Fidelis Network v9.0.3 Security Target

Version 1.0
24 August 2018

Prepared for:

Fidelis Cybersecurity
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 v9.0.3 provided by Fidelis Cybersecurity. The Fidelis Network product is a network security solution for advanced threat detection.

The focus of this evaluation is on the TOE functionality supporting the claims in the collaborative Protection Profile for Network Devices, Version 2.0 + Errata 20180314, 14-March-2018, [CPP_ND_V2.0E] (See section 1.2 for specific version information). The security functionality specified in [CPP_ND_V2.0E] 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 v9.0.3 Security Target

ST Version – Version 1.0

ST Date – 24 August 2018

TOE Identification – Fidelis Network v9.0.3

The TOE consists of the following Fidelis components:

- Fidelis K2 v9.0.3
- Fidelis Collector v9.0.3
- Fidelis Sensor v9.0.3
- Fidelis Sandbox v9.0.3

The K2, Collector and Sensor components are available in various form factors, as outlined in the following table:

Component	Appliance Models	Virtual Models
K2	K2 appliance	K2 VM
Collector	Collector SA2 Collector XA2 Collector XA4 Collector Controller 2 Collector Controller 10G	Collector SA VM
Sensor	Direct 50 Direct 100	Direct VM

Component	Appliance Models	Virtual Models
	Direct 250 Direct 500 Direct 1000 Direct 2500 Direct 5000 Direct 10G	
	Internal 1000 Internal 2500 Internal 5000 Internal 10G	Internal VM
	Web	Web VM
	Mail 250 Mail 500 Mail 1000	Mail VM 250 Mail VM 500 Mail VM 1000

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

Two further form factors combine three virtual models in a single hardware appliance:

- Fidelis XPS Scout+ AP v9.0.3 (includes a K2 VM, a Direct 1000 VM, and a Collector SA VM in one box)
- Fidelis XPS Scout+ IR v9.0.3 (includes a K2 VM, a Direct 1000 VM, and a Collector SA VM in one box).

Each virtual model in its evaluated configuration is installed on a hardware platform that includes VMware ESXi with vSphere 5.1, 5.5, 6.0 or 6.5 and an Intel Core or Xeon processor based on the Ivy Bridge or Haswell microarchitecture, which implement Intel Secure Key. The virtual module must be the only guest running in the virtual environment.

1.2 Conformance Claims

This TOE is conformant to the following CC specifications:

- collaborative Protection Profile for Network Devices, Version 2.0 + Errata 20180314, 14-March-2018, [CPP_ND_V2.0E] and including the following optional SFRs: FAU_STG.1, FCS_HTTPS_EXT.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.2, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1, FIA_X509_EXT.1/ITT, FPT_ITT.1/Join, and FCO_CPC_EXT.1. The following NIAP Technical Decisions apply to this PP and have been accounted for in the ST development and the conduct of the evaluation:
    - TD0228: NIT Technical Decision for CA certificates - basicConstraints validation
    - TD0256: NIT Technical Decision for Handling of TLS connections with and without mutual authentication
    - TD0257: NIT Technical Decision for Updating FCS_DTLSC_EXT.x.2/FCS_TLSC_EXT.x.2 Tests 1-4
    - TD0262: NIT Technical Decision for TLS server testing - Empty Certificate Authorities list
    - TD0289: NIT technical decision for FCS_TLSC_EXT.x.1 Test 5e
    - TD0290: NIT technical decision for physical interruption of trusted path/channel
    - TD0291: NIT technical decision for DH14 and FCS_CKM.1

  - TD0322: NIT Technical Decision for TLS server testing - Empty Certificate Authorities list
  - TD0324: NIT Technical Decision for Correction of section numbers in SD Table 1
- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 4, September 2012.
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1 Revision 4, September 2012.
  - 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, iteration is indicated by adding a string starting with "/" (e.g. "FCS_COP.1/Hash").
  - 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.
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 Network	The Fidelis Network components include several types of sensors, Fidelis Collectors, the Fidelis Sandbox, and K2. The sensors can be

	deployed to specific areas of the network as needed.
Fingerprint	The description of a specific kind of data based on particular characteristics. Fingerprints define either the 'content' within a transmission, the communication 'channel' of the transmission, or the sender or receiver of the transmission (e.g., 'location').
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
K2	Unique name for the Fidelis management console appliance of the TOE
Malware Detection Engine	The Malware Detection Engine (MDE) is included with K2 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 K2.
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 policies are composed of one or more rules, which in turn, contain one or more fingerprint definitions.
Postfix	An open source mail server alternative to the Sendmail program.
Rule	A rule is a logical combination of fingerprints that together are used by the event manager to generate alerts based on matches on combinations of fingerprints.
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
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
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 version 9.0.3 appliances. More specifically, the TOE consists of

- one or more Fidelis Network v9.0.3 K2 management consoles
- one or more Fidelis Network Collectors v9.0.3
- zero or more Fidelis Sandbox appliances,
- at least one of the following sensor appliances:
    - Fidelis Network Direct v9.0.3
    - Fidelis Network Internal v9.0.3
    - Fidelis Network Web v9.0.3
    - Fidelis Network Mail v9.0.3

The TOE also includes Fidelis Network Scout+ v9.0.3 which includes a K2; a Direct Sensor; and a Collector in one box.

2.1 Product Overview

This sub-section provides an overview of the capabilities of the Fidelis Network 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 monitors network traffic for malicious content coming into the network (intrusion) and for sensitive and secure data leaving the network (extrusion). 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 Network K2 is the management system for the Fidelis solution. The web-based, K2 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, and the K2 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.

