ID Technologies

GoSilent Cube + GoSilent Server v25.01

Assurance Activity Report

Version 0.5

December 2022

Document prepared by

www.lightshipsec.com

Table of Contents

1 INTRODUCTION 3
1.1 Evaluation Identifiers 3
1.2 Evaluation Methods 3
1.3 Summary of SFRs 6
1.4 Reference Documents 8

2 TOE DETAILS 10
2.1 Overview 10
2.2 TOE Models 10

3 EVALUATION ACTIVITIES FOR THE COLLABORATIVE PROTECTION PROFILE FOR NETWORK DEVICES 13
3.1 Security Audit (FAU) 13
3.2 Cryptographic Support (FCS) 19
3.3 Identification and Authentication (FIA) 36
3.4 Security management (FMT) 42
3.5 Protection of the TSF (FPT) 47
3.6 TOE Access (FTA) 57
3.7 Trusted path/channels (FTP) 60

4 EVALUATION ACTIVITIES FOR VPN GATEWAY PP-MODULE 64
4.1 Security Audit (FAU) 64
4.2 Cryptographic Support (FCS) 65
4.3 Security management (FMT) 67
4.4 Packet Filtering (FPF) 68
4.5 Protection of the TSF (FPT) 83
4.6 Trusted Path/Channels (FTP) 84

5 EVALUATION ACTIVITIES FOR STATEFUL TRAFFIC FILTER FIREWALLS PP-MODULE 85
5.1 Security Audit (FAU) 85
5.2 User Data Protection (FDP) 85
5.3 Firewall (FFW) 86
5.4 Security Management (FMT) 103

6 EVALUATION ACTIVITIES FOR OPTIONAL REQUIREMENTS 105
6.1 Identification and Authentication (FIA) 105
6.2 Protection of the TSF (FPT) 109
6.3 Communication (FCO) 111

7 EVALUATION ACTIVITIES FOR SELECTION-BASED REQUIREMENTS 117
7.1 Security Audit (FAU) 117
7.2 Cryptographic Support (FCS) 119
7.3 Identification and Authentication (FIA) 150
7.4 Security management (FMT) 158

8 EVALUATION ACTIVITIES FOR SECURITY ASSURANCE REQUIREMENTS 161
8.1 ASE: Security Target 161
8.2 ADV: Development 161
8.3 AGD: Guidance Documents 163
8.4 ALC: Life-cycle Support 166
8.5 ATE: Tests 166
8.6 Vulnerability Assessment 167
8.7 Evaluating additional components for a distributed TOE 169

1 Introduction

1 This Assurance Activity Report (AAR) documents the evaluation activities performed by Lightship Security for the evaluation identified in Table 1. The AAR is produced in accordance with National Information Assurance Program (NIAP) reporting guidelines.

1.1 Evaluation Identifiers

Table 1: Evaluation Identifiers

Scheme	NIAP
Evaluation Facility	Lightship Security USA 3600 O'Donnell St., Suite 2 Baltimore, MD 21224
Developer/Sponsor	ID Technologies 19980 Highland Vista Drive Suite 175 Ashburn, Virginia 20147 United States
TOE	GoSilent Cube + GoSilent Server v25.01 Build: 25.01.4 (GoSilent Server) Build: 25.01.3 (GoSilent Cube)
Security Target	ID Technologies GoSilent Cube + GoSilent Server v25.01 Security Target, v1.18, December 2022
Protection Profile	collaborative Protection Profile for Network Devices, v2.2e, 23-March-2020 (CPP_ND_V2.2E)
	PP-Module for Stateful Traffic Filter Firewalls, v1.4 + Errata 20200625, 25-June-2020 (MOD_CPP_FW_v1.4e)
	PP-Module for VPN Gateways, v1.2, 2022-03-31 (MOD_VPNGW_v1.2)

1.2 Evaluation Methods

2 The evaluation was performed using the methods and standards identified in Table 2.

Table 2: Evaluation Methods

Evaluation Criteria	CC v3.1R5
Evaluation Methodology	CEM v3.1R5

Supporting Documents	Evaluation Activities for Network Device cPP, December-2019, v2.2 (NDcPP-SD), Evaluation Activities for Stateful Traffic Filter Firewalls PP-Module, June-2020, v1.4 + Errata 20200625 (MOD_FW_v1.4e-SD) Supporting Document Mandatory Technical Document PP-Module for VPN Gateways, v1.2, March 31, 2022 (MOD_VPNGW_v1.2-SD)
Interpretations	**CPP_ND_v2.2E, MOD_CPP_FW_v1.4e, MOD_VPNGW_v1.2** TD0527: Updates to Certificate Revocation Testing (FIA_X509_EXT.1) TD0528: NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4 N/A—FCS_NTP_EXT.1 not claimed. TD0536: NIT Technical Decision for Update Verification Inconsistency TD0537: NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3 TD0538: NIT Technical Decision for Outdated link to allowed-with list TD0545: NIT Technical Decision for Conflicting FW rules cannot be configured (extension of Rfl#201837) TD0546: NIT Technical Decision for DTLS – clarification of Application Note 63 N/A—FCS_DTLSC_EXT.1 not claimed. TD0547: NIT Technical Decision for Clarification on developer disclosure of AVA_VAN TD0551: NIT Technical Decision for Incomplete Mappings of OEs in FW Module v1.4+Errata TD0555: NIT Technical Decision for RFC Reference incorrect in TLSS Test TD0556: NIT Technical Decision for RFC 5077 question TD0563: NiT Technical Decision for Clarification of audit date information TD0564: NiT Technical Decision for Vulnerability Analysis Search Criteria TD0569: NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7 TD0570: NiT Technical Decision for Clarification about FIA_AFL.1

TD0571: NiT Technical Decision for Guidance on how to handle FIA_AFL.1
TD0572: NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers
TD0580: NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e
TD0581: NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3
TD0591: NIT Technical Decision for Virtual TOEs and hypervisors
TD0592: NIT Technical Decision for Local Storage of Audit Records
TD0631: NIT Technical Decision for Clarification of public key authentication for SSH Server N/A—FCS_SSHS_EXT.1 not claimed.
TD0632: NIT Technical Decision for Consistency with Time Data for vNDs
TD0633: NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance
TD0634: NIT Technical Decision for Clarification required for testing IPv6
TD0635: NIT Technical Decision for TLS Server and Key Agreement Parameters
TD0636: NIT Technical Decision for Clarification of Public Key User Authentication for SSH N/A—FCS_SSHC_EXT.1 not claimed.
TD0638: NIT Technical Decision for Key Pair Generation for Authentication
TD0639: NIT Technical Decision for Clarification for NTP MAC Keys N/A—FCS_NTP_EXT.1 not claimed.
TD0656: Missing EAs for VPN GW Optional Headend SFRs N/A—FTA_SSL.3/VPN, FTA_TSE.1, FTA_VCM_EXT.1 not claimed.
TD0657: IPSEC_EXT.1.6 GCM support for VPN GW
TD0670: NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing

	TD683: RFC 2460 to be replaced with RFC 8200

1.3 Summary of SFRs

Table 3: Summary of SFRs

Requirement	Title	Source
FAU_GEN.1	Audit Data Generation	CPP_ND_V2.2E MOD_CPP_FW_v1.4e
FAU_GEN.1/VPN	Audit Data Generation	MOD_VPNGW_v1.2
FAU_GEN.2	User Identity Association	CPP_ND_V2.2E
FAU_GEN_EXT.1	Security Audit Generation	CPP_ND_V2.2E
FAU_STG_EXT.1	Protected Audit Event Storage	CPP_ND_V2.2E
FAU_STG_EXT.4	Protected Local Audit Event Storage for Distributed TOEs	CPP_ND_V2.2E
FCO_CPC_EXT.1	Component Registration Channel Definition	CPP_ND_V2.2E
FCS_CKM.1	Cryptographic Key Generation	CPP_ND_V2.2E
FCS_CKM.1/IKE	Cryptographic Key Generation (for IKE Peer Authentication)	MOD_VPNGW_v1.2
FCS_CKM.2	Cryptographic Key Establishment	CPP_ND_V2.2E
FCS_CKM.4	Cryptographic Key Destruction	CPP_ND_V2.2E
FCS_COP.1/DataEncryption	Cryptographic Operation (AES Data Encryption/Decryption)	CPP_ND_V2.2E MOD_VPNGW_v1.2
FCS_COP.1/SigGen	Cryptographic Operation (Signature Generation and Verification)	CPP_ND_V2.2E
FCS_COP.1/Hash	Cryptographic Operation (Hash Algorithm)	CPP_ND_V2.2E
FCS_COP.1/KeyedHash	Cryptographic Operation (Keyed Hash Algorithm)	CPP_ND_V2.2E
FCS_RBG_EXT.1	Random Bit Generation	CPP_ND_V2.2E
FCS_HTTPS_EXT.1	HTTPS Protocol	CPP_ND_V2.2E
FCS_IPSEC_EXT.1	IPsec Protocol	CPP_ND_V2.2E MOD_VPNGW_v1.2
FCS_TLSC_EXT.1	TLS Client Protocol	CPP_ND_V2.2E

Requirement	Title	Source
FCS_TLSS_EXT.1	TLS Server Protocol	CPP_ND_V2.2E
FDP_RIP.2	Full Residual Information Protection	MOD_CPP_FW_v1.4e
FFW_RUL_EXT.1	Stateful Traffic Filtering	MOD_CPP_FW_v1.4e
FIA_AFL.1	Authentication Failure Management	CPP_ND_V2.2E
FIA_PMG_EXT.1	Password Management	CPP_ND_V2.2E
FIA_UIA_EXT.1	User Identification and Authentication	CPP_ND_V2.2E
FIA_UAU_EXT.2	Password-based Authentication Mechanism	CPP_ND_V2.2E
FIA_UAU.7	Protected Authentication Feedback	CPP_ND_V2.2E
FIA_X509_EXT.1/ITT	X.509 Certificate Validation	CPP_ND_V2.2E
FIA_X509_EXT.1/Rev	X.509 Certificate Validation	CPP_ND_V2.2E MOD_VPNGW_v1.2
FIA_X509_EXT.2	X.509 Certificate Authentication	CPP_ND_V2.2E MOD_VPNGW_v1.2
FIA_X509_EXT.3	X.509 Certificate Requests	CPP_ND_V2.2E MOD_VPNGW_v1.2
FMT_MOF.1/ManualUpdate	Management of Security Functions Behaviour	CPP_ND_V2.2E
FMT_MOF.1/Services	Management of Security Functions Behaviour	CPP_ND_V2.2E
FMT_MTD.1/CoreData	Management of TSF Data	CPP_ND_V2.2E
FMT_MTD.1/CryptoKeys	Management of TSF Data	CPP_ND_V2.2E MOD_VPNGW_v1.2
FMT_SMF.1	Specification of Management Functions	CPP_ND_V2.2E
FMT_SMF.1/FFW	Specification of Management Functions	MOD_CPP_FW_v1.4e
FMT_SMF.1/VPN	Specification of Management Functions (VPN Gateway)	MOD_VPNGW_v1.2
FMT_SMR.2	Restrictions on Security Roles	CPP_ND_V2.2E
FPF_RUL_EXT.1	Rules for Packet Filtering	MOD_VPNGW_v1.2

Requirement	Title	Source
FPT_FLS.1/SelfTest	Failure with Preservation of Secure State (Self-Test Failures)	MOD_VPNGW_v1.2
FPT_ITT.1	Basic Internal TSF Data Transfer Protection	CPP_ND_V2.2E
FPT_SKP_EXT.1	Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)	CPP_ND_V2.2E
FPT_APW_EXT.1	Protection of Administrator Passwords	CPP_ND_V2.2E
FPT_TST_EXT.1	TSF Testing	CPP_ND_V2.2E MOD_VPNGW_v1.2
FPT_TST_EXT.3	TSF Self-Test with Defined Methods	MOD_VPNGW_v1.2
FPT_TUD_EXT.1	Trusted Update	CPP_ND_V2.2E MOD_VPNGW_v1.2
FPT_STM_EXT.1	Reliable Time Stamps	CPP_ND_V2.2E
FTA_SSL_EXT.1	TSF-initiated Session Locking	CPP_ND_V2.2E
FTA_SSL.3	TSF-initiated Termination	CPP_ND_V2.2E
FTA_SSL.4	User-initiated Termination	CPP_ND_V2.2E
FTA_TAB.1	Default TOE Access Banners	CPP_ND_V2.2E
FTP_ITC.1	Inter-TSF trusted channel	CPP_ND_V2.2E
FTP_ITC.1/VPN	Inter-TSF Trusted Channel (VPN Communications)	MOD_VPNGW_v1.2
FTP_TRP.1/Admin	Trusted Path	CPP_ND_V2.2E

1.4 Reference Documents

Table 4: List of Reference Documents

Ref	Document
[ST]	ID Technologies GoSilent Cube + GoSilent Server v25.01 Security Target, v1.18, December 2022
[AGD]	ID Technologies GoSilent Cube + GoSilent Server v25.01 Common Criteria Guide, v1.8, December 2022

