Fidelis Network and Fidelis Deception v9.3.3 Security Target

Version 1.0
16 February 2021

Prepared for:

Fidelis Cybersecurity Inc.
4500 East West Highway, Suite 400
Bethesda, Maryland 20814

Prepared by:

Common Criteria Testing Laboratory
6841 Benjamin Franklin Drive, Columbia, Maryland 21046

LIST OF TABLES

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization.

The TOE is the Fidelis Network and Fidelis Deception v9.3.3 provided by Fidelis Cybersecurity Inc. that includes modules to discover, monitor, and protect all digital assets in an enterprise.

The Fidelis Network and Fidelis Deception is a collection of network security appliances that detect inappropriate and malicious network data based on aspects of the network traffic such as content, source, destination, application, and aspects of the communication channel. The Fidelis Network and Fidelis Deception is used to prevent the intrusion of attacks and to prevent the transmission of sensitive data, either as a result of an attack or insider threat. The Fidelis Network and Fidelis Deception analyzes network activity and can issue alerts of significant events. The Fidelis Network and Fidelis Deception collects and stores metadata from the network to allow a security analyst to view the context associated with alerts and to analyze network activity.

The focus of this evaluation is on the TOE functionality supporting the claims in the collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020, [CPP_ND_V2.2E] (See section 1.2 for specific version information). The security functionality specified in [CPP_ND_V2.2E] includes protection of communications between TOE components and trusted IT entities, identification and authentication of administrators, auditing of security-relevant events, ability to verify the source and integrity of updates to the TOE, and specifies NIST-validated cryptographic mechanisms.

The Security Target contains the following additional sections:

- TOE Description (Section 2)
- Security Problem Definition (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
- TOE Summary Specification (Section 6)
- Protection Profile Claims (Section 7)
- Rationale (Section 8).

1.1 Security Target, TOE and CC Identification

ST Title – Fidelis Network and Fidelis Deception v9.3.3 Security Target

ST Version – Version 1.0

ST Date – 16 February 2021

TOE Identification – Fidelis Network and Fidelis Deception v9.3.3

The TOE consists of the following Fidelis components:

- one or more Fidelis Network v9.3.3 CommandPost management consoles
- one or more Fidelis Network Collectors v9.3.3
- zero or more Fidelis Sandbox appliances, v9.3.3
- zero or more Decoy Server appliances, v9.3.3
- at least one of the following sensor appliances:
    - Fidelis Network Direct v9.3.3
    - Fidelis Network Internal v9.3.3
    - Fidelis Network Web v9.3.3

- Fidelis Network Mail v9.3.3

The CommandPost, Decoy Server, Collector and Sensor components are available in the models, as outlined in the following table:

Component	Appliance Models	Virtual Models
CommandPost	CommandPost appliance	CommandPost VM
Collector	Collector SA2 Collector XA2 Collector XA4 Collector Controller 2 Collector Controller 10G	Collector SA VM
Sensor	Direct 50 Direct 100 Direct 250 Direct 500 Direct 1000 Direct 2500 Direct 5000 Direct 10G	Direct VM
	Internal 1000 Internal 2500 Internal 5000 Internal 10G	Internal VM
	Web	Web VM
	Mail 250 Mail 500 Mail 1000 Mail 5000	Mail VM 250 Mail VM 500 Mail VM 1000 Mail VM 5000
Decoy Server	Decoy Server FDH-3000 FDH-1000	Decoy Server VM
Sandbox	Sandbox	N/A

The Sandbox component is available in a single appliance form factor.

1.2 Conformance Claims

This TOE is conformant to the following CC specifications:

- The SFRs have all been drawn from the Protection Profile (PP): collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020, [CPP_ND_V2.2E] and including the following optional SFRs: FAU_STG.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.2, FCO_CPC_EXT.1, FIA_X509_EXT.1/ITT, FPT_ITT.1, FPT_ITT.1/Join and the following selection-based SFRs: FAU_GEN_EXT.1, FAU_STG_EXT.4, FAU_STG_EXT.5, FCS_HTTPS_EXT.1, FCS_NTP_EXT.1, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1, FIA_X509_EXT.1/Rev, FIA_X509_EXT.2, FIA_X509_EXT.3, FMT_MOF.1/Functions, FMT_MTD.1/CryptoKeys.

- The following NIAP Technical Decisions apply to this PP and have been accounted for in the ST development and the conduct of the evaluation:
  - TD0572: NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers
  - TD0571: NiT Technical Decision for Guidance on how to handle FIA_AFL.1
  - TD0570: NiT Technical Decision for Clarification about FIA_AFL.1
  - TD0569: NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7
  - TD0564: NiT Technical Decision for Vulnerability Analysis Search Criteria
  - TD0563: NiT Technical Decision for Clarification of audit date information
  - TD0556: NIT Technical Decision for RFC 5077 question
  - TD0555: NIT Technical Decision for RFC Reference incorrect in TLSS Test
  - TD0547: NIT Technical Decision for Clarification on developer disclosure of AVA_VAN
  - TD0538: NIT Technical Decision for Outdated link to allowed-with list
  - TD0537: NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3
  - TD0536: NIT Technical Decision for Update Verification Inconsistency
  - TD0528: NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4

  The following Technical Decision against the collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020, [CPP_ND_V2.2E] is not applicable to the TOE.

  - TD0546: NIT Technical Decision for DTLS - clarification of Application Note 63
    - The Technical Decision is not applicable to the TOE. The TOE does not use DTLS.
  - TD0527: Updates to Certificate Revocation Testing (FIA_X509_EXT.1)
    - The Technical Decision is not applicable to the TOE. The TOE does not use EC certificates as indicated in FCS_COP.1/SigGen).
- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017.
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1 Revision 5, April 2017.
  - Part 3 Conformant

1.3 Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
  - Iteration: allows a component to be used more than once with varying operations. In the ST iterations defined by the PP author are identified by adding a string starting with "/" (e.g. "FCS_COP.1/Hash"). While iterations made by the ST author are further identified by a number within parenthesis and includes descriptive text (e.g. FIA_X509_EXT.1/ITT(1) X.509 Certificate Validation (descriptive text).
  - Assignment: allows the specification of an identified parameter. Assignments are indicated using bold and are surrounded by brackets (e.g., [**assignment**]). Note that an assignment within a selection would be identified in italics and with embedded bold brackets (e.g., [***[selected-assignment]***]).
  - Selection: allows the specification of one or more elements from a list. Selections are indicated using bold italics and are surrounded by brackets (e.g., [***selection***]).

- Refinement: allows the addition of details. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …"). Note that 'cases' that are not applicable in a given SFR have simply been removed without any explicit identification.

- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.

1.3.1 Terminology

This section identifies TOE-specific terminology.

Alert	An alert is the recorded and displayed incident of a network event and is generated if the alert action for the rule has been configured to include an alert. Alerts are violations of advanced threat detection policies.
Collector	Unique name for the Fidelis Collector appliance. This may refer to a single appliance configuration or to a cluster of appliances which include a Fidelis Collector Controller and three or more Fidelis Collector XA nodes.
CommandPost	Unique name for the Fidelis management console appliance of the TOE
Decoy Server	A decoy is a system that emulates a real system in the enterprise. The decoy stores data, but none of it is real. An attacker can interact with the decoy to log in, access files, and store files on the system. All activity is recorded and all access to a decoy results in a decoy alert.
Event	A rule violation. One or more events are reported as an alert if the rule action is configured to alert.
Fidelis Insight Server	A non-TOE component which provides software and policy updates for the TOE. Fidelis Insight provides threat intelligence, an execution environment for malware and threat detection, and machine-learning algorithms applied to collected threat data.
Fidelis Network	The Fidelis Network and Deception v9.3.3 components include several types of sensors, Fidelis Collectors, the Fidelis Sandbox, and CommandPost. The sensors can be deployed to specific areas of the network as needed.
Indicator	Indicators are patterns for analyzing and matching the data flowing across the network (e.g. content of the files, attributes of the protocols or files) in various ways. Indicators are used to define an aspect of behavior, but do not indicate goodness or badness. There are several groups of indicators: Content indicators, Metadata indicators, and Other indicators (including Attribute indicators). Content indicators are used to detect the data within a data transmission. Metadata indicators are used to detect the source or destination of a data transmission. Other indicators include the feed indicators: Attribute, Reputation, Email, and Content URL.
ICAP	ICAP is a lightweight and extensible point-to-point protocol used for requesting services for content inspection.
ISO	An ISO image (or .ISO file) is a computer file that is an exact copy of an existing file system
Malware Detection Engine	The Malware Detection Engine (MDE) is included with CommandPost and Fidelis Direct, Internal, and Mail sensors. When enabled, the Malware Detection Engine

	will analyze all executable objects. If malware is detected, an action will be taken, as defined on the Malware Reaction page.
Metadata	Data collected by a Fidelis sensor for all network traffic, whether a rule violation occurs or not. Metadata is stored within a Fidelis Collector appliance and is available for analysis by a Fidelis CommandPost.
Milter Protocol	A protocol for e-mail traffic handling that receives e-mail traffic from an external MTA, reassembles the e-mail session and forwards to the next layer for protocol decoding.
Policy	Fidelis Network and Deception v9.3.3 policies are composed of one or more rules, which in turn, contain one or more indicator definitions.
Postfix	An open source mail server alternative to the Sendmail program.
Rule	A rule is a logical combination of indicators that together are used by the event manager to generate alerts based on matches on combinations of indicators.
SAMBA	A Windows interoperability suite of programs for Linux and Unix for stable and fast file and print services for all clients using the SMB/CIFS protocol, such as all versions of DOS and Windows, OS/2, Linux and many others.
Sensor	Refers to the Fidelis Direct, Fidelis Internal, Fidelis Web, and Fidelis Mail appliances (hardware or virtual) running the Fidelis software.

1.3.2 Abbreviations

This section identifies abbreviations and acronyms used in this ST.

AES	Advanced Encryption Standard
API	Application Programming Interface
CBC	Cipher-Block Chaining
CA	Certificate Authority
CIFS	Common Internet File System
CLI	Command Line Interface
CM	Configuration Management
CRL	Certificate Revocation List
CSP	Critical Security Parameter
DH	Diffie-Hellman
FIPS	Federal Information Processing Standard
GUI	Graphical User Interface
HMAC	Hashed Message Authentication Code
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICAP	Internet Content Adaptation Protocol
LDAP	Lightweight Directory Access Protocol
MTA	Mail Transfer Agent
MDE	Malware Detection Engine
NDPP	Protection Profile for Network Devices
OS	Operating System
PEM	Privacy Enhanced Email
PPS	Packets per second
RSA	Rivest, Shamir and Adleman (algorithm for public-key cryptography)
SAR	Security Assurance Requirement
SFR	Security Functional Requirement

SHA	Secure Hash Algorithm
SMB	Server Message Block
SPAN	Switched Port ANalyzer
SNMP	Simple Network Management Protocol
ST	Security Target
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functions
UAU	User Authentication
UDP	User Datagram Protocol
VM	Virtual Machine

2. TOE Description

The Target of Evaluation (TOE) is a combination of Fidelis Network and Deception version 9.3.3 components. More specifically, the TOE consists of:

- one or more Fidelis Network v9.3.3 CommandPost management consoles
- one or more Fidelis Network Collectors v9.3.3
- zero or more Fidelis Sandbox appliances, v9.3.3
- zero or more Decoy Server appliances, v9.3.3
- at least one of the following sensor appliances:
    - Fidelis Network Direct v9.3.3
    - Fidelis Network Internal v9.3.3
    - Fidelis Network Web v9.3.3
    - Fidelis Network Mail v9.3.3

2.1 Product Overview

This sub-section provides an overview of the capabilities of the Fidelis Network and Deception solution. The evaluated configuration of the TOE and the TOE functionality included within the scope of evaluation are described in the "TOE Overview" subsection that follows.

The Fidelis Network and Deception monitors network traffic for malicious content coming into the network (intrusion) and for sensitive and secure data leaving the network (extrusion). Threat analysis is performed by network sensors and intelligent decoys. Analytics run within Network Collectors. Threat analysis is automatically correlated and triangulated across the separate analysis engines and presented to the analyst in a single user interface known as CommandPost.

It is designed to operate continuously, observing network traffic as it is perceived on the attached networks. Traffic observed by a Fidelis Network sensor is reassembled into sessions; protocols are identified; applications are identified; and, contents are analyzed in order to determine whether they contain anything inappropriate based on the applicable (intrusion/extrusion) policy rules. When inappropriate content is identified, the sensor takes action, as defined by the rule which was violated. Actions include alert, prevent, throttle, tag metadata, flag host, MDE filtered, quarantine, reroute, notify sender, remove attachments, append message, X-header modification, whitelist, and malware exception. Additionally, packets can be captured in a .pcap file. A rule may invoke several actions for a single violation.

The Fidelis CommandPost is the management system for the Fidelis solution. The CommandPost GUI can be accessed from anywhere on the network to:

- Visually monitor and analyze network alerts and other metadata in real time.
- Enable, disable, or customize policies and analytics as required.
- Add, configure, and manage Sensors, Collectors, Decoy Servers, and the CommandPost management console itself.
- Collect, aggregate, and store data from the Fidelis Sensors and Collectors
- Create users using the access control capabilities in several user authentication mechanisms including integration with a user directory server.
- Export information to a third-party network alert aggregation system.
- Use the built-in reports or customize reports.

The Fidelis Network Collector captures and stores into metadata details about network transactions. The metadata includes all attributes of the analyzed network traffic but excludes any recording of the data. Metadata includes the identified protocol and application in addition to any attributes detected by the protocol, application, or files transferred. The tag action of a policy can be used to simply tag the metadata without taking any further action on the network data.

The Fidelis Network sensor software is designed around a series of layers that receive packets from the attached networks, perform session reassembly, and decode the payload. Authorized administrators configure policies that delineate exactly what the Fidelis Network will capture, analyze and monitor. Once content is identified, a set of rules is applied. When a rule indicates a violation, the sensor performs the action identified by the rule. The Fidelis Network Direct, Internal, and Mail sensors also include a Malware Detection Engine (MDE) that can examine files to determine malicious intent. The MDE uses intelligence obtained from the Fidelis Insight Server and uses internal and external sources for file examination and the determination of maliciousness.

The Fidelis Network Direct and Internal sensor appliances operate directly on Ethernet packets received from the wire. Packets are reassembled into TCP or UDP sessions and analyzed. The Direct and Internal modules can take alert, prevent, throttle, packet capture, flag host, MDE filtered, whitelist, malware exception, and tag metadata actions. Prevention is performed by dropping packets (if installed inline) and sending TCP reset packets to the source of the session. Throttling can only be performed when installed inline and is performed by randomly dropping packets and manipulating the TCP window size until the bandwidth is below the configured value.

The Fidelis Network Web sensor utilizes the standard Internet Content Adaptation Protocol (ICAP) to receive information from a web proxy server. Received packets are stripped of the ICAP layer and reassembled into application sessions, ready for the protocol decoding layer of software. The Web sensor can take alert and prevent actions. Prevention is performed by instructing the web proxy server to drop the session and either diverts the user‘s browser to a standard Error 403 (Forbidden) HTTP page or to a customized security violation page provided by the operating environment.

The Fidelis Network Mail sensor processes e-mail and can act as a Mail Transfer Agent (MTA) or utilize the milter protocol to receive messages from an external MTA. In either case, received traffic is handled by the milter protocol layer, which will reassemble the e-mail session and forward to the next layer for protocol decoding. When the Fidelis Network Mail sensor is running as an MTA, the e-mail handler is embedded on the appliance utilizing Postfix. The Mail module can take alert, prevent, quarantine, MDE filtered, tag metadata, whitelist, malware exception, reroute, notify sender, append message, remove attachments, and X-header modification actions. Prevention is performed by dropping the incoming e-mail message. Quarantine, in MTA mode, is performed by storing the message locally on the sensor until an authorized administrator reviews the message and decides to discard or forward the message.

