VECTRA PLATFORM SECURITY TARGET

VERSION 1.27
21 NOVEMBER 2025

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DOCUMENT CONTROL INFORMATION

Version	Date	Summary of Changes
1.0	2022-05-20	First version of the Security Target
1.1	2023-01-30	Updated introduction, objectives
1.2	2023-03-10	Updated SFRs and TSS
1.3	2023-03-14	Adjusted markings of selections, assignment and refinements in SFRs
1.4	2023-03-17	Updated ST based on feedback from evaluator
1.5	2024-08-23	Updated ST based on feedback from the certifier
1.6	2024-09-05	Updated ST based on feedback from evaluator
1.7	2024-09-26	Updated ST based on clarifications
1.8	2024-10-30	Updated ST based on feedback from evaluator
1.9	2025-02-03	Updated ST based on clarifications
1.10	2025-02-04	Updated ST
1.11	2025-03-04	Updated ST based on feedback from evaluators
1.12	2025-03-26	Updated ST based on feedback from evaluators
1.13	2025-03-28	Updated ST based on clarifications
1.14	2025-04-03	Update based on evaluator's feedback
1.15	2025-04-08	Addressed certifier comments and evaluator's feedback

1.16	2025-05-12	Addressed evaluator's feedback.
1.17	2025-06-17	Addressed evaluator's feedback.
1.18	2025-09-08	Addressed evaluator's feedback and updated TOE version
1.19	2025-09-15	Updated ST based on feedback from evaluators
1.20	2025-09-25	Addressed evaluator's feedback.
1.21	2025-10-01	Addressed evaluator's feedback.
1.22	2025-10-07	Addressed evaluator's feedback.
1.23	2025-10-14	Addressed evaluator's feedback.
1.24	2025-10-22	Addressed evaluator's feedback.
1.25	2025-10-27	Addressed evaluator's feedback.
1.26	2025-11-11	Addressed evaluator's feedback.
1.27	2025-11-21	Updated ST based on feedback from evaluators

ABBREVIATIONS

Abbreviation	Description
AI	Artificial Intelligence
API	Application Programming Interface
AWS	Amazon Web Services
CLI	Command Line Interface
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
ESXi	Elastic Sky X Integrated
GCP	Google Cloud Platform
HTTPS	Hyper-Text Transport Protocol Secure
IoT	Internet of Things
IP	Internet Protocol
KVM	Kernel-based Virtual Machine
NDR	Network Detection and Response
NTP	Network Time Protocol
RBAC	Role Based Access Control
REST API	Representational State Transfer API
SIEM	Security Information and Event Management
SPAN	Switched Port Analyzer
SSH	Secure Shell
Syslog	System log
TAP	Terminal Access Point
TLS	Transport Layer Security
TOE	Target of Evaluation
UI	User Interface

NOTATIONS AND FORMATTING

The notations and formatting used in this ST are consistent with version 3.1 Revision 5 of the Common Criteria (CC).

The **refinement** operation is used to add detail to a requirement, and thus further restricts a requirement. Refinement of security requirements is denoted by **bold text**. Deleted words are denoted by ~~strike-through text~~.

The **selection** operation is used to select one or more options provided by the CC in stating a requirement. Selections are denoted by *italicized* text in square brackets, [*Selection value*].

The **assignment** operation is used to assign a specific value to an unspecified parameter, such as the length of a password. Assignment is indicated by showing the value with bold face in square brackets, [**Assignment_value**].

The **iteration** operation is used when a component is repeated with varying operations. Iteration is denoted by showing the iteration number in parenthesis following the component identifier, (iteration_number).

Assets: Assets to be protected by the TOE are given names beginning with "AS." – e.g., AS.CLASSIFIED_INFO.

Assumptions: TOE security environment assumptions are given names beginning with "A."- e.g., A.Security_Procedures.

Threats: Threats to the TOE are given names beginning with "T." – e.g., T.Filter_Fails.

Policies: TOE security environment policies are given names beginning with "P."—e.g., P.Information_AC.

Objectives: Security objectives for the TOE and the TOE environment are given names beginning with "O." and "OE.", respectively, - e.g., O.Filter-msg and OE.Clearance.

1. Security Target Introduction

1.1. ST and TOE References

The following table identifies the Security Target (ST).

Item	Identification
ST Title	Vectra Platform Security Target
ST Version	1.27
ST Date	21 November 2025
ST Author	Nemko System Sikkerhet AS & atsec information security

Table 1: ST references

The following table identifies the Target of Evaluation (TOE).

Item	Identification
TOE Identifier	Vectra Platform
TOE Components	Vectra Brain (virtual)
	Vectra Sensor (virtual)
TOE Version	Release 9.3.0-13-36

Table 2: TOE references

The following table identifies common references for the ST and the TOE.

Item	Identification
CC Version	3.1 Revision 5
Assurance level	EAL2 augmented with ALC_FLR.1
Protection Profile	None

Table 3: CC identification and Evaluation Assurance Level

1.2. TOE Overview

The Vectra platform is a threat detection platform categorized as a Network Detection and Response (NDR) platform driven by AI to see and stop attacks - from hybrid and cloud-native apps, as well as AWS, Azure and GCP environments to data center workloads, IoT, and enterprise networks. The Vectra Platform uses AI to detect behaviors across the kill chain, prioritizes the threat behaviors that pose the highest risk to the organization and provides actionable data and automated response for investigation, hunting and containment.

The Vectra platform provides the ability to protect itself and associated data from unauthorized access or modification and provides an audit trail to ensure accountability for authorized actions. It provides the following capabilities that shall be satisfied for an enterprise with simplicity, scalability, and ease of management:

- End-to-end data protection.
- Secure access to the data.

The Vectra Platform deployment consists of two main components, i.e., Vectra Brain and Vectra Sensor. Both of the components can be deployed either in a data center or a cloud environment. Only the data center deployment scenario is included in the evaluated configuration. Additionally, the evaluated configuration requires the deployment to be air-gapped, ensuring that it operates in a physically isolated network environment with no direct connection to external networks, including the internet.

The following figure shows a logical view of the placement of the Vectra Platform components (Vectra Brain and Vectra Sensors) in the evaluated configuration, including the data flows.

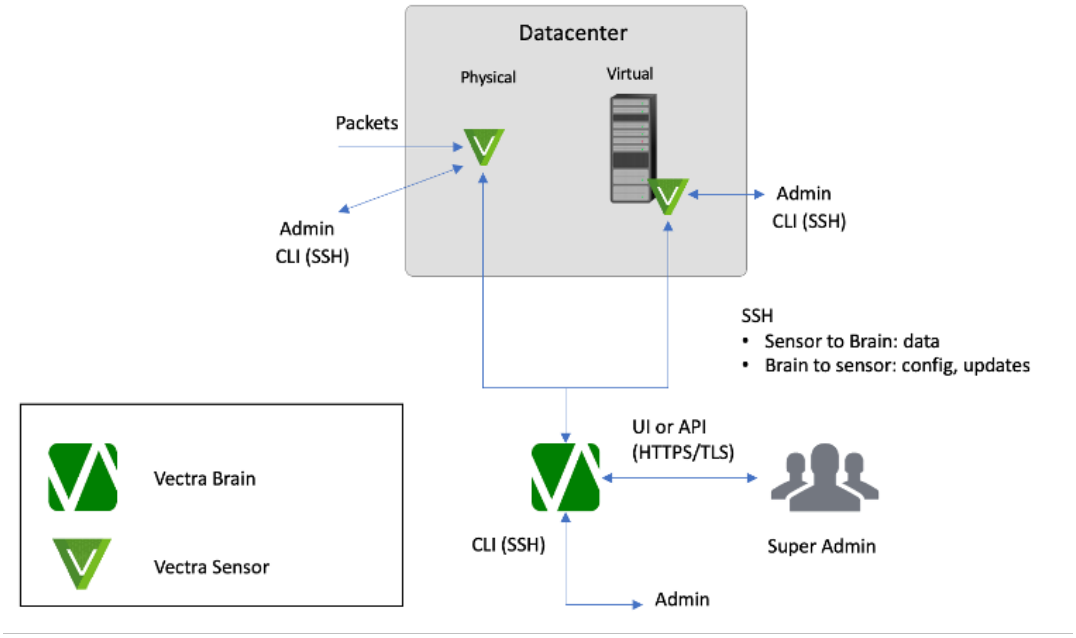


Figure 1: Vectra deployment

VECTRA BRAIN

The Brain is where all the algorithms are executed to detect and prioritize threats using AI, as well as interfaces to send the data out in syslog, email, or API.

The Brain has a management interface over which it communicates with the Sensors, receives software updates, and provides Web UI and REST API access. The Web UI and the REST API are used to manage the TOE and to view the surfaced threats and take action.

The Brain can be in a form of an appliance (dedicated hardware with software) or as a software on VMware ESXi or AWS and Azure environments and is deployed in a customer data center or a customer cloud environment. Only the scenario where the Brain is in the form of a software on VMware ESXi deployed in a customer data center is in the scope of the evaluation.

VECTRA SENSORS

The Sensors can be physical, virtual (deployed on ESXi, KVM or Hyper-V) and cloud (deployed in the customer's AWS, Azure or GCP environments). Only the virtual Sensor deployment (on ESXi, KVM or Hyper-V) will be in the scope of the evaluation.

The Sensors receive network traffic from SPAN, TAP or network visibility devices, where each Sensor has one or more capture interfaces ('packets') over which a copy of the traffic is received. Traffic packets are parsed by each Sensor into various metadata streams. The Sensor performs packet level de-duplication and sends the resulting metadata to the Brain over a secure connection using the machine-to-machine interface. Each Sensor has also a rolling buffer that stores a copy of the packets for the past 30 – 40 minutes.

The Sensors are deployed by customers in various locations where they want the network-based coverage ranging from data centers or remote offices, or cloud environments. Only the scenarios where

the Sensors are deployed virtually in a customer data center are in the scope of the evaluation. When the Sensors are deployed virtually, the Vectra Sensor is in the form of software.

DEPLOYMENT SETUP OF THE VECTRA PLATFORM

Before the Vectra Platform is operational, the Brain and the Sensors, (at least one Sensor), need to be deployed, where each component needs to be racked and stacked and provided power and an IP address (or DHCP configured). Once configured, the Brain can be accessed via the Web UI, where settings such as NTP server, DNS server, and IP addresses assignment in the internal network can be specified.

Vectra can be deployed either: online, with initial sensor pairing and updates managed via the Vectra cloud, offline, where both pairing and updates occur without cloud connectivity, or air-gapped, where the Brain operates in a fully isolated network segment with no external connectivity. In this configuration, Sensors are paired manually, and updates are applied manually via offline media. Air-gapped operation is the only configuration considered within the evaluated environment.

Virtual Sensors come pre-configured with Brain information embedded in the image downloaded from the Brain. Once deployed and assigned an IP address, the virtual Sensors reach out to the Brain. There is a setting that can be enabled for auto pairing of virtual Sensors. The SSH tunnel carries metadata from Sensors to the Brain and configuration/updates from the Brain to the Sensors.

The CLI environment is a restricted interface provided solely for initial device setup. It facilitates essential configuration tasks, such as IP assignment and device pairing, using the default user "vectra", whose password must be changed during the setup process. After the TOE is configured, the CLI is not intended for routine operation, remaining available only for occasional maintenance tasks (for example, forcing sensor-to-brain pairing if auto-pairing fails).

1.2.1. TOE TYPE

The Vectra platform (TOE), consisting of Vectra Brain and Vectra Sensor, is a threat detection platform categorized as a Network Detection and Response (NDR) platform.