Ref	Document
[DTR]	ID Technologies GoSilent Cube + GoSilent Server v25.01 Detailed Test Report, Version 0.5, December 2022 ID Technologies GoSilent Cube + GoSilent Server v25.01 Detailed Test Report Evidence, Version 0.5, December 2022
[ETR]	ID Technologies GoSilent Cube + GoSilent Server v25.01 Evaluation Technical Report, v0.7, December 2022
[VULN]	ID Technologies GoSilent Cube + Go Silent Server v25.01 Vulnerability Assessment, Version 0.5, December 2022
[Base-PP]	collaborative Protection Profile for Network Devices, v2.2e, 23-March-2020 (CPP_ND_V2.2E)
[PP-TFFW]	PP-Module for Stateful Traffic Filter Firewalls, v1.4 + Errata 20200625, 25-June-2020 (MOD_CPP_FW_v1.4e)
[PP-VPN]	PP-Module for VPN Gateways, v1.2, 2022-03-31 (MOD_VPNGW_v1.2)
[SD-Base-PP]	Evaluation Activities for Network Device cPP, December-2019, v2.2 (NDcPP-SD)
[SD-TFFW]	Evaluation Activities for Stateful Traffic Filter Firewalls PP-Module, June-2020, v1.4 + Errata 20200625 (MOD_FW_v1.4e-SD)
[SD-VPN]	Supporting Document Mandatory Technical Document PP-Module for VPN Gateways, v1.2, March 31, 2022 (MOD_VPNGW_v1.2-SD)

2 TOE Details

2.1 Overview

3 The GoSilent Server acts as the centralized management and external access system for the TOE integrated into one system. The GoSilent Server provides the management GUI for configuration of GoSilent Server as well as the GoSilent Cube user devices. Operationally, it acts as the central peer for all IPsec connections from the GoSilent Cube devices and provides gateway connectivity to external systems located behind a GoSilent Cube.

4 The GoSilent Cube is a portable enterprise-grade firewall and VPN, ideal for sensitive communications, secure remote network access, and IoT deployments. GoSilent Cube can be setup within minutes by non-technical users. Physical Ethernet connections are supported on both the user side and VPN side of GoSilent Cube.

5 Together, GoSilent Cube and GoSilent Server provide a secure communications path to one or more systems located "behind" each GoSilent Cube. These systems connect to GoSilent Cube via physical Ethernet. Each GoSilent Cube establishes an IPsec VPN to the GoSilent Server, and all traffic from the user systems is routed over that VPN. Physical Ethernet is supported on the VPN side of GoSilent Cube.

6 The TOE applies policies to restrict the traffic that is permitted to pass between the user systems and external networks.

7 Management of the system is performed via a GUI provided by GoSilent Server, accessed via a browser on remote PCs using TLS/HTTPS. Authorized administrators may configure GoSilent Server as well as the GoSilent Cubes.

8 GoSilent Cubes also provide a GUI accessed from a browser on a user system via a TLS/HTTPS connection. This GUI only provides authorized administrators with the ability to configure that specific GoSilent Cube with enough information to connect to GoSilent Server. All additional configuration information is downloaded from GoSilent Server once the IPsec VPN is established.

9 Both GoSilent Server and GoSilent Cube generate audit records that are stored locally as well as sent to a remote syslog server via a TLS connection.

2.2 TOE Models

10 The physical boundary of the TOE includes all software and hardware included in the models shown in Table 5. The TOE is delivered to the customer via commercial courier.

Table 5: TOE models

Type	Model	CPU	Memory	Storage
GoSilent Server	Virtual Appliance	Intel Xeon E3-1270 v5 (Skylake) w/ ESXi 6.5	16 GB UDIMM	8 GB SD Card (hypervisor) 1 TB SATA HDD
GoSilent Cube	GSC-100	AllWinner H5/ Cortex A-53 (ARM v8-A)	1 GB DRAM	8 GB eMMC
	GSC-120		512 MB DRAM	

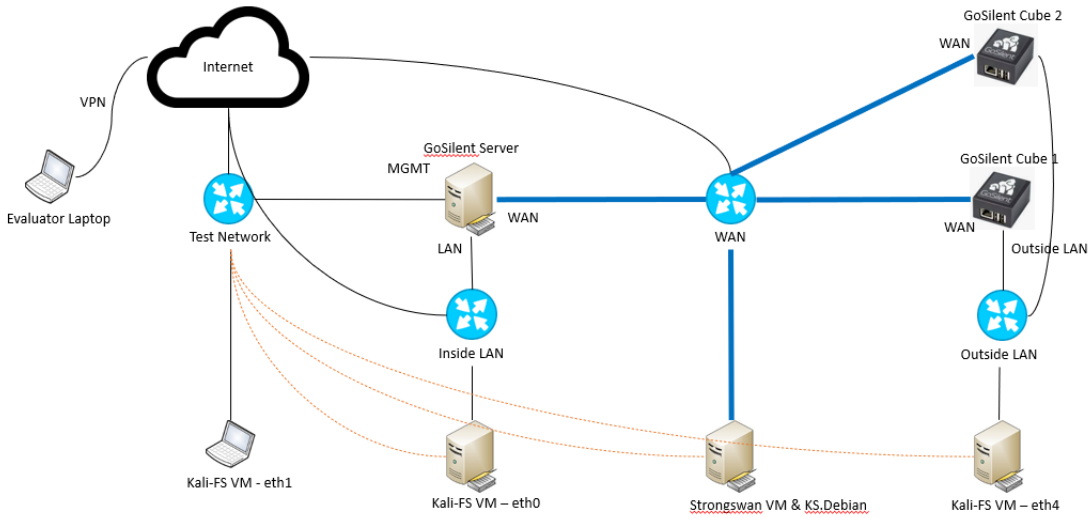
2.2.1 Test Platform Equivalency

11 The evaluation team selected GoSilent Server and GoSilent Cube (GSC-100) to be the models under test. Full testing was performed on each selected model. The GSC-120 only differs from the GSC-100 in the amount of memory. The evaluator determined the amount of memory to not impact evaluated functions therefore, these models are equivalent.

2.2.2 TOE Test Configuration (testing environment)

12 Figure 1 below the TOE test configuration.

Figure 1: TOE Test Configuration



2.2.3 Tools used in the test environment

13 Table 6 below depicts the tools used in the test environment.

Table 5: Tools used in the test environment

Tool name	Version	Description
Lightship Greenlight	3.0.34	Tool used for TLS/X509 certificate modification as well as handshake modification
Lightship Strongswan	5.7.1	Tool used for IPsec modification for X.509 Certificates
Scapy	2.4.4	Packet generation tool
Packeth	1.6	Packet generation tool
Openssl	1.1.1k	Openssl was used for simple TLS server or TLS client connections and as an OCSP responder

StrongSwan	Linux StrongSwan U5.7.2/K4.19.0-22-amd64	Used for IPsec peer connections and algorithm testing
Wireshark	3.4.4 (Linux) & 3.6.5 (Windows)	Used for packet capture and analysis
Tcpdump	4.9.3	Used for packet capture and analysis
Apache	2.4.46	Web server for hosting CRLs
Hping3	3.0.0	Firewall testing
Google Chrome	108.0.5359.125	Access to TOE GUI

3 Evaluation Activities for the collaborative Protection Profile for Network Devices

3.1 Security Audit (FAU)

3.1.1 FAU_GEN.1 Audit data generation

3.1.1.1 TSS

14 For the administrative task of generating/import of, changing, or deleting of cryptographic keys as defined in FAU_GEN.1.1c, the TSS should identify what information is logged to identify the relevant key.

