thus you will be required to have an SFTP capable server to receive both audit logs and the traditional Zeek logs. We discuss configuration of audit log export later in this document.

11.2 Connection Loss Recovery

Upon connection loss, the exporter process is going to detect the failure and will immediately restart itself, which will happen in a continuous loop as long as the remote end is not available. Each time the exporter is unable to connect, it will terminate, wait for a period of time and start-up again. There are no security concerns with the strategy. Any sensitive information in flight between the appliance and the audit server was encrypted on the wire, and any information in memory is destroyed as soon as the exporter exits, which happens as quickly as a disconnected connection is detected. Upon recovery there is no possibility for any data to be observed in clear text coming from

the appliance. The appliance will attempt to authenticate with the key in the same way after an outage recovery as it does normally.

11.3 Auditable Events

Table 5 – Auditable Events

Requirement	Audit Event	Audit Content
FAU_GEN.1	Startup and shutdown of the audit function.	None.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session.	Reason for failure.
FCS_SSHC_EXT.1	Failure to establish an SSH session.	Reason for failure.
FCS_SSHS_EXT.1	Failure to establish an SSH session.	Reason for failure.
FCS_TLSS_EXT.1	Failure to establish a TLS Session.	Reason for failure.
FIA_AFL.1	Unsuccessful login attempt limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure).	
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
FTA_SSL_EXT.1 (if 'terminate the session' is selected)	The termination of a local session by the session locking mechanism.	
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	
FTA_SSL.4	The termination of an interactive session.	
NDcPP22e	Ability to configure the access banner	

Requirement	Audit Event	Audit Content
NDcPP22e	Ability to configure the session inactivity time before session termination or locking	
NDcPP22e	Ability to configure the authentication failure parameters for FIA_AFL.1	
NDcPP22e	Ability to configure audit behavior (e.g. changes to storage locations for audit; changes to behaviour when local audit storage space is full)	
NDcPP22e	Ability to enable or disable automatic checking for updates or automatic updates	
NDcPP22e	Ability to set the time which is used for time-stamps	
NDcPP22e	Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors	
	Ability to import X509v3 certificates to the TOE's trust store	

11.4 Sample Audit Events

The audit events will be written in batches, where every batch contains one hour of events. They are JSON-structured for ease of further processing and ingestion into other systems.

All events will contain a:

- Timestamp
- Source of the event
- Outcome, either *success* or *failure*
- Additional required details, and details to provide richer context

Depending upon the nature of the event and its origin, there may be more or less context available. These are examples of what the audit events will look like when the contents are examined after export.

FAU_GEN.1 Startup and shutdown of the audit function.

Startup

info Oct 14 15:56:00 AP200 Corelight Sensor: Startup of audit functions | details: (timestamp=2024-10-14T15:56:00Z)

Shutdown

info Oct 14 20:29:12 AP200 Corelight Sensor: Shutdown of audit functions | details: (timestamp=2024-10-14T20:29:12Z)

FCS_HTTPS_EXT.1 Failure to establish an HTTPS Session.
FCS_TLSS_EXT.1 Failure to establish a TLS Session.


```
{
 "source": "nginx",
 "type": "auditlog",
 "level": "info",
 "key": "SslCipherFailure",
 "msg": "Client 192.168.144.254 used unsupprted SSL cipher",
 "audit_event": {
  "outcome": "failure",
  "remote_host": "192.168.144.254",
  "host": "0.0.0.0",
  "port": "443",
  "timestamp": "2024-09-04T20:20:52.462440Z"
 },
 "time": 1725481252.4716802,
 "uid": "c8-48-0e-b2-1a-1f-83-5b",
 "ip": "192.168.144.240",
```

```
  "mac": "ac:1f:6b:47:34:f0",
  "telemetry-engine": "broalad"
}
```


FCS_SSHC_EXT.1 Failure to establish an SSH Session.


```
{
  "audit_event": {
    "outcome": "failure",
    "port": "22",
    "target": "192.168.144.254",
    "timestamp": "2024-10-09T16:41:21.091877Z",
    "user": "sftpuser"
  },
  "level": "error",
  "msg": "Error connecting to SSH server as sftpuser connecting to 192.168.144.254 port 22; kex error : no match for method encryption client->server: server [aes192-cbc], client [aes128-cbc,aes256-cbc,aes128-ctr,aes256-ctr]",
  "rid": "95ed9f739f",
  "source": "(corelight-sensor/bsftp-log-exporter/bsftp-log-exporter)",
  "telemetry-engine": "sensor-api",
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


FCS_SSHS_EXT.1 Failure to establish an SSH Session.


```
{
  "audit_event": {
    "outcome": "failure",
    "port": "57984",
    "proposed_cipher": "aes128-cbc",
    "remote_host": "192.168.144.254",
    "timestamp": "2024-09-11T14:36:46.774906Z"
  },
  "brolin": "brolin-dev/1.4461",
  "ip": "192.168.144.240",
  "key": "SSHAuthFail",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Client used invalid cipher",
  "rid": "6964a98715",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-7ac8403d",
  "source": "sshd",
  "telemetry-engine": "sensor-api",
  "time": 1726065406.7845654,
  "type": "auditlog",
```

```
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


FIA_AFL.1 Unsuccessful login attempt limit is met or exceeded.

Web UI


```
{
  "audit_event": {
    "host": "192.168.144.240",
    "method": "POST",
    "outcome": "failure",
    "path": "/fleet/v1/login",
    "referer": "https://192.168.144.240/login?nextPage=%2Fhome",
    "remote_host": "192.168.144.253",
    "timestamp": "2024-10-28T14:02:39.488794173Z",
    "user": "testgssuser",
    "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/129.0.0.0 Safari/537.36"
  },
  "brolin": "brolin-dev/1.4515",
  "ip": "192.168.144.240",
  "key": "UserMaxAuthenticationAttemptsFailed",
  "level": "error",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Maximum authentication attempts exceeded - Account has been locked",
  "rid": "e3809d3780",
  "seeded": "20240209140800",
  "sensor-ng-version": "27.13.3~dev.1497373135.git80dca2c1",
  "source": "fleetd",
  "telemetry-engine": "sensor-api",
  "time": 1730124159.726753,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


SSH


```
{
  "audit_event": {
    "outcome": "failure",
    "port": "50718",
    "remote_host": "192.168.144.253",
    "timestamp": "2024-10-28T14:25:33.141488Z",
    "user": "admin"
  },
  "brolin": "brolin-dev/1.4515",
  "ip": "192.168.144.240",
  "key": "SSHAuthFail",
```

```
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Maximum authentication attempts exceeded - Account has been locked",
  "rid": "a0ae45dd4b",
  "seeded": "20240209140800",
  "sensor-ng-version": "27.13.3~dev.1497373135.git80dca2c1",
  "source": "sshd",
  "telemetry-engine": "sensor-api",
  "time": 1730125533.150087,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


FIA_UIA_EXT.1 All use of identification and authentication mechanism.

Console Login Success:


```
{
  "audit_event": {
    "login_source": "console-ui",
    "outcome": "success",
    "timestamp": "2024-08-29T14:08:38.341444Z",
    "user": "admin"
  },
  "brolin": "brolin-dev/1.4432",
  "ip": "192.168.144.240",
  "key": "Login",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "user admin has logged in through console-ui",
  "rid": "4e45ab6ec6",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-d9c49628",
  "source": "broala-at-login",
  "telemetry-engine": "sensor-api",
  "time": 1724940518.3586798,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


Console Login Failure:


```
{
  "audit_event": {
    "outcome": "failure",
    "timestamp": "2024-08-29T14:09:03.099663Z",
    "user": "admin"
  },
```

```
 "brolin": "brolin-dev/1.4432",
 "ip": "192.168.144.240",
 "key": "FailedConsoleLogin",
 "level": "error",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "failed login attempt with invalid password by user admin on console",
 "rid": "121490df23",
 "seeded": "20240209140800",
 "sensor-ng-version": "main-d9c49628",
 "source": "UI",
 "telemetry-engine": "sensor-api",
 "time": 1724940542.974391,
 "type": "auditlog",
 "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


WebUI Login Success:


```
{
 "audit_event": {
  "body": {
   "method": "local",
   "password": "**REDACTED**",
   "username": "admin"
  },
  "host": "192.168.144.240",
  "method": "POST",
  "outcome": "success",
  "path": "/fleet/v1/login",
  "referer": "https://192.168.144.240/login?nextPage=%2Fsensor%2Flocal%2Fadvanced",
  "remote_host": "192.168.144.253",
  "status": 200,
  "timestamp": "2024-08-29T16:09:19.016259682Z",
  "user": "admin",
  "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/127.0.0.0 Safari/537.36"
 },
 "brolin": "brolin-dev/1.4447",
 "ip": "192.168.144.240",
 "key": "UserLoginSucceeded",
 "level": "info",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "User login succeeded.",
 "rid": "7e6f3103dc",
 "seeded": "20240209140800",
 "sensor-ng-version": "main-d3e84d46",
 "source": "fleetd",
```

```
  "telemetry-engine": "sensor-api",
  "time": 1724947759.4298465,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


WebUI Login Failure:


```
{
  "audit_event": {
    "body": {
      "method": "local",
      "password": "**REDACTED**",
      "username": "admin"
    },
    "host": "192.168.144.240",
    "method": "POST",
    "outcome": "failure",
    "path": "/fleet/v1/login",
    "referer": "https://192.168.144.240/login?nextPage=%2Fhome",
    "remote_host": "192.168.144.253",
    "status": 401,
    "timestamp": "2024-08-27T15:21:55.061461622Z",
    "user": "admin",
    "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/127.0.0.0 Safari/537.36"
  },
  "brolin": "brolin-dev/1.4432",
  "ip": "192.168.144.240",
  "key": "UserLoginFailed",
  "level": "error",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "User login failed.",
  "rid": "284e476a39",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-d9c49628",
  "source": "fleetd",
  "telemetry-engine": "sensor-api",
  "time": 1724772115.3330107,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