2.2 TOE Overview

The TOE consists of:

- Fidelis Network v9.0.3 K2 management console component
- Fidelis Network v9.0.3 Collector component
- Fidelis Network v9.0.3 Sensor component
- Fidelis Network v9.0.3 Sandbox component.

A Fidelis Network system can be deployed entirely as hardware appliances, VM appliances, or a mixture, so long as there is a K2 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 K2 - Fidelis K2 appliance or Fidelis K2 VM
- Fidelis Sandbox – Fidelis Sandbox
- Fidelis Collector – Fidelis Collector SA2, Fidelis Collector XA2, Fidelis Collector XA4, and Fidelis Collector SA VM, Fidelis Collector Controller 2, or Fidelis Collector Controller 10G
- Fidelis Sensor

- 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

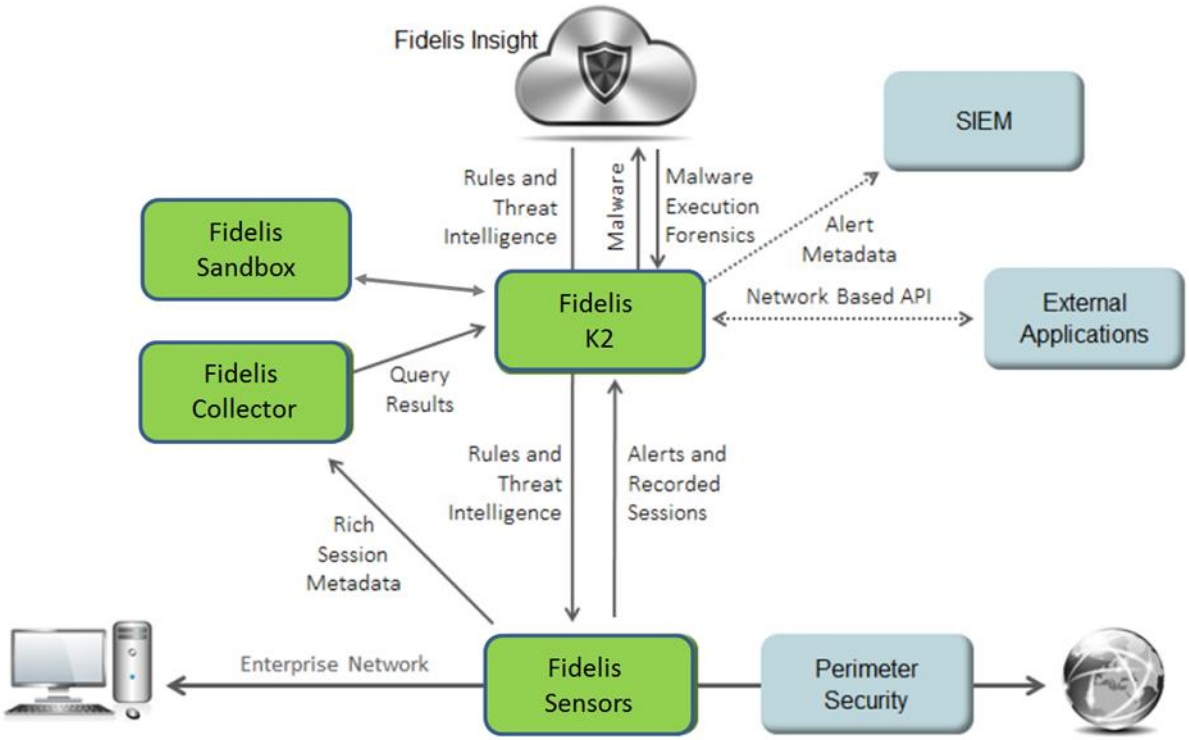

Figure 1 – Sample Fidelis Network Deployment Scenario

Initial configuration for each of the appliances is performed by directly 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 are required to be installed on the Collector, Sensor and Sandbox components before proceeding with registration to the K2.

After initial configuration and connecting each component to the network, the administrator adds all the components (Sensors, Collectors, Sandboxes) to the K2 to register them. The component name, IP address and description are entered into the K2. The component IP address must match the address established in the initial configuration and setup. After registration, the K2 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 K2 component via a web browser where the Open Secure Sockets Layer (OpenSSL) is used to implement Transport Layer Security (TLS) to secure the underlying communications. Similarly, K2 uses TLS to interact with the other components (Collectors, Sensors, Sandbox) in the deployment for the purposes of managing the components and receiving information from the components. Finally, the K2 uses TLS for communications with the following authorized IT entities: syslog server; LDAP server; Fidelis Insight

Server. The TOE implements NIST-validated cryptographic algorithms. Authorized administrators can also interact with each TOE component via a CLI using a directly connected console. However, once the TOE components have been installed and configured, it is intended that the TOE be managed remotely via the K2 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. K2 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 K2 audit log tracks all user activity. The Collectors, Sensors, and Sandbox forward all audit information to the K2 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 K2 audit log.

The K2 can communicate with Fidelis Insight Server to download policy and TOE updates. The K2 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 K2 v9.0.3 management console appliance
- One Fidelis SA2 Collector v9.0.3 appliance
- One Fidelis Direct 1000 Sensor v9.0.3 appliance
- One Fidelis Sandbox v9.0.3 appliance

Test Configuration 2 (virtual machines)

- One Fidelis K2 v9.0.3 VM management console appliance
- One Fidelis Collector SA v9.0.3 VM appliance
- One Fidelis Direct Sensor v9.0.3 VM appliance

Software

- Fidelis Network ™ v9.0.3

The following software was used in the test configurations to verify the TOE functionality:

Browsers

- Microsoft Internet Explorer v11.0
- Firefox v57.0.2, v58.0.1
- Google Chrome v64.0

Adobe Flash Player v28.0.0.161

Virtual appliances (K2 VM, Direct VM, Collector VM) were tested in an environment consistent with the requirements described in Section 2.3.1.1 of this document, including an Intel E7-4890 v2 @ 2.80G CPUs, DDR3 memory and 7200 RPM 2.5" SATA HDD in the host hardware system. More generally, the virtual appliances are supported on host hardware that includes Intel Core or Xeon processors based on the Ivy Bridge or Haswell microarchitecture, which implement Intel Secure Key.