1.2.2. TOE FEATURES

The TOE is comprised of several security features:

- **Security Audit**: The TOE provides extensive auditing capabilities.
- **Cryptographic Support**: The TOE provides cryptography in support of remote administrative management via SSH and TLS/HTTPS.
- **Identification and Authentication**: The TOE provides authentication services for local and remote users wishing to connect to the TOEs secure Web UI or REST API administrator interfaces.
- **Security Management**: The TOE provides secure administrative services for management of general TOE configuration and the security functionality provided by the TOE.
- **Protection of the TSF**: The TOE protects against interference and tampering by untrusted subjects by implementing identification, authentication, and access controls to limit configuration to only authorized administrators.
- **TOE Access**: Administrative interfaces present a configurable access banner before authentication. The banner is shown on all interactive connections and provides the organization's required warning or advisory notice.

- **Trusted channels**: The TOE provides trusted channel between the Sensors and the Brain (encrypted, authenticated over SSH). Also, the TOE use TLS 1.2 and TLS 1.3 to protect data-in-transit.

For detailed description of the security features, please refer to section 1.3.2.

1.3. TOE DESCRIPTION

The Vectra Platform elements are presented in the following figure.

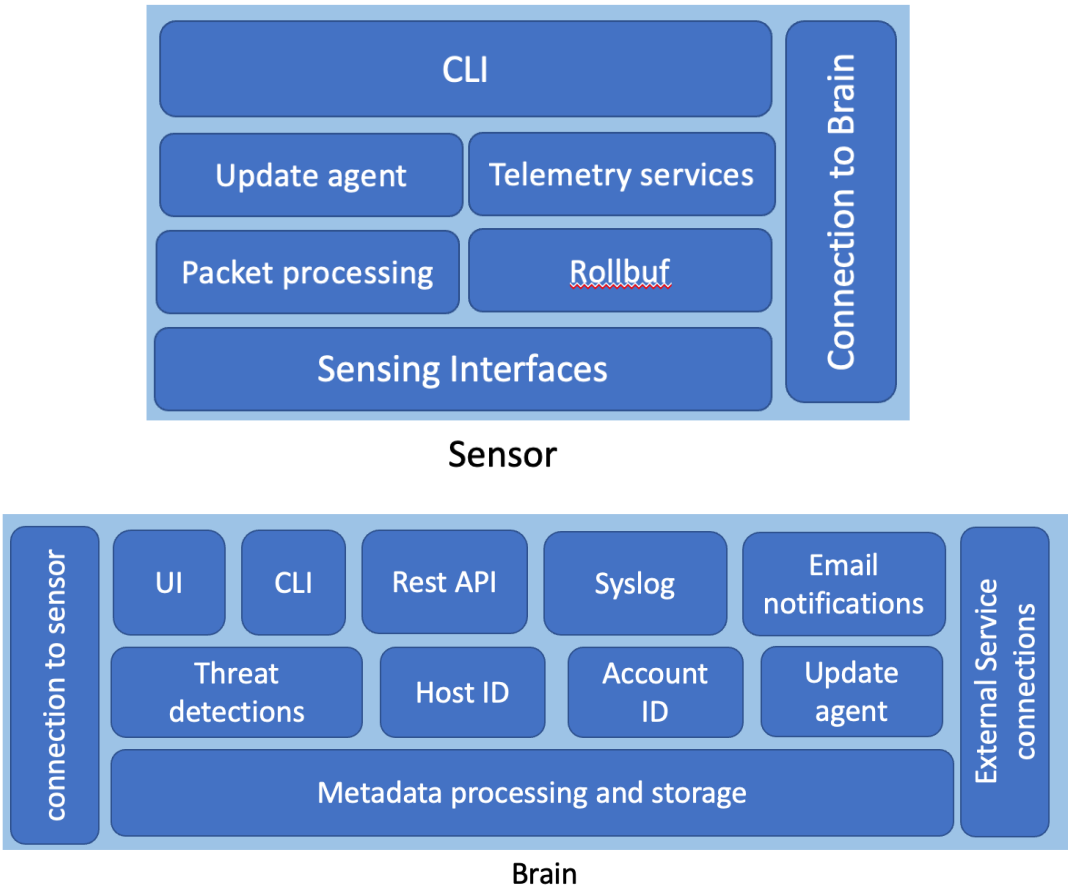


Figure 2: Vectra Platform

Sensing Interfaces
These are network interfaces on which packets are received.

Rollbuf
This is a temporary store of the raw packets on each Sensor. The store is limited in size and as new data comes in, older packets are removed. When a threat detection fires, the Brain asks the Sensor for corresponding packets from the rollbuf, and if available the Sensor sends them back. The associated packets are presented as a 'micro pcap' for the detection which analysts can access over UI or REST API.

Packet Processing
As the Sensors receive packets from the sensing interfaces, they process the packets and extract metadata. The extracted metadata is sent to the Brain for analysis.

Telemetry services
This service monitors the sensor telemetry (packets processed, errors) and sends it to the Brain.

Update agent
The update agent is responsible for receiving the updates and installing them on Brain and Sensors.

CLI
There is a default CLI password for the Brain and the Sensors, which admin users can change it from the Web UI. CLI (through SSH) only exposes a limited set of commands, for initial setup and other setup like backup and restore. CLI passwords are stored using SHA-512 hash with salt.

Data storage on the Brain
To keep TOE data secure, data at rest encryption is used on the Brain using a combination of RAID controller and self-encrypting drives.

Web UI
Web UI is accessed over HTTPS using TLS 1.2 and TLS 1.3. Web UI access is secured using username and password, where the Brain first ships with a default password, which the user has to change on first login. The Web UI session can be terminated by the user logging out. Web UI passwords for local users are stored using PBKDF2 SHA-256 hash with salt.

REST API
The REST API is secured over HTTPS using TLS 1.2 and TLS 1.3, with the same list of ciphers offered for both REST API and Web UI connections. Every user created on the Vectra platform has access to its API token. The admin can revoke tokens for any user, and users can regenerate their token if needed. By default, no token is generated; users must log into the Web UI and create one before accessing the REST API.

Syslog
The Brain has the ability to send syslog to external systems for audit log, system health or the threat detections.

Email notifications
The Brain also, optionally has the ability to send email notifications. This can be configured from the Web UI. Email notifications will not be in the scope of the evaluation.

Threat detections
The Brain runs AI algorithms to detect threats. These detections consume metadata and trigger detections when a behavior is observed. The behaviors surfaced along with all the contexts can be accessed via the Web UI or REST API.

Host ID
The Brain tracks an IP address and associates it to a host based on a variety of network artifacts. This allows for correlation of multiple behaviors to a specific host, allowing the analyst a more holistic view that viewing threat detections in isolation.

Account ID
For detections attributed to accounts, the Brain also tracks and correlates detections to the observed account.

Metadata processing and storage
This service on the Brain receives metadata from all connected sensors, performs flow level de-duplication and makes the metadata available to all detections in an efficient and scalable manner.

1.3.1. PHYSICAL SCOPE

The TOE (Vectra Platform) consists of the following two components:

- Vectra Brain, in the form of a virtual SW component deployed on Virtual Machines
- Vectra Sensor, in the form of a virtual SW component deployed on Virtual Machines

The TOE includes the following supporting documents:

- Vectra_AGD_v1.5.pdf
- Hyper-V vSensor Deployment Guide - 2025_Mar_5.pdf
- KVM vSensor Deployment Guide - 2025_Aug_27.pdf
- Permissions - Feb2025.xlsx
- SSL Certificate Installation - 2025_Nov_11.pdf
- Vectra Quadrant UX Deployment Guide - 2024_Oct_9.pdf
- Vectra_REST_API_Guide_v2.5 - Aug2025.pdf
- VMware Brain Deployment Guide - 2025_Oct_7.pdf
- VMware vSensor Deployment Guide - 2025_Oct_8.pdf

The TOE images can be downloaded from the Vectra Customer Support Portal.
The static documentation is available publicly without any restrictions, and static links can be requested for documentation access.

1.3.2. LOGICAL SCOPE

The TOE is comprised of several security features:

- Security Audit
- Cryptographic Support
- Identification and Authentication
- Security Management
- Protection of the TSF
- TOE Access
- Trusted channels

Each of the security features identified consists of several security functionalities, as identified below.

Security Audit
The TOE provides extensive auditing capabilities, by generating a comprehensive set of audit logs that identify specific TOE operations. For each event, the TOE records the date and time of each event, the type of event, the subject identity, and the outcome of the event. The audit logs are stored locally and can be sent out over syslog to an external system like a SIEM.

Cryptographic Support
The TOE provides cryptography in support of remote administrative management via SSH and TLS/HTTPS. TOE components communicate with each other using SSH.

Vectra Brain utilizes file integrity checking during boot where it verifies cryptographic checksums of critical system files against the known expected values. If verification succeeds the system proceeds to decrypt the encrypted file system and continue the boot process normally to enter operational mode. If the verification fails, the boot process is halted with error message, and the application code remains encrypted.

Vectra Sensor does not utilize Secure Boot, as it is considered an untrusted asset. Instead, it performs a file system integrity check as part of the Ubuntu operating system to verify the absence of file system or hardware corruption. If a file system integrity issue is discovered, it will run a File System Integrity repair attempt automatically. If it is successful it will boot, if not it will not. If there is Hardware corruption, it will fail to boot.

Identification and Authentication
The TOE provides authentication services for local and remote users wishing to connect to the TOEs secure Web UI or REST API administrator interfaces. The TOE requires authorized administrators to authenticate prior to being granted access to any of the management functionality. After successful authentication, the TOE determines the permitted level of access for a user based on the local authorization setting for that user and provides role-based access (RBAC).

Security Management
The TOE provides secure administrative services for management of general TOE configuration and the security functionality provided by the TOE. All TOE administration occurs either through a secure SSH or TLS/HTTPS session. The TOE supports various administrator roles out of the box and custom roles can be created as needed, but only the Super Admin role has a user defined out of the box.

Protection of the TSF
The TOE protects against interference and tampering by untrusted subjects by implementing identification, authentication, and access controls to limit configuration to only authorized administrators.

TOE Access
Administrative interfaces present a configurable access banner before authentication. The banner is shown on all interactive connections and provides the organization's required warning or advisory notice.

Trusted Channel
The TOE provides trusted channel between the Sensors and the Brain (encrypted, authenticated over SSH). The TOE also provides trusted channels on remote administrative management via SSH and TLS/HTTPS. SSHv2 and TLS 1.2 and TLS 1.3 are supported.

1.3.3. NON-TOE COMPONENTS

The following components are part of the Operational Environment:

- Underlying network infrastructure and IT components which send data to the Sensor and provides underlying communication between Sensor and Brain.
- Underlying infrastructure on which the virtual Vectra Sensor is deployed (ESXi, KVM or Hyper-V).
- DNS for lookups.
- Local management terminal to manage the TOE using CLI through a SSH communication
- Remote management terminal to manage the TOE using WebUI or REST API through a HTTPS/TLS communication.
- Syslog server for receiving audit logs.
- NTP Server for reliable time.

These components operate outside the TOE boundary. The confidentiality and integrity of audit data transmitted to the Syslog server are ensured by secure network configurations within the Operational Environment in accordance with OE.SYSLOG_PROTECTION. The Brain uses unauthenticated NTP

to synchronize time with external NTP servers; therefore, the authenticity of received time updates is ensured by the Operational Environment as described in OE.TIME. The NTP server provides additional benefit for maintaining accurate time, but its absence does not affect the TOE's ability to provide reliable timestamps.

1.3.4. TOE UPDATE

The TOE supports both automatic (using Vectra cloud) and manual (where customer obtains the update and uploads it in the Brain UI) secure update of the TOE software. Only manual secure update will be in the scope of this evaluation.

For manual update, the customer downloads an update file together with a SHA256 checksum from Vectra for an upgrade manually. After verification of the checksum, the file can be uploaded into the Brain from the Web UI. The Brain validates the package is from Vectra before applying the update. The update package contains updates for the Brain and Sensors.

Updating the TOE will result in deviation from the evaluated configuration.

2. CONFORMANCE CLAIMS

This TOE and ST are conformant with the following specifications.