SSH Login Success:


```
{
  "audit_event": {
    "outcome": "success",
    "port": "40738",
```

```
   "remote_host": "192.168.144.254",
   "timestamp": "2024-07-02T16:56:14.307917Z",
   "user": "diag-shell"
  },
  "brolin": "brolin-dev/1.4275",
  "ip": "192.168.144.240",
  "key": "SSHAuthSuccess",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Established new connection between 192.168.144.254 and sensor",
  "rid": "9c5f60ab29",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-54621c30",
  "source": "sshd",
  "telemetry-engine": "sensor-api",
  "time": 1719939374.309335,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


SSH Login Failure:


```
{
  "audit_event": {
    "outcome": "failure",
    "port": "40738",
    "remote_host": "192.168.144.254",
    "timestamp": "2024-07-02T16:56:10.913103Z",
    "user": "diag-shell"
  },
  "brolin": "brolin-dev/1.4275",
  "ip": "192.168.144.240",
  "key": "SSHAuthFail",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Login was attempted with incorrect password",
  "rid": "a13d7ce79a",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-54621c30",
  "source": "sshd",
  "telemetry-engine": "sensor-api",
  "time": 1719939370.915773,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


FPT_TUD_EXT.1 Initiation of update; result of the update attempt (success or failure).

Initiation:


```
{
  "audit_event": {
    "method": "POST",
    "outcome": "success",
    "params": {
      "dry-run": false,
      "repository": null,
      "skip-apply-config": false,
      "skip-reboot": false
    },
    "path": "/api/updates/actions/apply",
    "remote_host": "192.168.144.253",
    "status": 200,
    "timestamp": "2024-08-29T14:50:13.806972Z",
    "user": "admin"
  },
  "brolin": "brolin-dev/1.4432",
  "ip": "192.168.144.240",
  "key": "APIRequestSucceeded",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "requested application of updates on sensor",
  "rid": "6d406be6ae",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-d9c49628",
  "source": "broala-apid",
  "telemetry-engine": "sensor-api",
  "time": 1724943013.999239,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


Success:


```
{
  "audit_event": {
    "method": "POST",
    "outcome": "success",
    "params": {
      "dry-run": false,
      "repository": null,
      "skip-apply-config": false,
      "skip-reboot": false
    },
    "path": "/api/updates/actions/apply",
    "remote_host": "192.168.144.253",
```

```
   "status": 200,
   "timestamp": "2024-08-29T14:50:13.806972Z",
   "user": "admin"
  },
  "brolin": "brolin-dev/1.4432",
  "ip": "192.168.144.240",
  "key": "APIRequestSucceeded",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "requested application of updates on sensor",
  "rid": "6d406be6ae",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-d9c49628",
  "source": "broala-apid",
  "telemetry-engine": "sensor-api",
  "time": 1724943013.999239,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


Failure:


```
{
  "audit_event": {
    "outcome": "failure",
    "reason": "request too recent relative to last update request for this repository",
    "repository": "brolin-dev",
    "timestamp": "2024-08-29T14:56:12.805687Z",
    "user": "admin"
  },
  "brolin": "brolin-dev/1.4447",
  "ip": "192.168.144.240",
  "key": "APIRequestFailed",
  "level": "error",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "previously requested sensor update skipped",
  "rid": "c7399c6945",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-d3e84d46",
  "source": "broala-apid",
  "telemetry-engine": "sensor-api",
  "time": 1724943372.8097475,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


FPT_STM_EXT.1 Discontinuous changes to time - either Administrator actuated or changed via an automated process.

```
{
   "audit_event": {
   "delta": -40.055371,
   "method": "PUT",
   "new_time": "2024-08-24T06:58:50+04:00",
   "original_time": "2024-08-24T06:59:30.055371+04:00",
   "outcome": "success",
   "params": {
     "ascii": null,
     "dry-run": false,
     "unix": 1724468330
   },
   "path": "/api/system/status/clock",
   "remote_host": "127.0.0.1",
   "status": 200,
   "timestamp": "2024-08-24T02:58:50.517487Z",
   "user": "admin"
 },
 "brolin": "brolin-dev/1.4432",
 "ip": "192.168.144.240",
 "key": "APIRequestSucceeded",
 "level": "info",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "requested change to current local time",
 "rid": "db45335f33",
 "seeded": "20240209140800",
 "sensor-ng-version": "main-d9c49628",
 "source": "broala-apid",
 "telemetry-engine": "sensor-api",
 "time": 1724468330.5288022,
 "type": "auditlog",
 "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


FTA_SSL_EXT.1 The termination of a local session by the session locking mechanism if “terminate the session” is selected.


```
{
  "audit_event": {
    "outcome": "success",
    "remote_host": "console",
    "timestamp": "2024-08-24T03:44:48.529832Z",
    "user": "admin"
  },
  "brolin": "brolin-dev/1.4432",
  "ip": "192.168.144.240",
  "key": "UIIdleSessionTimeout",
  "level": "info",
```

```
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "idle session timeout",
  "rid": "f3c04b3b2d",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-d9c49628",
  "source": "UI",
  "telemetry-engine": "sensor-api",
  "time": 1724471088.7151582,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


FTA_SSL.3 The termination of a remote session by the session locking mechanism.

WebUI Session Lock:


```
{
  "audit_event": {
    "outcome": "success",
    "timestamp": "2024-08-23T00:11:45.801377752Z",
    "user": "FLEETD"
  },
  "brolin": "brolin-dev/1.4432",
  "ip": "192.168.144.240",
  "key": "UserSessionExpireSucceeded",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Session '**REDACTED**' for user 'admin' was evicted.",
  "rid": "4ab74ab88b",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-d9c49628",
  "source": "fleetd",
  "telemetry-engine": "sensor-api",
  "time": 1724371906.243164,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


FTA_SSL.4 The termination of an interactive session.

WebUI Logout:


```
{
  "audit_event": {
    "outcome": "success",
    "port": "53202",
    "remote_host": "192.168.144.254",
    "timestamp": "2024-08-29T16:00:17.839951Z",
    "user": "diag-shell"
```

```
 },
 "brolin": "brolin-dev/1.4447",
 "ip": "192.168.144.240",
 "key": "SSHAuthSuccess",
 "level": "info",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "Closed connection between 192.168.144.254 and sensor",
 "rid": "914c0d9066",
 "seeded": "20240209140800",
 "sensor-ng-version": "main-d3e84d46",
 "source": "sshd",
 "telemetry-engine": "sensor-api",
 "time": 1724947217.8579757,
 "type": "auditlog",
 "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


Console Logout:


```
{
 "audit_event": {
  "outcome": "success",
  "remote_host": "console",
  "timestamp": "2024-08-23T21:33:04.754729Z",
  "user": "admin"
 },
 "brolin": "brolin-dev/1.4432",
 "ip": "192.168.144.240",
 "key": "UIExiting",
 "level": "info",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "exiting UI",
 "rid": "ca5843496f",
 "seeded": "20240209140800",
 "sensor-ng-version": "main-d9c49628",
 "source": "UI",
 "telemetry-engine": "sensor-api",
 "time": 1724448784.9892561,
 "type": "auditlog",
 "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


SSH Logout:


```
{
 "audit_event": {
  "diag-shell": true,
  "outcome": "success",
```

```
   "port": "40740",
   "remote_host": "192.168.144.254",
   "timestamp": "2024-07-02T16:56:51.020138Z",
   "user": "diag-shell"
  },
  "brolin": "brolin-dev/1.4275",
  "ip": "192.168.144.240",
  "key": "SSHAuthSuccessDiagShell",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Exit from SSH diagnostic shell UI",
  "rid": "3213e3dc56",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-54621c30",
  "source": "sshd",
  "telemetry-engine": "sensor-api",
  "time": 1719939411.0215728,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


NDcPP22e Ability to configure the access banner


```
{
  "audit_event": {
    "body": {
      "banner_title": "GSS Test Banner Test"
    },
    "host": "192.168.144.240",
    "method": "PATCH",
    "outcome": "success",
    "path": "/fleet/v1/settings",
    "referer": "https://192.168.144.240/admin/access",
    "remote_host": "192.168.144.253",
    "status": 200,
    "timestamp": "2024-09-10T20:11:40.678385104Z",
    "user": "admin",
    "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/127.0.0.0 Safari/537.36"
  },
  "brolin": "brolin-dev/1.4461",
  "ip": "192.168.144.240",
  "key": "GlobalSettingsUpdateSucceeded",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Global settings update succeeded.",
  "rid": "7de1529896",
  "seeded": "20240209140800",
```

```
  "sensor-ng-version": "main-7ac8403d",
  "source": "fleetd",
  "telemetry-engine": "sensor-api",
  "time": 1725999100.9283147,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


NDcPP22e Ability to configure the session inactivity time before session termination or locking


```
{
  "audit_event": {
    "body": {
      "security.auto_logout.timeout": 5
    },
    "host": "192.168.144.240",
    "method": "PATCH",
    "outcome": "success",
    "path": "/fleet/v1/settings",
    "referer": "https://192.168.144.240/admin/access",
    "remote_host": "192.168.144.253",
    "status": 200,
    "timestamp": "2024-09-10T20:12:31.556454336Z",
    "user": "admin",
    "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/127.0.0.0 Safari/537.36"
  },
  "brolin": "brolin-dev/1.4461",
  "ip": "192.168.144.240",
  "key": "GlobalSettingsUpdateSucceeded",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Global settings update succeeded.",
  "rid": "b6741eeca7",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-7ac8403d",
  "source": "fleetd",
  "telemetry-engine": "sensor-api",
  "time": 1725999151.7145164,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


NDcPP22e Ability to configure the authentication failure parameters for FIA_AFL.1