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
K2	Dual Intel Xeon v3 8-core 3.2 GHz Intel Xeon E5-2667v3	3 TB 6x HDD, RAID-5	4x 1GbE	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - Apache httpd 2.4.25 - Apache tomcat 8.5.11 - Syslog-ng 3.7.3 - Mysql 5.6.36 - OpenSSL 1.0.1e-fips
Scout+ AP	Dual Intel Xeon 12-core 2.7 GHz Intel Xeon E5-2697 v2	1 x 200GB SSD 5 x 600GB HDD (RAID-5) (2.4 TB) 1 x 600 GB HDD Hot Spare	4x 1GbE	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 VMWare ESXi / vSphere 6.5
Scout+ IR	Dual Intel Xeon 12-core 2.7 GHz Intel Xeon E5-2697 v2	2 x 120GB SSD (RAID-1) 5 x 900GB HDD (RAID-5) (3.6 TB) 1 x 900 GB HDD Hot Spare	4x 1GbE	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 VMWare ESXi / vSphere 6.5
Direct 10G	Quad Intel Xeon v3 18-core 2.1 GHz Intel® Xeon® E5-4669v3 (on HP DL360/DL560 Gen9 server)	500 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Direct 5000	Dual Intel Xeon v3	300 GB	4x 1GbE	CentOS 6.8 with Linux kernel 2.6.32-

Device	Main Processor	Storage	Network Ports	Operating System / Software
	14-core 2.6 GHz Intel® Xeon® E5-2697v3 (on HP DL360/DL560 Gen9 server)	2x HDD, RAID-1	2x 10GbE optical (inline capable)	642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Direct 2500	Dual Intel Xeon v3 14-core 2.6 GHz Intel® Xeon® E5-2697v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Direct 1000	Dual Intel Xeon v3 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Direct 500	Dual Intel Xeon v3 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Direct 250	Intel Xeon v4 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Direct 100	Intel Xeon v4 10-core 2.6 GHz Intel® Xeon®	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64

Device	Main Processor	Storage	Network Ports	Operating System / Software
	E5-2660v3 (on HP DL360/DL560 Gen9 server)			- OpenSSL 1.0.1e-fips
Direct 50	Intel Xeon v4 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Internal 10G	Quad Intel Xeon v3 18-core 2.1 GHz Intel® Xeon® E5-4669v3 (on HP DL360/DL560 Gen9 server)	500 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Internal 5000	Dual Intel Xeon v3 14-core 2.6 GHz Intel® Xeon® E5-2697v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Internal 2500	Dual Intel Xeon v3 14-core 2.6 GHz Intel® Xeon® E5-2697v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Internal 1000	Dual Intel Xeon v3 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips

Device	Main Processor	Storage	Network Ports	Operating System / Software
	Gen9 server)			
Mail 1000	Dual Intel Xeon v3 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Mail 500	Dual Intel Xeon v3 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Mail 250	Dual Intel Xeon v3 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Web	Dual Intel Xeon v3 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL360/DL560 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 1GbE (inline capable)	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Sandbox Appliance	Dual Intel Xeon v3 10-core 3.1 GHz Intel® Xeon® E5-2687Wv3 (on HP DL380/DL360 Gen9 server)	3 TB 6x HDD, RAID-10	4x 1GbE	CentOS 7.3.1611 with Linux kernel 3.10.0-514.el7.x86_64 - OpenSSL 1.0.1e-fips
Collector SA2	Dual Intel Xeon v3 10-core 3.1 GHz	3 TB 6x HDD, RAID-	4x 1GbE	CentOS 6.8 with Linux kernel 2.6.32-

Device	Main Processor	Storage	Network Ports	Operating System / Software
	Intel® Xeon® E5-2687Wv3 (on HP DL380/DL360 Gen9 server)	10		642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Collector Controller 2	Dual Intel Xeon v3 10-core 2.6 GHz Intel® Xeon® E5-2660v3 (on HP DL380/DL360 Gen9 server)	300 GB 2x HDD, RAID-1	6x 1GbE	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Collector XA2	Dual Intel Xeon v3 8-core 3.2 GHz Intel® Xeon® E5-2667v3 (on HP DL380/DL360 Gen9 server)	300 GB 2x HDD, RAID1 ---------- 3.6 TB 6x HDD, RAID-10	4x 1GbE	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64
Collector Controller 10G	Dual Intel Xeon v3 14-core 2.6 GHz Intel® Xeon® E5-2697v3 (on HP DL380/DL360 Gen9 server)	300 GB 2x HDD, RAID-1	4x 1GbE 2x 10GbE optical	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Collector XA4	Dual Intel Xeon v3 10-core 3.1 GHz Intel® Xeon® E5-2687Wv3 (on HP DL380/DL360 Gen9 server)	300 GB 2x HDD, RAID1 ---------- 19.8 TB 22x HDD, RAID-10	4x 1GbE 2x 10GbE optical	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips

Table 1 TOE Hardware Appliances

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	Number of vCPUs	Memory	Disk	Operating System / Software
K2 VM	8	24 GB	100 GB	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - Apache httpd 2.4.25 - Apache tomcat 8.5.11 - Syslog-ng 3.7.3 - Mysql 5.6.36 - OpenSSL 1.0.1e-fips
Direct VM	14	24 GB	40 GB	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Internal VM	14	24 GB	40 GB	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Web VM	4	8 GB	40 GB	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Mail 250 VM	4	8 GB	100 GB	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Mail 500 VM	6	12 GB	120 GB	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Mail 1000 VM	8	14 GB	150 GB	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips
Collector SA VM	8	32 GB	200 GB	CentOS 6.8 with Linux kernel 2.6.32-642.13.1.el6.x86_64 - OpenSSL 1.0.1e-fips

Table 2 TOE Virtual Machine Appliances

The Fidelis Virtual Appliances have been tested and benchmarked by Fidelis on Dell Power Edge R920 servers installed with Intel E7-4890 v2 @ 2.80G CPUs, DDR3 memory and 7200 RPM 2.5" SATA HDD.

The following sub-sections identify the specific operating environment components required for the operation of the TOE.

2.3.1.1 Operational Environment Components

Administrators require a client computer with a web browser and Adobe Flash Player to remotely access the K2 GUI.

The virtual appliances are delivered as an installation disk (or ISO image). The virtual systems were tested by the evaluation team with VMWare ESXi / vSphere 6.5 installed.

Each virtual model in its evaluated configuration is installed on a hardware platform that includes VMware ESXi with vSphere 5.1, 5.5, 6.0 or 6.5 and an Intel Core or Xeon processor based on the Ivy Bridge or Haswell microarchitecture, which implement Intel Secure Key. 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 (including Scout) 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 (including Scout) 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_V2.0E]. The TOE stores the logs locally on the K2 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 implements NIST-validated cryptographic algorithms that provide 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 is deployed as a distributed TOE configuration. Initial configuration for each of the appliances is performed by directly attaching a keyboard and 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, and Sandboxes) to K2 to register them. After registration, K2 attempts to communicate to the newly registered component (the Sensor, the Collector, or the Sandbox or the Sensor) 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 K2 web-based GUI accessed via HTTPS or locally through the Command Line Interface using a directly connected console. The TOE supports the local (i.e., on device) definition of administrators with usernames and passwords. Additionally, the TOE can be configured 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. All administrative functionality available from the GUI is also available by directly attaching a keyboard and monitor to the appliance. However, the TOE is designed to be managed using the K2 GUI from a remote HTTPS/TLS client. Following the initial configuration, all changes should be performed by an authorized user from K2. The TOE provides the System Administrator role which corresponds to the [CPP_ND_V2.0E] 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.

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 HTTP over TLS. 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 Common Criteria Configuration Guide, Version 9.0.3, Revised August 2018
- Fidelis Network Enterprise Setup and Configuration Guide, Version 9.0.3, Revised 2017
- Fidelis Network User Guide, Version 9.0.3, Revised 2017
- Fidelis Network Guide to Creating Policies, Version 9.0.3, Revised 2017

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_V2.0E].

In general, the [CPP_ND_V2.0E] 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_V2.0E]. The [CPP_ND_V2.0E] 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_V2.0E] 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.
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.
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.

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 2.0 + Errata 20180314, 14-March-2018,, [CPP_ND_V2.0E] and including the following optional SFRs: FAU_STG.1, FCS_HTTPS_EXT.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.2, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1, FIA_X509_EXT.1/ITT, FPT_ITT.1/Join, and FCO_CPC_EXT.1.

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_V2.0E] 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_V2.0E] is not reproduced in ST.

The SARs are the set of SARs specified in [CPP_ND_V2.0E].

5.1 Extended Requirements

All extended requirements in this ST have been drawn from the [CPP_ND_V2.0E]. The [CPP_ND_V2.0E] defines the following extended SFRs and since they are not redefined in this ST, the [CPP_ND_V2.0E] should be consulted for more information regarding those CC extensions.

- FAU_STG_EXT.1: External Audit Event Storage
- 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
- FCS_TLSS_EXT.1: TLS Server Protocol
- FCS_TLSC_EXT.2: TLS Client Protocol with mutual authentication
- FCS_TLSS_EXT.2: TLS Server Protocol with 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: X.509 Certificate Validation
- 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: Extended: 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.2: User Identity Association
	FAU_STG.1: Protected Audit Trail Storage
	FAU_STG_EXT.1: Protected Audit Event Storage
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_RBG_EXT.1: Random Bit Generation
	FCS_TLSC_EXT.1: TLS Client Protocol
	FCS_TLSC_EXT.2: TLS Client Protocol with Mutual Authentication
	FCS_TLSS_EXT.1: TLS Server Protocol
	FCS_TLSS_EXT.2: TLS Server Protocol with 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
	FIA_X509_EXT.1/Rev: X.509 Certificate Validation
	FIA_X509_EXT.1/ITT: 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/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

Requirement Class	Requirement Component
	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 Symmetric 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.2	None.	None.
FAU_STG.1	None.	None.
FAU_STG_EXT.1	None.	None.
FCO_CPC_EXT.1	Enabling communications between a pair of components. Disabling communications between a	Identities of the endpoints pairs enabled or disabled.