Item	Identification
Part 2 of the ISO/IEC 15408 international standard	Common Criteria security functional components, April 2017, Version 3.1, Revision 5, extended
Part 3 of the ISO/IEC 15408 international standard	Common Criteria security assurance components, April 2017, Version 3.1, Revision 5, conformant
Protection Profiles	None
Packages	EAL2 augmented with ALC_FLR.1

Table 4: Conformance Claims

3. SECURITY PROBLEM DEFINITION

This chapter identifies the following:

- Threats that are to be countered by the TOE and its Operational Environment.
- Organizational security policies (OSPs) that are to be enforced by the TOE and its Operational Environment.
- Assumptions that are made on the Operational Environment in order to be able to provide security functionality.

3.1. THREATS

Potential assets:

- TSF data – software, configuration files, audit records, authentication information and cryptographic keys that are controlling the behaviour of the TSFs or is a result of a TSF, and relevant to determine if the TOE is in a secure state.
- Metadata – Data received to the Sensor which is transferred to the Brain.

Potential threat agents:

- External entities not authorized to access TSF services. Those may attempt to get access to TSF services either by masquerading as an authorized entity or by attempting to use TSF services without proper authorization

It is expected that the Vectra Platform (TOE) units will be protected to the extent necessary to ensure that they remain connected to the target server applications they protect.

The following table lists the threats addressed by the TOE and the TOE Environment.

Threat	Description
T.UNAUTHORIZEDACCESS	Threat agents may attempt to gain Administrator access to the TOE by nefarious means such as masquerading as an Administrator to the TOE, masquerading as the TOE to an Administrator, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session. Successfully gaining Administrator access allows malicious actions that compromise the security functionality of the TOE.
T.HACKACCESS	An attacker may get undetected system access to the TOE. The attacker may use hacking methods to exploit access control in the TOE.
T.MALFUNCTION	The TOE may malfunction that may compromise information and data processing, implying risk of data exploiting. The attacker may get unauthorized access to TOE resources. The TOE may malfunction that may compromise roles and permissions, implying risk of data exploiting. An administrator may gain unauthorized roles and permissions in TOE.

Table 5: Security Threats

3.2. Organizational Security Policies (OSP)

The following table lists the Organizational Security Policies imposed by an organization to address its security needs.

Organizational security Policies	Description
P.ACCOUNTABILITY	The authorized administrators of the TOE shall be held accountable for their actions.
P.ADMINACCESS	An authorized administrator must manage the TOE securely.
P.DETECT	To trace all security-related responsibilities, security-related events shall be documented, maintained, and analyzed, and such records can be checked.
P.M2MSECURE	The TOE shall ensure that all machine-to-machine communication between the Sensor and Brain is secure, authenticated, and encrypted to protect the confidentiality and integrity of transmitted data.

Table 6: Organizational Security Policies

3.3. Assumptions

The specific conditions listed in the following table are assumed to exist in the TOE's environment. These assumptions include both practical realities in the development of the TOE security requirements and the essential environmental conditions on the use of the TOE.

Assumptions	Description
A.NETWORK	There will be a network that supports communication between instances of the TOE and between the TOE and IT systems used to manage the TOE. This network functions properly.
A.NOGENPURP	There are no general-purpose computing capabilities (e.g., compilers or applications) available on the TOE, other than those services necessary for the operation, administration, and support of the TOE.
A.PHYSICAL	The TOE shall presumably be located in physically secure environment that can be accessed only by the authorized administrators.
A.TRUSTADMIN	The administrators of the TOE shall not have any malicious intention, shall receive proper training on the TOE management, and shall follow the administrator guidelines.
A.TIME	It is assumed that the Operational Environment provides the TOE with reliable time.
A.KEYS	It is assumed that random bits provided by the underlying platform are of good quality and have sufficient entropy.

Assumptions	Description
A.SYSLOG_PROTECTION	It is assumed that the Operational Environment ensures the confidentiality and integrity of audit data transmitted from the TOE to external Syslog servers through secure network configurations and mechanisms.

Table 7: TOE Environment Assumptions

4. SECURITY OBJECTIVES

This chapter defines the security objectives for the TOE and its supporting environment. The security objectives are intended to counter identified threats, comply with defined organizational security policies, and address applicable assumptions.

4.1. TOE SECURITY OBJECTIVES

This section defines the security objectives that are to be addressed by the TOE.

Security Objectives	Description
O.ACCESS	The TOE must allow only authorized administrators to access only appropriate TOE functions and data. Also, the TOE must display an initial banner before users log into the TOE. The initial banner must contain restrictions of use, legal agreements, or any other appropriate information to which users make consent by accessing the TOE.
O.AUDIT	The TOE shall record, and export security-related events associated with users to enable tracing of responsibilities for security-related events. The TOE is responsible for formatting and transmitting audit records to an external Syslog server via interfaces provided by the Operational Environment.
O.CRYPTO	The TOE must protect the confidentiality and integrity of data passed between itself and authorized administrators.
O.IDAUTH	The TOE must be able to identify and authenticate authorized administrators prior to allowing access to TOE security management functions.
O.MANAGE	The TOE must include a set of functions that allow effective management of its functions and data. The TOE will provide mechanisms to ensure that only administrators are able to log in and configure the TOE and provide protections for logged-in administrators.
O.PROTECT	The TOE must protect itself and its resources from unauthorized modifications and access to its functions and data.
O.TEST	The TOE will provide the capability to test its integrity to ensure it is operating properly.
O.TIME	The TOE shall provide reliable time stamps. The administrator can configure the TOE to synchronize its clocks with NTP servers using unauthenticated NTP protocol. The integrity and authenticity of the external time source are ensured by the Operational Environment.

Security Objectives	Description
O.M2MPROTECT	The TOE must protect machine-to-machine communication between the Sensor and Brain by ensuring secure transmission of metadata, telemetry, and configuration updates. This shall be achieved using cryptographic mechanisms, such as SSH, that provide authentication, encryption, and integrity protection.

Table 8: Security Objectives for the TOE

4.2. Operational Environment Security Objectives

This section defines the security objectives that are to be addressed by the Operational Environment of the TOE.

Security Objectives	Description
OE.NETWORK	The administrator will install and configure a network that supports communication between instances of the TOE and between the TOE and IT systems used to manage the TOE. The administrator will ensure that this network functions properly and that it is air-gapped via the firewall or other applicable method.
OE.NOGENPURP	There are no general-purpose computing capabilities (e.g., compilers or applications) available on the TOE, other than those services necessary for the operation, administration, and support of the TOE.
OE.PHYSICAL	Those responsible for the TOE must ensure that the hardware on which the TOE and OS are installed, is protected from any physical attack.
OE.TRUSTADMIN	The administrator of the TOE shall not have any malicious intention, shall receive proper training on the TOE management, and shall follow the administrator guidelines.
OE.TIME	The Operational Environment shall provide the TOE with reliable time source and it is responsible for ensuring the authenticity, integrity, and reliability of that NTP service.
OE.KEYS	The Operational Environment shall ensure that random bits provided by the underlying platform are of good quality and have sufficient entropy.
OE.SYSLOG_PROTECTION	The Operational Environment shall provide a secure and trusted communication channel between the TOE and the external Syslog server, ensuring the confidentiality and integrity of audit data during transmission.

Table 9: Security Objectives for the Operational Environment

4.3. Security Objectives Rationale

The following tracing shows which security objectives address which threats, policies (OSPs) and assumptions.

	Threats			Policies				Assumptions						
	T.UNAUTHORIZEDACCESS	T.HACKACCESS	T.MALFUNCTION	P.ACCOUNTABILITY	P.ADMINACCESS	P.DETECT	P.M2MSECURE	A.NETWORK	A.NOGENPURP	A.PHYSICAL	A.TRUSTADMIN	A:TIME	A.KEYS	A.SYSLOG_PROTECTION
TOE Security Objectives														
O.ACCESS		X			X									
O.AUDIT		X	X	X		X								
O.CRYPTO		X	X				X							
O.IDAUTH		X		X										
O.MANAGE		X			X									
O.PROTECT	X	X	X											
O.TEST			X											
O.TIME				X		X								
O.M2MPROTECT							X							
Operational Environment Security Objectives														
OE.NETWORK							X	X						
OE.NOGENPURP									X					
OE.PHYSICAL										X				
OE.TRUSTADMIN		X		X	X						X			
OE.TIME				X		X						X		
OE.KEYS							X						X	
OE:SYSLOG_PROTECTION			X			X								X

Table 10: Mapping of Objectives to Threats, Policies and Assumptions

The following table is a set of justifications that shows that all threats, policies (OSPs), and assumptions are effectively addressed by the security objectives.

Threat/Policy/Assumption	Security Objective Rationale
T.UNAUTHORIZEDACCESS	*Threat agents may attempt to gain Administrator access to the TOE by nefarious means such as masquerading as an Administrator to the TOE, masquerading as the TOE to an Administrator, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session. Successfully gaining Administrator access allows malicious actions that compromise the security functionality of the TOE.* O.PROTECT ensures that the TOE will have adequate protection from external sources and that all TOE Security Policy functions are invoked.
T.HACKACCESS	*An attacker may get undetected system access to the TOE due to missing, weak and/or incorrectly implemented access control.* O.ACCESS and O.IDAUTH provide the means to identify and authenticate the TOE administrators. The correct identity of the administrator is the basis for any decision of the TOE about an attempt of an administrator to access data.

	O.AUDIT provides the TOE the capability to detect and create records of security-relevant events associated with administrators and users. The TOE is responsible for formatting and transmitting audit records to an external Syslog server via interfaces provided by the Operational Environment. O.CRYPTO ensures the confidentiality and integrity of data passed between the TOE and the authorized administrator for management purposes. O.MANAGE restricts the ability to modify the security attributes associated with access control rules, access to authenticated and unauthenticated services, etc. to the authorized administrators. These objectives ensure that no other administrator can modify the information flow policy to bypass the intended TOE security policy. O.PROTECT ensures that the TOE will have adequate protection from external sources and that all TOE Security Policy functions are invoked. OE.TRUSTADMIN ensures that the TOE administrators have guidance that instructs them how to administer the TOE in a secure manner.
T.MALFUNCTION	*The TOE may malfunction that may compromise information and data processing and/or roles and permissions.* O.AUDIT provides the TOE the capability to detect and create records of security-relevant events associated with administrators and users. The TOE is responsible for formatting and transmitting audit records to an external Syslog server via interfaces provided by the Operational Environment. O.CRYPTO requires the TOE to implement cryptographic services to provide confidentiality protection of data in flight. O.PROTECT ensures that the TOE will have adequate protection from external sources and that all TOE Security Policy functions are invoked. O.TEST ensures that failure of mechanisms do not lead to a compromise in the TSF. OE.SYSLOG_PROTECTION ensures that any audit data transmitted externally to Syslog servers is protected for confidentiality and integrity by secure mechanisms provided by the Operational Environment.
P.ACCOUNTABILITY	*The authorized administrators of the TOE shall be held accountable for their actions.* O.AUDIT ensures that the administrator's user-identifier is recorded when any security relevant change is made to the TOE (e.g., modifying TSF data, login sessions). The TOE is responsible for formatting and transmitting audit records to an

	external Syslog server via interfaces provided by the Operational Environment. O.IDAUTH requires the TOE to identify and authenticate administrators prior to allowing any TOE access on behalf of those administrators. O.TIME ensures that audit logs have correct timestamps. OE.TRUSTADMIN ensures that the TOE administrators have guidance that instructs them how to administer the TOE in a secure manner.
P.ADMINACCESS	*An authorized administrator must manage the TOE securely.* O.ACCESS and O.MANAGE provide authorized administrators the capability to view and manage configuration settings. OE.TRUSTADMIN ensures that the administrators are non-hostile and are trained to appropriately manage and administer the TOE.
P.DETECT	*To trace all security-related responsibilities, security-related events shall be documented, maintained, and analyzed, and such records can be checked.* O.AUDIT ensures the collection of data on security relevant events. The TOE is responsible for formatting and transmitting audit records to an external Syslog server via interfaces provided by the Operational Environment. O.TIME ensures that the audit functionality can include reliable timestamps. OE.SYSLOG_PROTECTION ensures that the Operational Environment protects audit data transmitted to external Syslog servers, preserving confidentiality and integrity during export.
P.M2MSECURE	*The TOE shall ensure that all machine-to-machine communication between the Sensor and Brain is secure, authenticated, and encrypted to protect the confidentiality and integrity of transmitted data.* O.M2MPROTECT ensures the protection of machine-to-machine communication between the Sensor and Brain by ensuring secure transmission of metadata, telemetry, and configuration updates. O.CRYPTO ensures the protection of confidentiality and integrity of data passed between itself and authorized entities using cryptographic mechanisms. OE.NETWORK ensures that the network supports machine-to-machine communication functions properly. OE.KEYS ensures that the environment provides high-entropy random bits for cryptographic operations securing M2M communication.