The Fidelis Sandbox appliance provides a virtual environment that executes files to analyze their behavior. The Fidelis Sandbox appliance can execute approximately 20,000 samples per day. File submissions are based on the Malware Detection Engine and custom rules that use the sandbox action.

The Fidelis Decoy Server implements Fidelis Deception using the same Internal or Direct sensor as Fidelis Network. The sensors provide traffic sniffing capability to analyze all traffic and to detect and classify all assets communicated through the sensor. CommandPost will store an asset database to list all such assets and to provide information including the asset operating system, and role, including discovered IoT devices. The asset database is used to automate the creation and distribution of decoys on Decoy servers. Decoys can also be created manually in CommandPost. A decoy can be based on a virtual machine running an image of any software desired to use as a decoy. A golden image can be installed on the virtual decoy and monitored in the same manner as the emulated decoy. A decoy is a system that emulates a real system in the enterprise. The decoy stores data, but none of it is real. An attacker can interact with the decoy to log in, access files, and store files on the system. All activity is recorded and all access to a decoy results in a decoy alert.

2.2 TOE Overview

The TOE consists of:

- one or more Fidelis Network v9.3.3 CommandPost management consoles
- one or more Fidelis Network Collectors v9.3.3

- zero or more Fidelis Sandbox appliances, v9.3.3
- zero or more Decoy Server appliances, v9.3.3
- at least one of the following sensor appliances:
    - Fidelis Network Direct v9.3.3
    - Fidelis Network Internal v9.3.3
    - Fidelis Network Web v9.3.3
    - Fidelis Network Mail v9.3.3

A Fidelis Network and Deception system can be deployed entirely as hardware appliances, VM appliances, or a mixture, so long as there is a CommandPost and at least one Collector and Sensor.

A sample deployment scenario for the sensors is depicted in **Figure 1** as follows. TOE components are depicted in the green.

- Fidelis CommandPost - Fidelis CommandPost appliance or Fidelis CommandPost VM
- Fidelis Sandbox – Fidelis Sandbox
- Fidelis Collector – (one or more)
    - Collector SA2
    - Collector XA2
    - Collector XA4
    - Collector Controller 2
    - Collector SA VM
    - Collector Controller 10G
- Fidelis Sensor – (one or more)
    - Fidelis Direct 50, Fidelis Direct 100, Fidelis Direct 250, Fidelis Direct 500, Fidelis Direct 1000, Fidelis Direct 2500, Fidelis Direct 5000, Fidelis Direct 10G, or Fidelis Direct VM
    - Fidelis Internal 1000, Fidelis Internal 2500, Fidelis Internal 5000, Fidelis Internal 10G, or Fidelis Internal VM
    - Fidelis Web, Fidelis Web VM
    - Fidelis Mail 250, Fidelis Mail 500, Fidelis Mail 1000, Fidelis Mail VM 250, Mail VM 500, or Mail VM 1000
- Decoy Server - FDH-3000, FDH-1000

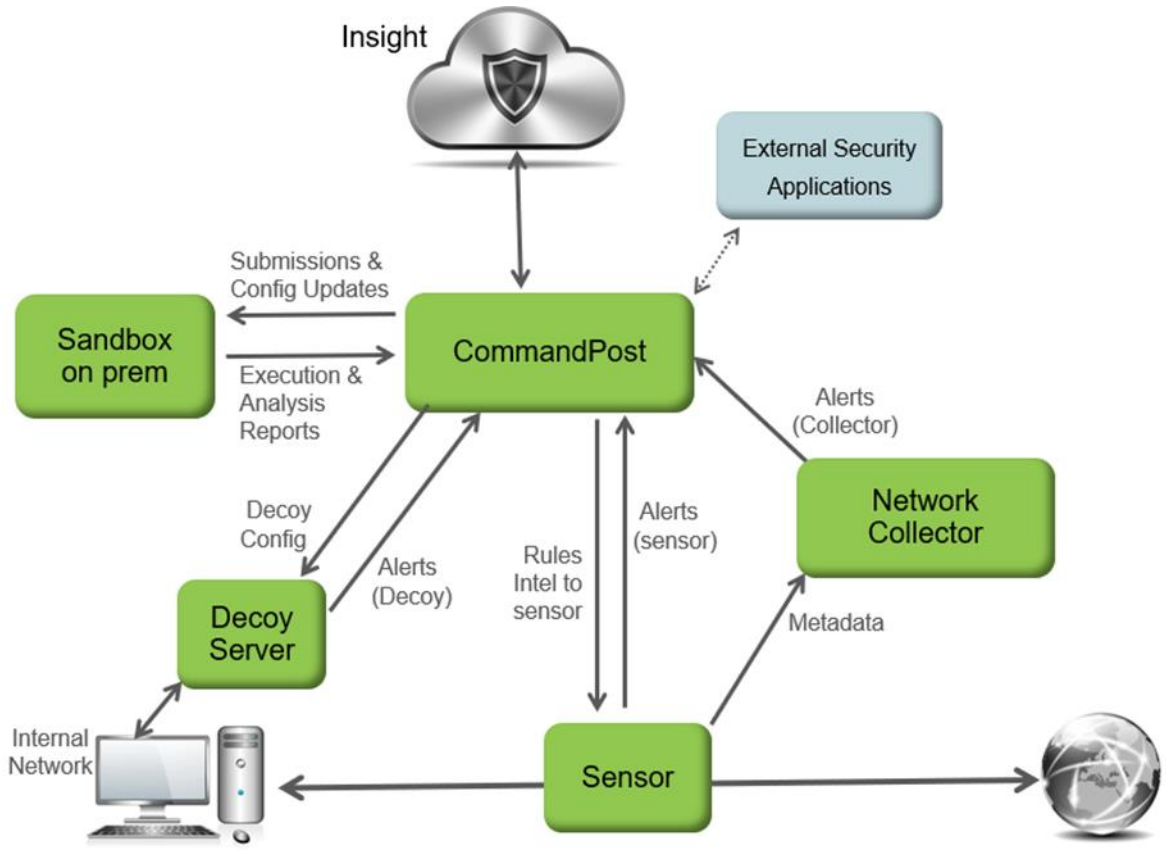

Figure 1 – Sample Fidelis Network Deployment Scenario

Initial configuration for each of the appliances is performed using the CLI by directly attaching a USB keyboard and VGA monitor to the appliance. The System Setup is used to set network parameters: the host name, IP address, IP mask, gateway, and primary (and secondary, if applicable) DNS, and the NTP server. Certificate files, CA-certificate files, CRL files are required to be installed on the Collector, Sensor, Sandbox, and Decoy Server components before proceeding with registration to the CommandPost.

After initial configuration and connecting each component to the network, the administrator adds all the components (Sensors, Collectors, Sandboxes, and Decoy Servers) to the CommandPost to register them. The component name, IP address and description are entered into the CommandPost. The component IP address must match the address established in the initial configuration and setup. After registration, the CommandPost attempts to communicate to the newly registered component at the specified IP address over a secure TLS tunnel.

The TOE requires users to be identified and authenticated before they can access any of the TOE functions. The only capabilities allowed prior to users authenticating are the display of the warning banner before authentication, and acceptance of the end-user license. The user only needs to accept the license once for each software release, after which the license acceptance message will not display. The banner is displayed on every login attempt.

Authorized administrators interact with the CommandPost component via a web browser where OpenSSL is used to implement Transport Layer Security (TLS) to secure the underlying communications. Similarly, CommandPost uses TLS to interact with the other components (Collectors, Sensors, Sandbox, and Decoy Server) in the deployment for the purposes of managing the components and receiving information from the components. Finally, the CommandPost uses TLS for communications with the following authorized IT entities: syslog server; LDAP server; Fidelis Insight Server. The TOE is operated in FIPS mode and includes an OpenSSL cryptographic module with CAVP approved algorithms. Authorized administrators can also interact with the CommandPost GUI component or by a using a directly connected console. However, once the TOE components have been installed and configured, it is intended that the TOE will be managed remotely via the CommandPost GUI.

The TOE provides several system functions that are controlled by an access privilege per user where a role is a collection of these functions. The levels of access are determined for TOE features such as Fidelis appliance configuration and user management. CommandPost includes several predefined roles, but only the System Administrators can manage all of the TOE security functions. Other roles only have a subset of TOE access capabilities.

The CommandPost audit log tracks all user activity. The Collectors, Sensors, Sandbox, and Decoy Server forward all audit information to the CommandPost which provides an internal log implementation that can be used to store audit records locally. Access is restricted to the System Administrator. The TOE can also be configured to send generated audit records to an external syslog server using TLS. When configured to send audit records to a syslog server, audit records are written to the external syslog as they are written locally to the CommandPost audit log.

The CommandPost can communicate with Fidelis Insight Server to download policy and TOE updates. The CommandPost GUI provides capabilities for administrators to update the TOE, and to query the currently executing software version of the TOE. Software updates are available as a tar package. The update package and its SHA256 hash are published on Fidelis support website. An administrator with proper credentials downloads the update via HTTPS.

Note: that hereinafter, the Fidelis Network sensor appliance identification will not include the specific type (Direct, Internal, Web, Mail), unless that has a direct impact on the specific Sensor functionality. Further, the Fidelis Network sensor(s) may also be referred to as just sensor(s), where all references pertain to the same TOE component providing this functionality.

The following two configurations of the TOE were covered by evaluation testing:

Test Configuration 1 (physical appliances)

- One Fidelis CommandPost v9.3.3 management console appliance
- One Fidelis SA2 Collector v9.3.3 appliance
- One Fidelis Direct 1000 Sensor v9.3.3 appliance
- One Fidelis Sandbox v9.3.3 appliance
- One Decoy Server v9.3.3 appliance

Test Configuration 2 (virtual machines)

- One Fidelis CommandPost v9.3.3 VM management console appliance
- One Fidelis Collector SA v9.3.3 VM appliance
- One Fidelis Direct Sensor v9.3.3 VM appliance
- One Decoy Server v9.3.3 VM appliance

Virtual appliances (CommandPost VM, Direct VM, Collector VM, Decoy Server VM) were tested in an environment consistent with the requirements described in Section 2.3.1.1 of this document, including CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake).

2.3 Physical Boundaries

2.3.1 TOE Components

Each TOE component is a self-contained hardware appliance or VM designed to interact with its environment via network connections.

The following table lists each of the hardware appliances of the TOE and identifies the following attributes of each: main processor; disk storage capacity; physical network ports; and operating system and software components.

Device	Main Processor	Storage	Network Ports	Operating System / Software
CommandPost	Dual Gold 6234 8/16-core 3.3Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	3 TB 6x HDD, RAID-5	4x 1GbE	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - Apache httpd 2.4.41 - Apache tomcat 9.0.33 - syslog-ng 3.7.3 - MariaDB 10.4.8 - OpenSSL 1.0.2k-fips
Direct 10G Internal 10G	Quad Gold 6248 20/80-core 2.5Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	500 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Direct 5000 Internal 5000	Dual Gold 6248 20/40-core 2.5Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	300 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical (inline capable)	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Direct 500 Direct 1000	Dual Silver 4214 12/24-core 2.2Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Internal 1000	Dual Silver 4214 12/24-core 2.2Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Mail 1000 Mail 500 Mail 250	Dual Silver 4214 12/24-core 2.2Ghz 2nd Generation Intel® Xeon® Scalable Processors	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips

Device	Main Processor	Storage	Network Ports	Operating System / Software
	Cascade Lake microarchitecture			
Web	Dual Silver 4214 12/24-core 2.2Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Sandbox Appliance	Dual Gold 6246 12/24-core 3.3Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	3 TB 6x HDD, RAID-10	4x 1GbE	CentOS 7.6 Linux kernel 3.10.0-957.10.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Collector SA2	Dual Gold 6246 12/24-core 3.3Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	3 TB 6x HDD, RAID-10	4x 1GbE	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Collector Controller 2	Dual Silver 4214 12/24-core 2.2Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	300 GB 2x HDD, RAID-1	6x 1GbE	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Collector XA2	Dual Gold 6234 8/16-core 3.3Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	300 GB 2x HDD, RAID1 ---------- 3.6 TB 6x HDD, RAID-10	4x 1GbE	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Collector Controller 10G	Dual Gold 6248 20/40-core 2.5Ghz 2nd Generation Intel® Xeon® Scalable Processors	300 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips

Device	Main Processor	Storage	Network Ports	Operating System / Software
	Cascade Lake microarchitecture			
Collector XA4	Dual Gold 6246 12/24-core 3.3Ghz 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	300 GB 2x HDD, RAID1 ---------- 19.8 TB 22x HDD, RAID-10	4x 1GbE 2x 10GbE optical	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - OpenSSL 1.0.2k-fips
Decoy Server FDH-3000	Dual Intel Xeon Gold 6234 (8 Cores, 16 threads each) 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	600GB Raid 1 1.8TB Raid 5	4x 1GbE	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - Stunnel 4.56 - Apache Tomcat/8.0.28 - Postgresql 9.2.24 - OpenSSL 1.0.2k-fips
Decoy Server FDH-1000	Dual Intel Xeon Silver 4214 Kit (12 Cores, 24 threads each) 2nd Generation Intel® Xeon® Scalable Processors Cascade Lake microarchitecture	300GB Raid 1	4x 1GbE	CentOS 7.7 with Linux kernel 3.10.0-1062.9.1.el7.x86_64 - Stunnel 4.56 - Apache Tomcat/8.0.28 - Postgresql 9.2.24 - OpenSSL 1.0.2k-fips

Table 1 TOE Hardware Components

The following table lists each of the virtual appliances of the TOE and identifies the following platform requirements: number of vCPUs; memory size; and disk capacity. The last column identifies the operating system and other software components included with the virtual appliance.

Device	Capacity	Number of vCPUs	Memory	Disk	Operating System / Software
CommandPost VM	Up to 4 alerts/sec Up to total 5 million alerts	Regular[1] 16	64 GB	1500 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) Linux kernel 3.10.0-1062.9.1.el7.x86_64

[1] Regular usage is maximum of 10 concurrent users with total alert volume up to 5 million alerts (depending on total average session size and the retention period).

Device	Capacity	Number of vCPUs	Memory	Disk	Operating System / Software
	Up to 10 alerts/sec Up to total 10 million alerts	Heavy[2] 32	128 GB	3000GB	httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
Direct/Internal VM	100 Mbps	8	16 GB	40 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
	500 Mbps (70k pps)	14	24 GB	80GB	
	1 Gbps (125k pps)	24	32Gb	100GB	
	2 Gbps (300K pps)	48	64	200 GB	
Web VM	100 Mbps	8	16 GB	40 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
	500 Mbps	14	24	80 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
	1 Gbps	24	32	100GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33

[2] Heavy usage is maximum of 20 concurrent users with total alert volume up to 10 million alerts (depending on total average session size and the retention period).

Device	Capacity	Number of vCPUs	Memory	Disk	Operating System / Software
					syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
Mail 250 VM	250k msg/day	6	12 GB	50 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
Mail 500 VM	500k msg/day	8	16 GB	100 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
Mail 1000 VM	1m msg/day	12	20 GB	200 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
Mail 5000 VM	5m msg/day	40	32	500 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
Collector SA VM	Minimal	4	28 GB	300 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon

Device	Capacity	Number of vCPUs	Memory	Disk	Operating System / Software
					Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
	Regular	16	64	1500 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
	Heavy	32	125	3000 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
Decoy Server	Low-End Virtual Machine	8 cores, 2.1 GHz and up	16 GB	250 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33 syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips
	High End Virtual Machine	24 cores, 2.4 GHz and up	32 GB	500 GB	CentOS 7.7 on VMware ESXi 6.7 on Intel Xeon Gold 6248 (Cascade Lake) with Linux kernel 3.10.0-1062.9.1.el7.x86_64 httpd 2.4.41 tomcat 9.0.33

Device	Capacity	Number of vCPUs	Memory	Disk	Operating System / Software
					syslog-ng 3.7.3 MariaDB 10.4.8 OpenSSL 1.0.2k-fips

Table 2 TOE Virtual Machine Appliances

2.3.1.1 Operational Environment Components

Administrators require a client computer with a web browser to remotely access the CommandPost GUI.