Requirement	Auditable Events	Additional Audit Record Contents
	pair of components.	
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_RBG_EXT.1	None.	None.
FCS_TLSC_EXT.1	Failure to establish a TLS Session	Reason for failure
FCS_TLSC_EXT.2	Failure to establish a TLS Session	Reason for failure
FCS_TLSS_EXT.1	Failure to establish a TLS Session	Reason for failure
FCS_TLSS_EXT.2	Failure to establish a TLS Session	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
FIA_X509_EXT.1 /Rev	Unsuccessful attempt to validate a certificate	Reason for failure
FIA_X509_EXT.1/ITT	Unsuccessful attempt to validate a certificate	Reason for failure
FIA_X509_EXT.2	None.	None.
FIA_X509_EXT.3	None.	None.
FMT_MOF.1(1)/ManualUpdate	Any attempt to initiate a manual update	None.
FMT_MTD.1/CoreData	All management activities of TSF data.	None.
FMT_MTD.1/CryptoKeys	Management of cryptographic keys.	None.
FMT_SMF.1	None.	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.

Requirement	Auditable Events	Additional Audit Record Contents
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).
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 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.

5.2.1.3 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.

5.2.1.4 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.

FAU_STG_EXT.1.3 The TSF shall [***overwrite previous audit records according to the following rule: [audit records exceed the administrator configured retention time frame, disk space reaches 80% capacity]***] when the local storage space for audit data is full.

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-256, SHA-384*]** and message digest sizes [***256, 384***] 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 require client authentication***] if the peer certificate is deemed invalid.

5.2.3.9 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] hardware-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.

5.2.3.10 TLS Client Protocol (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*****].**

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifier matches the reference identifier per RFC 6125 Section 6.

FCS_TLSC_EXT.1.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [***not establish the connection***].

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

5.2.3.11 TLS Client Protocol with Mutual Authentication (FCS_TLSC_EXT.2)

FCS_TLSC_EXT.2.1 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*****].**

FCS_TLSC_EXT.2.2 The TSF shall verify that the presented identifier matches the reference identifier per RFC 6125 Section 6.

FCS_TLSC_EXT.2.3 The TSF shall only establish a trusted channel if the server certificate is valid. If the server certificate is deemed invalid, then the TSF shall [***not establish the connection***].

FCS_TLSC_EXT.2.4 The TSF shall [***present the Supported Elliptic Curves Extension with the following NIST curves: [secp256r1, secp384r1, secp521r1] and no other curves***] in the Client Hello.

FCS_TLSC_EXT.2.5 The TSF shall support mutual authentication using X.509v3 certificates.

5.2.3.12 TLS Server Protocol (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*****].**

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 [***generate EC Diffie-Hellman parameters over NIST curves [secp256r1] and no other curves; generate Diffie-Hellman parameters of size*** [***3072 bits***]].

5.2.3.13 TLS Server Protocol with Mutual Authentication (FCS_TLSS_EXT.2)

FCS_TLSS_EXT.2.1 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*****]**.

FCS_TLSS_EXT.2.2 The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0, and [***TLS 1.1***].

FCS_TLSS_EXT.2.3 The TSF shall [***generate EC Diffie-Hellman parameters over NIST curves*** **[*secp256r1*] *and no other curves; generate Diffie-Hellman parameters of size*** [***3072 bits***]].

FCS_TLSS_EXT.2.4 The TSF shall support mutual authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.5 The TSF shall not establish a trusted channel if the client certificate is invalid. If the client certificate is deemed invalid, then the TSF shall [***not establish the connection***].

FCS_TLSS_EXT.2.6 The TSF shall not establish a trusted channel if the distinguished name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match the expected identifier for the peer.

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 to 999**] unsuccessful authentication attempts occur related to Administrators attempting to authenticate remotely.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met, the TSF shall [***prevent the offending remote Administrator from successfully authenticating until*** **[*password reset*] *is taken by a local 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 [**1**] and [**999**].

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, and [***[access to an external LDAP Server]***] 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)

FIA_X509_EXT.1.1/ITT 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 two certificates.
- The certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- 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:
  - 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 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.

5.2.4.7 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 certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- 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.

5.2.4.8 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 [***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***].

5.2.4.9 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/ManualUpdate)

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

5.2.5.2 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.

5.2.5.3 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.

5.2.5.4 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 the cryptographic functionality;***
- ***Ability to re-enable an Administrator account;***
- ***Ability to set the time which is used for time-stamps;***
- ***Ability to configure the reference identifier for the peer;***

].

5.2.5.5 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 passwords in non-plaintext form.

FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext 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***].

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***].

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 external time sources***].

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**

].

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 FTP_ITC.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_V2.0E].

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_V2.0E] 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 is a distributed TOE configuration. The following table identifies which TOE components satisfy the [CPP_ND_V2.0E] 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.2	None.	None.	All	All
FAU_STG.1	None.	None.	K2	None
FAU_STG_EXT.1	None.	None.	All	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

Requirement	Auditable Events	Additional Audit Record Contents	Component Implementing the SFR	Component Generating the Audit Record
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session.	Reason for failure	K2	K2
FCS_RBG_EXT.1	None.	None.	All	None
FCS_TLSC_EXT.1	Failure to establish a TLS Session	Reason for failure	K2	K2
FCS_TLSC_EXT.2	Failure to establish a TLS Session	Reason for failure	All	All
FCS_TLSS_EXT.1	Failure to establish a TLS Session	Reason for failure	K2	K2
FCS_TLSS_EXT.2	Failure to establish a TLS Session	Reason for failure	All	All
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	All
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).	K2	K2
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).	All	All
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).	All	All
FIA_UAU.7	None.	None.	All	None
FIA_X509_EXT.1/Rev	Unsuccessful attempt to validate a certificate	Reason for failure	K2	K2
FIA_X509_EXT.1/ITT	Unsuccessful attempt to validate a certificate	Reason for failure	All	All
FIA_X509_EXT.2	None.	None.	All	None
FIA_X509_EXT.3	None.	None.	All	None