A.NETWORK	*There will be a network that supports communication between instances of the TOE and between the TOE and IT systems used to manage the TOE.* OE.NETWORK restates it as an objective for the administrator to satisfy.
A.NOGENPURP	*There are no general-purpose computing capabilities (e.g., compilers or applications) available on the TOE, other than those services necessary for the operation, administration, and support of the TOE.* OE.NOGENPURP ensures that there are no general- purpose computing capabilities (e.g., the ability to execute arbitrary code or applications) on the TOE.
A.PHYSICAL	*The TOE shall presumably be located in physically secure environment that can be accessed only by the authorized administrators.* OE.PHYSICAL ensures that the environment provides physical security, commensurate with the value of the TOE and the data it contains.
A.TRUSTADMIN	*The administrators of the TOE shall not have any malicious intention, shall receive proper training on the TOE management, and shall follow the administrator guidelines.* OE.TRUSTADMIN ensures that the administrators of the TOE shall not have any malicious intention, shall receive proper training on the TOE management, and shall follow the administrator guidelines.
A.TIME	*It is assumed that the environment provides the TOE with reliable time.* OE.TIME ensures that the environment provides the TOE with reliable time.
A.KEYS	*It is assumed that random bits provided by the underlying platform are of good quality and have sufficient entropy.* OE.KEYS ensures that the platform provides random bits of good quality.
A.SYSLOG_PROTECTION	*It is assumed that the Operational Environment ensures the confidentiality and integrity of audit data transmitted from the TOE to external Syslog servers through secure network configurations and mechanisms.* *OE.SYSLOG_PROTECTION satisfies this assumption by requiring the environment to provide secure and trusted communication channels for audit data transmission.*

Table 11: Rationale between Objectives and SPDs

5. EXTENDED COMPONENTS DEFINITION

This chapter defines security components for the TOE not already defined in CC part 2. The following extended component has been included in this Security Target because the Common Criteria components were found to be insufficient as stated.

5.1. CLASS FAU: SECURITY AUDIT

5.1.1. PROTECTED AUDIT EVENT STORAGE (FAU_STG_EXT)

Family behaviour
This component defines the requirements for the TSF to be able to securely transmit audit data between the TOE and an external IT entity.

Component levelling

FAU_STG_EXT.1 Protected audit event storage requires the TSF to use a trusted channel implementing a secure protocol.
FAU_STG_EXT.2 Counting lost audit data requires the TSF to provide information about audit records affected when the audit log becomes full.
FAU_STG_EXT.3 Action in case of possible audit data loss requires the TSF to generate a warning before the audit trail exceeds the local storage capacity.
FAU_STG_EXT.4 Protected Local audit event storage for distributed TOEs requires the TSF to use a trusted channel to protect audit transfer to another TOE component.
FAU_STG_EXT.5 Protected Remote audit event storage for distributed TOEs requires the TSF to use a trusted channel to protect audit transfer to another TOE component.

Management: FAU_STG_EXT.1, FAU_STG_EXT.2, FAU_STG_EXT.3, FAU_STG_EXT.4, FAU_STG_EXT.5
The following actions could be considered for the management functions in FMT:

a) The TSF shall have the ability to configure the cryptographic functionality.

Audit: FAU_STG_EXT.1, FAU_STG_EXT.2, FAU_STG_EXT.3, FAU_STG_EXT.4. FAU_STG_EXT.5
The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) No audit necessary.

5.1.1.1. FAU_STG_EXT.1 PROTECTED AUDIT EVENT STORAGE

Hierarchical to:	No other components
Dependencies:	FAU_GEN.1 Audit data generation FTP_ITC.1 Inter-TSF Trusted Channel

FAU_STG_EXT.1.1 The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.1

FAU_STG_EXT.1.2 The TSF shall be able to store generated audit data on the TOE itself. In addition [selection:

- *The TOE shall consist of a single standalone component that stores audit data locally,*
- *The TOE shall be a distributed TOE that stores audit data on the following TOE components: [assignment: identification of TOE components],*
- *The TOE shall be a distributed TOE with storage of audit data provided externally for the following TOE components: [assignment: list of TOE components that do not store audit data locally and the other TOE components to which they transmit their generated audit data]*.

FAU_STG_EXT.1.3 The TSF shall [selection: *drop new audit data, overwrite previous audit records according to the following rule: [assignment: rule for overwriting previous audit records], [assignment: other action]*] when the local storage space for audit data is full.

5.2. CLASS FCS: CRYPTOGRAPHIC SUPPORT

5.2.1. HTTPS PROTOCOL (FCS_HTTPS_EXT.1)

Family behaviour
Components in this family define the requirements for protecting remote management sessions between the TOE and a Security Administrator. This family describes how HTTPS will be implemented. This is a new family defined for the FCS Class.

Component levelling

FCS_HTTPS_EXT.1 HTTPS requires that HTTPS be implemented according to RFC 2818 and supports TLS.

Management: FCS_HTTPS_EXT.1
The following actions could be considered for the management functions in FMT:

a) There are no management activities foreseen.

Audit: FCS_HTTPS_EXT.1
The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) There are no auditable events foreseen.

5.2.1.1. FCS_HTTPS_EXT.1 HTTPS PROTOCOL

Hierarchical to:	No other components
Dependencies:	[FCS_TLSC_EXT.1 TLS Client Protocol, or FCS_TLSS_EXT.1 TLS Server Protocol]

FCS_HTTPS_EXT.1.1 The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2 The TSF shall implement the HTTPS protocol using TLS.

5.2.2. SSH CLIENT PROTOCOL (FCS_SSHC_EXT)

Family behaviour
The component in this family addresses the ability for a client to use SSH to protect data between the client and a server using the SSH protocol.

Component level

FCS_SSHC_EXT.1 SSH Client requires that the client side of SSH be implemented as specified.

Management: FCS_SSHC_EXT.1
The following actions could be considered for the management functions in FMT:
a) There are no management activities foreseen.

Audit: FCS_SSHC_EXT.1
The following actions should be considered for audit if FAU_GEN Security audit data generation is included in the PP/ST:
a) There are no auditable events foreseen.

5.2.2.1. FCS_SSHC_EXT.1 SSH CLIENT PROTOCOL

Hierarchical to:	No other components
Dependencies:	FCS_CKM.1Cryptographic KeyGeneration FCS_CKM.2Cryptographic KeyEstablishment FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption) FCS_COP.1/SigGen Cryptographic operation (Signatur e Generation and Verification) FCS_COP.1/Hash Cryptographic operation (HashAlgorithm) FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm) FCS_RBG_EXT.1 Random Bit Generation

FCS_SSHC_EXT.1.1 The TSF shall implement the SSH protocol in accordance with RFCs 4251, 4252, 4253, 4254, 5656, and 6668.

FCS_SSHC_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based and no other method.

FCS_SSHC_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than 32 768 bytes in an SSH transport connection are dropped.

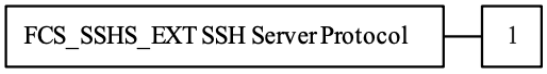
FCS_SSHC_EXT.1.4 The TSF shall ensure that within SSH connections the same session keys are used no longer than one hour or one gigabyte of data, whichever occurs first; rekey thereafter.

FCS_SSHC_EXT.1.5 The TSF shall authenticate the SSH server using a local database of host names and public keys and no other method as per RFC 4251 § 4.1.

5.2.3. SSH SERVER PROTOCOL (FCS_SSHS_EXT)

Family behaviour
The component in this family addresses the ability for a server to offer SSH to protect data between a client and the server using the SSH protocol.

Component levelling



FCS_SSHS_EXT.1 SSH Server requires that the server side of SSH be implemented as specified.

Management: FCS_SSHS_EXT.1
The following actions could be considered for the management functions in FMT: a) There are no management activities foreseen.

Audit: FCS_SSHS_EXT.1
The following actions should be considered for audit if FAU_GEN Security audit data generation is included in the PP/ST:
a) There are no auditable events foreseen.

5.2.3.1. FCS_SSHS_EXT.1 SSH SERVER PROTOCOL

Hierarchical to:	No other components
Dependencies:	FCS_CKM.1 Cryptographic Key Generation FCS_CKM.2 Cryptographic Key Establishment FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption) FCS_COP.1/SigGen Cryptographic operation (Signatur e Generation and Verification) FCS_COP.1/Hash Cryptographic operation (HashAlgorithm) FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm) FCS_RBG_EXT.1 Random Bit Generation

FCS_SSHS_EXT.1.1 The TSF shall implement the SSH protocol in accordance with RFCs 4251, 4252, 4253, 4254, 5656, and 6668.

FCS_SSHS_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, password-based, keyboard-interactive and no other method.

FCS_SSHS_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than 32 768 bytes in an SSH transport connection are dropped.

FCS_SSHS_EXT.1.4 The TSF shall accept only the encryption algorithms aes128-ctr, aes256-ctr, aes128-gcm, aes256-gcm, aes192-ctr, and chacha20-poly1305.

FCS_SSHS_EXT.1.5 The TSF shall use the public-key algorithms rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, and ssh-ed25519 and reject others.

FCS_SSHS_EXT.1.6 The TSF shall use the MAC algorithms hmac-sha2-256, hmac-sha2-512, hmac-sha2-256-etm, hmac-sha2-512-etm, umac-128-etm, and umac-128 and reject others.

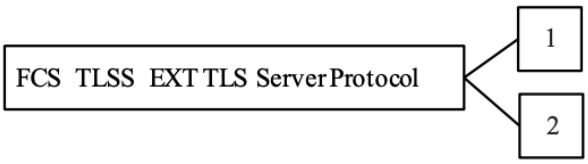
FCS_SSHS_EXT.1.7 The TSF shall allow only key exchange methods ecdh-sha2-nistp256, ecdh-sha2-nistp384, ecdh-sha2-nistp521, diffie-hellman-group-exchange-sha256, and curve25519-sha256.

FCS_SSHS_EXT.1.8 The TSF shall limit session key lifetime to one hour or one gigabyte of encrypted data, whichever comes first, and perform rekey after threshold.

5.2.4. TLS Server Protocol (FCS_TLSS_EXT)

Family behaviour
The component in this family addresses the ability for a server to use TLS to protect data between a client and the server using the TLS protocol.

Component levelling



FCS_TLSS_EXT.1 TLS Server requires that the server side of TLS be implemented as specified.
FCS_TLSS_EXT.2: TLS Server requires the mutual authentication be included in the TLS implementation.

Management: FCS_TLSS_EXT.1
The following actions could be considered for the management functions in FMT:
a) There are no management activities foreseen.

Audit: FCS_TLSS_EXT.1
The following actions should be considered for audit if FAU_GEN Security audit data generation is included in the PP/ST:
a) There are no auditable events foreseen.