Findings:	[ST] Section 6.1.1 - The following information is logged as a result of the Security Administrator generating/importing, changing, or deleting cryptographic keys: The audit record identifies the cryptographic key via reference to the certificate or key identifier associated with the key.

15 For distributed TOEs the evaluator shall examine the TSS to ensure that it describes which of the overall required auditable events defined in FAU_GEN.1.1 are generated and recorded by which TOE components. The evaluator shall ensure that this mapping of audit events to TOE components accounts for, and is consistent with, information provided in Table 1, as well as events in Tables 2, 4, and 5 (where applicable to the overall TOE). This includes that the evaluator shall confirm that all components defined as generating audit information for a particular SFR should also contribute to that SFR as defined in the mapping of SFRs to TOE components, and that the audit records generated by each component cover all the SFRs that it implements.

Findings:	[ST] Section 6.1.1 – The GoSilent Server and GoSilent Cube components generate audit records associated with the SFRs they implement as specified in Table 20 in section 7.4 of the ST.

3.1.1.2 Guidance Documentation

16 The evaluator shall check the guidance documentation and ensure that it provides an example of each auditable event required by FAU_GEN.1 (i.e. at least one instance of each auditable event, comprising the mandatory, optional and selection-based SFR sections as applicable, shall be provided from the actual audit record).

Findings:	[AGD] Section 4.2 – Table 5 provides an example of each auditable event required by FAU_GEN.1.

17 The evaluator shall also make a determination of the administrative actions related to TSF data related to configuration changes. The evaluator shall examine the guidance documentation and make a determination of which administrative commands, including subcommands, scripts, and configuration files, are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the requirements specified in the cPP. The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are related to TSF data related to configuration changes. The evaluator may perform this activity as part of the activities associated with ensuring that the corresponding guidance documentation satisfies the requirements related to it.

Findings:	The evaluator performed this activity as a part of those Assurance Activities associated with ensuring the corresponding guidance documentation satisfies their independent requirements. Overall, the evaluator considered the administrator guides published by the vendor. The evaluator reviewed the contents of these documents and looked specifically for functionality related to the scope of the evaluation.

3.1.1.3 Tests

18 The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records for the events listed in the table of audit events and administrative actions listed above. This should include all instances of an event: for instance, if there are several different I&A mechanisms for a system, the FIA_UIA_EXT.1 events must be generated for each mechanism. The evaluator shall test that audit records are generated for the establishment and termination of a channel for each of the cryptographic protocols contained in the ST. If HTTPS is implemented, the test demonstrating the establishment and termination of a TLS session can be combined with the test for an HTTPS session. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the guidance documentation, and that the fields in each audit record have the proper entries.

19 For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of auditable events to TOE components in the Security Target. For all events involving more than one TOE component when an audit event is triggered, the evaluator has to check that the event has been audited on both sides (e.g. failure of building up a secure communication channel between the two components). This is not limited to error cases but includes also events about successful actions like successful build up/tear down of a secure communication channel between TOE components.

20 Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly.

High-Level Test Description
Ensure that the TOE displays an audit record for each of the auditable events defined for this requirement.
Findings: PASS – The evaluator performed the testing in conjunction with the testing of the security mechanisms directly. The evaluator confirmed that the TOE correctly generates audit records for the events listed in the table of audit events and administrative actions. The evaluator performed tests on all TOE components and confirmed that all TOE components correctly generated audit events according to the mapping of auditable events to TOE components in the Security Target.

3.1.2 FAU_GEN.2 User identity association

3.1.2.1 TSS & Guidance Documentation

21 The TSS and Guidance Documentation requirements for FAU_GEN.2 are already covered by the TSS and Guidance Documentation requirements for FAU_GEN.1.

3.1.2.2 Tests

22 This activity should be accomplished in conjunction with the testing of FAU_GEN.1.1.

23 For distributed TOEs the evaluator shall verify that where auditable events are instigated by another component, the component that records the event associates the event with the identity of the instigator. The evaluator shall perform at least one

test on one component where another component instigates an auditable event. The evaluator shall verify that the event is recorded by the component as expected and the event is associated with the instigating component. It is assumed that an event instigated by another component can at least be generated for building up a secure channel between two TOE components. If for some reason (could be e.g. TSS or Guidance Documentation) the evaluator would come to the conclusion that the overall TOE does not generate any events instigated by other components, then this requirement shall be omitted.

High-Level Test Description
Ensure that the TOE displays an audit record for each of the auditable events defined for this requirement.
Finding: Pass. The evaluator confirmed that an audit record was generated by the TOE for each required auditable event. This activity was completed in conjunction with FAU_GEN.1.1. Note that the TOE is distributed, however each TOE component is responsible for generating audit events for itself. No TOE components generate audit events on behalf of other components.

3.1.3 FAU_STG_EXT.1 Protected audit event storage

3.1.3.1 TSS

24 The evaluator shall examine the TSS to ensure it describes the means by which the audit data are transferred to the external audit server, and how the trusted channel is provided.

Findings:	[ST] Section 6.1.2 – The Security Target details that each component of the TOE transmits a copy of each audit record to an external syslog server. Each component of the TOE opens a connection to the server using TLS.

25 The evaluator shall examine the TSS to ensure it describes the amount of audit data that are stored locally; what happens when the local audit data store is full; and how these records are protected against unauthorized access.

Findings:	[ST] Section 6.1.2 – The TSS details that each component of the TOE stores up to 50MB of audit information. When that space is exhausted, the oldest records are discarded so that new records can be saved. The records are stored in a series of 5 files, each of 10MB .

26 The evaluator shall examine the TSS to ensure it describes whether the TOE is a standalone TOE that stores audit data locally or a distributed TOE that stores audit data locally on each TOE component or a distributed TOE that contains TOE components that cannot store audit data locally on themselves but need to transfer audit data to other TOE components that can store audit data locally. The evaluator shall examine the TSS to ensure that for distributed TOEs it contains a list of TOE components that store audit data locally. The evaluator shall examine the TSS to ensure that for distributed TOEs that contain components which do not store audit data locally but transmit their generated audit data to other components it contains a mapping between the transmitting and storing TOE components.

Findings:	[ST] Section 2.1 specifies that the TOE is a distributed TOE which consists of the ID Technologies GoSilent Cube + GoSilent Server operating together to provide firewall and VPN capabilities. [ST] Section 6.1.2 – The TSS details that a copy of all audit records are stored on each component of the TOE.