```
{
  "audit_event": {
    "body": {
      "security.brute_force.max_attempts": 5
```

```
  },
  "host": "192.168.144.240",
  "method": "PATCH",
  "outcome": "success",
  "path": "/fleet/v1/settings",
  "referer": "https://192.168.144.240/admin/access",
  "remote_host": "192.168.144.253",
  "status": 200,
  "timestamp": "2024-09-10T20:13:28.111775159Z",
  "user": "admin",
  "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/127.0.0.0 Safari/537.36"
 },
 "brolin": "brolin-dev/1.4461",
 "ip": "192.168.144.240",
 "key": "GlobalSettingsUpdateSucceeded",
 "level": "info",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "Global settings update succeeded.",
 "rid": "4cc90cf617",
 "seeded": "20240209140800",
 "sensor-ng-version": "main-7ac8403d",
 "source": "fleetd",
 "telemetry-engine": "sensor-api",
 "time": 1725999208.35307,
 "type": "auditlog",
 "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


NDcPP22e Ability to configure audit behavior (e.g. changes to storage locations for audit; changes to behavior when local audit storage space is full)


```
{
 "audit_event": {
  "body": {
   "exceptions.current": {
    "remote.enable": true,
    "updates.no_impact": true
   },
   "filter-exceptions.current": {},
   "group.id": 0,
   "packages.enabled.exceptions": {}
  },
  "host": "192.168.144.240",
  "method": "PUT",
  "outcome": "success",
  "path": "/fleet/v1/sensor/catalog/local",
  "referer": "https://192.168.144.240/sensor/local/maintain",
```

```
    "remote_host": "192.168.144.253",
    "status": 200,
    "timestamp": "2024-09-11T15:19:36.558360278Z",
    "user": "admin",
    "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/127.0.0.0 Safari/537.36"
  },
  "brolin": "brolin-dev/1.4461",
  "ip": "192.168.144.240",
  "key": "SensorCatalogUpdateSucceeded",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "Sensor catalog update succeeded.",
  "rid": "42e386ff6a",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-7ac8403d",
  "source": "fleetd",
  "telemetry-engine": "sensor-api",
  "time": 1726067976.8448648,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


NDcPP22e Ability to set the time which is used for time-stamps


```
{
  "audit_event": {
    "body": {
      "exceptions.current": {
        "remote.enable": true,
        "updates.no_impact": true
      },
      "filter-exceptions.current": {},
      "group.id": 0,
      "packages.enabled.exceptions": {}
    },
    "host": "192.168.144.240",
    "method": "PUT",
    "outcome": "success",
    "path": "/fleet/v1/sensor/catalog/local",
    "referer": "https://192.168.144.240/sensor/local/maintain",
    "remote_host": "192.168.144.253",
    "status": 200,
    "timestamp": "2024-09-11T15:19:36.558360278Z",
    "user": "admin",
    "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/127.0.0.0 Safari/537.36"
  },
```

```
 "brolin": "brolin-dev/1.4461",
 "ip": "192.168.144.240",
 "key": "SensorCatalogUpdateSucceeded",
 "level": "info",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "Sensor catalog update succeeded.",
 "rid": "42e386ff6a",
 "seeded": "20240209140800",
 "sensor-ng-version": "main-7ac8403d",
 "source": "fleetd",
 "telemetry-engine": "sensor-api",
 "time": 1726067976.8448648,
 "type": "auditlog",
 "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


NDcPP22e Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors


```
{
 "audit_event": {
  "fingerprint": "31:02:D0:DF:D7:89:6B:47:30:B0:0E:AF:11:2A:70:3D:AB:86:B9:90",
  "method": "PUT",
  "outcome": "success",
  "params": {},
  "path": "/api/ssl/webserver-cert/root-ca",
  "remote_host": "127.0.0.1",
  "status": 201,
  "subject": "/C=US/ST=MD/L=Catonsville/O=GSS/CN=rootca-rsa",
  "task": "Root certificate installation",
  "timestamp": "2024-03-14T20:34:09.409057Z",
  "used_by": "webserver",
  "user": "admin"
 },
 "brolin": "brolin-dev/1.4120",
 "ip": "192.168.144.240",
 "key": "APIRequestSucceeded",
 "level": "info",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "installation of root certificate",
 "rid": "04d7137fa3",
 "seeded": "20240209140800",
 "sensor-ng-version": "main-9331b236",
 "source": "broala-apid",
 "telemetry-engine": "sensor-api",
 "time": 1710448449.4111297,
 "type": "auditlog",
 "uid": "c8-48-0e-b2-1a-1f-83-5b"
```

}

Import correctly signed certificate:


```
{
  "audit_event": {
    "certificate_id": "14f0e98031",
    "fingerprint": "7f:83:c1:d4:c2:19:d2:af:ac:2a:ef:ff:f8:00:c4:4c:9a:88:5d:97",
    "method": "put",
    "outcome": "success",
    "params": {},
    "path": "/api/ssl/webserver-cert/certificate",
    "remote_host": "127.0.0.1",
    "subject":
"/c=us/st=md/l=columbia/o=corelight/ou=engineering/cn=ap200/emailaddress=ap200@email.com",
    "task": "webserver certificate installation",
    "timestamp": "2024-03-14T20:34:27.160572Z",
    "used_by": "webserver",
    "user": "admin"
  },
  "brolin": "brolin-dev/1.4120",
  "ip": "192.168.144.240",
  "key": "APIRequestSucceeded",
  "level": "info",
  "mac": "ac:1f:6b:47:34:f0",
  "msg": "requested to install webserver signed cert",
  "rid": "6f7548c9e8",
  "seeded": "20240209140800",
  "sensor-ng-version": "main-9331b236",
  "source": "broala-apid",
  "telemetry-engine": "sensor-api",
  "time": 1710448467.169035,
  "type": "auditlog",
  "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


Attempt to import incorrectly signed certificate:


```
{
  "audit_event": {
    "method": "PUT",
    "outcome": "failure",
    "params": {},
    "path": "/api/ssl/webserver-cert/certificate",
    "reason": "Private key does not match certificate.",
    "remote_host": "127.0.0.1",
    "status": 400,
    "timestamp": "2024-10-28T14:52:03.584861Z",
    "user": "admin"
```