5.2.4.1. FCS_TLSS_EXT.1 TLS Server Protocol without Mutual Authentication

Hierarchical to:	No other components
Dependencies:	FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.2 Cryptographic Key Establishment
FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption)
FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)
FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation

FCS_TLSS_EXT.1.1 The TSF shall implement TLS 1.2 (RFC 5246) and TLS 1.3 (RFC 8446) and reject all other TLS and SSL versions.
The TLS 1.2 implementation shall support the following cipher suites:

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (RFC 5289)
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (RFC 5289)
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (RFC 5288)
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (RFC 5288)
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (RFC 7905)

and no other cipher suites.

The TLS 1.3 implementation shall support the following cipher suites:

- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_CHACHA20_POLY1305_SHA256

and no other cipher suites.

FCS_TLSS_EXT.1.2 The TSF shall deny connections from clients requesting **SSL 2.0**, **SSL 3.0**, **TLS 1.0**, and **TLS 1.1**.

FCS_TLSS_EXT.1.3 The TSF shall perform key establishment for TLS using:

- RSA with key sizes 2048 bits and 4096 bits
- Diffie-Hellman parameters with sizes 2048 bits and 4096 bits
- ECDHE using curves secp384r1, secp521r1, prime256v1 (NIST P-256), X25519, and X448

and no other curves.

Application note: prime256v1, X25519, and X448 were added; all provide equivalent or stronger security than the existing curves.

FCS_TLSS_EXT.1.4 The TSF shall support session resumption based on session IDs as defined in RFC 5246 (TLS 1.2).

5.3. CLASS FPT: PROTECTION OF THE TSF

5.3.1. TSF SELF-TEST (FPT_TST_EXT)

Family Behaviour
Components in this family address the requirements for self-testing the TSF for selected correct operation.

Component levelling

FPT_TST_EXT.1 TSF Self-Test requires a suite of self-tests to be run during initial start-up in order to demonstrate correct operation of the TSF.

Management: FPT_TST_EXT.1
The following actions could be considered for the management functions in FMT:
a) No management functions.

Audit: FPT_TST_EXT.1
The following actions should be considered for audit if FAU_GEN Security audit data generation is included in the PP/ST:
a) Indication that TSF self-test was completed
b) Failure of self-test

5.3.1.1. FPT_TST_EXT.1 TSF TESTING

Hierarchical to:	No other components
Dependencies:	No other components

FPT_TST_EXT.1.1 The TSF shall run a suite of the following self-tests [selection: *during initial start-up (on power on), periodically during normal operation, at the request of the authorised user, at the conditions [assignment: conditions under which self-tests should occur]*] to demonstrate the correct operation of the TSF: *[assignment: list of self-tests run by the TSF*].

5.4. RATIONAL FOR INCLUDING EXTENDED COMPONENTS

Explicit Component	Rationale
FAU_STG_EXT.1	The SFR is needed to describe the used protection of audit. It is taken from the well-established collaborative Protection Profile for Network Devices, version 2.2e.
FCS_HTTPS_EXT.1	The SFR is needed to describe the used HTTPS implementation.
FCS_SSHC_EXT.1	This SFR defines the security mechanisms for the SSH client functionality.
FCS_SSHS_EXT.1	This SFR defines the security mechanisms for the SSH server functionality.
FCS_TLSS_EXT.1	This SFR defines the security mechanisms for the TLS server functionality.
FPT_TST_EXT.1	The SFR is needed to describe the TSF testing. It is taken from the well-established collaborative Protection Profile for Network Devices, version 2.2e.

Table 12: Rationale for Extended Component

6. SECURITY REQUIREMENTS

This chapter describes the security requirements that consist of two groups of requirements:

- The security functional requirements (SFRs): a translation of the security objectives for the TOE into a standardized language;
- The security assurance requirements (SARs): a description of how assurance is to be gained that the TOE meets the SFRs.

6.1. SECURITY FUNCTIONAL REQUIREMENTS (SFRS)

The convention used for operations applied to the Security Functional Requirements is documented in section Notations and Formatting.

Functional Class	Functional Component	
FAU: Security audit	FAU_GEN.1	Audit data generation
	FAU_GEN.2	User identity association
	FAU_STG_EXT.1	External audit trail storage
FCS: Cryptographic support	FCS_CKM.1	Used to generate cryptographic keys for TLS and SSH
	FCS_CKM.2	Used to for cryptographic key establishment for TLS and SSH
	FCS_CKM.4	Used to for cryptographic key destruction for TLS and SSH
	FCS_COP.1/DataEncryption	Used for encrypting SSH and TLS traffic
	FCS_COP.1/SigGen	Used for signing SSH and TLS traffic
	FCS_COP.1/Hash	Used for hashing SSH and TLS traffic
	FCS_COP.1/KeyedHash	Used for hashing keys in SSH and TLS traffic
	FCS_HTTPS_EXT.1	HTTPS Protocol
	FCS_SSHC_EXT.1	SSH client requirements
	FCS_SSHS_EXT.1	SSH server requirements
	FCS_TLSS_EXT.1	TLS server requirements
FIA: Identification and authentication	FIA_ATD.1	User attribute definition
	FIA_UAU.2	User authentication before any action
	FIA_UAU.7	Protected authentication feedback

Functional Class	Functional Component	
	FIA_UID.2	User identification before any action
	FIA_USB.1	User-subject binding
FMT: Security management	FMT_MOF.1	Management of Security Functions Behaviour
	FMT_MTD.1	Management of TSF data
	FMT_SMF.1	Specification of Management Functions
	FMT_SMR.1	Security roles
FPT: Protection of the TSF	FPT_ITT.1	Basic internal TSF data transfer protection
	FPT_STM.1	Reliable time stamps
	FPT_TST_EXT.1	TSF testing
FTA: TOE access	FTA_TAB.1	Display banner
FTP: Trusted Path/Channels	FTP_TRP.1	Trusted path

Table 13: Security Functional Requirements

6.1.1. SECURITY AUDIT (FAU)

6.1.1.1. FAU_GEN.1 AUDIT DATA GENERATION

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;
b) All auditable events for the [*not specified*] level of audit; and
c) [**Additional auditable events in Table 14 Auditable Events**].

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Time/date, Type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and
b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [**additional information listed in the following table**].

SFR	Auditable Event	Additional Audit Record Contents
FAU_GEN.1	None.	
FAU_GEN.2	None.	
FAU_STG_EXT.1	None.	

SFR	Auditable Event	Additional Audit Record Contents
FCS_CKM.1	None.	
FCS_CKM.2	None.	
FCS_CKM.4	None.	
FCS_COP.1/DataEncryption	None.	
FCS_COP.1/SigGen	None.	
FCS_COP.1/Hash	None.	
FCS_COP.1KeyedHash	None.	
FCS_HTTPS_EXT.1	None.	
FCS_SSHC_EXT.1	None.	
FCS_SSHS_EXT.1	None.	
FCS_TLSS_EXT.1	None.	
FIA_ATD.1	None.	
FIA_UAU.2	All use of the authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None.	
FIA_UID.2	All use of the user identification mechanism, including the user identity provided.	Provided user identity (for HTTPS and SSH interfaces only). For REST API authentication failures, user identity is not logged.
FIA_USB.1	Successful and failure of binding of user security attributes to a subject (e.g., creation of a subject).	Provided user identity whose attributes are attempting to be bound.
FMT_MOF.1	All modifications in the behavior of the functions in the TSF.	Provided user identity who performs the function.
FMT_MTD.1	All modifications to the values of TSF data.	Provided user identity who performs the function.
FMT_SMF.1	Use of the management functions.	Provided user identity who performs the function.
FMT_SMR.1	Every use of the rights of role.	Provided user identity who performs the function. Provided user identity who performs the modifications.
FPT_ITT.1	None.	
FPT_STM.1	Providing a timestamp.	None.
FPT_TST_EXT.1	Execution of the TSF self-tests and the results of the tests.	None.
FTA_TAB.1	None.	
FTP_TRP.1	All attempted uses of the trusted path functions. Identification of the user associated with all trusted path invocations, if available.	None.

Table 14: Auditable Events

6.1.1.2. FAU_GEN.2 User identity association

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

6.1.1.3. FAU_STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1.1 The TSF shall be able to transmit the generated audit data to an external IT entity using a communication channel protected by the Operational Environment**,** which ensures confidentiality and integrity of the transmitted data according to {OE.SYSLOG_PROTECTION}.

FAU_STG_EXT.1.2 The TSF shall be able to store generated audit data on the TOE itself. In addition [

- *The TOE shall consist of a single standalone component that stores audit data locally*

].

FAU_STG_EXT.1.3 The TSF shall [*overwrite previous audit records according to the following rule: [**overwrite the data present in the oldest audit file**]]* when the local storage space for audit data is full.

Application note: The SFR is taken from cPP ND v2.2E. The TOE generates and stores audit data locally and supports configuration to forward this data to an external Syslog server. The confidentiality and integrity of audit data in transit to the external Syslog server are provided by the Operational Environment, which is responsible for establishing a trusted communication channel. The TOE does not itself implement TLS for Syslog transmission. If the secure channel between the TOE and the Syslog server becomes unavailable, the TOE continues to store audit events locally according to FAU_STG_EXT.1.2 and FAU_STG_EXT.1.3. The administrator may later retrieve or re-export these events once the external secure channel is restored.

6.1.2. Cryptographic Support (FCS)

6.1.2.1. FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1.1 The TSF shall generate asymmetric cryptographic keys in accordance with a specified cryptographic key generation algorithm: [

- *RSA schemes using cryptographic key sizes of 2048-bit ~~or greater~~ and 4096-bit that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;*
- *ECC schemes using 'NIST curves' [P-256, P-384, P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4;*

] ~~and specified cryptographic key sizes [assignment: *cryptographic key sizes*] that meet the following: [assignment: *list of standards*]~~.

Application note: The SFR is taken from cPP ND v2.2E.

6.1.2.2. FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1 The TSF shall **perform** cryptographic **key establishment** in accordance with a specified cryptographic key **establishment** method: [

- *RSA-based key establishment schemes that meet the following: RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1";*
- *Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";*

] ~~that meets the following: [assignment: *list of standards*]~~.

Application note: The SFR is taken from cPP ND v2.2E.

6.1.2.3. FCS_CKM.4 Cryptographic Key Destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method

- For plaintext keys in volatile storage, the destruction shall be executed by a [*single overwrite consisting of **[zeroes]***];
- For plaintext keys in non-volatile storage, the destruction shall be executed by the invocation of an interface provided by a part of the TSF that [
    - *logically addresses the storage location of the key and performs a* [*single*], ***[one]**-pass*] overwrite consisting of [*zeroes*]];

that meets the following: *No Standard*.

Application note: The SFR is taken from cPP ND v2.2E.

6.1.2.4. FCS_COP.1/DataEncryption Cryptographic Operation (AES Data Encryption/Decryption)

FCS_COP.1.1/DataEncryption The TSF shall perform *encryption/decryption* in accordance with a specified cryptographic algorithm *AES used in [CTR, GCM*] *mode* and cryptographic key sizes [*128 bits, 192 bits, 256 bits*] that meet the following: *AES as specified in ISO 18033-3,* [*CTR as specified in ISO 10116, GCM as specified in ISO 19772*].

Application note: The SFR is taken from cPP ND v2.2E.

6.1.2.5. FCS_COP.1/SigGen Cryptographic Operation (Signature Generation and Verification)

FCS_COP.1.1/SigGen The TSF shall perform *cryptographic signature services (generation and verification)* in accordance with a specified cryptographic algorithm [

- *RSA Digital Signature Algorithm and cryptographic key sizes (modulus)* [***2048 bits and 4096 bits***],
- *Elliptic Curve Digital Signature Algorithm and cryptographic key sizes* [***256 bits, 384 bits and 512 bits***]

]

that meet the following: [

- *For RSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3,*

- *For ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" [P-256, P-384, P-521]; ISO/IEC 14888-3, Section 6.4*

].

Application note: The SFR is taken from cPP ND v2.2E.