The following browsers are supported by the TOE:

- Mozilla Firefox v78
- Google Chrome v83

The virtual appliances are delivered as an installation disk (or ISO image). The virtual systems were tested by the evaluation team with CentOS 7.7 on VMware ESXi 6.7 with an Intel Xeon Gold 6248 processor based on the Cascade Lake microarchitecture. The virtual module must be the only guest running in the virtual environment.

The following components are supported in the operational environment of the TOE:

- External authentication methods require the use of LDAP servers.
- External audit storage requires the use of syslog servers.
- An NTP Server is required for proper clock synchronization for use in creating reliable timestamps.
- Fidelis Insight Server which provides software and policy updates for the TOE.

The TOE's (unevaluated) monitoring capability performs differently depending on whether sensors are connected by Network Taps or SPAN Ports.

- **Network Taps—**required for lossless network monitoring by Fidelis Direct and internal sensors in an out-of-band deployment. A network tap will replicate all network traffic with no data loss or performance degradation. Network taps guarantee complete traffic replication.
- **SPAN Ports—**connecting the Fidelis Direct or internal sensors to the SPAN ports on the router or switch can be done, but unlike Network Taps do not guarantee complete traffic replication and/or processing of all data due to traffic volumes. While they can be used, they are not recommended since the applicable network router or other device supporting SPAN ports generally treat SPAN ports with low priority and may not send all packets when under load.

Initial configuration of the TOE appliances requires local access. A keyboard and monitor are connected to the appliances for initial network setup.

2.3.2 Logical Boundaries

This section summarizes the security functions provided by the TOE:

- Security audit
- Cryptographic support
- Communication
- Identification and authentication
- Security management
- Protection of the TSF
- TOE access
- Trusted path/channels

2.3.2.1 Security audit

The TOE is able to generate logs of security relevant events including the events specified in [CPP_ND_V2.2E]. The TOE stores the logs locally on the CommandPost so they can be accessed by an administrator. The TOE can also be configured to send the logs to a designated external log server.

2.3.2.2 Cryptographic support

The TOE is operated in FIPS mode and includes an OpenSSL cryptographic module with CAVP approved algorithms. The module provides key management, random bit generation, encryption/decryption, digital signature and cryptographic hashing and keyed-hash message authentication features in support of higher level cryptographic protocols, including TLS and HTTPs.

2.3.2.3 Communication

The Fidelis Network and Deception v9.3.3 is deployed as a distributed TOE configuration. Initial configuration for each of the appliances is performed using the CLI by directly attaching a USB keyboard and VGA monitor to the appliance. The System Setup is used to set network parameters and certificate files. After initial configuration and connecting each appliance to the network, the administrator adds all the components (Sensors, Collectors, Decoy Server, and Sandboxes) to the CommandPost to register them. After registration, CommandPost attempts to communicate to the newly registered component (the Sensor, the Collector, or the Sandbox or the Decoy Server) at the specified IP address over a secure TLS tunnel as described in FPT_ITT.1/Join.

2.3.2.4 Identification and authentication

The TOE requires users (i.e., administrators) to be successfully identified and authenticated before they can access any security management functions available in the TOE. Administrators manage the TOE remotely using the CommandPost web-based GUI accessed via HTTPS or locally using the CLI by a directly connected USB keyboard and a monitor to the appliance VGA connector. The TOE supports the local (i.e., on device) definition of administrators with usernames and passwords on all of the TOE components. Additionally, the TOE can be configured to authenticate remote administrators to use the services of trusted LDAP servers in the operational environment.

2.3.2.5 Security management

The TOE provides a GUI to access its security management functions. Security management commands are limited to administrators and are available only after they have provided acceptable user identification and authentication data to the TOE.

The TOE also provides the ability to manage the TOE locally using the CLI by directly attaching a keyboard and monitor to the appliance. However, the TOE is designed to be managed using the CommandPost GUI from a remote HTTPS/TLS client. Following the initial configuration, all changes should be performed by an authorized user from CommandPost. The TOE provides the System Administrator role which corresponds to the [CPP_ND_V2.2E] Security Administrator.

2.3.2.6 Protection of the TSF

The TOE implements a number of features designed to protect itself to ensure the reliability and integrity of its security features.

It protects particularly sensitive data such as stored passwords and cryptographic keys so that they are not accessible even by an administrator. The TOE includes a hardware-based real-time clock that in conjunction with an NTP server in the operational environment ensure that reliable time information is available (e.g., for log accountability).

The TOE includes functions to perform self-tests so that it might detect when it is failing.

An administrator with proper credentials can download product updates from the Fidelis support website. The administrator verifies the published hash of the download to ensure that the update will not introduce malicious or other unexpected changes in the TOE.

Secure communication between the TOE components is provided by HTTPS/TLS.

2.3.2.7 TOE access

The TOE can be configured to display an informative banner that will appear prior to an administrator being permitted to establish an interactive session. Prior to a user logging in, the user must indicate whether he/she wants to continue with the authentication process. The TOE subsequently will enforce an administrator-defined inactivity timeout value after which the inactive session will be terminated.

2.3.2.8 Trusted path/channels

The TOE protects interactive communication with remote administrators using HTTPS. TLS ensures both integrity and disclosure protection.

The TOE protects communication with network peers, such as log server, Fidelis Insight Server and authentications servers, using TLS connections to prevent unintended disclosure or modification of the transferred data. The communication between the distributed TOE components is protected by TLS.

2.4 TOE Documentation

Fidelis Security Systems offers a series of documents that describe the installation process for the TOE, as well as guidance for subsequent use and administration of the system security features.

- Fidelis Network® Fidelis Deception® Enterprise Setup and Configuration Guide, Version 9.3.3
- Fidelis Network® Fidelis Deception® User Guide, Version 9.3.3
- Fidelis Network® Fidelis Deception® Guide to Creating Policies, Version 9.3.3 CC
- Fidelis Network® Fidelis Deception Common Criteria Configuration Guide v9.3.3, Revised 2021

3. Security Problem Definition

This security target includes by reference the Security Problem Definition (composed of organizational policies, threat statements, and assumptions) from the [CPP_ND_V2.2E].

In general, the [CPP_ND_V2.2E] has presented a Security Problem Definition appropriate for network infrastructure devices, and as such is applicable to the Fidelis TOE.

4. Security Objectives

Like the Security Problem Definition, this security target includes by reference the Security Objectives from the [CPP_ND_V2.2E]. The [CPP_ND_V2.2E] security objectives for the operational environment are reproduced below, since these objectives characterize technical and procedural measures each consumer must implement in their operational environment.

In general, the [CPP_ND_V2.2E] has presented a Security Objectives statement appropriate for network infrastructure devices, and as such is applicable to the Fidelis TOE.

4.1 Security Objectives for the Operational Environment

OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE. For vNDs the TOE includes only the contents of the its own VM, and does not include other VMs or the VS.
OE.NO_THRU_TRAFFIC_PROTECTION	The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment.
OE.TRUSTED_ADMIN	Security Administrators are trusted to follow and apply all guidance documentation in a trusted manner. For vNDs, this includes the VS Administrator responsible for configuring the VMs that implement ND functionality. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are assumed to monitor the revocation status of all certificates in the TOE's trust store and to remove any certificate from the TOE's trust store in case such certificate can no longer be trusted.
OE.UPDATES	The TOE firmware and software is updated by an administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
OE.ADMIN_CREDENTIALS_SECURE	The administrator's credentials (private key) used to access the TOE must be protected on any other platform on which they reside.
OE.COMPONENTS_RUNNING	For distributed TOEs the Security Administrator ensures that the availability of every TOE component is checked as appropriate to reduce the risk of an undetected attack on (or failure of) one or more TOE components. The Security Administrator also ensures that it is checked as appropriate for every TOE component that the audit functionality is running properly.
OE.RESIDUAL_INFORMATION	The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information

	(e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment. For vNDs, this applies when the physical platform on which the VM runs is removed from its operational environment.
OE.VM_CONFIGURATION	For vNDs, the Security Administrator ensures that the VS and VMs are configured to • reduce the attack surface of VMs as much as possible while supporting ND functionality (e.g., remove unnecessary virtual hardware, turn off unused inter-VM communications mechanisms), and • correctly implement ND functionality (e.g., ensure virtual networking is properly configured to support network traffic, management channels, and audit reporting). The VS should be operated in a manner that reduces the likelihood that vND operations are adversely affected by virtualisation features such as cloning, save/restore, suspend/resume, and live migration. If possible, the VS should be configured to make use of features that leverage the VS's privileged position to provide additional security functionality. Such features could include malware detection through VM introspection, measured VM boot, or VM snapshot for forensic analysis.

5. IT Security Requirements

This section defines the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) that serve to represent the security functional claims for the Target of Evaluation (TOE) and to scope the evaluation effort.

The SFRs have all been drawn from the Protection Profile (PP): collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020, [CPP_ND_V2.2E] and including the following optional SFRs: FAU_STG.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.2, FCO_CPC_EXT.1, FIA_X509_EXT.1/ITT, FPT_ITT.1, FPT_ITT.1/Join and the following selection-based SFRs: FAU_GEN_EXT.1, FAU_STG_EXT.4, FAU_STG_EXT.5, FCS_HTTPS_EXT.1, FCS_NTP_EXT.1, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1, FIA_X509_EXT.1/Rev, FIA_X509_EXT.2, FIA_X509_EXT.3, FMT_MOF.1/Functions, FMT_MTD.1/CryptoKeys.

As a result, refinements and operations already performed in that PP are not identified (e.g., highlighted) here, rather the requirements have been copied from that PP and any residual operations have been completed herein. Of particular note, the [CPP_ND_V2.2E] made a number of refinements and completed some of the SFR operations defined in the CC and that PP should be consulted to identify those changes if necessary. Text deleted from SFRs by a refinement in [CPP_ND_V2.2E] is not reproduced in ST.

The SARs are the set of SARs specified in [CPP_ND_V2.2E].

5.1 Extended Requirements

All extended requirements in this ST have been drawn from the [CPP_ND_V2.2E]. The [CPP_ND_V2.2E] defines the following extended SFRs and since they are not redefined in this ST, the [CPP_ND_V2.2E] should be consulted for more information regarding those CC extensions.

- FAU_GEN_EXT.1 Security Audit Generation
- FAU_STG_EXT.1: Protected Audit Event Storage
- FAU_STG_EXT.4: Protected Local Audit Event Storage for Distributed TOEs
- FAU_STG_EXT.5: Protected Remote Audit Event Storage for Distributed TOEs
- FCS_NTP_EXT.1: NTP Protocol
- FCS_HTTPS_EXT.1: HTTPS Protocol
- FCO_CPC_EXT.1: Component Registration Channel Definition
- FCS_RBG_EXT.1: Random Bit Generation
- FCS_TLSC_EXT.1: TLS Client Protocol Without Mutual Authentication
- FCS_TLSS_EXT.1: TLS Server Protocol without Mutual Authentication
- FCS_TLSC_EXT.2: TLS Client Protocol for Mutual Authentication
- FCS_TLSS_EXT.2: TLS Server Support for Mutual Authentication
- FIA_PMG_EXT.1: Password Management
- FIA_UIA_EXT.1: User Identification and Authentication
- FIA_UAU_EXT.2: Password-based Authentication Mechanism
- FIA_X509_EXT.1/Rev: X.509 Certificate Validation
- FIA_X509_EXT.1/ITT(1) X.509 Certificate Validation (CommandPost Collectors, Sensors, and Sandbox)
- FIA_X509_EXT.1/ITT(2) X.509 Certificate Validation (Decoy Server)
- FIA_X509_EXT.2: X509 Certificate Authentication
- FIA_X509_EXT.3: X.509 Certificate Requests

- FPT_APW_EXT.1: Protection of Administrator Passwords
- FPT_SKP_EXT.1: Protection of TSF Data (for Reading of all Pre-shared, Symmetric and Private Keys)
- FPT_STM_EXT.1: Reliable Time Stamps
- FPT_TST_EXT.1: TSF Testing
- FPT_TUD_EXT.1: Trusted Update
- FTA_SSL_EXT.1: TSF-initiated Session Locking

5.2 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by the Fidelis TOE.

Requirement Class	Requirement Component
FAU: Security audit	FAU_GEN.1: Audit Data Generation
	FAU_GEN_EXT.1: Audit Data Generation
	FAU_GEN.2: User Identity Association
	FAU_STG.1: Protected Audit Trail Storage
	FAU_STG_EXT.1: Protected Audit Event Storage
	FAU_STG_EXT.4: Protected Local Audit Event Storage for Distributed TOEs
	FAU_STG_EXT.5: Protected Remote Audit Event Storage for Distributed TOEs
FCS: Cryptographic support	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)
	FCS_HTTPS_EXT.1: HTTPS Protocol
	FCS_NTP_EXT.1: NTP Protocol
	FCS_RBG_EXT.1: Random Bit Generation
	FCS_TLSC_EXT.1: TLS Client Protocol Without Mutual Authentication
	FCS_TLSC_EXT.2: TLS Client Protocol for Mutual Authentication
	FCS_TLSS_EXT.1: TLS Server Protocol without Mutual Authentication
	FCS_TLSS_EXT.2: TLS Server Support for Mutual Authentication
FCO: Communication	FCO_CPC_EXT.1: Component Registration Channel Definition
FIA: Identification and authentication	FIA_AFL.1: Authentication Failure Management
	FIA_PMG_EXT.1: Password Management
	FIA_UIA_EXT.1: User Identification and Authentication
	FIA_UAU_EXT.2: Password-based Authentication Mechanism
	FIA_UAU.7: Protected Authentication Feedback

Requirement Class	Requirement Component
	FIA_X509_EXT.1/Rev: X.509 Certificate Validation
	FIA_X509_EXT.1/ITT(1): X.509 Certificate Validation
	FIA_X509_EXT.1/ITT(2): X.509 Certificate Validation
	FIA_X509_EXT.2: X.509 Certificate Authentication
	FIA_X509_EXT.3: X.509 Certificate Requests
FMT: Security management	FMT_MOF.1/Functions: Management of Security Functions Behaviour
	FMT_MOF.1/ManualUpdate : Management of Security Functions Behaviour
	FMT_MTD.1/CoreData : Management of TSF Data
	FMT_MTD.1/CryptoKeys : Management of TSF Data
	FMT_SMF.1: Specification of Management Functions
	FMT_SMR.2: Restrictions on Security Roles
FPT: Protection of the TSF	FPT_APW_EXT.1: Protection of Administrator Passwords
	FPT_ITT.1: Basic Internal TSF Data Transfer Protection
	FPT_ITT.1/Join: Basic Internal TSF Data Transfer Protection
	FPT_SKP_EXT.1: Protection of TSF Data (for reading of all pre-shared, symmetric and private keys".)
	FPT_TST_EXT.1: TSF Testing
	FPT_TUD_EXT.1: Trusted Update
	FPT_STM_EXT.1: Reliable Time Stamps
FTA: TOE access	FTA_SSL_EXT.1: TSF-initiated Session Locking
	FTA_SSL.3: TSF-initiated Termination
	FTA_SSL.4: User-initiated Termination
	FTA_TAB.1: Default TOE Access Banners
FTP: Trusted path/channels	FTP_ITC.1: Inter-TSF Trusted Channel
	FTP_TRP.1/Admin: Trusted Path

Table 3 TOE Security Functional Components

5.2.1 Security audit (FAU)

5.2.1.1 Audit Data Generation (FAU_GEN.1)

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

a) Start-up and shut-down of the audit functions;

b) All auditable events for the not specified level of audit; and

c) All administrative actions comprising:

- Administrative login and logout (name of user account shall be logged if individual user accounts are required for Administrators).
- Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).
- Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).
- Resetting passwords (name of related user account shall be logged).
- [***no other actions***];

d) Specifically defined auditable events listed in Table **4**.