```
 },
 "brolin": "brolin-dev/1.4515",
 "ip": "192.168.144.240",
 "key": "APIRequestFailed",
 "level": "error",
 "mac": "ac:1f:6b:47:34:f0",
 "msg": "requested to install webserver signed cert",
 "rid": "8d1966710f",
 "seeded": "20240209140800",
 "sensor-ng-version": "27.13.3~dev.1497373135.git80dca2c1",
 "source": "broala-apid",
 "telemetry-engine": "sensor-api",
 "time": 1730127123.593879,
 "type": "auditlog",
 "uid": "c8-48-0e-b2-1a-1f-83-5b"
}
```


11.5 Role Based Access Control (RBAC)

The TOE implements Role Based Access Control (RBAC). Administrative users are required to authenticate before being granted access to any administrative functions. The TOE restricts the ability to manage the TOE to Security Administrators, otherwise referred to as the Administrator role.
The TOE maintains the following roles: Administrator, Network and Monitor. Each role defined has a set of permissions that will grant them access to the TOE data. Security functions and data are restricted to the Administrator.

For the purposes of the Common Criteria certification, only the Administrator role is supported with a single admin account.

Table 2 – Roles and Permissions

Roles	**Permissions**
Administrator (admin)	Can configure user accounts and manage users and their associated privileges.
	Ability to administer the TOE locally and remotely
	Ability to configure the access banner
	Ability to configure the session inactivity time before session termination or locking

	Ability to configure account lockout
	Ability to update the TOE, and to verify the updates using digital signatures capability prior to installing those updates
	Ability to configure the authentication failure parameters
	Ability to configure audit behavior
	Ability to set the time which is used for timestamps
	Ability to configure the reference identifier for the peer
Network Administrator (netconfig)	Ability to change network settings of the TOE locally and remotely
	Can change their own password, but not other users' passwords
User (monitor)	Able to carry out system monitoring and gather information about the configuration and performance of the system.
	Can change their own password, but not other users' passwords

12 Accessing the appliance to make configuration changes

All configuration changes here assume that the administrator has previously launched the diagnostic shell (diag-shell) via the textual UI on the sensor, or by establishing a connection via **ssh** or has connected to the Web UI of the sensor. No functionality aside from login prompts and screens are available until after authentication on all supported interfaces.

12.1 Local Console

It is not necessary to enable the local console. It is automatically enabled out of the box. To connect to the sensor via the local console, the keyboard and monitor must be connected first. A warning banner appears for the administrator to ensure the interface is local, press **Enter** login prompt will normally be seen on the console. Login as the admin user and after successfully authenticating, select **Maintain** from the left-hand side menu, then hit the **right arrow key** on the keyboard and select **Enter Diagnostic Shell**. You are now ready to configure the sensor.

Example

12.2 Remote SSH

To connect to the sensor remotely, you will be required to use the SSH protocol, and therefore will need some SSH client on the management workstation. In order to enable SSH access to the sensor it is necessary to connect via the console in order to configure required settings. This step requires the keyboard and monitor to be connected to the sensor. After successfully authenticating, select **Maintain** from the left-hand side menu, then hit the **right arrow key** on the keyboard and select

Enter Diagnostic Shell. You are now ready to configure the sensor. The following command is going to enable remote access via ssh.

corelight-client configuration update --security.ssh.enable=True

It is likewise possible to enable remote access via SSH directly from the textual interface without entering the diagnostic shell. After successfully authenticating, select **Access** from the left-hand side menu, then hit the **right arrow key** on the keyboard and scroll down to the checkbox labeled **Enable access via SSH**. Check the previously unchecked box by hitting the spacebar key once. Hit the **left arrow key** once to return to the main menu. Then, scroll down and select either **Save configuration** or **Save and exit**.

If the admin user's ssh public key was previously configured on the sensor, and the administrator possesses the matching private key, authentication via a key-exchange will simply work.

To establish a management connection to the sensor, use the ssh client utility such as the OpenSSH client simply named ssh or a utility with a graphical interface such as PuTTY. The following syntax

is appropriate for the OpenSSH version of the ssh client. Replace the <address> placeholder with the IP address or hostname of the sensor: ssh admin@<address>. Adjust this accordingly for whatever ssh client utility you are using. After successfully authenticating, select **Maintain** from the left-hand side menu, then hit the **right arrow key** on the keyboard and select **Enter Diagnostic Shell**. You are now ready to configure the sensor.

12.3 Session Termination

To terminate a local or remote session from the diagnostic shell:

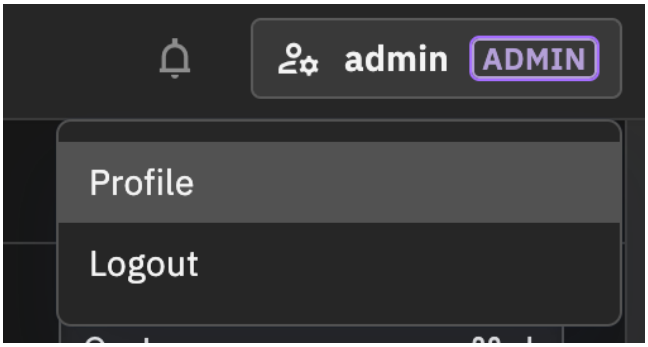
- type exit or exit diag and hit the Return/Enter key.

To terminate local or remote session from the textual user interface, select from two options in the left-hand navigation with the arrow keys, and hit the Return/Enter key:

- **Save and exit**
- **Exit without saving**

To terminate a Web UI session, click on the user name in the upper right hand corner, then click *Logout*

from the drop-down menu as illustrated below.



12.4 FIPS mode requirement

FIPS compliance is a prerequisite for the Common Criteria certification. The sensor must be configured to operate in FIPS mode, which among other things imposes limitations on availability of algorithms from the cryptographic module and by extension constrains those components which depend on it, imposes mandatory cryptographic module initialization health testing and enables protections which go beyond the normal security mechanisms such as keeping sensitive details protected in memory and securely wiping memory after it has been freed.
corelight-config fips enable

The system will validate that it is FIPS-capable and then enable FIPS mode and reboot.

12.5 Enabling Common Criteria mode

It is necessary to enable Common Criteria mode in order to make adjustments to allowable algorithms as well as other security parameters which put the system into a state that is compliant with the certification. When the Common Criteria mode is enabled, a limited set of algorithms will be enforced by the system. This is entirely governed by the appliance and there are no external controls. Only those algorithms claimed as part of the certification will be available.

corelight-client configuration update --mode.common_criteria.enable=True

13 Configuring the system clock

The Corelight appliance allows for the system clock to be set manually. It is possible to keep the clock synchronized with NTP. If the system clock is very inaccurate, it is possible to set the clock manually, and then make sure that NTP is configured, which will continue to maintain accurate time. Note that NTP is not evaluated.

13.1 Setting the clock manually

It is possible to manually set the clock, but NTP is strongly recommended. To set the clock manually, using the following command, replacing <time now> with a UNIX epoch time value.

corelight-client clock update --help --unix <time now>

You may want to change the configured time zone to **Etc/GMT+0** in order to have universal coordinated time across your infrastructure. Perform the following to change the timezone to UTC.

corelight-client configuration update --system.timezone=Etc/GMT+0

14 Enabling Audit Log Export

14.1 Command Line Enablement of Audit Log Export

This is necessary to enable export of audit records as batches using the same mechanism as that used for Zeek logs.

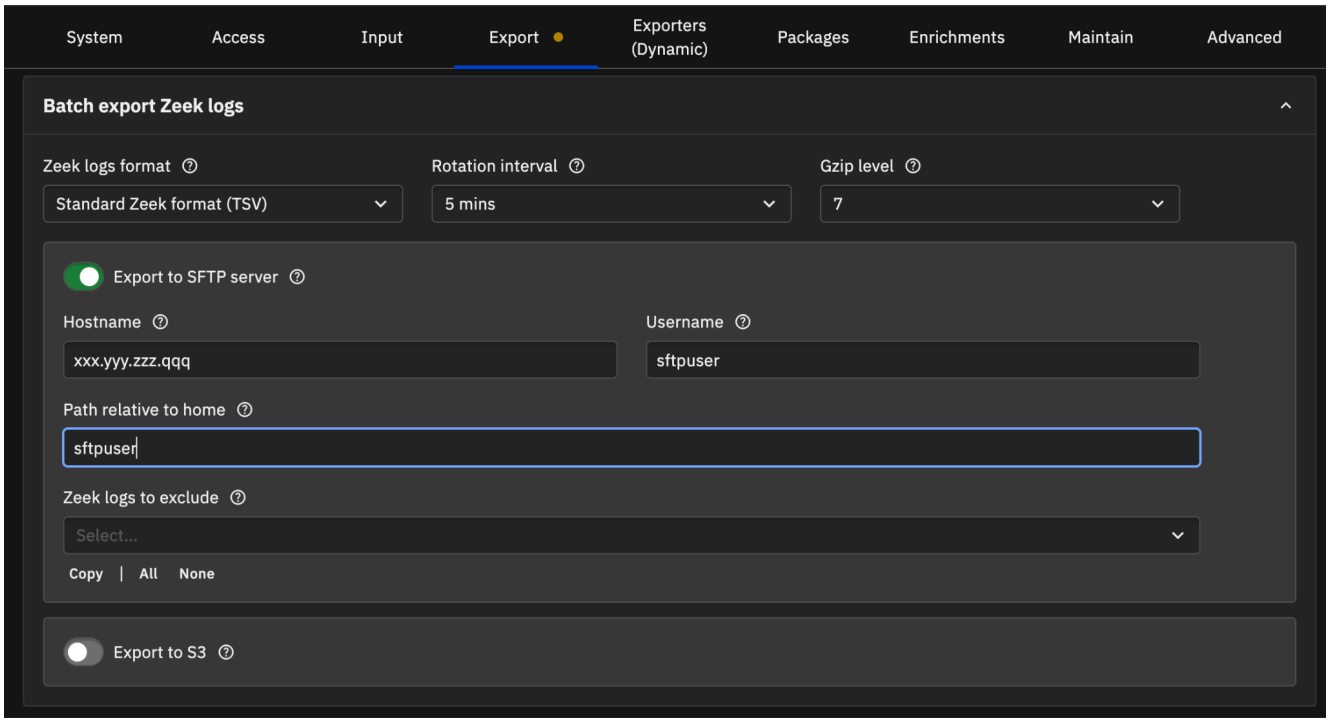
```
corelight-client configuration update \
        --security.auditlog_export.enable=True
```

In order for the records to be exported, the SFTP exporter must be configured and working correctly. To enable the exporter perform the following steps. Replace <remote path> with the correct target path on the SFTP server. The sensor must be able to create directories and files in this path. Replace <destination address> with the IP address or resolvable hostname of the SFTP server. Replace <username> with the username with which the sensor should be authenticated.

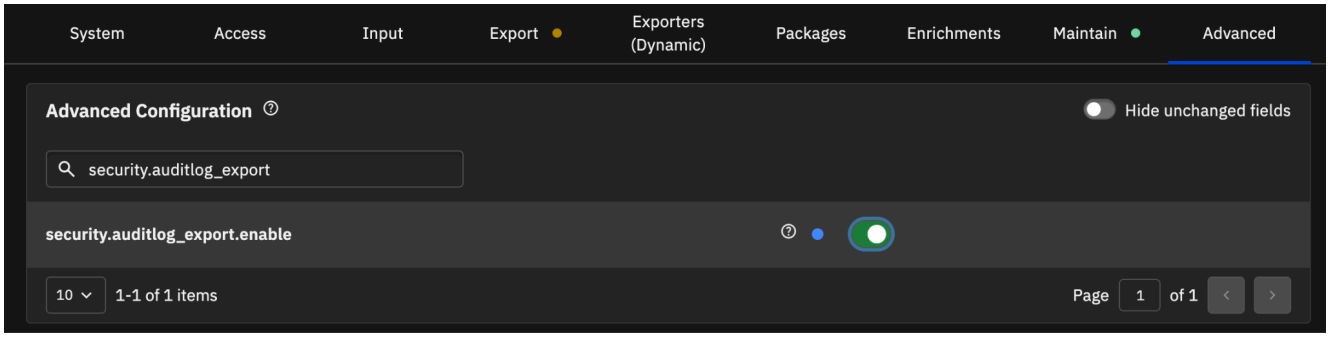
```
corelight-client configuration update --bro.export.logs.enable=True
corelight-client configuration update \
        --bro.export.sftp.log.path=<remote path> \
        --bro.export.sftp.log.server=<destination address> \
        --bro.export.sftp.log.user=<username>
```

14.2 Web UI Enablement of Audit Log Export

From the *Configuration* section select the *Export* tab, enable *Export to SFTP server* and fill in the SFTP exporter parameters as shown below.



Select the *Advanced* tab and toggle-on the *security.auditlog_export.enable* option.



14.3 SFTP Authentication

The Corelight appliance does not allow the administrator to provide a private SSH key for authentication with the remote SFTP server, but it does allow an administrator to generate a new key and set its type. The default type is ED25519, which is not allowed when FIPS is enabled, so in order to be Common Criteria compliant, an administrator will have to generate a key that uses either RSA or ECDSA encryption. Then the administrator has to retrieve the key from the sensor, which they will then have to authorize on the SFP server. The details around this are discussed in a later section.

In order to generate a new key, the administrator can use the command

corelight-client keys exporter generate –type <type>

where type is either “rsa” or “ecdsa”.

The SFTP batch exporter only supports public key exchange for authentication, thus it is necessary for the SFTP server to support public key authentication, and it must be configured correctly for the user specified in the previous step. The Corelight sensor does not allow the user to provide keys for key exchanges between it and the SFTP server, instead it generates its own keypair. In order to authorize the sensor to access the remote SFTP destination, the public key part of the generated key pair will be required on the remote end. The public key is obtained from the sensor with the following command.

corelight-client keys exporter get

This public key which must be added to the authorized_keys file or its equivalent, associated with the remote user with which the sensor is going to be authenticating. Once the public key is configured on the SFTP server, establishing a trust, the sensor will create a connection automatically.

14.3.1 Supported ciphers in Common Criteria mode

- aes128-cbc
- aes256-cbc
- aes128-ctr
- aes256-ctr

14.3.2 Supported public key types in Common Criteria mode

- ecdsa-sha2-nistp256
- ecdsa-sha2-nistp384

- ecdsa-sha2-nistp521
- rsa-sha2-256
- rsa-sha2-512

14.3.3 Supported host key types in Common Criteria mode

- ecdsa-sha2-nistp256
- ecdsa-sha2-nistp384
- ecdsa-sha2-nistp521
- rsa-sha2-256
- rsa-sha2-512

14.3.4 Supported MAC algorithms in Common Criteria mode

- hmac-sha2-256
- hmac-sha2-512

14.35 Supported kex algorithms in Common Criteria mode

- diffie-hellman-group14-sha1
- ecdh-sha2-nistp256

14.3.6 Rekeying

SFTP exporter connections to the sensor will rekey after processing increments of 1GB of data or at intervals of 1 hour.

14.4 Disabling Corelight Cloud Service connectivity

Disabling remote connectivity will mean that updates must be performed via an offline updater mechanism. Corelight's Support and Customer Success teams will be able to assist with this.