6.1.2.6. FCS_COP.1/HASH CRYPTOGRAPHIC OPERATION (HASH ALGORITHM)

FCS_COP.1.1/Hash The TSF shall perform *cryptographic hashing services* in accordance with a specified cryptographic algorithm [*SHA-1, SHA-256*] ~~and cryptographic key sizes~~ and message digest sizes [*160, 256*] bits that meet the following: *ISO/IEC10118-3:2004.*

Application note: The SFR is taken from cPP ND v2.2E.

6.1.2.7. FCS_COP.1/KEYEDHASH CRYPTOGRAPHIC OPERATION (KEYED HASH ALGORITHM)

FCS_COP.1.1/KeyedHash The TSF shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm [*HMAC-SHA-256*, ***HMAC-SHA-256-ETM, HMAC-SHA-512-ETM, UMAC-128, UMAC-128-ETM***] and cryptographic key sizes [*256 bits for HMAC-SHA-256*, ***HMAC-SHA-256-ETM, HMAC-SHA-512-ETM, 128 bits for UMAC-128 and UMAC-128-ETM***] and message digest sizes [*256 bits for HMAC-SHA-256*, ***HMAC-SHA-256-ETM, HMAC-SHA-512-ETM, 128 bits for UMAC-128 and UMAC-128-ETM***] that meet the following: ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".

Application note: The SFR is taken from cPP ND v2.2E. The additional hmac-sha2-256-etm, hmac-sha2-512-etm, umac-128-etm, and umac-128 have been added as refinements. These added data integrity MAC algorithms do not provide less security compared to the selected options in the cPP and are therefore acceptable refinements.

6.1.2.8. FCS_HTTPS_EXT.1 HTTPS PROTOCOL

FCS_HTTPS_EXT.1.1 The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2 The TSF shall implement the HTTPS protocol using TLS.

6.1.2.9. FCS_SSHC_EXT.1 SSH CLIENT PROTOCOL

FCS_SSHC_EXT.1.1 The TSF shall implement the SSH protocol in accordance with RFCs 4251, 4252, 4253, 4254, 5656, and 6668.

FCS_SSHC_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based and no other method.

FCS_SSHC_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than 32 768 bytes in an SSH transport connection are dropped.

FCS_SSHC_EXT.1.4 The TSF shall ensure that within SSH connections the same session keys are used no longer than one hour or one gigabyte of data, whichever occurs first; rekey thereafter.

FCS_SSHC_EXT.1.5 The TSF shall authenticate the SSH server using a local database of host names and public keys and no other method as per RFC 4251 § 4.1.

6.1.2.10. FCS_SSHS_EXT.1 SSH Server Protocol

FCS_SSHS_EXT.1.1 The TSF shall implement the SSH protocol in accordance with RFCs 4251, 4252, 4253, 4254, 5656, and 6668.

FCS_SSHS_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, password-based, keyboard-interactive and no other method.

FCS_SSHS_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than 32 768 bytes in an SSH transport connection are dropped.

FCS_SSHS_EXT.1.4 The TSF shall accept only the encryption algorithms aes128-ctr, aes256-ctr, aes128-gcm, aes256-gcm, aes192-ctr, and chacha20-poly1305.

FCS_SSHS_EXT.1.5 The TSF shall use the public-key algorithms rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, and ssh-ed25519 and reject others.

FCS_SSHS_EXT.1.6 The TSF shall use the MAC algorithms hmac-sha2-256, hmac-sha2-512, hmac-sha2-256-etm, hmac-sha2-512-etm, umac-128-etm, and umac-128 and reject others.

FCS_SSHS_EXT.1.7 The TSF shall allow only key exchange methods ecdh-sha2-nistp256, ecdh-sha2-nistp384, ecdh-sha2-nistp521, diffie-hellman-group-exchange-sha256, and curve25519-sha256.

FCS_SSHS_EXT.1.8 The TSF shall limit session key lifetime to one hour or one gigabyte of encrypted data, whichever comes first, and perform rekey after threshold.

6.1.2.11. FCS_TLSS_EXT.1 TLS Server Protocol without Mutual Authentication

FCS_TLSS_EXT.1.1 The TSF shall implement TLS 1.2 (RFC 5246) and TLS 1.3 (RFC 8446) and reject all other TLS and SSL versions.
The TLS 1.2 implementation shall support the following cipher suites:

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (RFC 5289)
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (RFC 5289)
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (RFC 5288)
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (RFC 5288)
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (RFC 7905)

and no other cipher suites.

The TLS 1.3 implementation shall support the following cipher suites:

- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_CHACHA20_POLY1305_SHA256

and no other cipher suites.

FCS_TLSS_EXT.1.2 The TSF shall deny connections from clients requesting **SSL 2.0**, **SSL 3.0**, **TLS 1.0**, and **TLS 1.1**.

FCS_TLSS_EXT.1.3 The TSF shall perform key establishment for TLS using:

- RSA with key sizes 2048 bits and 4096 bits
- Diffie-Hellman parameters with sizes 2048 bits and 4096 bits
- ECDHE using curves secp384r1, secp521r1, prime256v1 (NIST P-256), X25519, and X448

and no other curves.

Application note: prime256v1, X25519, and X448 were added; all provide equivalent or stronger security than the existing curves.

FCS_TLSS_EXT.1.4 The TSF shall support session resumption based on session IDs as defined in RFC 5246 (TLS 1.2).

6.1.3. IDENTIFICATION AND AUTHENTICATION (FIA)

6.1.3.1. FIA_ATD.1 USER ATTRIBUTE DEFINITION

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: [

- **Authentication configuration (either remote authentication or local password);**
- **Role permissions**.

].

6.1.3.2. FIA_UAU.2 USER AUTHENTICATION BEFORE ANY ACTION

FIA_UAU.2.1 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

6.1.3.3. FIA_UAU.7 PROTECTED AUTHENTICATION FEEDBACK

FIA_UAU.7.1 The TSF shall provide only [**obscured feedback**] to the user while the authentication is in progress.

6.1.3.4. FIA_UID.2 USER IDENTIFICATION BEFORE ANY ACTION

FIA_UID.2.1 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

6.1.3.5. FIA_USB.1 USER-SUBJECT BINDING

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: [**Role**].

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: [**A user can only have one role, but a role can be assigned to many users**].

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: [**The role permissions do not change during a session**].

6.1.4. SECURITY MANAGEMENT (FMT)

6.1.4.1. FMT_MOF.1 MANAGEMENT OF SECURITY FUNCTIONS BEHAVIOUR

FMT_MOF.1.1 The TSF shall restrict the ability to [*determine the behaviour of, disable, enable, and modify the behaviour of*] the functions [**management functions identified in FMT_SMF.1**] to [**management roles identified in FMT_SMR.1**].

6.1.4.2. FMT_MTD.1 MANAGEMENT OF TSF DATA

FMT_MTD.1.1 The TSF shall restrict the ability to [***manage***] the [**general TSF data**] to [**authorized administrators**].

6.1.4.3. FMT_SMF.1 SPECIFICATION OF MANAGEMENT FUNCTIONS

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [

- **Ability to configure the access banner**
- **Ability to modify the behaviour of the transmission of audit data to an external IT entity;**
- **Ability to configure and modify roles;**
- **Ability to configure and modify user accounts;**
- **Ability to manage threat detections;**
- **Ability to configure and modify system configurations/settings**.

].

6.1.4.4. FMT_SMR.1 SECURITY ROLES

FMT_SMR.1.1 The TSF shall maintain the roles [**Super Admin, Auditor, Read Only, Restricted Admin, Security Analyst, Admin, Settings Admin**].

FMT_SMR.1.2 The TSF shall be able to associate users with roles.

6.1.5. PROTECTION OF THE TSF (FPT)

6.1.5.1. FPT_ITT.1 BASIC INTERNAL TSF DATA TRANSFER PROTECTION

FPT_ITT.1.1 The TSF shall protect TSF data from [*disclosure*] **and detect its modification** when it is transmitted between separate parts of the TOE **through the use of SSH**.

6.1.5.2. FPT_STM.1 RELIABLE TIME STAMPS

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps.

6.1.5.3. FPT_TST_EXT.1 TSF TESTING

FPT_TST_EXT.1.1 The TSF shall run a suite of the following self-tests [*during initial start-up (on power on)*] to demonstrate the correct operation of the TSF: [**Integrity check to verify that core libraries have not been altered**].

6.1.6. TOE ACCESS (FTA)

6.1.6.1. FTA_TAB.1 TOE ACCESS BANNERS

FTA_TAB.1.1 Before establishing an administrative user session the TSF shall display a Security Administrator-specified advisory notice and consent warning message regarding use of the TOE.

Application note: The SFR is taken from cPP ND v2.2E.

6.1.7. TRUSTED PATH/CHANNELS (FTP)

6.1.7.1. FTP_TRP.1 TRUSTED PATH

FTP_TRP.1.1 The TSF shall provide a communication path between itself and [*remote, local*] users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from [*disclosure, **[and detection of modification of the communicated data]***].

FTP_TRP.1.2 The TSF shall permit [*local users, remote users*] to initiate communication via the trusted path.

FTP_TRP.1.3 The TSF shall require the use of the trusted path for [*initial user authentication, **[and all local administration over SSH and all remote administration over HTTPS/TLS]***].

6.2. SECURITY REQUIREMENTS RATIONALE

6.2.1. RELATION BETWEEN SFRS AND SECURITY OBJECTIVES

The following tracing shows which SFRs address which security objectives for the TOE.

Objectives / Requirements	O.ACCESS	O.AUDIT	O.CRYPTO	O.IDAUTH	O.MANAGE	O.PROTECT	O.TEST	O.TIME	O.M2MPROECT
FAU_GEN.1		X							
FAU_GEN.2		X							
FAU_STG_EXT.1		X							
FCS_CKM.1			X						
FCS_CKM.2			X						
FCS_CKM.4			X						
FCS_COP.1/DataEncryption			X						

Objectives / Requirements	O.ACCESS	O.AUDIT	O.CRYPTO	O.IDAUTH	O.MANAGE	O.PROTECT	O.TEST	O.TIME	O.M2MPROECT
FCS_COP.1/SigGen			X						
FCS_COP.1/Hash			X						
FCS_COP.1/KeyedHash			X						
FCS_SSHC_EXT.1			X			X			X
FCS_SSHS_EXT.1			X			X			X
FCS_TLSS_EXT.1			X			X			
FIA_ATD.1				X					
FIA_UAU.2	X			X					
FIA_UAU.7				X					
FIA_UID.2	X			X					
FIA_USB.1				X					
FMT_MOF.1	X				X	X			
FMT_MTD.1					X				
FMT_SMF.1					X	X			
FMT_SMR.1				X		X			
FPT_ITT.1						X			
FPT_STM.1		X						X	
FPT_TST_EXT.1							X		
FTA_TAB.1	X								
FTP_TRP.1			X						

Table 15: Tracing of functional requirements to Objectives

The following set of justifications shows that all security objectives for the TOE are effectively addressed by the SFRs.

Security Objectives	Security Functional Requirement Rationale
O.ACCESS	*The TOE must allow only authorized administrators to access only appropriate TOE functions and data. Also, the TOE must display an initial banner before users log into the TOE. The initial banner must contain restrictions of use, legal agreements, or any other appropriate information to which users make consent by accessing the TOE.* FIA_UID.2 and FIA_UAU.2 ensure that administrators are identified and authenticated prior to being allowed access to TOE security management functionality. FMT_MOF.1 ensures that only authorized administrators have access to security management functions. FTA_TAB.1 addresses the display of the banner before a session is established.