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

a) Date and time of the event, type of event, subject identity, 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 cPP/ST, information specified in column three of Table **4**.

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	None.
FAU_GEN_EXT.1	None.	None.
FAU_GEN.2	None.	None.
FAU_STG.1	None.	None.
FAU_STG_EXT.1	None.	None.
FAU_STG_EXT.4	None.	None.
FAU_STG_EXT.5	None.	None.
FCO_CPC_EXT.1	Enabling communications between a pair of components. Disabling communications between a pair of components.	Identities of the endpoints pairs enabled or disabled.
FCS_CKM.1	None.	None.
FCS_CKM.2	None.	None.
FCS_CKM.4	None.	None.
FCS_COP.1/DataEncryption	None.	None.
FCS_COP.1/SigGen	None.	None.
FCS_COP.1/Hash	None.	None.
FCS_COP.1/KeyedHash	None.	None.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session	Reason for failure
FCS_NTP_EXT.1	Configuration of a new time server Removal of configured time server	Identity if new/removed time server
FCS_RBG_EXT.1	None.	None.
FCS_TLSC_EXT.1	Failure to establish a TLS Session	Reason for failure
FCS_TLSC_EXT.2	None.	None.
FCS_TLSS_EXT.1	Failure to establish a TLS Session	Reason for failure
FCS_TLSS_EXT.2	Failure to authenticate the client	Reason for failure
FIA_PMG_EXT.1	None.	None.
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None.	None.

Requirement	Auditable Events	Additional Audit Record Contents
FIA_X509_EXT.1/Rev	Unsuccessful attempt to validate a certificate Any addition, replacement or removal of trust anchors in the TOE's trust store	Reason for failure of certificate validation Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store
FIA_X509_EXT.1/ITT(1)	Unsuccessful attempt to validate a certificate Any addition, replacement or removal of trust anchors in the TOE's trust store	Reason for failure of certificate validation Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store
FIA_X509_EXT.1/ITT(2)	Unsuccessful attempt to validate a certificate Any addition, replacement or removal of trust anchors in the TOE's trust store	Reason for failure of certificate validation Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store
FIA_X509_EXT.2	None.	None.
FIA_X509_EXT.3	None.	None.
FMT_MOF.1/Functions	None.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.
FMT_MTD.1/CoreData	None.	None.
FMT_MTD.1/CryptoKeys	None.	None.
FMT_SMF.1	All management activities of TSF data.	None.
FMT_SMR.2	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_ITT.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FPT_ITT.1/Join	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FPT_SKP_EXT.1	None.	None.
FPT_TST_EXT.1	None.	None.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).

Requirement	Auditable Events	Additional Audit Record Contents
FTA_SSL_EXT.1 (if "terminate the session" is selected)	The termination of a local session by the session locking mechanism.	None.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.
FTA_SSL.4	The termination of an interactive session.	None.
FTA_TAB.1	None.	None.
FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_TRP.1/Admin	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions	None.

Table 4 Auditable Events

5.2.1.2 Security Audit Generation (FAU_GEN_EXT.1)

FAU_GEN_EXT.1.1 The TSF shall be able to generate audit records for each TOE component. The audit records generated by the TSF of each TOE component shall include the subset of security relevant audit events which can occur on the TOE component.

5.2.1.3 User Identity Association (FAU_GEN.2)

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.

5.2.1.4 Protected Audit Trail Storage (FAU_STG.1)

FAU_STG.1.1 The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.

FAU_STG.1.2 The TSF shall be able to prevent unauthorised modifications to the stored audit records in the audit trail.

5.2.1.5 Protected Audit Event Storage (FAU_STG_EXT.1)

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 [

- ***The TOE shall be a distributed TOE that stores audit data on the following TOE components: [CommandPost server]***,
- ***The TOE shall be a distributed TOE with storage of audit data provided externally for the following TOE components: [Direct Sensor, Internal Sensor, Mail Sensor, Web Sensor, Sandbox Appliance, Collector, Collector Controller, and Decoy Server]***

].

FAU_STG_EXT.1.3 The TSF shall [***[delete any audit record older than the administrator configured number of retention days, delete 20 of the oldest audit events once the disk space reaches 80% capacity]***] when the local storage space for audit data is full.

5.2.1.6 Protected Local Audit Event Storage for Distributed TOEs (FAU_STG_EXT.4)

FAU_STG_EXT.4.1 The TSF of each TOE component which stores security audit data locally shall perform the following actions when the local storage space for audit data is full:

[**CommandPost:** ***[[delete any audit record older than the administrator configured number of retention days, delete 20 of the oldest audit events once the disk space reaches 80% capacity]]***].

5.2.1.7 Protected Remote Audit Event Storage for Distributed TOEs (FAU_STG_EXT.5)

FAU_STG_EXT.5.1 Each TOE component which does not store security audit data locally shall be able to buffer security audit data locally until it has been transferred to another TOE component that stores or forwards it. All transfer of audit records between TOE components shall use a protected channel according to [***FPT_ITT.1***].

5.2.2 Communication (FCO)

5.2.2.1 Component Registration Channel Definition (FCO_CPC_EXT.1)

FCO_CPC_EXT.1.1 The TSF shall require a Security Administrator to enable communications between any pair of TOE components before such communication can take place.

FCO_CPC_EXT.1.2 The TSF shall implement a registration process in which components establish and use a communications channel that uses [***a channel that meets the secure channel requirements in*** [***FPT_ITT.1***]] for at least [**TSF**] data.

FCO_CPC_EXT.1.3 The TSF shall enable a Security Administrator to disable communications between any pair of TOE components.

Application Note: *The registration channel is identified in FPT_ITT.1/Join. The channel set up and used for registration is adopted as a continuing internal communication channel between different TOE components.*

5.2.3 Cryptographic support (FCS)

5.2.3.1 Cryptographic Key Generation (FCS_CKM.1)

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 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;***
- ***FFC schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.1***

].

5.2.3.2 Cryptographic Key Establishment (FCS_CKM.2)

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

- ***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";***

- ***Finite field-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";***

].

5.2.3.3 Cryptographic Key Destruction (FCS_CKM.4)

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*], *destruction of reference to the key directly followed by a request for garbage collection*];**
- 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-pass] overwrite consisting of [zeroes]*]**

that meets the following: No Standard.

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

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

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

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) [3072 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***

].

5.2.3.6 Cryptographic Operation (Hash Algorithm) (FCS_COP.1/Hash)

FCS_COP.1.1/Hash The TSF shall perform cryptographic hashing services in accordance with a specified cryptographic algorithm **[*SHA-1, SHA-256, SHA-384, SHA-512*]** and message digest sizes [***160, 256, 384, 512***] bits that meet the following: ISO/IEC 10118-3:2004.

5.2.3.7 Cryptographic Operation (Keyed Hash Algorithm) (FCS_COP.1/KeyedHash)

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-384***] and cryptographic key sizes [**256, 384 bits**] and message digest sizes [***256, 384***] bits that meet the following: ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".

5.2.3.8 HTTPS Protocol (FCS_HTTPS_EXT.1)

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 HTTPS using TLS.

FCS_HTTPS_EXT.1.3 If a peer certificate is presented, the TSF shall [***not establish the connection,***] if the peer certificate is deemed invalid.

5.2.3.9 NTP Protocol (FCS_NTP_EXT.1)

FCS_NTP_EXT.1.1 The TSF shall use only the following NTP version(s) [***NTP v4 (RFC 5905)***].

FCS_NTP_EXT.1.2 The TSF shall update its system time using [
Authentication using [***SHA1***] as the message digest algorithm(s).
].

FCS_NTP_EXT.1.3 The TSF shall not update NTP timestamp from broadcast and/or multicast addresses.

FCS_NTP_EXT.1.4 The TSF shall support configuration of at least three (3) NTP time sources in the Operational Environment.

5.2.3.10 Random Bit Generation (FCS_RBG_EXT.1)

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using **[*CTR_DRBG (AES)*]**.

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [***[one] platform-based noise source***] with a minimum of [***256 bits***] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions", of the keys and hashes that it will generate.

5.2.3.11 TLS Client Protocol Without Mutual Authentication (FCS_TLSC_EXT.1)

FCS_TLSC_EXT.1.1 The TSF shall implement [***TLS 1.2 (RFC 5246)***] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

- ***TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*]** and no other ciphersuites**.**

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifier matches [***the reference identifier per RFC 6125 section 6***, ***IPv4 address in SAN***, ***IPv6 address in the SAN***] and no other attribute types].

FCS_TLSC_EXT.1.3 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the server certificate is invalid. The TSF shall also [

- ***Not implement any administrator override mechanism***

].

FCS_TLSC_EXT.1.4 The TSF shall [***present the Supported Elliptic Curves/Supported Groups Extension with the following curves/groups:* [*secp256r1, secp384r1, secp521r1*] *and no other curves/groups***] in the Client Hello.

5.2.3.12 TLS Client Support for Mutual Authentication (FCS_TLSC_EXT.2)

FCS_TLSC_EXT.2.1 The TSF shall support TLS communication with mutual authentication using X.509v3 certificates.

5.2.3.13 TLS Server Protocol Without Mutual Authentication (FCS_TLSS_EXT.1)

FCS_TLSS_EXT.1.1 The TSF shall implement [***TLS 1.2 (RFC 5246)***] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

- ***TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246***

- ***TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246***
- ***TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*****]** and no other ciphersuites**.**

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 [***Diffie-Hellman parameters with size [3072 bits], ECDHE curves [secp256r1] and no other curves***].

FCS_ TLSS_EXT.1.4 The TSF shall support [***no session resumption or session tickets***].

5.2.3.14 TLS Server Support for Mutual Authentication (FCS_TLSS_EXT.2)

FCS_TLSS_EXT.2.1 The TSF shall support TLS communication with mutual authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.2 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the client certificate is invalid. The TSF shall also [

- ***Not implement any administrator override mechanism***

].

FCS_TLSS_EXT.2.3 The TSF shall not establish a trusted channel if the identifier contained in a certificate does not match an expected identifier for the client. If the identifier is a Fully Qualified Domain Name (FQDN), then the TSF shall match the identifiers according to RFC 6125, otherwise the TSF shall parse the identifier from the certificate and match the identifier against the expected identifier of the client as described in the TSS.

5.2.4 Identification and authentication (FIA)

5.2.4.1 Authentication Failure Management (FIA_AFL.1)

FIA_AFL.1.1 The TSF shall detect when an Administrator configurable positive integer within [**1 - 999**] unsuccessful authentication attempts occur related to Administrators attempting to authenticate remotely using a password.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met, the TSF shall [***prevent the offending Administrator from successfully establishing a remote session using any authentication method that involves a password until [password reset] is taken by an Administrator***].

5.2.4.2 Password Management (FIA_PMG_EXT.1)

FIA_PMG_EXT.1.1 The TSF shall provide the following password management capabilities for administrative passwords:

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [***"!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [blank space, "~", "`", "_", "+", "-", "=", "{", "}", "|", "[", "]", ":", ";", "<", ">", ",", ".", "/"]***];

b) Minimum password length shall be configurable to between [**1**] and [**999**] characters.

5.2.4.3 Protected Authentication Feedback (FIA_UAU.7)

FIA_UAU.7.1 The TSF shall provide only obscured feedback to the administrative user while the authentication is in progress at the local console.

5.2.4.4 Password-based Authentication Mechanism (FIA_UAU_EXT.2)

FIA_UAU_EXT.2.1 The TSF shall provide a local [***password-based***] authentication mechanism to perform local administrative user authentication.

5.2.4.5 User Identification and Authentication (FIA_UIA_EXT.1)

FIA_UIA_EXT.1.1 The TSF shall allow the following actions prior to requiring the non-TOE entity to initiate the identification and authentication process:

- Display the warning banner in accordance with FTA_TAB.1;
- **[*[Acceptance of the end user license]*]**.

FIA_UIA_EXT.1.2 The TSF shall require each administrative user to be successfully identified and authenticated before allowing any other TSF-mediated actions on behalf of that administrative user.

5.2.4.6 X.509 Certificate Validation (FIA_X509_EXT.1/ITT(1))

FIA_X509_EXT.1.1/ITT(1) The ~~TSF~~ **CommandPost, Collectors, Sensors, and Sandbox** shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation supporting a minimum path length of two certificates.
- The certification path must terminate with a trusted CA certificate designated as a trust anchor.
- The ~~TSF~~ **CommandPost, Collectors, Sensors, and Sandbox** shall validate a certification path by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag set to TRUE.
- The ~~TSF~~ **CommandPost, Collectors, Sensors, and Sandbox** shall validate the revocation status of the certificate using [***a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3***].
- The ~~TSF~~ **CommandPost, Collectors, Sensors, and Sandbox** shall validate the extendedKeyUsage field according to the following rules:
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
  - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

FIA_X509_EXT.1.2/ITT(1) The ~~TSF~~ **CommandPost, Collectors, Sensors, and Sandbox** shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

5.2.4.7 X.509 Certificate Validation (FIA_X509_EXT.1/ITT(2))

FIA_X509_EXT.1.1/ITT(2) The ~~TSF~~ **Decoy Server** shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation supporting a minimum path length of two certificates.

- The certification path must terminate with a trusted CA certificate designated as a trust anchor.
- The ~~TSF~~ **Decoy Server** shall validate a certification path by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag set to TRUE.
- The ~~TSF~~ **Decoy Server** shall validate the revocation status of the certificate using [***no revocation method]***].
- The ~~TSF~~ **Decoy Server** shall validate the extendedKeyUsage field according to the following rules:
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
  - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

FIA_X509_EXT.1.2/ITT(2) The ~~TSF~~ **Decoy Server** shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

5.2.4.8 X.509 Certificate Validation (FIA_X509_EXT.1/Rev)

FIA_X509_EXT.1.1/Rev The TSF shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation supporting a minimum path length of three certificates.
- The certification path must terminate with a trusted CA certificate designated as a trust anchor.
- The TSF shall validate a certification path by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag set to TRUE.
- The TSF shall validate the revocation status of the certificate using [***a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3***].
- The TSF shall validate the extendedKeyUsage field according to the following rules:
  - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
  - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

FIA_X509_EXT.1.2/Rev The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

5.2.4.9 X.509 Certificate Authentication (FIA_X509_EXT.2)

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [***HTTPS, TLS***], and [***no additional uses].***

FIA_X509_EXT.2.2 When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [***not accept the certificate***].

5.2.4.10 X.509 Certificate Requests (FIA_X509_EXT.3)

FIA_X509_EXT.3.1 The TSF shall generate a Certificate Request Message as specified by RFC 2986 and be able to provide the following information in the request: public key and [***Common Name, Organization, Organizational Unit, Country***].

FIA_X509_EXT.3.2 The TSF shall validate the chain of certificates from the Root CA upon receiving the CA Certificate Response.

5.2.5 Security management (FMT)

5.2.5.1 Management of Security Functions Behaviour (FMT_MOF.1/Functions)

FMT_MOF.1.1/Functions The TSF shall restrict the ability to [***modify the behaviour of***] the functions [***handling of audit data***] to Security Administrators.

5.2.5.2 Management of Security Functions Behaviour (FMT_MOF.1/ManualUpdate)

FMT_MOF.1.1/ManualUpdate The TSF shall restrict the ability to enable the functions to perform manual updates to Security Administrators.