27 The evaluator shall examine the TSS to ensure that it details the behaviour of the TOE when the storage space for audit data is full. When the option 'overwrite previous audit record' is selected this description should include an outline of the rule for overwriting audit data. If 'other actions' are chosen such as sending the new audit data to an external IT entity, then the related behaviour of the TOE shall also be detailed in the TSS.

Findings:	[ST] Section 6.1.2 – The TSS details that when the TOE audit data is full, the oldest records are discarded so that new records can be saved. The records are stored in a series of 5 files, each of 10MB for the GSS and 300KB for the Cube. When the current 10MB or 300KB file (named 'messages') is full, it is then renamed 'messages1' and a new empty file with the same name ('messages') takes its place for new audit messages. As additional files fill, the number appended to it is incremented. In the case of the 5th audit file, 'messages4', the number is not incremented and it is instead removed to limit the saved audit data to 50MB or 1.5MB for the GSS and Cube, respectively.

28 The evaluator shall examine the TSS to ensure that it details whether the transmission of audit information to an external IT entity can be done in real-time or periodically. In case the TOE does not perform transmission in real-time the evaluator needs to verify that the TSS provides details about what event stimulates the transmission to be made as well as the possible as well as acceptable frequency for the transfer of audit data.

Findings:	[ST] Section 6.1.2 – The TSS details that each component also transmits a copy of each audit record to an external syslog server in real time.

29 For distributed TOEs the evaluator shall examine the TSS to ensure it describes to which TOE components this SFR applies and how audit data transfer to the external audit server is implemented among the different TOE components (e.g. every TOE components does its own transfer or the data is sent to another TOE component for central transfer of all audit events to the external audit server).

Findings:	[ST] Section 6.1.2 – The TSS details that each component opens a connection to the server using TLS; GoSilent Cube sends the syslog server traffic through the established IPsec tunnel and the datagrams are then forwarded by GoSilent Server to the syslog server.

30 For distributed TOEs the evaluator shall examine the TSS to ensure it describes which TOE components are storing audit information locally and which components are buffering audit information and forwarding the information to another TOE component for local storage. For every component the TSS shall describe the behaviour when local storage space or buffer space is exhausted.

Findings:	[ST] Section 6.1.2 – The TSS details that each TOE component stores audit event records that are generated on that component. Each component can store up to 50MB of audit information.

3.1.3.2 Guidance Documentation

31 The evaluator shall also examine the guidance documentation to ensure it describes how to establish the trusted channel to the audit server, as well as describe any requirements on the audit server (particular audit server protocol, version of the protocol required, etc.), as well as configuration of the TOE needed to communicate with the audit server.

Findings:	[AGD] Section 3.10 describes how to configure the GoSilent Server (GSS) to enable audit transmission to the syslog server using TLS. Additionally, this section also

	provides configuration of the supported ciphers and certificate checking to be enabled. Section 3.10 also states that the GSS then communicates the configuration to all GoSilent Cubes. Each Cube will transmit audit log traffic via TLS through the established IPsec tunnel. The IPSec header is then stripped by the GSS and is forwarded to the syslog server. The GSS initiates the connection with the syslog server upon startup and the Cube will initiate the connection to the syslog server once the IPsec tunnel to the GSS is established.

32 The evaluator shall also examine the guidance documentation to determine that it describes the relationship between the local audit data and the audit data that are sent to the audit log server. For example, when an audit event is generated, is it simultaneously sent to the external server and the local store, or is the local store used as a buffer and "cleared" periodically by sending the data to the audit server.

Findings:	[AGD] Section 3.10 states that all components store up to 50MB of audit data locally and each component sends a copy of each audit record to the external syslog server in real time.

33 The evaluator shall also ensure that the guidance documentation describes all possible configuration options for FAU_STG_EXT.1.3 and the resulting behaviour of the TOE for each possible configuration. The description of possible configuration options and resulting behaviour shall correspond to those described in the TSS.

Findings:	FAU_STG_EXT.1.3 in the [ST] only selects the option to overwrite previous audit records by discarding the oldest audit files. This option is not configurable on the TOE and [AGD] section 3.10 states that the audit information is stored across 5 files in which the oldest is deleted first.

3.1.3.3 Tests

34 Testing of the trusted channel mechanism for audit will be performed as specified in the associated assurance activities for the particular trusted channel mechanism. The evaluator shall perform the following additional tests for this requirement:

a) Test 1: The evaluator shall establish a session between the TOE and the audit server according to the configuration guidance provided. The evaluator shall then examine the traffic that passes between the audit server and the TOE during several activities of the evaluator's choice designed to generate audit data to be transferred to the audit server. The evaluator shall observe that these data are not able to be viewed in the clear during this transfer, and that they are successfully received by the audit server. The evaluator shall record the particular software (name, version) used on the audit server during testing. The evaluator shall verify that the TOE is capable of transferring audit data to an external audit server automatically without administrator intervention.

High-Level Test Description
The evaluator shall enable the TOE for log shipping to a remote device and capture the log shipping traffic via a Wireshark session. The evaluator used openssl (1.1.1k) s_server to accept secure syslog connections via TLS.
Finding: Pass. The evaluator confirmed that audit data is successfully sent to the remote device and encrypted via TLS.

b) Test 2: The evaluator shall perform operations that generate audit data and verify that this data is stored locally. The evaluator shall perform operations that

generate audit data until the local storage space is exceeded and verifies that the TOE complies with the behaviour defined in FAU_STG_EXT.1.3. Depending on the configuration this means that the evaluator has to check the content of the audit data when the audit data is just filled to the maximum and then verifies that

1) The audit data remains unchanged with every new auditable event that should be tracked but that the audit data is recorded again after the local storage for audit data is cleared (for the option 'drop new audit data' in FAU_STG_EXT.1.3).

2) The existing audit data is overwritten with every new auditable event that should be tracked according to the specified rule (for the option 'overwrite previous audit records' in FAU_STG_EXT.1.3)

3) The TOE behaves as specified (for the option 'other action' in FAU_STG_EXT.1.3).

High-Level Test Description
Verify that when the log file is filled on the TOE, the log file rolls over and a new one is created. When the (5) file limit is met, the TOE removes the oldest to make space for the new file.
Findings: PASS – The evaluator confirmed when the log file is filled up, the log file rolls over to the next one. Once 5 long files are full, the oldest is removed to make space for the new file.

c) Test 3: If the TOE complies with FAU_STG_EXT.2/LocSpace the evaluator shall verify that the numbers provided by the TOE according to the selection for FAU_STG_EXT.2/LocSpace are correct when performing the tests for FAU_STG_EXT.1.3

High-Level Test Description
FAU_STG_EXT.2/LocSpace is not claimed by the TOE. N/A
Findings: N/A

d) Test 4: For distributed TOEs, Test 1 defined above should be applicable to all TOE components that forward audit data to an external audit server. For the local storage according to FAU_STG_EXT.1.2 and FAU_STG_EXT.1.3 the Test 2 specified above shall be applied to all TOE components that store audit data locally. For all TOE components that store audit data locally and comply with FAU_STG_EXT.2/LocSpace Test 3 specified above shall be applied. The evaluator shall verify that the transfer of audit data to an external audit server is implemented.