Requirement	Auditable Events	Additional Audit Record Contents	Component Implementing the SFR	Component Generating the Audit Record
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.	K2	K2
FMT_MTD.1/CoreData	All management activities of TSF data.	None.	All	All
FMT_MTD.1/CryptoKeys	Management of Cryptographic keys.	None.	All	All
FMT_SMF.1	None.	None.	All	None
FMT_SMR.2	None.	None.	All	None
FPT_SKP_EXT.1	None.	None.	All	None
FPT_APW_EXT.1	None.	None.	All	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.	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.	K2	K2
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

Requirement	Auditable Events	Additional Audit Record Contents	Component Implementing the SFR	Component Generating the Audit Record
FTA_SSL_EXT.1 (if "terminate the session" is selected)	The termination of a local session by the session locking mechanism.	None.	All	All
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.	K2	K2
FTA_SSL.4	The termination of an interactive session.	None.	All	All
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.	K2	K2
FTP_TRP.1/Admin	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	None.	K2	K2

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 K2, 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: 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 K2 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 K2 administrator actions are stored locally on the K2 appliance. The TOE components (Sensor, Collector, or Sandbox) are registered with the K2 and forward log records generated on those appliances to the K2 over a secure TLS connection.

All of the TOE audit events are accessed via the K2 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 K2 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, K2 GUI Page Access". For example:
  - logging in is "Access";
  - changing a configuration parameter is "Modification";
  - accessing any K2 GUI page in a browser is "K2 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, and Sandbox components do not store any audit records, but rather forward all audit events securely over a TLS connection to the K2. The K2 locally stores the audit records and forwards the audit record in real time to an externa audit server.

Table 4 corresponds to the audit events specified in Table 1 of the [CPP_ND_V2.0E] and includes the audit events specified in the [CPP_ND_V2.0E] 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

The TOE includes an internal log implementation that can be used to store and review audit records locally on the K2. The local audit logs are stored on the K2 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 K2 are protected by database access control and there are no interfaces to delete individual audit records.

6.1.4 FAU_STG_EXT.1: Protected Audit Event Storage

The TOE Sensor, Collector, and Sandbox 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 TLS connection to the K2 for storage. If the communication link to the K2 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 dropped. Once communication is re-established, the audit records will be transmitted to the K2 for central local storage and the buffer cleared.

The TOE can 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 also written to the external syslog as they are written locally to the K2 audit log.

6.2 Cryptographic support

The TOE includes the FIPS-capable OpenSSL 1.0.1e-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 #1590 Sandbox Appliance RSA #1878
• FFC key pair (3072 bits)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.1	DSA #903 Sandbox Appliance DSA #1038
• ECDSA (P-256, P-384, P-521 curves)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4;	ECDSA #564 Sandbox Appliance ECDSA #757
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	CVL #380 Sandbox Appliance CVL #657

Functions	Standards	Certificates
• 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";	CVL #380 Sandbox Appliance CVL #657
Encryption/Decryption (FCS_COP.1/Data Encryption)		
• AES CBC (128 and 256 bits)	ISO 18033-3, CBC as specified in ISO 10116	AES #3119 Sandbox Appliance AES #3641
• AES GCM (128 and 256 bits)	ISO 18033-3, GCM as specified in ISO 19772	AES #3119 Sandbox Appliance AES #3641
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 #1590 Sandbox Appliance RSA #1878
Cryptographic hashing (FCS_COP.1/Hash)		
• SHA-256 (digest size 256 bits) • SHA-384 (digest size 384 bits)	ISO/IEC 10118-3:2004	SHS #2577 Sandbox Appliance SHS #3061
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 #1958 Sandbox Appliance HMAC #2394
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 #631 Sandbox Appliance DRBG #970

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 K2 TLS management interface, and with components of distributed TOE. The TOE acts as both a sender and receipt. It acts as a client for an external audit server, authentication server, and Fidelis Insight Server and as a server for it K2 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

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 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 cryptomodule. 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 K2, 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 SHA-256 hash in a MySQL database located on the K2.

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 2048 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. User passwords for users with local authentication are stored as SHA-256. The SHA hash algorithm is used as part of HMAC, but is also used as part of RSA digital signature creation and 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

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 K2 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.

6.2.9 FCS_RBG_EXT.1: Random Bit Generation

The TOE uses a software-based 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 Ivy Bridge and newer processors and assumed to generate at least 0.5 bits of entropy per sample.

6.2.10 FCS_TLSC_EXT.1: TLS Client Protocol

The K2 acts as a TLS client for secure communications with a LDAP server, audit server, and the Fidelis Insight Server.

The TOE verifies that the presented identifier matches the reference identifier according to RFC 6125.

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 K2 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 K2. A hostname check in the certificates is performed on Subject Alternative Names and Subject Common Name. The TOE will only establish a trusted channel if the peer certificate is valid.

Certificate pinning is not supported.

6.2.11 FCS_TLSC_EXT.2: TLS Client Protocol with 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 K2 acting as a TLS client supports mutual authentication for secure communications with an external audit server and can optionally be configured to support mutual authentication for secure communications with an authentication 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, and the NTP server. Certificate files, CA-certificate files, and CRL files are then installed on each of the appliances before proceeding with registration to the K2. The FQDN or the IP address of the audit server is configured as the reference identifier on the K2. 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) to K2 and successfully registers them.

The TOE verifies that the presented identifier matches the reference identifier according to RFC 6125.

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.

6.2.12 FCS_TLSS_EXT.1: TLS Server Protocol

The TOE acts as a TLS Server when remote administrators connect to the TOE's GUI using HTTPS. 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.

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.