```
corelight-client configuration update \
        --remote.enable=False \
        --remote.download.license=False
```

15 Key-based Authentication with SSH

As already mentioned the administration of the sensor is limited to a single user. This account is named **admin**, and this default administrative account ships with the sensor Cryptographic

functions are restricted to security administrators (admin). While the sensor is not limited to a single user, our Common Criteria compliance implementation choice is enforcing this.

15.1 Using key-based authentication

It is possible to switch off password-based authentication on the sensor and enable public key based authentication, which eliminates the need for passing the administrator's password between the remote ssh client and sensor. Public key authentication is a more secure authentication mechanism relative to password based, and will not be subject to account locking. The public key string should be taken verbatim from the public key file, most commonly generated via the ssh keygen command. The sensor can only support keys compatible with OpenSSH. There are two formats that OpenSSH supports, a PEM format, which has been used historically and a more recent OpenSSH-specific format. Both formats are allowed. The sensor supports ED25519, RSA, and ECDSA keys, and defaults to using ED25519. ED25519 is not a valid key type when operating in FIPS mode.

If you are unsure about the format of the private key, or if this key is used with an ssh implementation other than OpenSSH, you should be able to tell if it is compatible by inspecting the first or last lines of the key. All compatible RSA keys will have one of the following two first lines.

-----BEGIN OPENSSH PRIVATE KEY-----
-----BEGIN RSA PRIVATE KEY-----

The last line of the key will have one of the following two lines.

-----END OPENSSH PRIVATE KEY-----
-----END RSA PRIVATE KEY-----

To generate a compliant RSA key pair, the following command may be used, on a trusted, and secure system. Because versions of the ssh-keygen command vary, and are affected by the version of the OpenSSH package and the operating system, it is important to consult the documentation for the specific version of the OpenSSH package on the system where the keys are actually generated. Always use the latest version of the OpenSSH package, and by extension latest version of the ssh keygen utility.

ssh-keygen -m PEM -t rsa -f id_rsa
It is possible to specify the size of the modulus, which is the equivalent of the key's size, and cryptographic strength. The only supported sizes are a 2048-bit and 3072-bit moduli. Replace the <modulus bits> in the following command with 2048 or 3072, which corresponds to a 2048-bit or 3072-bit modulus.

ssh-keygen -t rsa -b <modulus bits> id_rsa

This will generate a key with 3072-bit long modulus. The output of the above command should look similar to the following example. Note that -m PEM will result in the creation of the traditional PEM formatted private key. This may be necessary if the same key is going to be used from a system with older version of the ssh command, which does not understand the more recent OpenSSH format.

```
$ ssh-keygen -m PEM -t rsa -f ./id_rsa
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ./id_rsa
Your public key has been saved in ./id_rsa.pub
The key fingerprint is:
SHA256:l5vKJzgOcAaa/+rswFsKIcVYvISSaJQi1Z8c5BM2nfI
demouser@demo-machine.local The key's randomart image is:
+---[RSA 3072]----+
|o*+. .=. . |
|B*o .oooo |
|*.+. oo= |
| +.. +.E . |
|= . o S o |
|oo + . o |
|o.... . o |
|.o+. .o.... |
| ==.....oo |
+----[SHA256]-----+
```

It is also possible to generate a key pair in a similar fashion with the openssl command if for any reason ssh-keygen is not available or from a very old version of the OpenSSH package. Depending on the version of openssl, the method is going to vary. For this reason there is no example provided.

Elliptic Curve Cryptography based keys are generated in a similar fashion. Replace the <prime curve> in this command with your chosen prime curve, which will be one of 256, 384 or 521, corresponding to P256, P384, and P521 prime curves. Replace <curve size> with one of 256, 384 or 521 in the following command.

```
ssh-keygen -t ecdsa -b <curve size> -f id_ecdsa
```

The output of the above command should look similar to the following examples of generating Elliptic Curve key pairs with the three supported prime curve sizes: 256, 384 and 512 on a system with a recent version of OpenSSH.

```
$ ssh-keygen -t ecdsa -b 256 -f id_ecdsa
```

```
Generating public/private ecdsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_ecdsa
Your public key has been saved in id_ecdsa.pub
The key fingerprint is:
SHA256:4kCPAl/zCIzsYcx0Q0fFsRV9taMpu0FYIyVAByTrE6c demouser@demohost
The key's randomart image is:
+---[ECDSA 256]---+
| ..+.+=B+=o. .. |
|=o. o o.+ o. . .|
|o*o +o o . o. o |
|oo.+.*+ + . o .|
| .o +E+ S. o o |
| . o.. . o |
| . o |
| o |
| . |
+----[SHA256]-----+

$ ssh-keygen -t ecdsa -b 384 -f id_ecdsa
Generating public/private ecdsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_ecdsa
Your public key has been saved in id_ecdsa.pub
The key fingerprint is:
SHA256:i+fuzTJ2orEc72AEgmMn85caa7RpOGL0lj7CnLIxobM
demouser@demohost The key's randomart image is:
+---[ECDSA 384]---+
| |
| . |
|o+... |
|..=. .. |
|.. + o. S |
|o.+ O. . . |
|O+.@ * o |
|+X*. o O=o. |
|E.... +**=o |
+----[SHA256]-----+

ssh-keygen -t ecdsa -b 521 -f id_ecdsa
```

```
Generating public/private ecdsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_ecdsa
Your public key has been saved in id_ecdsa.pub
The key fingerprint is:
SHA256:T3DmPQO5G2Z5kG1kCMkxycUDiSFZ0ozXB+GooWSwf/s
demouser@demohost The key's randomart image is:
+---[ECDSA 521]---+
|. o*o*B%..o |
| o oo=oB.=* |
|. o ... o.B.o |
| + . o = B |
| o o S O = |
| . . = + o |
| . o |
| . |
| E |
+----[SHA256]-----+
```

The OpenSSH public key file will normally contain a single line, with two or three columns. The first two columns identify the type of key as well as provide the actual content of the public key. The third column is a comment and strictly optional. The most convenient way to extract the exact content that needs to be passed to the sensor, in order to set up key exchange is to do the following, where /path/to/ssh/private/key is the path to the private key. Frequently, this will be in your home directory, under the .ssh directory.

```
ssh-keygen -m PEM -y -f /path/to/ssh/private/key
```

Finally, pass the output of the previous command to the sensor with the following command. Be sure to quote the string as shown below to prevent token splitting by the shell.

```
corelight-client configuration update \
        --security.user.admin.ssh_public_key='<SSH public key string>'
```

To comply with the Common Criteria certification, RSA keys must have 2048-bit or 3072-bit modulus. If you are unsure about previously generated keys meeting this requirement, you can discover the modulus size of the key using one of the following methods. In the first example, the openssl command is used, and in the second ssh-keygen is used. In both cases the file name of the private key is id_rsa, and it is located in the same directory from which the command is executed. Notice that in this example the size is 1024-bit as reported by both commands, albeit with quite

different output formats. This is a smaller modulus than what is compliant with the Common Criteria certification. If using RSA keys, the recommended size is 3072. All reasonably modern versions of the OpenSSH package, and therefore the ssh-keygen tool will automatically choose to use 3072 with RSA keys.