5.2.5.3 Management of TSF Data (FMT_MTD.1/CoreData)

FMT_MTD.1.1/CoreData The TSF shall restrict the ability to manage the TSF data to Security Administrators.

5.2.5.4 Management of TSF Data (FMT_MTD.1/CryptoKeys)

FMT_MTD.1.1/CryptoKeys The TSF shall restrict the ability to manage the cryptographic keys to Security Administrators.

5.2.5.5 Specification of Management Functions (FMT_SMF.1)

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

- Ability to administer the TOE locally and remotely;
- Ability to configure the access banner;
- Ability to configure the session inactivity time before session termination or locking;
- Ability to update the TOE, and to verify the updates using [***hash comparison***] capability prior to installing those updates;
- Ability to configure the authentication failure parameters for FIA_AFL.1;

[

- ***Ability to configure audit behaviour (e.g. changes to storage locations for audit; changes to behaviour when local audit storage space is full);***
- ***Ability to manage the cryptographic keys;***
- ***Ability to configure the cryptographic functionality;***
- ***Ability to configure the interaction between TOE components;***
- ***Ability to import X.509v3 certificates to the TOE's trust store;***
- ***Ability to re-enable an Administrator account;***
- ***Ability to configure NTP;***

].

5.2.5.6 Restrictions on Security Roles (FMT_SMR.2)

FMT_SMR.2.1 The TSF shall maintain the roles:

- Security Administrator.

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions

- The Security Administrator role shall be able to administer the TOE locally;

- The Security Administrator role shall be able to administer the TOE remotely

are satisfied.

5.2.6 Protection of the TSF (FPT)

5.2.6.1 Protection of Administrator Passwords (FPT_APW_EXT.1)

FPT_APW_EXT.1.1 The TSF shall store administrative passwords in non-plaintext form.

FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext administrative passwords.

5.2.6.2 Basic Internal TSF Data Transfer Protection (FPT_ITT.1)

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 [***TLS, HTTPS***].

5.2.6.3 Basic Internal TSF Data Transfer Protection (FPT_ITT.1/Join)

FPT_ITT.1.1/Join 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 [***TLS, HTTPS***].

5.2.6.4 Protection of TSF Data (for Reading of all Pre-shared, Symmetric and Private Keys) (FPT_SKP_EXT.1)

FPT_SKP_EXT.1.1 The TSF shall prevent reading of all pre-shared keys, symmetric key, and private keys.

5.2.6.5 Reliable Time Stamps (FPT_STM_EXT.1)

FPT_STM_EXT.1.1 The TSF shall be able to provide reliable time stamps for its own use.

FPT_STM_EXT.1.2 The TSF shall [***synchronise time with an NTP server***].

5.2.6.6 TSF Testing (FPT_TST_EXT.1)

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: [

- **software module integrity tests**
- **cryptographic known answer tests**
- **entropy source online health tests**

].

5.2.6.7 Trusted Update (FPT_TUD_EXT.1)

FPT_TUD_EXT.1.1 The TSF shall provide Security Administrators the ability to query the currently executing version of the TOE firmware/software and [***no other TOE firmware/software version***].

FPT_TUD_EXT.1.2 The TSF shall provide Security Administrators the ability to manually initiate updates to TOE firmware/software and [***no other update mechanism***].

FPT_TUD_EXT.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a [***published hash***] prior to installing those updates.

5.2.7 TOE access (FTA)

5.2.7.1 TSF-initiated Termination (FTA_SSL.3)

FTA_SSL.3.1 The TSF shall terminate a remote interactive session after a Security Administrator-configurable time interval of session inactivity.

5.2.7.2 User-initiated Termination (FTA_SSL.4)

FTA_SSL.4.1 The TSF shall allow Administrator-initiated termination of the Administrator's own interactive session.

5.2.7.3 TSF-initiated Session Locking (FTA_SSL_EXT.1)

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions, [
- ***terminate the session***

]
after a Security Administrator-specified time period of inactivity.

5.2.7.4 Default TOE Access Banners (FTA_TAB.1)

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.

5.2.8 Trusted path/channels (FTP)

5.2.8.1 Inter-TSF Trusted Channel (FTP_ITC.1)

FTP_ITC.1.1 The TSF shall be capable of using [***TLS***] to provide a trusted communication channel between itself and authorized IT entities supporting the following capabilities: audit server, [***authentication server, [Fidelis Insight Server]***] that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure and detection of modification of the channel data.

FTP_ITC.1.2 The TSF shall permit the TSF or the authorized IT entities to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for [**transmitting audit records to an audit server, obtaining TOE updates, and external authentication functions**].

5.2.8.2 Trusted Path (FTP_TRP.1/Admin)

FTP_TRP.1.1/Admin The TSF shall be capable of using [***HTTPS***] to provide a communication path between itself and authorized remote Administrators 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 provides detection of modification of the channel data.

FTP_TRP.1.2/Admin The TSF shall permit remote Administrators to initiate communication via the trusted path.

FTP_TRP.1.3/Admin The TSF shall require the use of the trusted path for initial Administrator authentication and all remote administrative actions.

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are included by reference from the [CPP_ND_V2.2E].

Requirement Class	Requirement Component
ADV: Development	ADV_FSP.1 Basic functional specification
AGD: Guidance documents	AGD_OPE.1: Operational user guidance
	AGD_PRE.1: Preparative procedures
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE
	ALC_CMS.1 TOE CM coverage
ATE: Tests	ATE_IND.1 Independent testing - conformance
AVA: Vulnerability assessment	AVA_VAN.1 Vulnerability survey

Table 5 Assurance Components

Consequently, the assurance activities specified in [CPP_ND_V2.2E] apply to the TOE evaluation.

6. TOE Summary Specification

This chapter describes the security functions:

- Security audit
- Cryptographic support
- Communication
- Identification and authentication
- Security management
- Protection of the TSF
- TOE access
- Trusted path/channels

The Fidelis Network and Deception v9.3.3 is a distributed TOE configuration. The following table identifies which TOE components satisfy the [CPP_ND_V2.2E] requirements. The table also identifies which auditable events are generated and recorded by which TOE component.

Requirement	Auditable Events	Additional Audit Record Contents	Component Implementing the SFR	Component Generating the Audit Record
FAU_GEN.1	None.	None.	All	All
FAU_GEN_EXT.1	None.	None.	All	All
FAU_GEN.2	None.	None.	All	All
FAU_STG.1	None.	None.	CommandPost	None.
FAU_STG_EXT.1	None.	None.	CommandPost	None.
FAU_STG_EXT.4	None.	None.	CommandPost	None.
FAU_STG_EXT.5	None.	None.	All except CommandPost	None.
FCS_CKM.1	None.	None.	All	None.
FCS_CKM.2	None.	None.	All	None.
FCS_CKM.4	None.	None.	All	None.
FCS_COP.1/DataEncryption	None.	None.	All	None.
FCS_COP.1/SigGen	None.	None.	All	None.
FCS_COP.1/Hash	None.	None.	All	None.
FCS_COP.1/KeyedHash	None.	None.	All	None.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session.	Reason for failure	All	All
FCS_NTP_EXT.1	Configuration of a new time server Removal of configured time server	Identity if new/removed time server	CommandPost	CommandPost
FCS_RBG_EXT.1	None.	None.	All	None.
FCS_TLSC_EXT.1	Failure to establish a TLS Session	Reason for failure	All	All
FCS_TLSC_EXT.2	None.	None	All	None
FCS_TLSS_EXT.1	Failure to establish a TLS Session	Reason for failure	All	All
FCS_TLSS_EXT.2	Failure to authenticate the client	Reason for failure	All	All

Requirement	Auditable Events	Additional Audit Record Contents	Component Implementing the SFR	Component Generating the Audit Record
FCO_CPC_EXT.1	Enabling communications between a pair of components. Disabling communications between a pair of components.	Identities of the endpoints pairs enabled or disabled.	All components participate in the registration process. Enable/disable communications is performed from the CommandPost"	CommandPost
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).	CommandPost GUI	CommandPost
FIA_PMG_EXT.1	None.	None.	All	None.
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).	CommandPost	CommandPost
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).	CommandPost	CommandPost
FIA_UAU.7	None.	None.	All	None.
FIA_X509_EXT.1/Rev	Unsuccessful attempt to validate a certificate Any addition, replacement or removal of trust anchors in the TOE's trust store	Reason for failure of certificate validation Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store	CommandPost	CommandPost

Requirement	Auditable Events	Additional Audit Record Contents	Component Implementing the SFR	Component Generating the Audit Record
FIA_X509_EXT.1/ITT(1)	Unsuccessful attempt to validate a certificate Any addition, replacement or removal of trust anchors in the TOE's trust store	Reason for failure of certificate validation Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store	CommandPost, Collectors, Sensors, and Sandbox	CommandPost, Collectors, Sensors, and Sandbox
FIA_X509_EXT.1/ITT(2)	Unsuccessful attempt to validate a certificate Any addition, replacement or removal of trust anchors in the TOE's trust store	Reason for failure of certificate validation Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store	Decoy Server	Decoy Server
FIA_X509_EXT.2	None.	None.	All	None.
FIA_X509_EXT.3	None.	None.	All	None.
FMT_MOF.1/Functions	None.	None.	CommandPost	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.	CommandPost	CommandPost
FMT_MTD.1/CoreData	None.	None.	All	None.
FMT_MTD.1/CryptoKeys	None.	None.	CommandPost	None.
FMT_SMF.1	All management activities of TSF data.	None.	CommandPost	CommandPost
FMT_SMR.2	None.	None.	CommandPost	None.
FPT_SKP_EXT.1	None.	None.	All	None.
FPT_APW_EXT.1	None.	None.	All	None.

Requirement	Auditable Events	Additional Audit Record Contents	Component Implementing the SFR	Component Generating the Audit Record
FPT_ITT.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.	All	All
FPT_ITT.1/Join	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.	CommandPost	CommandPost
FPT_TST_EXT.1	None.	None.	All	None.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.	All	All
FPT_STM_EXT.1	Discontinuous changes to time – either Administrator actuated or changed via an automated process.	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).	All	All
FTA_SSL_EXT.1 (if "terminate the session" is selected)	The termination of a local session by the session locking mechanism.	None.	CommandPost	CommandPost

Requirement	Auditable Events	Additional Audit Record Contents	Component Implementing the SFR	Component Generating the Audit Record
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.	CommandPost	CommandPost
FTA_SSL.4	The termination of an interactive session.	None.	CommandPost	CommandPost
FTA_TAB.1	None.	None.	All	None.
FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.	CommandPost	CommandPost
FTP_TRP.1/Admin	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	None.	CommandPost	CommandPost

Table 6 SFR Allocation Requirements in the distributed TOE

6.1 Security audit

The TOE generates security relevant audit records including administrative activity. The audit records are stored on the CommandPost, protected from unauthorized deletion and can be sent to a remote audit server for storage. The connection for transmission of audit records uses TLS.

6.1.1 FAU_GEN.1, FAU_GEN_EXT.1: Audit Data Generation

The TOE is able to generate log records for security relevant and other events as they occur. The events that can cause an audit record to be logged include starting and stopping the audit function, any use of an administrator action via the CommandPost GUI comprising:

- Administrative login and logout (including the name of the user account).
- Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).
- Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).

- Resetting passwords (name of related user account is logged).

The CommandPost administrator actions are stored locally on the CommandPost appliance. The TOE components (Sensor, Collector, Sandbox, or Decoy Server) are registered with the CommandPost and forward log records generated on those appliances to the CommandPost over a secure TLS connection.

All of the TOE audit events are accessed via the CommandPost GUI via an authorized administrator. The audit records include the following fields:

- ID - The audit log ID number.
- Timestamp - The date and time when the action occurred.
- User - The user who performed the action.
- Category - The general type of action that occurred. For example, roles, users, and audit.
- Action - The specific action that occurred. Most actions relate to the section of the CommandPost used to trigger the action. For example, Alerts, Policies, and Reports. The Action column may also include information about what occurred, such as a login.
- Effect – The field provides additional details of administrator actions such as "Access, Modification, Addition, Deletion, Data Extraction, CommandPost GUI Page Access". For example:
  - logging in is "Access";
  - changing a configuration parameter is "Modification";
  - accessing any CommandPost GUI page in a browser is "CommandPost GUI Page Access";
  - downloading debug logs is "Data Extraction";
  - creating a new report is "Addition";
  - deleting a report is "Deletion".
- Description – Provides a high level description of the audit record. The field may contain IP addresses of the endpoint(s) involved in the generated audit event.
  For example, in case of TLS errors, the description field may contain the following:
  TLS ERROR: Local: ::ffff:10.89.113.69, Remote: ::ffff:10.89.184.31, error:1408A0C1:SSL routines:SSL3_GET_CLIENT_HELLO:no shared cipher
- Sensor - If audit event was generated on the remote component of the TOE, "Sensor". "Sensor" field contains remote component hostname/FQDN. Additionally, "Description" field may contain IP addresses of the endpoint(s) involved in the generated audit event.

The audit records capture the administrative task of generating/import of, changing, or deleting of cryptographic keys. The Description field of the audit record identifies the SHA-256 hash of the corresponding public key as well as the filename of the key file for all private RSA keys.

The above **Table 6** SFR Allocation Requirements in the distributed TOE identifies which auditable events are generated by each of the distributed TOE components. The Sensor, Collector, Decoy Server, and Sandbox components do not store any audit records, but rather forward all audit events securely over a TLS connection to the CommandPost. The CommandPost locally stores the audit record and also forwards the audit record in real time to an external audit server.

Table 4 corresponds to the audit events specified in Table 1 of the [CPP_ND_V2.2E and includes the audit events specified in the [CPP_ND_V2.2E] for optional and selected SFRs as selected in this ST.

6.1.2 FAU_GEN.2: User Identity Association

The logged audit records identify the date and time, the nature or type of the triggering event, an indication of whether the event succeeded or failed, and the identity of the user responsible for the event. The logged audit records also include event-specific content that includes at least all of the content required in **Table 4**.

6.1.3 FAU_STG.1: Protected Audit Trail Storage, FAU_STG_EXT.1: Protected Audit Event Storage, FAU_STG_EXT.4: Protected Local Audit Event Storage for Distributed TOEs

The TOE includes an internal log implementation that can be used to store and review audit records locally on the CommandPost. The local audit logs are stored on the CommandPost hard drive. The TOE is designed to retain audit records for an administrator configurable number of days (default is 190 days). Any audit record older than this

configured number of days will be removed. There is no enforced limit on the size of the audit table, but system disk space is monitored and once disk space reaches 80% capacity, the cleanup process will begin. The cleanup process is more aggressive than the configured number of days for storage retention. The audit cleanup involves a check for disk space at the time of adding an audit event. If disk is low, the 20 oldest audit events are deleted. If any events are deleted due to disk shortage, a status message is sent to the console, and an audit event to the effect is also logged. Authorized administrators can configure the storage time to help control how often audit records get overwritten. Only authorized administrators can access the local audit trail.

The audit records on the CommandPost are protected by database access control and there are no interfaces to delete individual audit records.

6.1.4 FAU_STG_EXT.5: Protected Remote Audit Event Storage for Distributed TOEs

The TOE Sensor, Collector, Sandbox, and Decoy server components do not locally store audit records. The generated audit records are buffered as files on the file system of each component, protected by strict file system permissions and transmitted over a secure HTTPS connection to the CommandPost for storage. If the communication link to the CommandPost is inadvertently broken, the components will buffer the audit records up to 80% of the available disk space. Once the 80% disk usage is reached, the buffering of the new audit records is not possible and the new audit records are overwritten. Once communication is re-established, the audit records will be transmitted to the CommandPost for central local storage and the buffer cleared.

The CommandPost can be configured to send collected and generated audit records to an external syslog server using TLS. When configured to send audit records to a syslog server, audit records are also written to the external syslog as they are written locally to the CommandPost audit log.