6.2.13 FCS_TLSS_EXT.2: TLS Server Protocol with 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.

The TOE secure communication is implemented using TLS 1.2 (RFC 5246) and supports the following 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 denies connections from clients requesting connections using SSL 2.0, SSL 3.0, TLS 1.0, and TLS 1.1.

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.

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

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 component host name 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 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 is deployed as a distributed TOE configuration. Users must install and set up X.509 certificates and enable FIPS 140-2 encryption for data storage on the K2. Fidelis components enforce FIPS 140-2 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 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 K2. 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) to K2 and successfully register them. The appliance name, IP address and description are entered into the K2. The appliance IP address must match the address provided in the initial configuration and setup. After registration, K2 attempts to communicate to the newly registered component (the Sensor, the Collector, or the Sandbox or the Sensor) at the specified IP address over a secure TLS tunnel as described in FPT_ITT.1/Join.

Communication between K2 and each component is verified by running the following command on K2 for every other component:

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

The System Administrator disables the communication between the K2 and the distributed components by remotely authenticating to the K2, navigating to the Components page, and deleting the desired 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 the local administrator resets the password. Authentication failures by remote Administrators cannot lead to a situation where no Administrator access is available to the TOE. If remote administrators are locked out, administrator access is still available via local console, and this preventing any condition where no administrator access is available.

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 K2 or a CLI connected to the distributed TOE components locally using a directly connected console. However the TOE is not intended to be managed locally. 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.

In order to log in, the user must provide an identity and also authentication data that matches the provided identity. Users can be defined locally within the TOE with a user identity, password, and user role. Alternately, users can be defined within an external LDAP (e.g. Active Directory) server configured to be used by the TOE that also defines 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 server and the result is enforced by the TOE. In either case, any resulting session is dependent upon successful authentication and established sessions are associated with the role(s) (see Section 6.4) 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.

6.4.6 FIA_X509_EXT.1/ITT: X.509 Certificate Validation

The TOE uses 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 TOE validates the revocation status of the certificate using a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3.

The TOE uses 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.

6.4.7 FIA_X509_EXT.2: X.509 Certificate Authentication

The TOE uses X.509v3 certificates as defined by RFC 5280 to support the TLS connection with all intra-TOE communication, remote administrative communication, external syslog server, authentication server, and Fidelis Insight 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. If a CRL is unavailable during the authentication process, the TOE will assume that the certificate is not revoked and operate as if the CRL was verified and the certificate is in good standing (from a revocation perspective only).

6.4.8 FIA_X509_EXT.3: X.509 Certificate Requests

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

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/ManualUpdate: Management of Security Functions Behaviour Requests

The initiation of manual TOE updates is restricted to System Administrators.

6.5.2 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.

6.5.3 FMT_MTD.1/CryptoKeys: Management of TSF Data

The ability to manage the cryptographic keys (e.g. modified, deleted or generated/imported) is restricted to System Administrators.

6.5.4 FMT_SMF.1: Specification of Management Functions

The TOE also provides the ability to manage the TOE locally. All administrative functionality available from the GUI is also available via a USB keyboard and a monitor to the appliance VGA connector using special debug account. However, the TOE in the distributed configuration is designed to be managed using the K2 GUI from a remote HTTPS/TLS client. Following the initial configuration, all changes should be performed by an authorized user from the K2 GUI.

The following management functions are performed locally on the K2, Sensor, Collector, and Sandbox components:

- Configure the access banner;
- Configure the cryptographic functionality;
- Set the time which is used for time-stamps;
- Configure the reference identifier for the peer.

The following management functions are performed by a remote administrator accessing the K2 GUI:

- Update the TOE, and to verify the updates using the hash comparison capability prior to installing those updates;
- Ability to re-enable an Administrator account;
- Ability to configure the authentication failure parameters for FIA_AFL.1;
- Ability to configure the session inactivity time before session termination or locking.

6.5.5 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_V2.0E]. 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, except configuration of the advisory banner, generating a CSR, and importing a certificate which is done by directly attaching a keyboard and monitor to the appliance.

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 provides its own internal clock which it uses to provide a reliable time source for audit records.

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 by saving a SHA-256 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.

6.6.2 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, and Sandboxes) to K2 to register them. The components IP addresses are manually entered and registered with the K2. The TOE does not provide an automated discovery process. The registration channel is protected by TLS as identified in FPT_ITT.1/Join. The registration channel is adopted as a continuing internal communication channel between different TOE components. The K2 will communicate to the newly registered components (the Sensor, the Collector, or the Sandbox or the Sensor) 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 SHA-256 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 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 and cryptographic known answer tests) 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 K2 Version Control page will list all components accessible from the K2. This includes the K2 Management Console (the K2 that you are currently logged into), and all registered sensors and Collectors. 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 K2 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 K2 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 K2.
8) K2 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, K2 ensures that it performs updates of TOE components in the right order (Sensors, Collectors, Sandboxes first, then K2 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 K2 version may be lower than that of other components.

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 K2 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 CLI session by entering the “exit” command.

6.7.3 FTA_SSL_EXT.1: TSF-initiated Session Locking

The TOE can be configured by an administrator 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.

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 authentication servers 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 authentication servers. The TOE will automatically re-establish communications in the event of a temporary network outage.

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), 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 ST conforms to the collaborative Protection Profile for Network Devices, Version 2.0 + Errata 20180314, 14-March-2018, [CPP_ND_V2.0E] and including the following optional SFRs: FAU_STG.1, FCS_HTTPS_EXT.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.2, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1, FIA_X509_EXT.1/ITT, FPT_ITT.1/Join, and FCO_CPC_EXT.1.

As explained in Section 3, Security Problem Definition, the Security Problem Definition of the [CPP_ND_V2.0E] has been included by reference into this ST.