```
$ ls id_rsa*
id_rsa id_rsa.pub

$ openssl rsa -text -noout -in id_rsa | head -1
RSA Private-Key: (1024 bit, 2 primes)

$ ssh-keygen -lf id_rsa
1024 SHA256:bZoaqqcZAdJuP1eeL4qhtdazxHUlmwPT2FCafxdHVaE demouser@demo
machine.local (RSA)
```

Private keys must be treated with the same care and attention as passwords. Generation and secure handling of private keys are a complex topic and beyond the scope of this document. Corelight's Support and Customer Success teams will be able to assist with this.

15.2 Sensor SSH host key types

The sensor has one SSH host key, and it supports host keys that use RSA, ECDSA, and ED25519 crypto algorithms. The default is ED25519, which is not allowed when operating in FIPS mode.

Before operating in FIPS mode, the administrator must generate a host key of type RSA or ECDSA, using the command:

corelight-client keys host generate –type <type>

where type is "rsa" or "ecdsa".

15.3 Supported kex algorithms in Common Criteria mode

- ecdh-sha2-nistp256
- ecdh-sha2-nistp384
- ecdh-sha2-nistp521
- diffie-hellman-group16-sha512
- diffie-hellman-group14-sha256

15.4 Supported ciphers in Common Criteria mode

- aes128-ctr
- aes256-ctr

15.5 Supported authentication key types supported in Common Criteria mode

- rsa
- ecdsa-sha2-nistp256

15.6 Supported MAC algorithms in Common Criteria mode

- hmac-sha2-256
- hmac-sha2-512

15.7 Rekeying

SSH connections to the sensor will rekey after processing increments of 1GB of data or at intervals of 1 hour.

16 Enabling Inactivity Timeout

The value for the inactivity timeout is in minutes, not seconds, please adjust accordingly if the existing policy is in seconds. This setting will cause a session to be disconnected after this many minutes of inactivity.

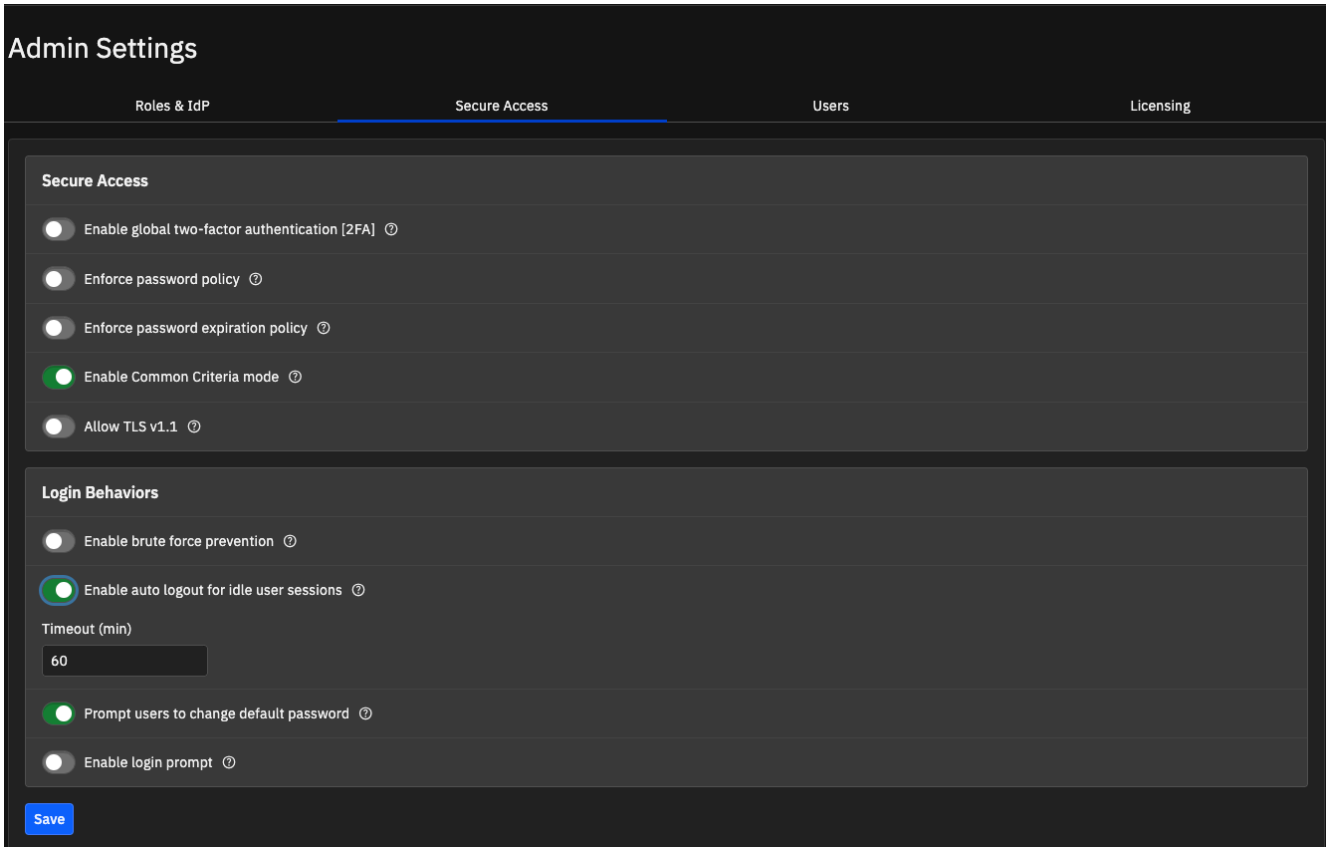
16.1 Enabling Inactivity Timeout from the CLI

corelight-client configuration update
    --security.auto_logout.enable=True \
    --security.auto_logout.timeout=<idle minutes>

The default inactivity time period is 60 minutes for both the CLI and SSH interfaces.

16.2 Enabling Inactivity Timeout from the Web UI

You can enable Inactivity Timeout from the Web UI with the *Enable auto logout for idle user sessions* setting on the *Secure Access* tab of the *Admin Settings* page, as illustrated below, and you can configure the timeout value in the *Timeout (min)* text box below it.



16.3 Enabling temporary account lockout for remote connections from the CLI

This mechanism is intended to lock out the **admin** user logging in remotely (ssh), for some period of time after a threshold of unsuccessful authentication attempts has been exceeded. Default value is 600 seconds. The admin user is never completely locked out. While the account may be locked out from authenticating remotely, it is still possible to login with the same account at the local console. The account is never locked if logging in locally, no matter how many consecutive unsuccessful authentication attempts have been made.

```
corelight-client configuration update \
        --authentication.local.lockout.enable=True
```

To adjust the time the account remains locked, once the threshold has been reached, change the value of authentication.local.lockout.unlock_after_secs. The range for this value is from 180 to

86400 seconds (24 hours).

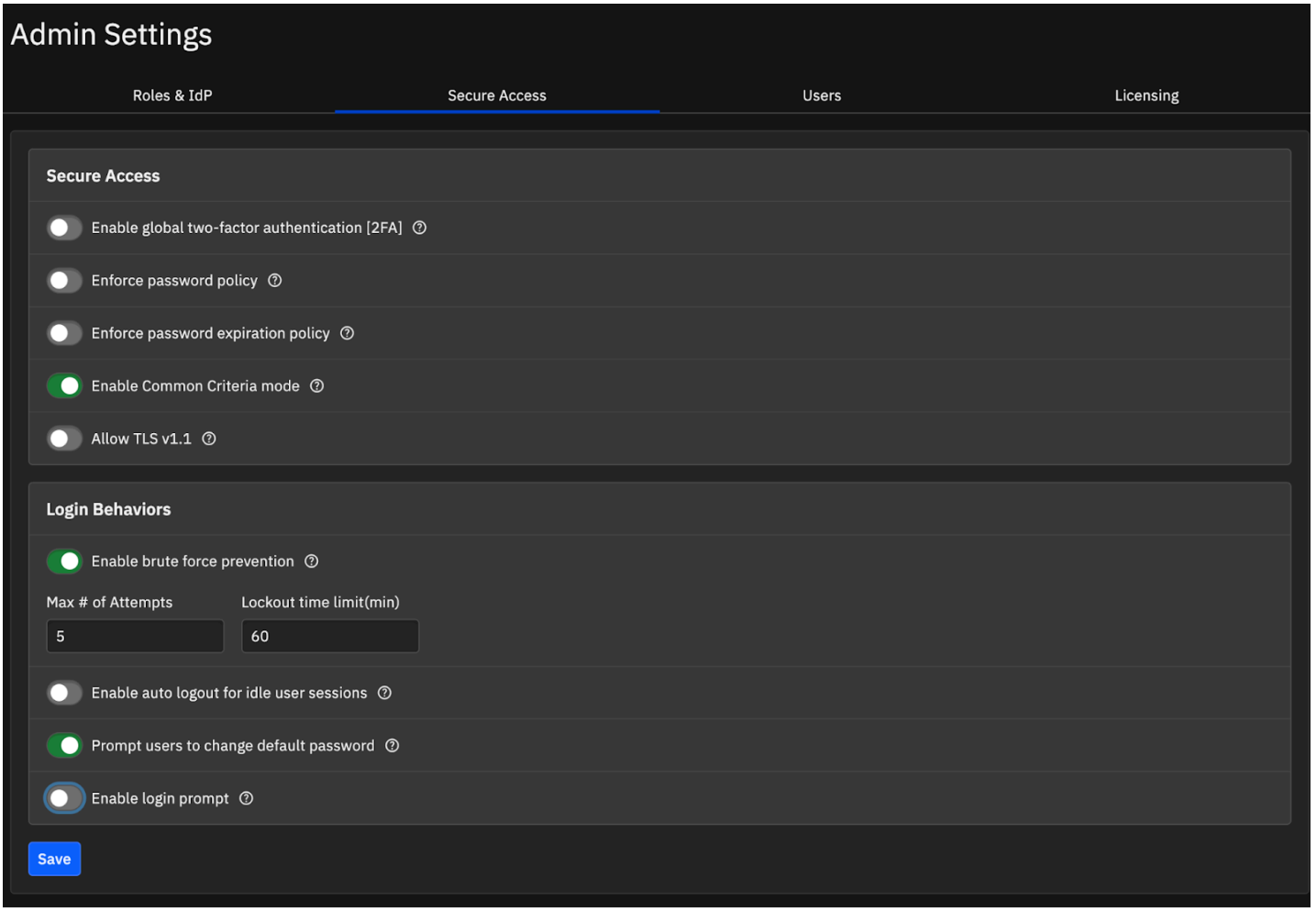
```
corelight-client configuration update \
        --authentication.local.lockout.unlock_after_secs=<lockout secs>
```

To adjust the number of consecutive unsuccessful authentication attempts, after which the account is locked out prohibiting further remote authentications (even with correct password, until the lockout time runs out), change the value of authentication.local.lockout.max_attempts. The range for this value is from 3 to 15 within 60 minutes.