6.2 Cryptographic support

The TOE includes the OpenSSL 1.0.2k-fips library. The module provides implementations of all required cryptographic algorithms and mechanisms. The TOE includes NIST-validated cryptographic algorithms providing supporting cryptographic functions. The following functions have been certified in accordance with the identified standards.

Functions	Standards	Certificates
Asymmetric key generation (FCS_CKM.1)		
• RSA (3072 bits)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3	RSA C1950
• FFC key pair (3072 bits)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.1	DSA # C1950
• ECDSA (P-256, P-384, P-521 curves)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4;	ECDSA # C1950
Key Establishment (FCS_CKM.2)		
• FFC key pair (3072 bits)	NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography	KAS-FFC # C1950 KAS-FFC #A880
• ECDSA (P-256, P-384, P-521 curves)	NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”;	KAS-ECC # C1950
Encryption/Decryption (FCS_COP.1/Data Encryption)		

Functions	Standards	Certificates
• AES CBC (128 and 256 bits)	ISO 18033-3, CBC as specified in ISO 10116	AES # C1950
• AES GCM (128 and 256 bits)	ISO 18033-3, GCM as specified in ISO 19772	AES C1950
Cryptographic signature services (Signature Generation and Verification) (FCS_COP.1/SigGen)		
• RSA Digital Signature Algorithm (rDSA) (modulus 3072)	FIPS PUB 186-4 “Digital Signature Standard (DSS)”	RSA # C1950
Cryptographic hashing (FCS_COP.1/Hash)		
• SHA-1 (digest size 160 bits) • SHA-256 (digest size 256 bits) • SHA-384 (digest size 384 bits) • SHA-512 (digest size 512 bits)	ISO/IEC 10118-3:2004	SHS # C1950
Keyed-hash message authentication (FCS_COP.1/KeyedHash)		
• HMAC-SHA2-256 (key size 256 bits and digest size 256 bits) • HMAC-SHA2-384 (key size 384bits and digest size 384 bits)	ISO/IEC 9797-2:2011	HMAC # C1950
Random bit generation (FCS_RBG_EXT.1)		
• CTR-DRBG(AES) with one independent hardware-based noise source of 256 bits of non-determinism	ISO/IEC 18031:2011	DRBG # C1950

Table 7 Cryptographic Functions

6.2.1 FCS_CKM.1: Cryptographic Key Generation

Each TOE component generates RSA asymmetric keys using cryptographic key sizes of 3072 bits according to FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3. The TOE uses the RSA asymmetric keys for certificate based device authentication. No administrative configuration is required to generate the default length 3072-bit RSA keys. See the table above for Asymmetric key generation: RSA (3072-bit).

Each TOE component generates finite field-based key pairs (3072 bits) for key establishment that meet the following: NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography.

The TOE generates elliptic curve keys using the P-256, P-384, P-521 curves when an ECDHE TLS ciphersuite is negotiated that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4.

6.2.2 FCS_CKM.2: Cryptographic Key Establishment

The TOE performs finite field-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”. The TOE generally fulfills all of the NIST SP 800-56A requirements without extensions. The TOE does not perform any operations marked as “shall not” or “should not” and performs all operations marked as “shall” or “should”. The TOE utilizes elliptic curve key agreement in accordance with NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”; using the P-256, P-384, and P-521 curves when an ECDHE TLS ciphersuite is negotiated.

The TOE implements elliptic curve-based key establishment schemes and finite field-based key establishment schemes to communicate with an external audit server, the authentication server, the Fidelis Insight server, and the CommandPost TLS management interface, and with components of distributed TOE. The TOE acts as both a sender

and receiver. The TOE acts as a client for an external audit server, authentication server, and Fidelis Insight Server and as a server for the CommandPost TLS management interface.

See **Table 7 Cryptographic Functions** above for detail.

6.2.3 FCS_CKM.4: Cryptographic Key Destruction

The TOE uses the following secret keys, private keys and CSPs.

Key/CSP Name	Algorithm/Key Size	Description
RSA SGK	RSA 3072 bits	RSA signature generation key
RSA KDK	RSA 3072 bits	RSA key decryption key
FFC Keys	FFC key pair (3072 bits)	TLS session keys
AES EDK	AES 128, 256 bits	AES encrypt/decrypt key
HMAC Key	HMAC-SHA2-256 256 bits HMAC-SHA2-384 384 bits	HMAC keyed hash key
CTR_DRBG Key	AES 256 bits	Internal CTR_DRBG key variable
CommandPost Database Encryption Key	AES 256 bits	CommandPost data base encryption

Table 8 Secret keys, Private keys and CSPs

The TOE incorporates OpenSSL, which provides implementation of the cryptographic algorithms specified in **Table 7**. The TOE operates in FIPS mode and invokes the OpenSSL cryptomodule APIs to set up and maintain the full TLS session, using the underlying cryptographic algorithms as identified in **Table 7**. Therefore, all key generation, negotiation of session keys, and packet authentication is performed by the OpenSSL cryptomodule. Files such as private keys and certificates are manually uploaded to the TOE during initial setup. Changes can be performed by remote access (GUI) or locally on the appliance with physical access. To perform local changes a USB keyboard and VGA monitor is connected to the appliance to access the CLI.

All secret keys, plaintext private keys, CTR_DRBG state values, and CSPs (see **Table** 8 above) are managed by the cryptomodule and stored in RAM. The cryptomodule does not store any other secret or private keys or CSPs persistently (beyond the lifetime of an API call). All secret keys, plaintext private keys, CTR_DRBG state values, and CSPs are destroyed automatically by the API when no longer required by overwriting once with zeroes, destroying the reference to the key followed by a request for garbage collection. A delay in the destruction may occur when the TOE is writing zeros into the CSP file before it's been flushed. This is mitigated by calling file flush immediately after zeroizing it.

The TOE stores the Certificate files, CA-Certificate files, Private-Key files, and CRL files used in communication between TOE components, and in user communication with CommandPost, in PEM format. The TOE stores PEM format files in plaintext in non-volatile memory. The destruction method of the PEM format files consists of overwriting once with zeros and then deleting the file using the OS file system APIs. The keys, key material, and authentication credentials are protected from unauthorized disclosure.

The files are stored on the file system and in all cases the files are passed to OpenSSL via API calls that pass in the complete filename including full path. Each API call return is checked to make sure there were no errors. The cryptomodule itself does not return sensitive data values and is responsible for ensuring the memory that held those file contents gets zeroized. User passwords for users with local authentication are stored as iterated (multiple rounds) salted SHA-512 hashes in the OS password store (/etc/shadow).

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

The TOE performs 128/256-bit AES encryption/decryption as specified in ISO 18033-3, CBC mode as specified in ISO 10116 and GCM mode as specified in ISO 19772.

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

The TOE will provide cryptographic signature services using RSA Digital Signature Algorithm with key size of 3072 bits that meets the FIPS 186-4 Digital Signature Standard.

6.2.6 FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm)

The TOE performs SHA-256 and SHA-384 cryptographic hashing services in accordance with ISO/IEC 10118-3:2004 for TLS operations. User passwords for users with local authentication are stored as iterated (multiple rounds) salted SHA-512 hashes. The SHA-256 hash algorithm is used as part of HMAC, but is also used as part of RSA digital signature creation and verification. SHA-256 is used for software updates. SHA1 is used as the message digest algorithm for NTP verification.

6.2.7 FCS_COP.1/KeyedHash Cryptographic Operation (Keyed Hash Algorithm)

The TOE performs keyed-hash message authentication that meets the following: ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2. The key length, hash function used, block size, and output MAC lengths are identified in the table below.

Algorithm	Key Size	Block Size	Message Digest Size
SHA-256	256	512	256
SHA-384	384	1024	384

Table 9 Keyed Hash Description

Keyed-hashing message authentication services HMAC-SHA-256 and HMAC-SHA-384 are supported for TLS.

6.2.8 FCS_HTTPS_EXT.1: HTTPS Protocol

The TOE uses HTTPS when remote administrators connect to the TOE's CommandPost GUI. The TOE's HTTPS protocol complies with RFC 2818 by implementing an industry-standard HTTP web server together with an industry-standard TLS implementation: Apache httpd and OpenSSL.

Client authentication is required because of mutual authentication. The connection will be rejected if the certificate is invalid for any reason. The TOE implements the HTTPS/TLS protocols for the following communication links:

- CommandPost to audit server (TLS, TOE is client, optional mutual authentication)
  - Transmission of audit data
- CommandPost to LDAP server (TLS, TOE is client, no mutual authentication)
  - Admin authentication
- CommandPost to Fidelis Insight server (TLS, TOE is client, no mutual authentication)
  - Acquisition of software updates and threat intelligence
- Remote admin to CommandPost (HTTPS, TOE is server, no mutual authentication)
- Sensor to CommandPost (TLS, CommandPost is server, mutual authentication)
  - Registration and management
- Sensor to CommandPost (HTTPS, CommandPost is server, mutual authentication)
  - Transmission of audit data
- Collector to CommandPost (TLS, CommandPost is server, mutual authentication)
  - Registration and management

- Collector to CommandPost (HTTPS, CommandPost is server, mutual authentication)
  - Transmission of audit data
- CommandPost to Decoy Server (TLS, Decoy Server is server, mutual authentication)
  - Management
- CommandPost to Decoy Server (HTTPS, Decoy Server is server, mutual authentication)
  - Registration, transmission of audit data, and transmission of data samples used for analysis/reporting
- Sandbox to CommandPost (TLS, CommandPost is server, mutual authentication)
  - Management
- Sandbox to CommandPost (HTTPS, CommandPost is server, mutual authentication)
  - Registration, transmission of audit data, and transmission of data samples used for analysis/reporting
- Sensor to Collector (TLS, Collector is server, mutual authentication)
  - Transmission of metadata for aggregation

6.2.9 FCS_NTP_EXT.1: The NTP Protocol

The NTP server is configured during initial setup using the `System Setup` command from the `Fidelis Setup` screen. The CommandPost supports up to four NTP servers, the remaining TOE components sync their time to the CommandPost. The IP address of the NTP server is entered. The TOE supports NTP v4 (RFC 5905) with SHA1 as the message digest algorithm. The TOE will not update an NTP timestamp originating from broadcast and/or multicast addresses.

6.2.10 FCS_RBG_EXT.1: Random Bit Generation

The TOE implements a deterministic random bit generator that complies with ISO/IEC 18031:2011, using CTR_DRBG (AES). The entropy source is a 256-bit value derived from a hardware based noise source on Intel processors with Intel Secure Key technology. The Entropy Source provided by the Intel Secure Key RDRAND functionality of Cascade Lake and newer processors and assumed to generate at least 0.5 bits of entropy per sample.

6.2.11 FCS_TLSC_EXT.1: TLS Client Protocol Without Mutual Authentication

Section 6.2.8 identifies which TOE components acts as a TLS client and which TOE components acts as a TLS Server. The Decoy server does not function as a TLS client. The Decoy server will never initiate a connection to the CommandPost. When a Decoy alert is triggered, the Decoy server will wait for the CommandPost to connect and only then send the alerts and other information.
The TOE verifies that the presented identifier matches the reference identifier per RFC 6125 Section 6, IPv4 address in the SAN or IPv6 address in the SAN.

Supported reference identifiers include IP address or FQDN as the identifier. The hostname is also checked using SAN and CN.

The TOE only supports TLS 1.2. No other TLS protocol versions, such as, TLS 1.0, TLS 1.1, SSL 3.0, or SSL 2.0 are offered.

The TOE uses TLS 1.2 and supports the ciphersuites:

- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289.

The TOE presents the Supported Elliptic Curves Extension in the Client Hello with the secp256r1, secp384r1, and secp521r1 NIST curves and is enabled by default.

The CommandPost compares the authentication server, audit server, and the Fidelis Insight Server's IP address or FQDN as the identifier, presented in the TLS server certificate during TLS handshake, to the reference identifier of the respective server stored on CommandPost. A hostname check in the certificates is performed on Subject Alternative Names and Common Name. The TOE verifies that the presented identifier matches the reference identifier per RFC 6125 section 6

The TOE will only establish a trusted channel if the peer certificate is valid. There is no administrative override for allowing a connection to be accepted if the certificate is not validated.

Certificate pinning is not supported. Wildcards are supported for internal and external communications.

6.2.12 FCS_TLSC_EXT.2: TLS Client Support for Mutual Authentication

All intra-TOE communication requires mutual authentication between the components including the use of client-side certificates for TLS mutual authentication. Additionally, the CommandPost acting as a TLS client supports mutual authentication for secure communications with an external audit server. Section 6.2.8 identifies which TOE components acts as a TLS client and which TOE components acts as a TLS Server.

Initial configuration for each of the appliances is used to set network parameters: the host name, IP address, IP mask, gateway, and primary (and secondary, if applicable) DNS. Certificate files, CA-certificate files, and CRL files are then installed on each of the appliances before proceeding with registration to the CommandPost. The FQDN or the IP address of the audit server is configured as the reference identifier on the CommandPost. This reference identifier must match the CN in the audit server's certificate. For intra-TOE communication, the component matches the Common Name (CN) and/or Subject Alternative Name (SAN) with the endpoint's IP address. The IP address must match the CN or SAN advertised in the certificate.

After initial configuration and connecting each appliance to the network, the administrator adds all the components (Sensors, Collectors, Sandboxes, or Decoy Server) to CommandPost and successfully registers them.

The TOE verifies that the presented identifier matches the reference identifier according to RFC 6125 Section 6, IPv4 address in the SAN or IPv6 address in the SAN.

The Decoy server does not function as a TLS client. The Decoy server will never initiate a connection to the CommandPost. When a Decoy alert is triggered, the Decoy server will wait for the CommandPost to connect and only then send the alerts and other information.

6.2.13 FCS_TLSS_EXT.1: TLS Server Protocol Without Mutual Authentication

The TOE acts as a TLS Server when remote administrators connect to the TOE's GUI using HTTPS/TLS. Section 6.2.8 identifies which TOE components acts as a TLS client and which TOE components acts as a TLS Server. The TOE's HTTPS protocol complies with RFC 2818 and is implemented using TLS 1.2 (RFC 5246) supporting the following the ciphersuites:

The TOE uses TLS 1.2 and supports the ciphersuites:

- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289.

The key agreement parameters of the server key exchange message consist of Diffie-Hellman parameters with the key size of 3072 bits. The TOE generates EC Diffie-Hellman parameters over the NIST secp256r1 curve.

The TOE does not support session resumption or session tickets.

Keyed-hashing message authentication services HMA HMAC-SHA-256 and HMAC-SHA-384 are supported for TLS.

The TOE denies connections from clients requesting connections using SSL 2.0, SSL 3.0, TLS 1.0, and TLS 1.1.

6.2.14 FCS_TLSS_EXT.2: TLS Server Support for Mutual Authentication

The TOE additionally supports TLS server communication with mutual authentication. The TOE is deployed in a distributed architecture. All intra-TOE communication requires mutual authentication between the components including the use of client-side certificates for TLS mutual authentication. Section 6.2.8 identifies which TOE components acts as a TLS client and which TOE components acts as a TLS Server.

Certificate validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280. The TOE shall not establish a trusted channel if the client certificate is invalid. The fully qualified domain name (FQDN) must match the Common Name (CN) in the subject field of the component's certificate. The presented identifier has to match the reference identifier in order to establish the connection. The TSF uses the Common Name (CN) as the Subject Name and the DNS name as the Subject Alternative Name (SAN). The TOE does not support any fallback authentication functions such as username/password.

The TOE supports wildcards and IP addresses reference identifiers on internal TLS communication links (intra-TOE communications).

Certificate pinning is not supported.

6.3 Communication

The System Administrator joins the TOE components together to create the distributed TOE.