O.AUDIT	*The TOE shall record, and export security-related events associated with users to enable tracing of responsibilities for security-related events.* FAU_GEN.1 defines the set of events that the TOE must be capable of recording. This requirement ensures that the administrator can audit any security relevant event that takes place in the TOE. FAU_GEN.2 ensures that the audit records associate an administrator identity with the auditable event. In the case of authorized administrators, the association is accomplished with the username. In all other cases, the association is based on the source identifier, which is presumed to be the correct identity, but cannot be confirmed since these subjects are not authenticated. FAU_STG_EXT.1 requires that the audit logs are transmitted securely to an external Syslog server via interfaces provided by the Operational Environment. The TOE is responsible for generating and exporting audit records; confidentiality and integrity of the transmitted audit data are ensured by the Operational Environment as defined in OE.SYSLOG_PROTECTION. FPT_STM.1 supports the audit functionality by ensuring that the TOE can obtain a time stamp for use in recording audit events.
O.CRYPTO	*The TOE must protect the confidentiality and integrity of data passed between itself and authorized administrators.* FCS_CKM.1, FCS_CKM.2, FCS_CKM.4 ensures that cryptographic keys are correctly generated, established and destroyed, which is the foundation for secure communication. FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash specifies the cryptographic operations. FCS_HTTPS_EXT.1, FCS_TLSS_EXT.1 ensures that the TLS communications between the TOE and the endpoints are secure by implementing the secure communication as defined in the SFRs. FTP_TRP.1 specifies the use of that cryptography between the TOE and the remote administrators. Protection of audit data transmitted to the Syslog server is not covered by these SFRs and is instead ensured by the Operational Environment through OE.SYSLOG_PROTECTION.
O.IDAUTH	*The TOE must be able to identify and authenticate authorized administrators prior to allowing access to TOE security management functions.* FIA_ATD.1 ensures that the data required to identify and authenticate administrators is maintained by the TOE. FIA_UID.2 and FIA_UAU.2 ensure that administrators are identified and authenticated prior to being granted access to TOE security management functionality. FIA_UAU.7 ensures obscured password feedback when the administrator is logging in, thus serving to protect that data.

	FIA_USB.1 requires the TOE to bind the username to each management session upon successful I&A. FMT_SMR.1 supports the objective by providing roles which are used to provide administrators access to TOE security functionality.
O.MANAGE	*The TOE must include a set of functions that allow effective management of its functions and data. The TOE will provide mechanisms to ensure that only administrators are able to log in and configure the TOE and provide protections for logged-in administrators.* FMT_MOF.1 provides functionality to manage the behaviour of the functions of the TOE restricted to authorized administrators and identifies the role required for specific actions. FMT_MTD.1 requires that the ability to manipulate TOE content is restricted to authorized administrators and identifies the role required for specific actions. FMT_SMF.1 provides the management functions supporting the specific security management claims and limiting access to that functionality to authorized administrators.
O.PROTECT	*The TOE must protect itself and its resources from unauthorized modifications and access to its functions and data.* FCS_HTTPS_EXT.1, FCS_SSHC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSS_EXT.1 ensures that the SSH and TLS communications between the TOE and the endpoints are secure by implementing the secure communication as defined in the SFRs. FMT_MOF.1, FMT_SMF.1 and FMT_SMR.1 ensure that access to TOE security functions is limited to authorized administrators. FPT_ITT.1 requires the TOE to protect TSF the data and ensure its confidentiality and integrity when the data is transmitted between components of the TOE. Protection of audit data sent externally to Syslog servers is provided by the Operational Environment (OE.SYSLOG_PROTECTION).
O.TEST	*The TOE will provide the capability to test some subset of its security functionality to ensure it is operating properly.* FPT_TST_EXT.1 performs self-test to ensure the TOE is operating correctly and all functions are available and enforced.
O.TIME	*The TOE shall provide reliable time stamps. The administrator can configure the TOE to synchronize its clocks with trusted NTP servers using unauthenticated NTP protocol. The integrity and authenticity of the external time source are ensured by the Operational Environment.* FPT_STM.1 requires that the TOE be able to provide reliable time stamps for its own use. Time stamps include date and time and are reliable in that they are always available to the TOE.
O.M2MPROTECT	*The TOE must protect machine-to-machine communication between the Sensor and Brain by ensuring secure transmission of metadata, telemetry, and configuration updates. This shall be achieved using cryptographic*

	mechanisms, such as SSH, that provide authentication, encryption, and integrity protection. FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 require that the TOE uses SSH to protect machine-to-machine communication between the Sensor and Brain.

Table 16: Rationale between Objectives and SFRs

6.3. SFR DEPENDENCIES

The table below shows the dependencies of the security functional requirement of the TOE and gives a rationale for each of them if they are included or not.

SFR	Dependency	Dependency Rationale
FAU_GEN.1	FPT_STM.1	Included
FAU_GEN.2	FAU_GEN.1	Included
	FIA_UID.1	Fulfilled by FIA_UID.2
FAU_STG_EXT.1	FAU_GEN.1	Included
	FTP_ITC.1	Fulfilled by OE.SYSLOG_PROTECTION
FCS_CKM.1	FCS_CKM.2 or FCS_COP.1	Included both FCS_CKM.2 and FCS_COP.1
	FCS_CKM.4	Included
FCS_CKM.2	FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1	Included FCS_CKM.1
	FCS_CKM.4	Included
FCS_CKM.4	FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1	Included FCS_CKM.1
FCS_COP.1/DataEncryption	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1	Included FCS_CKM.1
	FCS_CKM.4	Included
FCS_COP.1/SigGen	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1	Included FCS_CKM.1
	FCS_CKM.4	Included
FCS_COP.1/Hash	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1	Not fulfilled/applicable, since the Hash algorithms does not use any key.
	FCS_CKM.4	Not fulfilled/applicable, since the Hash algorithms does not use any key.
FCS_COP.1/KeyedHash	FTP_ITC.1 or FTP_ITC.2 or FCS_CKM.1	Included FCS_CKM.1
	FCS_CKM.4	Included
FCS_HTTPS_EXT.1	FCS_TLSC_EXT.1 or FCS_TLSS_EXT.1	Fulfilled by FCS_TLSS_EXT.1
FCS_SSHC_EXT.1	FCS_CKM.1	Included
	FCS_CKM.2	Included

SFR	Dependency	Dependency Rationale
	FCS_COP.1/DataEncryption	Included
	FCS_COP.1/SigGen	Included
	FCS_COP.1/Hash	Included
	FCS_COP.1/KeyedHash	Included
	FCS_RBG_EXT.1	Fulfilled by OE.KEYS
FCS_SSHS_EXT.1	FCS_CKM.1	Included
	FCS_CKM.2	Included
	FCS_COP.1/DataEncryption	Included
	FCS_COP.1/SigGen	Included
	FCS_COP.1/Hash	Included
	FCS_COP.1/KeyedHash	Included
	FCS_RBG_EXT.1	Fulfilled by OE.KEYS
FCS_TLSS_EXT.1	FCS_CKM.2	Included
	FCS_COP.1/DataEncryption	Included
	FCS_COP.1/SigGen	Included
	FCS_COP.1/Hash	Included
	FCS_COP.1/KeyedHash	Included
	FCS_RBG_EXT.1	Fulfilled by OE.KEYS
FIA_ATD.1	None	
FIA_UAU.2	FIA_UID.1	Fulfilled by FIA_UID.2
FIA_UAU.7	FIA_UAU.1	Included
FIA_UID.2	None	
FIA_USB.1	FIA_ATD.1	Included
FMT_MOF.1	FMT_SMR.1, FMT_SMF.1	Included
FMT_MTD.1	FMT_SMR.1, FMT_SMF.1	Included
FMT_SMF.1	None	
FMT_SMR.1	FIA_UID.1	Fulfilled by FIA_UID.2
FPT_ITT.1	None	
FPT_STM.1	None	
FPT_TST_EXT.1	None	
FTA_TAB.1	None	
FTP_TRP.1	None	

Table 17: SFR's dependencies and rationale

6.4. SECURITY ASSURANCE REQUIREMENTS (SARS)

The TOE assurance requirements for this ST consist of the requirements corresponding to the EAL2 level of assurance augmented with ALC_FLR.1, as defined in the CC Part 3.

The assurance components are summarized in the table below.

Assurance Class	Assurance Components
ADV: Development	ADV_ARC.1 Security architecture description
	ADV_FSP.2 Security-enforcing functional specification
	ADV_TDS.1 Basic design
AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	AGD_PRE.1 Preparative procedures

Assurance Class	Assurance Components
ALC: Life-cycle support	ALC_CMC.2 Use of a CM system
	ALC_CMS.2 Parts of the TOE CM coverage
	ALC_DEL.1 Delivery procedures
	ALC_FLR.1 Basic Flaw Remediation
ASE: Security Target evaluation	ASE_CCL.1 Conformance claims
	ASE_ECD.1 Extended components definition
	ASE_INT.1 ST introduction
	ASE_OBJ.2 Security objectives
	ASE_REQ.2 Derived security requirements
	ASE_SPD.1 Security problem definition
	ASE_TSS.1 TOE summary specification
ATE: Tests	ATE_COV.1 Evidence of coverage
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing - sample
AVA: Vulnerability assessment	AVA_VAN.2 Vulnerability analysis

Table 18: Assurance requirements

6.4.1. SECURITY ASSURANCE REQUIREMENTS RATIONAL

Dependencies within the EAL package selected (EAL2) for the security assurance requirements have been considered by the authors of CC Part 3 and are not analysed here again. The augmentation by flaw remediation, ALC_FLR.1, has no dependencies on other requirements. The security functional requirements in this Security Target do not introduce dependencies on any security assurance requirement; neither do the security assurance requirements in this Security Target introduce dependencies on any security functional requirement

The EAL2 level was also deemed sufficient because this will provide a necessary assurance for a product that is not directly exposed to external attackers, but still able to resist attacker with basic attack potential.

The assurance requirements of the EAL2 package provides a full Security Target and requires an analysis using a functional and interface specification and a basic description of the architecture of the TOE, which would give sufficient confidence in the design and architecture and for the evaluator to perform an analysis of the design and architecture for the vulnerability analysis

7. TOE SUMMARY SPECIFICATION

This chapter provides potential consumers of the TOE with a description of how the TOE satisfies all the Security Functional Requirements (SFRs) and is divided into the following security functions:

- Security Audit
- Cryptographic Support
- Identification and Authentication
- Security Management
- Protection of the TSF
- TOE Access
- Trusted channels

7.1. SECURITY AUDIT

The TOE records all events specified in Table 14, including all actions performed by the administrator. The identity is stored for each logged action; therefore, each action is tracible to an individual user. The records are stored in log files with default log file size of 10MB, and after that it is rolled over. Up to 7 files are allowed to be rolled over, and after that the oldest file is trashed. Size and number of log rollover are not configurable by the end user.

The local TOE storage has audit trail of all write action on important entities on TSF. This storage record will have the same level of auditing as the log file, with the added advantage of no log rollover and possible display of trail on TSF web view itself with no option for modification.

The following information is audited:

- Username
- Role
- Source IP
- Headend_addr
- Dvchost
- Version
- Result
- Message
- Timestamp

The Brain can be configured to export syslog records to an administrator-specified, external syslog server through interfaces provided by the Operational Environment. The Brain is responsible for generating and formatting audit records and invoking the export function. Protection of the transmitted audit data is ensured by the secure network mechanisms of the Operational Environment, as described in OE.SYSLOG_PROTECTION.

This section implements the following SFRs:

- FAU_GEN.1 Audit data generation
- FAU_GEN.2 User identity association
- FAU_STG_EXT.1 Protected Audit Event Storage

7.2. CRYPTOGRAPHIC SUPPORT

The TOE implements SSH and TLS/HTTPS which is covered in 7.7 Trusted channels.

The TOE includes OpenSSL 1.1.1f-1ubuntu2.24 which is used for the TLS implementation and underlying cryptographic operations for OpenSSH 8.2p1-4ubuntu0.13 which is used for SSH.