As explained in Section 4, Security Objectives, the Security Objectives of the [CPP_ND_V2.0E] 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_V2.0E] and operations completed as appropriate.

Requirement Class	Requirement Component	Source
FAU: Security audit	FAU_GEN.1: Audit Data Generation	CPP_ND_V2.0E
	FAU_GEN.2: User Identity Association	CPP_ND_V2.0E
	FAU_STG.1: Protected Audit Trail Storage	CPP_ND_V2.0E
	FAU_STG_EXT.1: External Audit Trail Storage	CPP_ND_V2.0E
FCS: Cryptographic support	FCS_CKM.1: Cryptographic Key Generation (for asymmetric keys)	CPP_ND_V2.0E
	FCS_CKM.2: Cryptographic Key Establishment (Refined)	CPP_ND_V2.0E
	FCS_CKM.4: Cryptographic Key Destruction	CPP_ND_V2.0E
	FCS_COP.1(1) Cryptographic Operation (AES Data Encryption/Decryption)	CPP_ND_V2.0E
	FCS_COP.1(2) Cryptographic Operation (Signature Generation and Verification)	CPP_ND_V2.0E
	FCS_COP.1(3) Cryptographic Operation (Hash Algorithm)	CPP_ND_V2.0E
	FCS_COP.1(4) Cryptographic Operation (Keyed Hash Algorithm)	CPP_ND_V2.0E
	FCS_HTTPS_EXT.1: HTTPS Protocol	CPP_ND_V2.0E
	FCS_RBG_EXT.1: Random Bit Generation	CPP_ND_V2.0E
	FCS_TLSC_EXT.1: TLS Client Protocol	CPP_ND_V2.0E
	FCS_TLSS_EXT.1: TLS Server Protocol	CPP_ND_V2.0E
FIA: Identification and authentication	FIA_AFL.1: Authentication Failure Management	CPP_ND_V2.0E
	FIA_PMG_EXT.1: Password Management	CPP_ND_V2.0E
	FIA_UAU.7: Protected Authentication Feedback	CPP_ND_V2.0E
	FIA_UAU_EXT.2: Password-based Authentication Mechanism	CPP_ND_V2.0E
	FIA_UIA_EXT.1: User Identification and Authentication	CPP_ND_V2.0E
	FIA_X509_EXT.1/Rev: X.509 Certificate Validation	CPP_ND_V2.0E
	FIA_X509_EXT.1/ITT: X.509 Certificate Validation	CPP_ND_V2.0E
	FIA_X509_EXT.2: X.509 Certificate Authentication	CPP_ND_V2.0E
	FIA_X509_EXT.3: X.509 Certificate Requests	CPP_ND_V2.0E
FMT: Security management	FMT_MOF.1/ManualUpdate: Management of security functions behaviour	CPP_ND_V2.0E

Requirement Class	Requirement Component	Source
	FMT_MTD.1/CoreData: Management of TSF Data	CPP_ND_V2.0E
	FMT_MTD.1/CryptoKeys: Management of TSF Data	CPP_ND_V2.0E
	FMT_SMF.1:Specification of Management Functions	CPP_ND_V2.0E
	FMT_SMR.2: Restrictions on Security Roles	CPP_ND_V2.0E
FPT: Protection of the TSF	FPT_SKP_EXT.1: Extended: Protection of TSF Data (for reading of all symmetric keys)	CPP_ND_V2.0E
	FPT_APW_EXT.1: Protection of Administrator Passwords	CPP_ND_V2.0E
	FPT_ITT.1: Basic internal TSF data transfer protection	CPP_ND_V2.0E
	FPT_ITT.1/Join: Basic internal TSF data transfer protection	CPP_ND_V2.0E
	FPT_STM_EXT.1: Reliable Time Stamps	CPP_ND_V2.0E
	FPT_TST_EXT.1: TSF Testing	CPP_ND_V2.0E
	FPT_TUD_EXT.1: Trusted Update	CPP_ND_V2.0E
FTA: TOE access	FTA_SSL.3: TSF-initiated Termination	CPP_ND_V2.0E
	FTA_SSL.4: User-initiated Termination	CPP_ND_V2.0E
	FTA_SSL_EXT.1: TSF-initiated Session Locking	CPP_ND_V2.0E
	FTA_TAB.1: Default TOE Access Banners	CPP_ND_V2.0E
FTP: Trusted path/channels	FTP_ITC.1: Inter-TSF trusted channel (Refined)	CPP_ND_V2.0E
	FTP_TRP.1: Trusted Path	CPP_ND_V2.0E

Table 9 SFR Protection Profile Sources

8. Rationale

This security target includes by reference the [CPP_ND_V2.0E] Security Problem Definition, Security Objectives, and Security Assurance Requirements. The security target makes no additions to the [CPP_ND_V2.0E] assumptions. [CPP_ND_V2.0E] security functional requirements have been reproduced with the Protection Profile operations completed. Operations on the security requirements follow [CPP_ND_V2.0E] application notes and assurance activities. Consequently, [CPP_ND_V2.0E] 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 10 Security Functions vs. Requirements Mapping** 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.2	X							
FAU_STG.1	X							
FAU_STG_EXT.1	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_RBG_EXT.1		X						
FCS_TLSC_EXT.1		X						
FCS_TLSC_EXT.2		X						
FCS_TLSS_EXT.1		X						
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				X				
FIA_X509_EXT.2				X				
FIA_X509_EXT.3				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		

	Security audit	Cryptographic support	Communication	Identification and authentication	Security management	Protection of the TSF	TOE access	Trusted path/channels
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 10 Security Functions vs. Requirements Mapping