6.3.1 FCO_CPC_EXT.1 Component Registration Channel Definition

The Fidelis Network and Fidelis Deception v9.3.3 is deployed as a distributed TOE configuration. Users must install and set up X.509 certificates and enable NIST CAVP validated encryption for data storage on the CommandPost. Fidelis components enforce NIST CAVP validated security requirements for Cryptographic Modules, and only accept certificates that satisfy its requirements. The TOE can be configured to support public key, certificate-based authentication for all TLS-based communication. Fidelis requires X.509 certificates to be signed by an external Certificate Authority (CA) to import them. The TOE stores the Certificate files, CA-Certificate files, Private-Key files, and CRL files used in communication between TOE components in PEM format. This includes both the end point and all other certificates in the trust chain. Each component in the distributed TOE configuration will require its own unique private key and a corresponding public key certificate.

Initial configuration for each of the appliances is performed by directly using the CLI by attaching a keyboard and monitor to the appliance. The System Setup is used to set network parameters: the host name, IP address, IP mask, gateway, and primary (and secondary, if applicable) DNS, and the NTP server. Certificate files, CA-certificate files, CRL files should be installed on each of the appliances before proceeding with registration to the CommandPost. The Fully Qualified Domain Name (FQDN) component host name must match the Common Name (CN) in the subject field of the component's certificate(s).

After initial configuration and connecting each appliance to the network, the administrator must add all the components (Sensors, Collectors, Sandboxes, and Decoy Servers) to the CommandPost and successfully register them. The appliance name, IP address and description are entered into the CommandPost. The appliance IP address must match the address provided in the initial configuration and setup. After registration, CommandPost attempts to communicate to the newly registered component (the Sensor, the Collector, or the Sandbox or the Sensor, or the Decoy Server) at the specified IP address over a secure TLS tunnel as described in FPT_ITT.1/Join.

Communication between CommandPost and each component is verified by running the following command on CommandPost for every other component:

```
/FSS/sbin/fping -s [fully-qualified sensor hostname] -k
```

The System Administrator disables the communication between the CommandPost and the distributed components by remotely authenticating to the CommandPost, navigating to the **System > Components** page of the GUI, which an administrator clicks on the **Unregister** button to disable a registered component.

6.4 Identification and authentication

The TOE requires users to be identified and authenticated before they can access any of the TOE functions.

6.4.1 FIA_AFL.1 Authentication Failure Management

The TOE can detect when an Administrator configurable number (range = 1 to 999) of failed remote authentication attempts has been reached. When the defined number of unsuccessful authentication attempts has been reached, the remote administrator accessing the TOE via HTTPS is locked out until another administrator resets the password.

To reactivate the locked-out account, an administrator using the CommandPost GUI must reset that account's password. Authentication failures by remote Administrators cannot lead to a situation where no Administrator access is available to the TOE. The TOE includes a default "admin" user account that is not used after initial configuration for normal operation. This user account cannot be deleted and can be used to log in to unlock other administrative accounts in the event that all other remote user accounts are locked.

6.4.2 FIA_PMG_EXT.1: Password Management

The TOE supports passwords composed from any combination of upper and lower case letters, numbers, and the following special characters: "!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [blank space, "~", "`", "_", "+", "-", "=", "{", "}", "|", "[", "]", ":", ";", "<", ">", ",", ".", and "/". Single and double quotes or back slashes are not allowed.

The minimum password length is administrator configurable from 1 to 999 characters. The recommended value is 8.

6.4.3 FIA_UAU.7: Protected Authentication Feedback

When logging in, the TOE will not echo passwords so that passwords are not inadvertently displayed to the user and any other users that might be able to view the login display.

6.4.4 User FIA_UIA_EXT.1: Identification and Authentication, FIA_UAU_EXT.2: Password-based Authentication Mechanism

Administrators manage the TOE remotely using a web-based GUI accessed via HTTPS to the CommandPost or locally on the CommandPost using the CLI by a directly connected USB keyboard and a monitor to the appliance's VGA connector. However the TOE is not intended to be managed locally after initial configuration. For each session, the user is required to log in prior to successfully establishing a session through which TOE functions can be exercised. Note that the only capabilities allowed prior to users authenticating are the display of the warning banner before authentication, and acceptance of the end-user license. The user only needs to accept the license once for each software release, after which the license acceptance message will not display. The banner is displayed on every login attempt.

Users can be defined locally within all of the TOE components with a user identity, password, and user role. Alternately, remote web-based GUI CommandPost users can be defined within an external LDAP (e.g. Active Directory) server for authentication and to define the user's role in the TOE.

Locally defined users are authenticated directly by the TOE, while remotely defined users are authenticated by the external LDAP server and the result is enforced by the TOE. In both authentication methods the administrator must enter a valid username and password that matches those defined for that user. Any resulting session is dependent upon successful authentication and established sessions are associated with the roles (see Section 6.5.6) assigned to the user.

Note also that should a user have their session terminated (e.g., due to inactivity), they are required to successfully authenticate, by reentering their identity and authentication data, in order to establish a new session.

6.4.5 FIA_X509_EXT.1/Rev: X.509 Certificate Validation

The TOE uses X.509v3 certificates as defined by RFC 5280 to support the TLS connection with external syslog server, authentication server, and Fidelis Insight Server. The TOE validates the revocation status of the certificate using a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3.

The validity check of the certificates is performed during the TLS connection for remote administrative access, for communication with external authentication servers, with syslog audit servers, with the Fidelis Insight Server, and by internal processes for intra-TOE communications.

The certificate path terminates with a trusted CA certificate. The certificate path is validated by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.

The X.509 certificates are validated using the certificate path validation algorithm defined in RFC 5280, which can be summarized as follows:

- The public key algorithm and parameters are checked;
- The current date/time is checked against the validity period of the certificate;
- The revocation status is checked, whether by CRL to ensure the certificate is not revoked;
- The issuer name is checked to ensure that it equals the subject name of the previous certificate in the path;
- Name constraints are checked, to make sure the subject name is within the permitted subtrees list of all previous CA certificates and not within the excluded subtrees list of any previous CA certificate;
- The asserted certificate policy OIDs are checked against the permissible OIDs as of the previous certificate, including any policy mapping equivalencies asserted by the previous certificate;
- Policy constraints and basic constraints are checked, to ensure that any explicit policy requirements are not violated and that the certificate is a CA certificate, respectively. This step is crucial in preventing some man in the middle attacks;
- The path length is checked to ensure that it does not exceed any maximum path length asserted in this or a previous certificate;
- The key usage extension is checked to ensure that is allowed to sign certificates; and
- Any other critical extensions are recognized and processed.

The certificate chain is validated to the root, and each certificate is checked against CRL. The TOE supports a hierarchy comprising of at least a self-signed root certificate, a subordinate CA certificate and a TOE identity certificate signed by an external Certificate Authority (CA).

The TOE uses the following rules for validating the extendedKeyUsage field:

- Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
- Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
- Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.

The TOE does not use certificates for trusted updates and executable code integrity verification. Therefore, the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field is not applicable and therefore is trivially satisfied.

6.4.6 FIA_X509_EXT.1/ITT(1), FIA_X509_EXT.1/ITT(2): X.509 Certificate Validation

The CommandPost, Collectors, Sensors, Sandbox, and Decoy Server use X.509v3 certificates for peer authentication as defined by RFC 5280 to support the TLS connection for the distributed TOE communication. The certificate validation is performed during initial configuration as well as during the TLS connection establishment. The CommandPost, Collectors, Sensors, and Sandbox validate the revocation status of the certificate using a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3.

The CommandPost, Collectors, Sensors, Sandbox, and Decoy Server use the following rules for validating the extendedKeyUsage field:

- Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
- Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.

The TOE does not use certificates presented for OCSP responses. Therefore, the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field is not applicable and therefore is trivially satisfied.

The Decoy Server does not perform revocation checking. The Decoy Server communicates only with CommandPost, so the only certificate it will ever receive is the one from Command Post as part of establishing a TLS connection between the two. Therefore, the Decoy Server is not required to do revocation checking of that certificate.

6.4.7 FIA_X509_EXT.2: X.509 Certificate Authentication

The TOE implements X.509v3 certificates as defined by RFC 5280 to support the HTTPS/TLS connections for all intra-TOE communication and optionally for an external syslog server,

During initial configuration, the certificate files, CA-certificate files, CRL files are installed on each of the appliances. The audit server certificate is subsequently installed on the audit server along with the CA certificate, and CRL. Each TOE device contains only one end point certificate which is stored in a trusted store of the device.

The TOE can be configured for automatic periodic CRL updates as part of the initial setup process. The following subsystems that support mutual authentication can be configured to automatically update CRLs:

1. CommandPost webserver,
2. Internal TLS communications (rconfig) on all components.

If a CRL is unavailable during the authentication process, the certificate will be rejected. If the TOE cannot establish a connection to determine the validity of a certificate, the TSF shall will not accept the certificate.

6.4.8 FIA_X509_EXT.3: X.509 Certificate Requests

The TOE generates a Certificate Request Message as specified by RFC 2986. The TOE provides the Common Name, Organization, Organizational Unit, and Country information in the request.

Each TOE component must generate their own CSRs locally as part of initial setup prior to registration so that registration is done using certificates. The TOE's validation of the certificate chain is based on a trusted CA which must be present on the TOE. Otherwise, the TOE will reject the certificate and will not associate it with the CSR.

6.5 Security management

The TOE provides a GUI to access security management functions. Security management commands are limited to administrators and are available only after they have provided acceptable user identification and authentication data to the TOE. The TOE controls user access to the TOE and resources based on user role. Users are given permission to access a set of commands and resources based on their user role.

6.5.1 FMT_MOF.1/Functions Management of Security Functions Behaviour

The local audit logs are stored on the CommandPost hard drive. The TOE is designed to retain the local audit records for an authorized administrator configurable number of days (default is 190 days). Any audit record older than this configured number of days will be removed.

6.5.2 FMT_MOF.1/ManualUpdate: Management of Security Functions Behaviour Requests

The initiation of manual TOE updates is restricted to System Administrators. Manual updates are initiated on the CommandPost node, which can be used to push updates to distributed nodes.

6.5.3 FMT_MTD.1/CoreData: Management of TSF Data

The ability to manage the TSF data is restricted to System Administrators. No administrative functions are accessible prior to administrator log-in.

6.5.4 FMT_MTD.1/CryptoKeys: Management of TSF Data

The ability to manage the cryptographic keys, Database encryption key, and certificates (e.g. generating/import of, changing, or deleting of cryptographic keys) is restricted to System Administrators.

6.5.5 FMT_SMF.1: Specification of Management Functions

The TOE provides the ability for administrators to remotely manage the TOE from the CommandPost GUI component or locally by the CLI using a USB keyboard and a monitor to the appliance VGA connector on the CommandPost. However, the TOE in the distributed configuration is designed to be managed using the CommandPost GUI from a remote HTTPS/TLS client. Following the initial configuration, all changes should be performed by an authorized user from the CommandPost GUI.

The following management functions are performed locally on the CommandPost:

- Configure the access banner,
- Ability to manage the cryptographic keys,
- Ability to configure the cryptographic functionality,
- Ability to configure the interaction between TOE components,
- Ability to import X.509v3 certificates to the TOE's trust store,
- Ability to configure NTP,
- Ability to configure the session inactivity time before session termination or locking.

The following management functions are performed by a remote administrator accessing the CommandPost GUI:

- Ability to configure the session inactivity time before session termination or locking,
- Update the TOE, and to verify the updates using the hash comparison capability prior to installing those updates,
- Ability to configure the authentication failure parameters for FIA_AFL.1,
- Ability to manage the cryptographic keys,
- Ability to configure the cryptographic functionality,
- Ability to configure the interaction between TOE components,
- Ability to import X.509v3 certificates to the TOE's trust store,
- Ability to re-enable an Administrator account,
- Ability to configure audit behaviour (configure the retention period for audit records).

The administrator has the ability to configure the interaction between TOE components. The administrator can disable the communication between the CommandPost and the distributed components by remotely authenticating to the CommandPost, navigating to the **System** > **Components** page of the GUI, and clicking on the **Unregister** button.

6.5.6 FMT_SMR.2: Restrictions on Security Roles

The TOE includes pre-defined user roles, of which only the user role: System Administrator is considered a ‘Security Administrator’ as defined in the [CPP_ND_V2.2E]. Users with the System Administrator role are capable of managing the security functions of the TOE. The security management functions required by the PP are accessible via the GUI or by directly attaching a keyboard and monitor to the CommandPost to access the CLI.

The TOE includes other pre-defined roles that represent logical subsets of the System Administrator role. Only users with the System Administrator role can manage all aspects of the TOE.

6.6 Protection of the TSF

The TOE ensures that sensitive information such as passwords and cryptographic keys are stored such that they are not accessible even to an administrator. The TOE relies on the use of an NTP server in its operational environment for clock synchronization to ensure that reliable time information is available (e.g., for log accountability).

The TOE includes functions to perform self-tests and mechanisms for the update of the TOE software/firmware and verification of the cryptographic functions.

6.6.1 FPT_APW_EXT.1: Protection of Administrator Passwords

The TOE prevents access to locally-stored cryptographically protected passwords and does not disclose any keys stored in the TOE. The TOE protects user passwords stored in /etc/shadow using a salted iterated SHA-512 hash of the password. The TOE protects the LDAP credentials stored in/FSS/etc/ldap.cf using AES-CBC 256-bit based encryption pseudo-random salt. The TOE does not offer any functions that will disclose to any users a stored cryptographic key or password. See Section 6.2 for more information about stored keys and passwords.

Credentials Purpose	Username Obfuscation	Password Obfuscation	Storage Location	Component
Command Post Web Server Access	None.	Salted iterated SHA-512	Maria DB /etc/shadow	Command Post
LDAP Server Access	None.	AES-CBC 256-bit based encryption pseudo-random salt	/FSS/etc/ldap.cf	Command Post

Table 10 User Credentials

6.6.2 FPT_ITT.1 / FPT_ITT.1/Join: Basic Internal TSF Data Transfer Protection

The TOE is deployed in a distributed TOE environment. The TOE components are manually pre-configured with information necessary to build the inter-TOE communications channel. After initial configuration and connecting each appliance to the network, the administrator adds all the components (Sensors, Collectors, Sandboxes, and Decoy Servers) to the CommandPost to register them. The components IP addresses are manually entered and registered with the CommandPost. The TOE does not provide an automated discovery process. The registration channel is protected by TLS or HTTPS as identified in FPT_ITT.1/Join. The registration channel is adopted as a continuing internal communication channel between different TOE components. The CommandPost will communicate to the newly registered components (the Sensor, the Collector, or the Sandbox or the Sensor or Decoy Server) at the specified IP address over a secure TLS tunnel.

6.6.3 FPT_SKP_EXT.1: Protection of TSF Data (for Reading of all Pre-shared, Symmetric and Private Keys)

While the administrative interface is function rich, the TOE is designed specifically to prevent access to locally-stored cryptographically protected passwords and does not disclose any keys stored in the TOE. The TOE protects user passwords by saving a salted iterated SHA-512 hash of the password. The TOE does not offer any functions that will disclose to any users a stored cryptographic key or password. See Section 6.2 for more information about stored keys and passwords.

6.6.4 FPT_STM_EXT.1: Reliable Time Stamps

The TOE is a hardware appliance or a virtual appliance image installed on a hardware appliance that includes a hardware-based real-time clock. The TOE relies on the use of an NTP server in its operational environment for clock synchronization. The TOE's embedded OS in conjunctions with the NTP Server manages the clock and exposes

administrator clock-related functions. The CommandPost server collects time data from external NTP server(s) and then syncs the time with the other components. The clock is used for audit record time stamps, measuring session activity for termination, and for cryptographic operations based on time/date.

6.6.5 FPT_TST_EXT.1: TSF Testing

Every TOE component performs all self-tests (software module integrity tests, cryptographic known answer tests, and entropy source online health tests) on start-up. The hardware entropy source (RDRAND) performs online health tests for every entropy sample returned while OpenSSL performs self-tests of FIPS DRBG on start-up.