```
corelight-client configuration update \
        --authentication.local.lockout.max_attempts=<number of allowed failures>
```

16.4 Enabling temporary account lockout for remote connections from the Web UI

You can enable temporary lockout from the Web UI with the *Enable brute force prevention* setting on the *Secure Access* tab of the *Admin Settings* page, as illustrated below.



17 Password Requirements

To prevent administrators from choosing insecure passwords, each password must meet the following requirements:

1. Minimum password length shall be configurable to between [8] and [64] characters. The default minimum password length is 8 characters.
2. Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, [“~”, “ ”, “””]];

In order to remain in compliance with the Common Criteria certification, the admin password must not be shorter than 8 characters. This setting is expected to be modified according to the security

requirements. The following command enables changing this default. Replace the string <chosen length> with the value which corresponds to your security policy.

```
corelight-client configuration update \
 --mode.password.strict=1 \
 --strict_pw.min_length.enable=1 \
 --strict_pw.min_length=<chosen length>
```

It is not possible to adjust this minimum length above 64. Anything above 64 will result in the command failing.

18 Login Banner

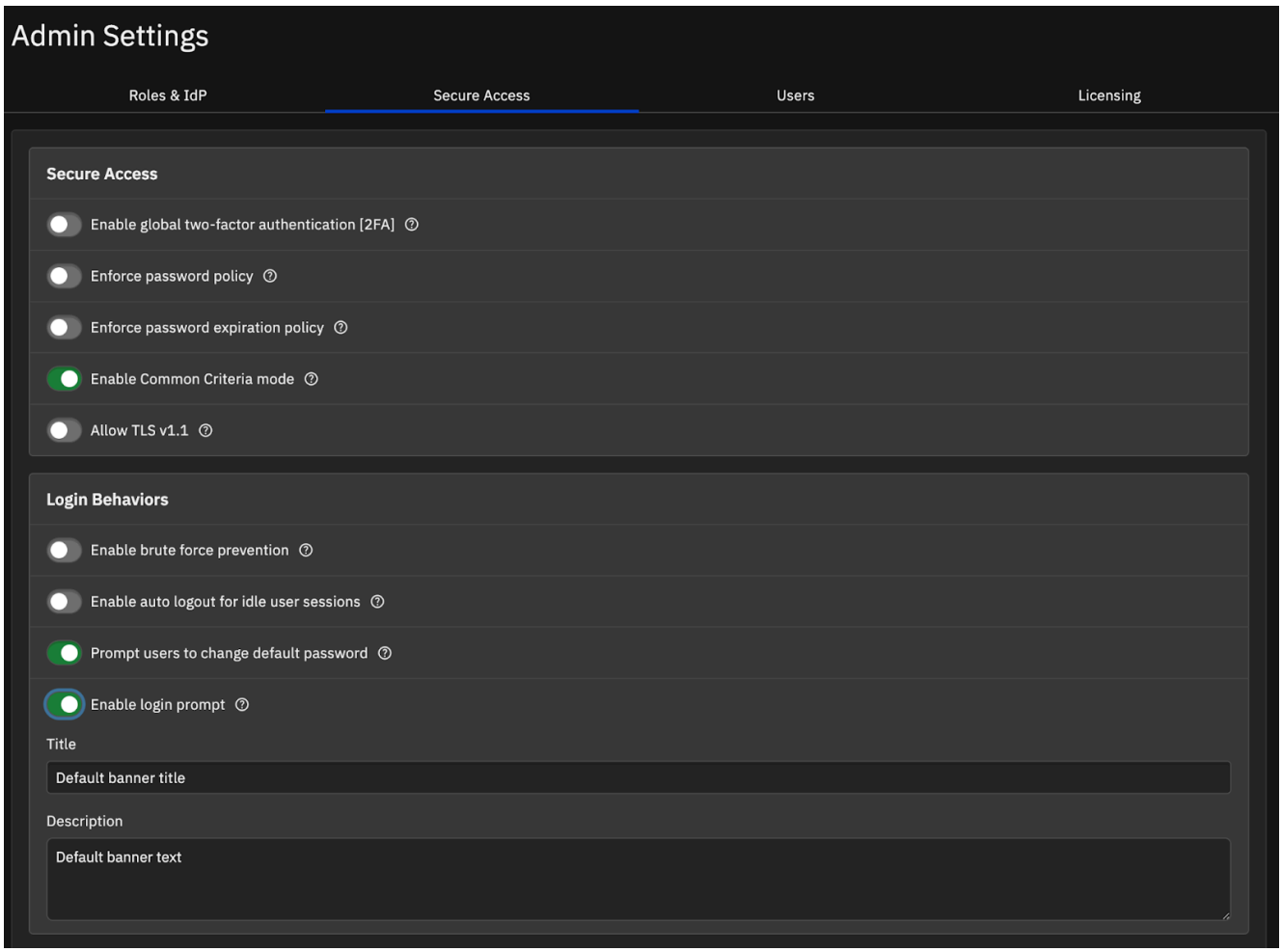
18.1 Setting the login banner from the CLI

Enabling and setting a pre-login banner is required to be in compliance with the Common Criteria certification. A security banner is simply a text file with the desired contents of the banner, which will be displayed whenever ssh or local connections are made to the sensor. The following command places the banner on the sensor and enables it automatically. Replace **<filename>** with the path to the file containing contents for the banner. It may be a bit awkward to deal with content via ssh. The simplest approach is to ssh as the diag-shell user, create the banner file with vi or cat and then upload using the following command. To change the local login message and to change the remote login message:

```
corelight-client banner pre-login upload --file <filename>
```

18.1 Setting the login banner from the Web UI

You can set the login banner from the Web UI on the Secure Access tab on the Admin page as illustrated below.


Admin Settings
Roles & IdP
Secure Access
Users
Licensing
Secure Access
Enable global two-factor authentication [2FA]
Enforce password policy
Enforce password expiration policy
Enable Common Criteria mode
Allow TLS v1.1
Login Behaviors
Enable brute force prevention
Enable auto logout for idle user sessions
Prompt users to change default password
Enable login prompt
Title
Default banner title
Description
Default banner text

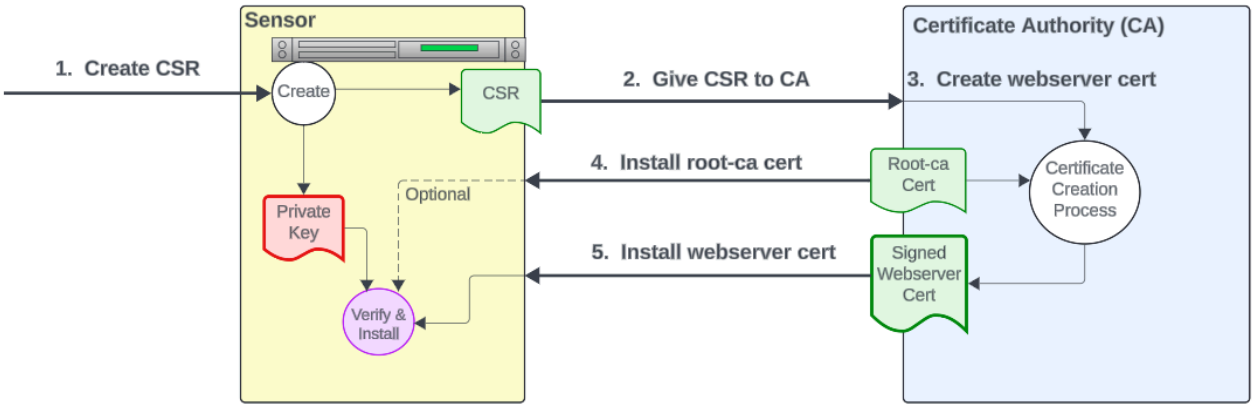
19 CSR generation

19.1 Overview

The secure webserver certificate installation process is designed to support creation and installation of the Sensor's webserver crtificate while ensuring private keys remain secure within the Sensor. The legacy webserver cert installation process required the user to install both the certificate and a private key. This new process never exposes the webserver certificate's private key outside the Sensor. The key is not readable by the user.

The following diagram shows the process of creating and installing the secure webserver certificate. This process is outlined below:

1. A user requests a Certificate Signing Request (CSR) to be created. To create the CSR, the user must specify the certificate attributes. e.g. Common Name, Organization, County, etc
2. The user gets the CSR and provides it to the Certificate Authority (CA).
3. The CA creates the webserver certificate. The CA uses the CSR, their root certificate, and other internal tools/artifacts to generate the new webserver certificate.
4. The user gets the root-ca cert from the CA and installs it on the sensor. The root-ca cert is used to verify the webserver cert before installation. This step is required for a Common Criteria enabled Sensor. For other Sensors, the root-ca cert verification can be skipped.
5. The user gets the webserver cert from the Certificate Authority and installs it on the Sensor. Server software uses the private key and the root-ca cert to verify the webserver cert. Only if verification is successful, will the webserver cert installed. Note, root-ca is required to verify the cert on Common Criteria enabled Sensors. On other Sensors, the root-ca verification step can be skipped.



19.2 Process Details

19.2.1. Create CSR

The Certificate Signing Request (CSR) contains all the information needed for a Certificate Authority (CA) to create the webserver certificate. Creating the CSR results in the creation of the CSR itself (in PEM format), and an associated private key. This private key is not readable by the user.

To create the CSR, the user should run this command:

corelight-client webserver-cert csr create [<attributes>]

The attributes are described below. Most attributes are required information for the CSR.

- **common-name** - The fully qualified domain name of the Sensor
- **organization** - The legal name of your organization
- **org-unit** - The division of your organization handling the cert
- **country** - The two-letter country code where your organization is located
- **state** - The state/region where your organization is located
- **locality** - The city or locality where your organization is located.
- **email** - The email address used to contact your organization
- **password** (optional) - The challenge password for the CSR. If provided, this passphrase must be given to the CA so they can create the certificate
- **subject-alt-name** (optional) - List of additional host names to be protected by this certificate. List of host names must be comma-separated. The Subject Alternate Name list allows the resulting certificate to secure multiple systems.

19.2.2. Get CSR

The user must get the created CSR from the Sensor. After requesting the CSR, the user should copy the contents into a file and remove any whitespace from the beginning of all lines. The file must be in a valid PEM format. Use the following command to get the CSR contents:

corelight-client webserver-cert csr get

Note: Be careful to not create a new CSR or delete the CSR if the original CSR is being used to generate a certificate. Creating a new CSR will replace the previous one. Once a CSR has been deleted or replaced, it is not possible to restore the previous CSR. The CSR and associated private key are needed to validate the certificate during installation.

Here is an example of CSR content in a PEM format:

```
Unset

-----BEGIN CERTIFICATE REQUEST-----
```

```
MIICxzCCAa8CAQAwgYExCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJPSDERMA8GA1UE
BwwIQ29sdW1idXMxEjAQBgNVBAoMCUNvcmVsaWdodDEMMAoGA1UECwwDZW5nMQ0w
CwYDVQQDDAR0ZXN0MSEwHwYJKoZIhvcNAQkBFhJ0ZXN0QGNvcmVsaWdodC5jb20w
ggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCtcBDgd6fV/euDxUSjj/he
wsFIfPjC/YgeP+nuk/BezrBE63EySlIRZiIC7k/KvM44Xi5+WFMPxEzI9D8KBIAI
cp6rPMDAW3Lgb/z+dhPNPq3vl+aI0iz0XfXd9hSyNpuqFtWF8NU/D7lFfomOeMA8
Me28NtG0uPckD5PqxGqcubDJkDUt4Sb/t/Vz8XBUBqNhIVb9lvVnDT9cgXslBxqp
DBWDyfUUeNkThEljvgVDH2Jr/2zaWspj4pnHiA2qBTqZ25GmCsAr28TLwSvg3zsK
xeCExeJCYjGVzA4Gw79G7QhmD4TvXObww2flJoExGmNmSHeNxpot8y1U48FvF4gn
AgMBAAGgADANBgkqhkiG9w0BAQsFAAOCAQEAlvcPWtLqcABD8sS3xJHa1wJUAfTq
gjrHmXs/S9I8C9ZrklZLZDqcxpnxJkQGyMDmpzXl7uwtjoRcarrA+eKdbADsQSyN
IUXk71pGH/neiQEFqjcQmdPl2wNyHpY38xkQQhh4KFCi3WR1czozmg2ukNKJvn6k
8UEkoCURJDXOr9lqh1iFwNqstsTorzdNxBnixDcqDkiU13bCwWei+W2uulbrouCY
Vrd0YTEXV64smapnOpXrMCGMTwxLXjf05RQSKFTC5Isd9zRQlkGnInme+LdRCInN
6IOeC7aC6s39x6IiDQX9BghToRzUX3a56UUy1u0HKUzkNi7byrCpfuBO/w==
-----END CERTIFICATE REQUEST-----
```

19.2.3. Create Webserver Certificate

The user must give the CSR PEM file to the Certificate Authority (CA), along with the challenge password if one

was provided in step 1. The CA will generate a certificate from the CSR. If root-ca certificate validation is needed, also request the root-ca cert from the CA.

It is also possible to generate the certificate from an intermediate cert. If using an intermediate cert, all intermediate certificates must be obtained to properly verify the new certificate during installation.

19.2.4. Install Root CA Certificate

This step is only needed if the root-ca certificate will be used validate the webserver cert. This additional validation step is optional, though it is required for Common Criteria enabled systems. This validation step ensures the webserver certificate being installed was in fact signed by the desired CA. To install the root-ca cert, copy the certificate contents into a PEM formatted file on the Sensor. Then install the root-ca cert with this command:

corelight-client webserver-cert root-ca upload --certificate=<path_to_PEM_file>

19.2.5. Install Webserver Certificate

The installation process of the webserver certificate will first verify the certificate is valid. The provided certificate must be PEM formatted; the previously created CSR's private key and root-ca certificate are then used to verify the given certificate. If the Sensor is not Common Criteria enabled, it is possible to skip root-ca certificate verification. The webserver certificate is only installed if it passes all verification checks; otherwise, an error message is returned. Once the webserver cert is installed, the CSR and root-ca are not needed; they are only used during the verification/installation process.

If an intermediate certificate is used to generate the server certificate, then all intermediate certificates in the chain must also be provided to properly verify the server cert. Place all intermediate certificates in a file to be uploaded with the server certificate.

To install the webserver certificate (also the intermediate certificate chain if signed by an intermediate certificate), copy the certificate contents into a PEM formatted file on the Sensor. Then install the webserver certificate with this command:

corelight-client webserver-cert certificate upload --certificate=<path_to_0PEM_file>
[--chain=<path_to_intermediate_file>]

To skip validation of the certificate chain (including the root-ca cert), provide the following flag:
--skip-cert-chain-validation.

19.3 Additional Information

19.3.1 Show Component Status

It is possible to show the status of the core components of this process: CSR, root-ca cert, webserver cert. This is helpful in understanding if all steps of the process have been completed. Execute the following command to get the status of all components:

corelight-client webserver-cert all list

19.3.2 Uninstall Components

It is possible to uninstall each component individually. Deleting the CSR or root-ca does not affect the installed webserver certificate. Keep in mind, deleting either the CSR or root-ca will prevent future installation of a signed webserver certificate.

Commands to delete each component:

corelight-client webserver-cert csr delete

corelight-client webserver-cert root-ca delete

corelight-client webserver-cert certificate delete

19.3.3 Multi-Delete

The 'all' sub-command also allows deletion of multiple components. This will delete both the webserver certificate and the CSR. The root-ca certificate is not deleted with this command; it may be used to install a new webserver certificate in the future. If the user wants to delete this component, they should use the individual component delete above.

Command to delete both the CSR and webserver certificate:

corelight-client webserver-cert all delete

20 Self-Tests

20.1 Cryptographic POST

Upon initialization of the cryptographic module several self-tests are performed by the module to assure proper function of the cryptographic components, the DRBG, etc. If any one of these tests does not pass, the module will refuse to perform any further work, which will prevent any application attempting to use the module from using possibly compromised cryptography. When the device detects a failure during one or more of the self-tests, an audit failure event will be raised. The administrator can attempt to reboot the TOE to clear the error. If rebooting the device does not resolve the issue, then the administrator should contact Corelight support for further assistance. All power up self-tests execution is logged for both successful and unsuccessful completion.

20.2 Appliance Software Updates

Normally the Corelight appliance may be updated via an automatic process, where updates are retrieved by the sensor from a repository hosted by Corelight. However, to comply with the choices we made as part of the certification process, only offline updates will be permitted. Offline updates are delivered to the appliance via an archive, which will contain encrypted contents, and upon successful validation of authenticity via signature validation, will be installed on the appliance.

In order to determine the running version of the appliance, the following command may be used. corelight-client information get | grep 'os\.'
Typical delivery of the offline update image is via a USB stick, which the administrator must insert into one of the available USB ports on the appliance. The following command may be used to install the update from the offline updater which was previously inserted into one of the USB ports on the appliance.

The following command will list all available updates.
corelight-client updates list

The following command will actually install the pending updates.
corelight-client updates apply

After the update is complete, unmount the previously mounted media with the following command. corelight-client updates unmount

At this point the appliance should be rebooted with the following command. You will be prompted to confirm this action.
corelight-client system reboot

21 Sensitive material zeroization

The appliance will automatically zero out any data on persistent storage when that data is destroyed. Key material, such as SSH keys is destroyed in this manner any time the Administrator triggers re-keying operation.

Sensitive material in memory, such as ephemeral keys, are zeroized as well, before the memory is freed.

22 Rekey Default

The rekey values are by default set in the device and cannot be changed. The time rekey of 1 hour and volume rekey of 1 GB is set in the device and cannot exceed this value.

23 Obscured Password

No specific configuration is required to ensure data is not revealed with entering local CLI login. Passwords are obscured to the users. For all authentication at the local CLI the TOE displays only "*" characters when the administrative password is entered.

24 Web UI Connections

The TOE offers a Web UI that is protected with HTTPS such that the TOE acts as an HTTPS server with no additional configuration required by the user.

To access the Web UI from a browser, enter the following URL in your browser's address bar:

https://<ip address>:8443

where <ip address> is the IP address assigned to the management interface of the TOE when it was connected to your network.

The TOE supports the following ciphersuites:

- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492,
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492,
- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289,
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289,
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

The TOE's server includes a non-modifiable configuration that prohibits all TLS versions other than v1.2 and v1.3. If the TOE receives a TLS attempt using a version other than v1.2 or v1.3, the connection attempt will be rejected.

The TOE includes ECDHE ciphersuites and supports curves secp256r1, secp384r1, secp521r1.

The TOE supports session resumption based on session tickets that adhere to the format provided in section 4 RFC 5077.