High-Level Test Description
Test 1 and Test 2 has been applied to all the TOE components.
Findings: PASS – Test 1 and 2 have been performed all TOE components.

3.2 Cryptographic Support (FCS)

3.2.1 FCS_CKM.1 Cryptographic Key Generation

3.2.1.1 TSS

35 The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Findings:	[ST] Section 6.3.1 – The TOE generates 2048-bit RSA keys and P-384 ECC keys in support of TLS client connections with the syslog server, and P-384 ECC keys in support of Administrative GUI TLS server sessions. ECDSA key pairs for X509 certificates can also be generated through the TOE GUI where the option of keys using P-256 or P-384 curves can be selected.

3.2.1.2 Guidance Documentation

36 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key generation scheme(s) and key size(s) for all cryptographic protocols defined in the Security Target.

Findings:	[AGD] Section 3.11 states the key establishment parameters for TLS Client, TLS Server, and IKE/IPsec. Section 3.11 also points to [AGD] sections 3.8 and 3.10 which contain instructions on how to configure the TLS and IPsec parameters.

3.2.1.3 Tests

37 Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products. Generation of long-term cryptographic keys (i.e. keys that are not ephemeral keys/session keys) might be performed automatically (e.g. during initial start-up). Testing of key generation must cover not only administrator invoked key generation but also automated key generation (if supported).

Key Generation for FIPS PUB 186-4 RSA Schemes

38 The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent *e*, the private prime factors *p* and *q*, the public modulus *n* and the calculation of the private signature exponent *d*.

39 Key Pair generation specifies 5 ways (or methods) to generate the primes *p* and *q*. These include:

a) Random Primes:

- Provable primes
- Probable primes

b) Primes with Conditions:

- Primes p1, p2, q1,q2, p and q shall all be provable primes
- Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes
- Primes p1, p2, q1,q2, p and q shall all be probable primes

40 To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation.

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

41 For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-4 Public Key Verification (PKV) Test

42 For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Key Generation for Finite-Field Cryptography (FFC)

43 The evaluator shall verify the implementation of the Parameters Generation and the Key Generation for FFC by the TOE using the Parameter Generation and Key Generation test. This test verifies the ability of the TSF to correctly produce values for the field prime p, the cryptographic prime q (dividing p-1), the cryptographic group generator g, and the calculation of the private key x and public key y.

44 The Parameter generation specifies 2 ways (or methods) to generate the cryptographic prime q and the field prime p:

- Primes q and p shall both be provable primes
- Primes q and field prime p shall both be probable primes

45 and two ways to generate the cryptographic group generator g:

- Generator g constructed through a verifiable process
- Generator g constructed through an unverifiable process.

46 The Key generation specifies 2 ways to generate the private key x:

- len(q) bit output of RBG where 1 <=x <= q-1
- len(q) + 64 bit output of RBG, followed by a mod q-1 operation and a +1 operation, where 1<= x<=q-1.

47 The security strength of the RBG must be at least that of the security offered by the FFC parameter set.

48 To test the cryptographic and field prime generation method for the provable primes method and/or the group generator g for a verifiable process, the evaluator must seed

the TSF parameter generation routine with sufficient data to deterministically generate the parameter set.

49 For each key length supported, the evaluator shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm

- g != 0,1
- q divides p-1
- g^q mod p = 1
- g^x mod p = y

50 for each FFC parameter set and key pair.

NIAP TD0580

FFC Schemes using "safe-prime" groups

51 Testing for FFC Schemes using safe-prime groups is done as part of testing in CKM.2.1.

Findings:	The vendor uses the CAVP certificate C1159 and C1104 for ECDSA and A2980 and A2979 for RSA key generation. These are described in [ST] Table 4.

3.2.2 FCS_CKM.2 Cryptographic Key Establishment

3.2.2.1 TSS

52 The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme. It is sufficient to provide the scheme, SFR, and service in the TSS.

NIAP TD0580

53 **Removed:** ~~If Diffie-Hellman group 14 is selected from FCS_CKM.2.1, the TSS shall claim the TOE meets RFC 3526 Section 3.~~

Findings:	This activity was removed by TD0580

54 The intent of this activity is to be able to identify the scheme being used by each service. This would mean, for example, one way to document scheme usage could be:

Scheme	**SFR**	**Service**
RSA	FCS_TLSS_EXT.1	Administration
ECDH	FCS_SSHC_EXT.1	Audit Server
~~Diffie-Hellman (Group 14)~~	~~FCS_SSHC_EXT.1~~ **Removed per TD0580**	~~Backup Server~~ **Removed per TD0580**

Scheme	SFR	Service
Removed per TD0580		
ECDH	FCS_IPSEC_EXT.1	Authentication Server

55 The information provided in the example above does not necessarily have to be included as a table but can be presented in other ways as long as the necessary data is available.

Findings:	[ST] section 6.3.1 states the TOE support for key establishment and parameters: TLS Client to syslog server – RSAES-PKCS1-v1_5 and Elliptic-curve (P-384) TLS Server for administrative GUI - Elliptic-curve (P-384) IKE/IPsec - Elliptic-curve (P-256, P-384) and FFC safe-primes (RFC 3526 DH14) [ST] Section 6.3.8 FCS_TLSC_EXT.1 states that the TOE acts as a TLS client to provide a trusted channel to the syslog server. [ST] Section 6.3.9 FCS_TLSS_EXT.1, FCS_HTTPS_EXT.1 describes that the TOE acts as a TLS/HTTPS server to provide an Administrative GUI to administrators. [ST] Section 6.3.7 FCS_IPSEC_EXT.1 specifies that it is used for IPSec VPN connections. The evaluator confirmed that the claimed key establishment schemes correspond to the claimed key generation schemes in FCS_CKM.1.1.

3.2.2.2 Guidance Documentation

56 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).

Findings:	[AGD] Section 3.8 specifies how to configure the IPsec cipher suites and key establishment schemes on the TOE. [AGD] section 3.10 specifies how to configure TLS ciphers and key establishment schemes on the TOE to use as a TLS client for remote syslog. The TOE only supports Elliptic-curve (P-384) for the Administrative GUI TLS server and no further configuration is required.

3.2.2.3 Tests

Key Establishment Schemes

57 The evaluator shall verify the implementation of the key establishment schemes of the supported by the TOE using the applicable tests below.

SP800-56A Key Establishment Schemes

58 The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of

the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag.

Function Test

59 The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement scheme-key agreement role combination, KDF type, and, if supported, key confirmation role- key confirmation type combination, the tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested.