The TOE components process manager service is responsible for bringing up and verifying integrity of all relevant TOE components processes. If a system daemon fails to start for other reasons than integrity check failure, the event will be logged in /var/log/messages. Depending on the daemon, TOE component, and the reason for its failure, more detailed information may be found in the corresponding log in /FSS/log/. The Power On Self-Test (POST) failure messages include identification of the distributed component that sustained the failure.

The TOE includes CAVP certified OpenSSL binaries which are included in the POST to ensure the correct operation of all relevant cryptographic functions. In case of fatal POST failures, the administrator should contact Fidelis Support immediately.

6.6.6 FPT_TUD_EXT.1: Trusted Update

The TOE provides graphical user interfaces for administrators to update the TOE, and to query the currently executing software version of the TOE. The CommandPost Version Control GUI page will list all currently active components accessible from the CommandPost and display the installed version of the TOE. This includes the CommandPost Management Console (the CommandPost that you are currently logged into), and all registered TOE components. Software updates are available as a tar package. The update package and its SHA256 hash are published on Fidelis support website. An administrator with proper credentials downloads the update via HTTPS.

The administrator performs the following steps from the System / Version Control configuration screen of the CommandPost Management Console GUI to ensure the integrity of the TOE update package and to update the TOE:

1) Scheduled Installs are disabled.
2) Download Control is set to ‘When a new version is available’ to ‘Notify Only’
3) When a new update package is available, download the Fidelis update installation file from: www.fidelissecurity.com/support to a folder on the local workstation.
4) Verify the SHA256 hash of the package (outside the TOE) for each TOE component.
5) If the hash values agree, upload the package to the CommandPost using the File Management configuration in System / Version Control.
6) The TOE calculates the SHA256 hash and displays it to the administrator.
7) If the hash values agree; initiate installation to the distributed components from the CommandPost.
8) CommandPost will copy the package to the desired component.
9) When the package reaches the intended component, the component will then be shut down, the update installed, and restored to functionality at the new version.

The TOE provides mechanisms that support the continuous proper functioning during the trusted update of the distributed TOE components. When all components are selected for a trusted software update, CommandPost ensures that it performs updates of TOE components in the right order (Sensors, Collectors, Sandboxes first, then CommandPost itself). Only tested and verified update paths are allowed: usually, only one major version at a time. Random version jumps are not permitted. The software is written and tested in the assumption that CommandPost version may be lower than that of other components.

The Decoy Server is updated separately from the other components.

1) When a new update package is available, download the Fidelis update installation file from: www.fidelissecurity.com/support to a folder on the local workstation.
2) Verify the SHA256 hash of the package (outside the TOE) for Decoy Server update.
3) If the hash values agree, upload the package to the CommandPost using the File Management configuration in System / Version Control.
4) The TOE calculates the SHA256 hash and displays it to the administrator.

5) If the hash values agree; initiate installation to the Decoy Server component from the CommandPost.
6) CommandPost will copy the package to the Decoy Server component.

6.7 TOE access

The TOE can be configured to display an informative banner when an administrator establishes an interactive session. The TOE can also enforce an administrator-defined inactivity timeout value after which the inactive session (local or remote) will be terminated. Finally, the TOE allows administrators to terminate their own session.

6.7.1 FTA_SSL.3: TSF-initiated Termination

The TOE can be configured by an administrator to set an interactive remote session timeout value (any integer value greater than zero in minutes) for user sessions. The default timeout is 15 minutes. Note also that should a user have their session terminated (e.g., due to inactivity), they are required to successfully authenticate, by reentering their identity and authentication data, in order to establish a new session.

6.7.2 FTA_SSL.4: User-initiated Termination

A user can terminate their own session and securely log out of CommandPost by moving the mouse over the User Account box at the top of the page. A dropdown will appear showing the logout function. A user can terminate their local administrative session by entering the “exit” command.

6.7.3 FTA_SSL_EXT.1: TSF-initiated Session Locking

The CommandPost can be configured by an administrator using the CLI by directly attaching a USB keyboard and VGA monitor to the appliance to set an interactive local session timeout value (any integer value greater than zero in minutes) for user sessions. The default timeout is 15 minutes. Note also that should a user have their session terminated (e.g., due to inactivity), they are required to successfully authenticate, by reentering their identity and authentication data, in order to establish a new session.

6.7.4 FTA_TAB.1: Default TOE Access Banners

The TOE can be configured by an administrator to display advisory banners prior to allowing an administrator to establish an administrative user session. The banner will be displayed when accessing the TOE locally or via the GUI. The banner on remote nodes has to be configured individually.

6.8 Trusted path/channels

To support secure remote administration, the TOE includes implementations of HTTPS. An authorized administrator can establish secure remote connections with the TOE using HTTP over TLS.

The TOE protects communication with an external log server, the Fidelis Insight Server, and an LDAP authentication server to prevent unintended disclosure or modification of audit records.

6.8.1 FTP_ITC.1: Inter-TSF trusted channel

The TOE can be configured to export audit records to an external audit server. The TOE uses TLS v1.2, to protect communications between itself and the audit server. The TOE initiates communication via the trusted channel for the audit server.

The TOE uses TLS to protect communications between itself and components in the operational environment including the audit server, Fidelis Insight Server, and an LDAP authentication server. The TOE will automatically re-establish communications in the event of a temporary network outage. An administrator with proper credentials downloads the TOE updates via HTTPS from the Fidelis Insight Server.

The TOEs secure protocols are supported by NIST-validated cryptographic mechanisms included in the TOE implementation.

6.8.2 FTP_TRP.1/Admin: Trusted Path

The TOE protects administrator communications from network workstations using HTTPS. To successfully establish an interactive administrative session, the administrator must be able to provide acceptable user credentials (e.g., user id and password) either locally or to a remote LDAP server after which they will be able to access the GUI features. The secure protocols are supported by NIST-validated cryptographic mechanisms included in the TOE implementation.

7. Protection Profile Claims

The SFRs have all been drawn from the Protection Profile (PP): collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020, [CPP_ND_V2.2E] and including the following optional SFRs: FAU_STG.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.2, FCO_CPC_EXT.1, FIA_X509_EXT.1/ITT, FPT_ITT.1, FPT_ITT.1/Join and the following selection-based SFRs: FAU_GEN_EXT.1, FAU_STG_EXT.4, FAU_STG_EXT.5, FCS_HTTPS_EXT.1, FCS_NTP_EXT.1, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1, FIA_X509_EXT.1/Rev, FIA_X509_EXT.2, FIA_X509_EXT.3, FMT_MOF.1/Functions, FMT_MTD.1/CryptoKeys.

As explained in Section 3, Security Problem Definition, the Security Problem Definition of the [CPP_ND_V2.2E] has been included by reference into this ST.

As explained in Section 4, Security Objectives, the Security Objectives of the [CPP_ND_V2.2E] have been included by reference into this ST.

The following table identifies all the Security Functional Requirements (SFRs) in this ST. Each SFR is reproduced from the [CPP_ND_V2.2E] and operations completed as appropriate.

Requirement Class	Requirement Component	Source
FAU: Security audit	FAU_GEN.1: Audit Data Generation	CPP_ND_V2.2E
	FAU_GEN_EXT.1: Audit Data Generation	CPP_ND_V2.2E
	FAU_GEN.2: User Identity Association	CPP_ND_V2.2E
	FAU_STG.1: Protected Audit Trail Storage	CPP_ND_V2.2E
	FAU_STG_EXT.1: External Audit Trail Storage	CPP_ND_V2.2E
	FAU_STG_EXT.4: Protected Local Audit Event Storage for Distributed TOEs	CPP_ND_V2.2E
	FAU_STG_EXT.5: Protected Remote Audit Event Storage for Distributed TOEs	CPP_ND_V2.2E
FCS: Cryptographic support	FCS_CKM.1: Cryptographic Key Generation (for asymmetric keys)	CPP_ND_V2.2E
	FCS_CKM.2: Cryptographic Key Establishment (Refined)	CPP_ND_V2.2E
	FCS_CKM.4: Cryptographic Key Destruction	CPP_ND_V2.2E
	FCS_COP.1(1) Cryptographic Operation (AES Data Encryption/Decryption)	CPP_ND_V2.2E
	FCS_COP.1(2) Cryptographic Operation (Signature Generation and Verification)	CPP_ND_V2.2E
	FCS_COP.1(3) Cryptographic Operation (Hash Algorithm)	CPP_ND_V2.2E
	FCS_COP.1(4) Cryptographic Operation (Keyed Hash Algorithm)	CPP_ND_V2.2E
	FCS_HTTPS_EXT.1: HTTPS Protocol	CPP_ND_V2.2E
	FCS_NTP_EXT.1: NTP Protocol	CPP_ND_V2.2E
	FCS_RBG_EXT.1: Random Bit Generation	CPP_ND_V2.2E
	FCS_TLSC_EXT.1: TLS Client Protocol Without Mutual Authentication	CPP_ND_V2.2E
	FCS_TLSC_EXT.2: TLS Client Support for Mutual Authentication	CPP_ND_V2.2E
	FCS_TLSS_EXT.1: TLS Server Protocol Without Mutual Authentication	CPP_ND_V2.2E

Requirement Class	Requirement Component	Source
	FCS_TLSS_EXT.2: TLS Server Support for Mutual Authentication	
FIA: Identification and authentication	FIA_AFL.1: Authentication Failure Management	CPP_ND_V2.2E
	FIA_PMG_EXT.1: Password Management	CPP_ND_V2.2E
	FIA_UAU.7: Protected Authentication Feedback	CPP_ND_V2.2E
	FIA_UAU_EXT.2: Password-based Authentication Mechanism	CPP_ND_V2.2E
	FIA_UIA_EXT.1: User Identification and Authentication	CPP_ND_V2.2E
	FIA_X509_EXT.1/Rev: X.509 Certificate Validation	CPP_ND_V2.2E
	FIA_X509_EXT.1/ITT(1): X.509 Certificate Validation	CPP_ND_V2.2E
	FIA_X509_EXT.1/ITT(2): X.509 Certificate Validation	CPP_ND_V2.2E
	FIA_X509_EXT.2: X.509 Certificate Authentication	CPP_ND_V2.2E
	FIA_X509_EXT.3: X.509 Certificate Requests	CPP_ND_V2.2E
FMT: Security management	FMT_MOF.1/Functions: Management of Security Functions Behaviour	CPP_ND_V2.2E
	FMT_MOF.1/ManualUpdate: Management of security functions behaviour	CPP_ND_V2.2E
	FMT_MTD.1/CoreData: Management of TSF Data	CPP_ND_V2.2E
	FMT_MTD.1/CryptoKeys: Management of TSF Data	CPP_ND_V2.2E
	FMT_SMF.1:Specification of Management Functions	CPP_ND_V2.2E
	FMT_SMR.2: Restrictions on Security Roles	CPP_ND_V2.2E
FPT: Protection of the TSF	FPT_SKP_EXT.1: Extended: Protection of TSF Data (for reading of all symmetric keys)	CPP_ND_V2.2E
	FPT_APW_EXT.1: Protection of Administrator Passwords	CPP_ND_V2.2E
	FPT_ITT.1: Basic internal TSF data transfer protection	CPP_ND_V2.2E
	FPT_ITT.1/Join: Basic internal TSF data transfer protection	CPP_ND_V2.2E
	FPT_STM_EXT.1: Reliable Time Stamps	CPP_ND_V2.2E
	FPT_TST_EXT.1: TSF Testing	CPP_ND_V2.2E
	FPT_TUD_EXT.1: Trusted Update	CPP_ND_V2.2E
FTA: TOE access	FTA_SSL.3: TSF-initiated Termination	CPP_ND_V2.2E
	FTA_SSL.4: User-initiated Termination	CPP_ND_V2.2E
	FTA_SSL_EXT.1: TSF-initiated Session Locking	CPP_ND_V2.2E
	FTA_TAB.1: Default TOE Access Banners	CPP_ND_V2.2E
FTP: Trusted path/channels	FTP_ITC.1: Inter-TSF trusted channel (Refined)	CPP_ND_V2.2E
	FTP_TRP.1: Trusted Path	CPP_ND_V2.2E

Table 11 SFR Protection Profile Sources

8. Rationale

This security target includes by reference the [CPP_ND_V2.2E Security Problem Definition, Security Objectives, and Security Assurance Requirements. The security target makes no additions to the [CPP_ND_V2.2E] assumptions.

[CPP_ND_V2.2E] security functional requirements have been reproduced with the Protection Profile operations completed. Operations on the security requirements follow [CPP_ND_V2.2E] application notes and assurance activities. Consequently, [CPP_ND_V2.2E] rationale applies but is incomplete. The TOE Summary Specification rationale below serves to complete the rationale required for the security target.

8.1 TOE Summary Specification Rationale

Each subsection in Section 6, the TOE Summary Specification, describes a security function of the TOE. Each description is followed with rationale that indicates which requirements are satisfied by aspects of the corresponding security function. The security functions work together to satisfy all of the security functional requirements. Furthermore, all of the security functions are necessary in order for the TSF to provide the required security functionality.

This Section in conjunction with Section 6, the TOE Summary Specification, provides evidence that the security functions are suitable to meet the TOE security requirements. The collection of security functions work together to provide all of the security requirements. The security functions described in the TOE summary specification are all necessary for the required security functionality in the TSF, **Table 12 Security Functions vs. Requirements Mapping 12**. demonstrates the relationship between security requirements and security functions.

	Security audit	Cryptographic support	Communication	Identification and authentication	Security management	Protection of the TSF	TOE access	Trusted path/channels
FAU_GEN.1	X							
FAU_GEN_EXT.1	X							
FAU_GEN.2	X							
FAU_STG.1	X							
FAU_STG_EXT.1	X							
FAU_STG_EXT.4	X							
FAU_STG_EXT.5	X							
FCS_CKM.1		X						
FCS_CKM.2		X						
FCS_CKM.4		X						
FCS_COP.1/DataEncryption		X						
FCS_COP.1/SigGen		X						
FCS_COP.1/Hash		X						
FCS_COP.1/KeyedHash		X						
FCS_HTTPS_EXT.1		X						
FCS_NTP_EXT.1		X						
FCS_RBG_EXT.1		X						
FCS_TLSC_EXT.1		X						
FCS_TLSC_EXT.2		X						
FCS_TLSS_EXT.1		X						

	Security audit	Cryptographic support	Communication	Identification and authentication	Security management	Protection of the TSF	TOE access	Trusted path/channels
FCS_TLSS_EXT.2		X						
FCO_CPC_EXT.1			X					
FIA_AFL.1				X				
FIA_PMG_EXT.1				X				
FIA_UAU.7				X				
FIA_UAU_EXT.2				X				
FIA_UIA_EXT.1				X				
FIA_X509_EXT.1/Rev				X				
FIA_X509_EXT.1/ITT(1)				X				
FIA_X509_EXT.1/ITT(2)				X				
FIA_X509_EXT.2				X				
FIA_X509_EXT.3				X				
FMT_MOF.1.1/Functions					X			
FMT_MOF.1/ManualUpdate					X			
FMT_MTD.1/CoreData					X			
FMT_MTD.1/CryptoKeys					X			
FMT_SMF.1					X			
FMT_SMR.2					X			
FPT_APW_EXT.1						X		
FPT_ITT.1						X		
FPT_ITT.1/Join						X		
FPT_SKP_EXT.1						X		
FPT_STM_EXT.1						X		
FPT_TST_EXT.1						X		
FPT_TUD_EXT.1						X		
FTA_SSL.3							X	
FTA_SSL.4							X	
FTA_SSL_EXT.1							X	
FTA_TAB.1							X	
FTP_ITC.1								X
FTP_TRP.1								X

Table 12 Security Functions vs. Requirements Mapping