Cryptographic service	Cryptographic algorithms and key sizes	Usage / Purpose
Key generation	RSA 2048-bit and 4096-bit keys	TLS 1.2 TLS 1.3 SSHv2
	ECDSA P-256 keys	SSHv2
	EdDSA ed25519	SSHv2
	Diffie-Hellman keys	SSHv2
Key establishment	RSA-based, 2048-bit and 4096-bit keys	TLS 1.2 TLS 1.3 SSHv2
	ECDSA-based, P-256, P-384, P-521 keys	SSHv2
	diffie-hellman-group-exchange-sha256, curve25519-sha256	SSHv2
Encryption / Decryption	AES in CTR mode, 128, 192 and 256 bit keys	TLS 1.2 TLS 1.3 SSHv2
	AES in GCM mode, 128, 192 and 256 bit keys	TLS 1.2 TLS 1.3 SSHv2
	chacha20-poly1305	SSHv2
Signature Generation and Verification	RSA PKCS#1 v1.5 with SHA-1, SHA-256, using 2048-bit and 4096-bit keys	TLS 1.2 TLS 1.3
	ECDSA with SHA-256, SHA-384 and SHA-512 using NIST P-256, P-384, and P-521 curves.	TLS 1.2 TLS 1.3
Message Digest	SHA-1 & SHA-256	Signature Generation and Verification Keyed Hashing
Keyed Hashing	HMAC-SHA-256 HMAC-SHA-512 HMAC-SHA-256etm HMAC-SHA-512-etm UMAC-128-etm UMAC-128	TLS 1.2 TLS 1.3 SSHv2

Table 19: Cryptographic service mapping

This section implements the following SFRs:

- FCS_CKM.1 Cryptographic Key Generation
- FCS_CKM.2 Cryptographic Key Establishment
- FCS_CKM.4 Cryptographic Key Destruction

- FCS_COP.1/DataEncryption Cryptographic Operation (AES Data Encryption/Decryption)
- FCS_COP.1/SigGen Cryptographic Operation (Signature Generation and Verification)
- FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm)
- FCS_COP.1/KeyedHash Cryptographic Operation (Keyed Hash Algorithm)

7.2.1. SECURE SHELL VERSION 2 (SSHV2)

The TOE implements SSHv2 (telnet is disabled in the evaluated configuration) using OpenSSH 8.2p1-4ubuntu0.13, operating both as SSH server and SSH client.

When acting as an SSH server, the TOE negotiates connections according to the SSH protocol specifications defined in RFCs 4251, 4252, 4253, 4254, 5656, and 6668. During key exchange, the SSH client advertises all supported algorithms, and the TOE, acting as the SSH server, restricts negotiation to a defined secure subset in accordance with FCS_SSHS_EXT.1.

The TOE also enforces session rekeying as specified in RFC 4253 when the session reaches one hour of lifetime or one gigabyte of encrypted data, whichever occurs first. SSH connections are dropped if the TOE receives any packet larger than 32,768 bytes. After 4 to 6 consecutive failed authentication attempts (depending on the SSH client configuration), the TOE terminates the session and displays the message: “Too many authentication failures.” If a user fails the SSH login more than 12 times within 10 minutes, the TOE blocks the corresponding source IP address for 10 minutes and displays the message: “Connection reset by peer.”

When the TOE acts as an SSH server, it uses password authentication for interactive user sessions (CLI used for maintenance purposes) and key-based authentication for machine-to-machine interfaces (SSH tunnels). When the TOE acts as an SSH client (SSH tunnel), it uses key-based authentication. The TOE creates session keys following SSHv2 protocol and destroys sessions keys when the session is terminated by zeroization and deallocation in the RAM.

The SSHv2 implementation complies with the followings RFC: 4251, 4252, 4253, 4254, 5656, 6668.

The Vectra Brain SSHv2 implementation supports:

- Chacha20-poly1305
- encryption algorithms to ensure confidentiality of the session:
  - AES-128-GCM
  - AES-256-GCM
  - AES-128-CTR
  - AES-192-CTR
  - AES-256-CTR
- RSA 2048 and 4096-bits
- ecdsa-sha2-nistp256
- ssh-ed25519

This section implements the following SFRs:

- FCS_SSHC_EXT.1 SSH Client Protocol
- FCS_SSHS_EXT.1 SSH Server Protocol
- FPT_ITT.1 Basic internal TSF data transfer protection

- FTP_TRP.1 Trusted path

7.2.2. TRANSPORT LAYER SECURITY (TLS)

The TOE implements TLS 1.2 (RFC 5246) and TLS 1.3 (RFC 8446) using OpenSSL 1.1.1f-1ubuntu2.24 and supports HTTPS. TLS/HTTPS is used to secure Web UI connections and REST API communications.

The Brain implements HTTPS over TLS to support remote administration. TLS 1.2 (RFC 5246) and 1.3 (RFC 8446) are supported by the TOE.

The TOE implements the following ciphersuites:

- ECDHE-ECDSA-AES128-GCM-SHA256,
- ECDHE-RSA-AES128-GCM-SHA256,
- ECDHE-ECDSA-AES256-GCM-SHA384,
- ECDHE-RSA-AES256-GCM-SHA384,
- ECDHE-ECDSA-CHACHA20-POLY1305,
- ECDHE-RSA-CHACHA20-POLY1305,
- DHE-RSA-AES128-GCM-SHA256,
- DHE-RSA-AES256-GCM-SHA384

The TOE implements the following elliptic curves:

- secp384r1
- secp521r1
- prime256v1
- X25519
- X448

This section implements the following SFRs:

- FCS_TLSS_EXT.1 TLS Server Protocol without Mutual Authentication

7.3. IDENTIFICATION AND AUTHENTICATION

Administrators are provided with:

- Unique username
- Password
- Action based roles

The TOE can identify administrators by a unique ID and enforces their authentication before granting them access to any TSF management interfaces. The TOE is by default defined with the initial administrator role Super Admin.

The TOE requires each user to successfully identify and authenticate before access is granted to any management function. Each action is only allowed if the administrator is authorized to perform the actions based on roles assigned to them. A user can only have one role, but a role can be assigned to many users. A user role cannot be changed by the administrator while the user is still logged in.

Administrative access to the TOE is facilitated through the GUI, and API. The TOE mediates all administrative actions through these interfaces. Once a potential administrative user

attempts to access an administrative interface, the TOE prompts the user for a username and password. Only after the administrative user presents the correct authentication credentials will access to the TOE administrative functionality be granted.

When an administrator enters their password using the GUI, or API client, the password is not echoed back. The TOE displays only characters in “non-readable” form, so that the administrator password is obscured.

When the TOE acts as SSH server, it retrieves and presents its host public key for client validation. The TOE enforces mutual authentication when configured to do so by verifying the client’s public key against the locally stored list of authorized keys.

ECDA-based, RSA-based and EdDSA-based certificates can be generated on the TOE, or imported to the TOE.

The administrator can replace the existing key and certificate in the key store using the CLI. Only one key pair exists at any time. The TOE does not provide an explicit command to delete the key pair, as removal without replacement would invalidate the TLS configuration and disrupt access to the WebUI.

This section implements the following SFRs:

- FIA_ATD.1 User attribute definition
- FIA_UAU.2 User authentication before any action
- FIA_UID.2 User identification before any action
- FIA_UAU.7 Protected authentication feedback
- FIA_USB.1 User-subject binding

7.4. Security Management

The privileges for management of security functions for each role are clearly defined and enforced by the TOE. All administrative accounts are assigned one role, and every role must at least possess the “read-only” privilege, so all accounts are able to access TSF data and read the audit logs. When the administrator is created, the administrator will be mapped to one of the roles.

Abilities to disable, enable, determine, and modify configuration settings is determined by the roles (and the privileges therein) assigned to each account.

The TOE is configured to restrict the ability to perform the following management functions to authorized administrators:

- Management of roles
- Management of user accounts
- Management of threat detections
- System configurations/settings
- Configuration of access banner

The TOE is pre-configured with 7 default roles: Admin, Auditor, Read-Only, Restricted Admin, Security Analyst, Setting Admin, and Super Admin. The 'Super Admin' role is the only role with a user defined out of the box and grants permissions to create, edit, and delete other roles. These can be customized, or the customer can choose to create their own.

The following is a summary of each role:

- Super Admin: This role has all permissions enabled (both to view and edit) and is the only role that has a user assigned to it on initial configuration, that being the only predefined user, the ‘admin’ user.
- Read Only: This role has no edit permissions but does give the user the permissions to view detections and a majority of the settings within the system, though they cannot make any changes.
- Restricted Admin: This user can view/edit the detections and most of the configuration, but they do not have permissions to add/remove users or change roles, and certain setting that can impact the functionality of the system.
- Security Analyst: This user has the permissions to view/edit detections and have read-only views on the system configuration, but they are restricted from changing the system configuration.
- Admin: This user can view/edit most of the configuration with the exception of the User Accounts and Roles.
- Settings Admin: This user can make changes to the settings on the system but have no permissions to view/edit the detections on the system.
- Auditor: This role is a read-only role that can access Audit Events in the system, and objects such as Hosts and Detections, which are referenced in the Audit Events. This role is intended primarily for programmatic API access.

This section implements the following SFRs:

- FMT_MOF.1 Management of Security Functions Behaviour
- FMT_MTD.1 Management of TSF data
- FMT_SMF.1 Specification of Management Functions
- FMT_SMR.1 Security roles

7.5. PROTECTION OF THE TSF

The TOE is configured to prevent disclosure or modification of TSF data for connections between the TOE components, by using SSH. This TSF data would include data/config/updates packets transmitted between the TOE parts Vectra Brain and Sensor. More information about the SSH implementation can be found in the section 7.2.1.

The TOE provides a reliable source of date and time information used in audit event timestamps. All timestamps are in epoch (seconds), in UTC. The reliability of the time information depends on the accuracy of the underlying system clock on which the TOE operates.

The Brain component can be configured by an authorized administrator to synchronize its system clock with an external NTP server. By default, the Brain is configured to use ntp.ubuntu.com, though administrators may specify a different NTP server as needed. The Brain uses standard, unauthenticated NTP communication for time synchronization; therefore, the authenticity and integrity of NTP updates are ensured by the Operational Environment in accordance with OE.TIME. Each Sensor synchronizes its clock with the Brain using tlsdate over the SSH tunnel.

The TOE performs file system integrity checks during boot to verify that core libraries have not been altered. If a discrepancy is detected, the TOE transitions to a setup and provisioning state, presenting an error code for further analysis. This error code must be sent to Vectra Support for decryption and assessment. Based on the findings, Vectra may provide a one-time

whitelist code to allow the system to proceed with the boot process. Subsequent integrity check failures require new whitelist codes, and coordination with Vectra Support is necessary to ensure compliance and system functionality.

This section implements the following SFRs:

- FPT_ITT.1 Basic internal TSF data transfer protection
- FPT_STM.1 Reliable time stamps
- FPT_TST_EXT.1 TSF testing

7.6. TOE Access

The administrator can configure a banner which is displayed for interactive connections before signing in. It can display warnings or advisory notices that reflect the security policy of the organization.

This section implements the following SFRs:

- FTA_TAB.1 TOE access banners

7.7. Trusted channels

The TOE uses both SSH and TLS/HTTPS to protect data in transit. OpenSSH 8.2p1-4ubuntu0.13 is used for protecting SSH communication and OpenSSL 1.1.1f-1ubuntu2.24 is used for protecting TLS/HTTPS communication.

The TOE uses these mechanisms for the following purposes:

- - SSH protects communication between TOE components (Sensor and Brain).
- - TLS/HTTPS protects communication between the TOE and remote administrators accessing the Web UI or REST API.

The TOE can be configured to export audit data to an external Syslog server. The TOE is responsible only for generating and transmitting audit records through an export interface. The confidentiality and integrity protection of audit data transmitted to the external Syslog server are provided by the secure network mechanisms of the Operational Environment, as defined in OE.SYSLOG_PROTECTION.

The TOE uses SSH to provide the trusted path (with protection from disclosure and detection of modification) for all local administration sessions. The Brain uses TLS/HTTPS to provide the trusted path (with protection from disclosure and detection of modification) for all remote administration sessions.

All administrators have to present valid login credentials to perform any read and write operation on the TOE.

This section implements the following SFRs:

- FTP_TRP.1 Trusted path