60 The evaluator shall obtain the DKM, the corresponding TOE's public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information field OI and TOE id fields.

61 If the TOE does not use a KDF defined in SP 800-56A, the evaluator shall obtain only the public keys and the hashed value of the shared secret.

62 The evaluator shall verify the correctness of the TSF's implementation of a given scheme by using a known good implementation to calculate the shared secret value, derive the keying material DKM, and compare hashes or MAC tags generated from these values.

63 If key confirmation is supported, the TSF shall perform the above for each implemented approved MAC algorithm.

Validity Test

64 The Validity test verifies the ability of the TOE to recognize another party's valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize. The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved curves, the evaluator's public keys, the TOE's public/private key pairs, MACTag, and any inputs used in the KDF, such as the other info and TOE id fields.

65 The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

66 The TOE shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator shall compare the TOE's results

with the results using a known good implementation verifying that the TOE detects these errors.

RSA-based key establishment schemes

67 The evaluator shall verify the correctness of the TSF's implementation of RSAES-PKCS1-v1_5 by using a known good implementation for each protocol selected in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses RSAES-PKCS1-v1_5.

High-Level Test Description
Testing evidence for this can be found in FPT_ITT.1, FTP_TRP.1/Admin and FTP_ITC.1.
Findings: PASS – The evaluator verified correctness of the TSF's implementation of RSAES-PKCS1-v1_5 using a known good implementation of the TLS protocol as part of the testing activities in FCS_TLSC_EXT.1.1 Test 1.

NIAP TD0580 Removed:

~~***Diffie-Hellman Group 14***~~

68 ~~The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 by using a known good implementation for each protocol selected in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses Diffie-Hellman group 14.~~

69 ***FFC Schemes using "safe-prime" groups***

70 The evaluator shall verify the correctness of the TSF's implementation of safe-prime groups by using a known good implementation for each protocol selected in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses safe-prime groups. This test must be performed for each safe-prime group that each protocol uses.

High-Level Test Description
Testing evidence for this can be found in FPT_ITT.1, FTP_TRP.1/Admin and FTP_ITC.1.
Findings: PASS – The evaluator verified the correctness of the TSF's implementation of safe-prime groups using a known good implementation as part of the testing activities in FCS_IPSEC_EXT.1.11 Test 1.

3.2.3 FCS_CKM.4 Cryptographic Key Destruction

3.2.3.1 TSS

71 The evaluator examines the TSS to ensure it lists all relevant keys (describing the origin and storage location of each), all relevant key destruction situations (e.g. factory reset or device wipe function, disconnection of trusted channels, key change as part of a secure channel protocol), and the destruction method used in each case. For the purpose of this Evaluation Activity the relevant keys are those keys that are relied upon to support any of the SFRs in the Security Target. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE (e.g. that all keys for the TOE-specific secure channels and protocols, or that support FPT_APW.EXT.1 and FPT_SKP_EXT.1, are accounted

for[1]). In particular, if a TOE claims not to store plaintext keys in non-volatile memory then the evaluator checks that this is consistent with the operation of the TOE.

Findings:	[ST] Section 6.3.2 – The TSS outlines the following relevant keys and their destruction methods. Cryptographic keys in volatile and non-volatile memory are overwritten with zeroes when a key is deleted. Keys in volatile memory are also destroyed when a component is powered down or rebooted. The following keys and other sensitive information are maintained on each component: ECDSA private key – stored on the hard drive and overwritten when a factory reset is performed IPsec session key – stored in volatile memory and overwritten when a session is terminated TLS session key - stored in volatile memory and overwritten when a session is terminated Administrator passwords – Plaintext value is stored in volatile memory when supplied by a user and overwritten after validation; configured administrator passwords are stored on the hard drive as hashed (SHA-256) values only.

72 The evaluator shall check to ensure the TSS identifies how the TOE destroys keys stored as plaintext in non-volatile memory, and that the description includes identification and description of the interfaces that the TOE uses to destroy keys (e.g., file system APIs, key store APIs).

Findings:	[ST] Section 6.3.2 – The TSS details the destruction of cryptographic keys by using the underlying filesystem to destroy keys in non-volatile memory as being overwritten with zeroes after deletion.

73 Note that where selections involve '*destruction of reference*' (for volatile memory) or '*invocation of an interface'* (for non-volatile memory) then the relevant interface definition is examined by the evaluator to ensure that the interface supports the selection(s) and description in the TSS. In the case of non-volatile memory, the evaluator includes in their examination the relevant interface description for each media type on which plaintext keys are stored. The presence of OS-level and storage device-level swap and cache files is not examined in the current version of the Evaluation Activity.

Findings:	[ST] Section 6.3.2 states that only ECDSA private keys are stored on the non-volatile hard drive and are overwritten when a factory reset is performed.

74 Where the TSS identifies keys that are stored in a non-plaintext form, the evaluator shall check that the TSS identifies the encryption method and the key-encrypting-key used, and that the key-encrypting-key is either itself stored in an encrypted form or that it is destroyed by a method included under FCS_CKM.4.

Findings:	[ST] Section 6.3.2 does not identify any keys that are stored in a non-plaintext form.

75 The evaluator shall check that the TSS identifies any configurations or circumstances that may not conform to the key destruction requirement (see further discussion in the Guidance Documentation section below). Note that reference may be made to

[1] Where keys are stored encrypted or wrapped under another key then this may need to be explained in order to allow the evaluator to confirm the consistency of the description of keys with the TOE functions.

the Guidance Documentation for description of the detail of such cases where destruction may be prevented or delayed.

Findings:	[ST] Section 6.3.2 – The TSS does not identify any configurations or circumstances that may not conform to the key destruction requirement.

76 Where the ST specifies the use of “a value that does not contain any CSP” to overwrite keys, the evaluator examines the TSS to ensure that it describes how that pattern is obtained and used, and that this justifies the claim that the pattern does not contain any CSPs.

Findings:	The ST does not specify the use of “a value that does not contain any CSP” to overwrite keys.

3.2.3.2 Guidance Documentation

77 A TOE may be subject to situations that could prevent or delay key destruction in some cases. The evaluator shall check that the guidance documentation identifies configurations or circumstances that may not strictly conform to the key destruction requirement, and that this description is consistent with the relevant parts of the TSS (and any other supporting information used). The evaluator shall check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer.

78 For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-levelling and garbage collection. This may result in additional copies of the key that are logically inaccessible but persist physically. Where available, the TOE might then describe use of the TRIM command[2] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

Findings:	N/A—The TOE is not subject to any situations that prevent or delay key destruction.

3.2.3.3 Tests

79 None

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

3.2.4.1 TSS

80 The evaluator shall examine the TSS to ensure it identifies the key size(s) and mode(s) supported by the TOE for data encryption/decryption.

Findings:	[ST] Section 6.3.3 – The TSS identifies the supported key size(s) as the following: TLS Client to syslog server - AES-GCM, 128 or 256 bit keys

[2] Where TRIM is used then the TSS and/or guidance documentation is also expected to describe how the keys are stored such that they are not inaccessible to TRIM, (e.g. they would need not to be contained in a file less than 982 bytes which would be completely contained in the master file table).

	TLS Server for administrative GUI - AES-GCM, 256 bit keys IKE/IPsec – AES-CBC or AES-GCM, 128 or 256 bit keys

3.2.4.2 Guidance Documentation

81 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected mode(s) and key size(s) defined in the Security Target supported by the TOE for data encryption/decryption.

Findings:	[AGD] Section 3.8 specifies how the TOE can be configured for IPsec using the AES-CBC or AES-GCM modes and 128 or 256 bit key sizes. [AGD] Section 3.10 specifies the AES ciphers and key sizes for TLS client connections for remote syslog. The TLS server only supports one AES mode and key size, so no further configuration is required.

3.2.4.3 Tests

AES-CBC Known Answer Tests

82 There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

83 **KAT-1.** To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all-zeros key.

84 To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption.

85 **KAT-2.** To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys.

86 To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption.

87 **KAT-3.** To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall

have 256 256-bit keys. Key *i* in each set shall have the leftmost *i* bits be ones and the rightmost *N-i* bits be zeros, for *i* in [1,N].

88 To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key.

89 **KAT-4.** To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 128-i bits be zeros, for i in [1,128].

90 To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input and AES-CBC decryption.

AES-CBC Multi-Block Message Test

91 The evaluator shall test the encrypt functionality by encrypting an *i*-block message where 1 < *i* <=10. The evaluator shall choose a key, an IV and plaintext message of length *i* blocks and encrypt the message, using the mode to be tested, with the chosen key and IV. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation.

92 The evaluator shall also test the decrypt functionality for each mode by decrypting an *i*-block message where 1 < *i* <=10. The evaluator shall choose a key, an IV and a ciphertext message of length *i* blocks and decrypt the message, using the mode to be tested, with the chosen key and IV. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key and IV using a known good implementation.

AES-CBC Monte Carlo Tests

93 The evaluator shall test the encrypt functionality using a set of 200 plaintext, IV, and key 3-tuples. 100 of these shall use 128 bit keys, and 100 shall use 256 bit keys. The plaintext and IV values shall be 128-bit blocks. For each 3-tuple, 1000 iterations shall be run as follows:

```
# Input: PT, IV, Key
for i = 1 to 1000:
        if i == 1:
                CT[1] = AES-CBC-Encrypt(Key, IV, PT)
```

PT = IV

else:

CT[i] = AES-CBC-Encrypt(Key, PT)

PT = CT[i-1]

94 The ciphertext computed in the 1000th iteration (i.e., CT[1000]) is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.

95 The evaluator shall test the decrypt functionality using the same test as for encrypt, exchanging CT and PT and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

AES-GCM Test

96 The evaluator shall test the authenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths:

128 bit and 256 bit keys

a) **Two plaintext lengths**. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported.

a) **Three AAD lengths**. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported.

b) **Two IV lengths**. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested.

97 The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM authenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known.

98 The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on authentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail.

99 The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

AES-CTR Known Answer Tests

100 The Counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. Since the Counter Mode does not specify the counter that is used, it is not possible to implement an automated test for this mode. The generation and management of the counter is tested through FCS_SSH*_EXT.1.4. If CBC and/or GCM are selected in FCS_COP.1/DataEncryption, the test activities for those modes sufficiently demonstrate the correctness of the AES algorithm. If CTR is the only selection in FCS_COP.1/DataEncryption, the AES-CBC Known Answer Test, AES-

GCM Known Answer Test, or the following test shall be performed (all of these tests demonstrate the correctness of the AES algorithm):

101 There are four Known Answer Tests (KATs) described below to test a basic AES encryption operation (AES-ECB mode). For all KATs, the plaintext, ~~IV,~~ and ciphertext values shall be 128-bit blocks. The results from each test may either be obtained by the validator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

102 KAT-1 To test the encrypt functionality, the evaluator shall supply a set of 5 plaintext values for each selected keysize and obtain the ciphertext value that results from encryption of the given plaintext using a key value of all zeros.

103 KAT-2 To test the encrypt functionality, the evaluator shall supply a set of 5 key values for each selected keysize and obtain the ciphertext value that results from encryption of an all zeros plaintext using the given key value.

104 KAT-3 To test the encrypt functionality, the evaluator shall supply a set of key values for each selected keysize as described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the given key values. A set of 128 128-bit keys, a set of 192 192-bit keys, and/or a set of 256 256-bit keys. Key_i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1, N].

105 KAT-4 To test the encrypt functionality, the evaluator shall supply the set of 128 plaintext values described below and obtain the ciphertext values that result from encryption of the given plaintext using each selected keysize with a key value of all zeros (e.g. 256 ciphertext values will be generated if 128 bits and 256 bits are selected and 384 ciphertext values will be generated if all keysizes are selected). Plaintext value i in each set shall have the leftmost bits be ones and the rightmost 128-i bits be zeros, for i in [1, 128].

AES-CTR Multi-Block Message Test

106 The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 less-than i less-than-or-equal to 10 (test shall be performed using AES-ECB mode). For each i the evaluator shall choose a key and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key using a known good implementation. The evaluator shall perform this test using each selected keysize.

AES-CTR Monte-Carlo Test

107 The evaluator shall test the encrypt functionality using 100 plaintext/key pairs. The plaintext values shall be 128-bit blocks. For each pair, 1000 iterations shall be run as follows:

Input: PT, Key

for i = 1 to 1000:

CT[i] = AES-ECB-Encrypt(Key, PT) PT = CT[i]

108 The ciphertext computed in the 1000th iteration is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation. The evaluator shall perform this test using each selected keysize.

109 There is no need to test the decryption engine.

Findings:	The vendor uses the CAVP certificates C1159 and C1104 for AES. This is described in [ST] Table 4.

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

3.2.5.1 TSS

110 The evaluator shall examine the TSS to determine that it specifies the cryptographic algorithm and key size supported by the TOE for signature services.

Findings:	[ST] Section 6.3.4 – The TSS identifies the cryptographic algorithms and key sizes supported by the TOE for signature services as the following: ECDSA P-256 and P-384 SigGen to support IPsec and TLS functions ECDSA P-256 and P-384 SigVer to support IPsec, TLS, X.509, and trusted update functions including firmware integrity checking. RSA SigGen and SigVer with key sizes of 2048 bits to support TLS client connections.

3.2.5.2 Guidance Documentation

111 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected cryptographic algorithm and key size defined in the Security Target supported by the TOE for signature services.

Findings:	[AGD] Section 3.11 specifies the cryptographic algorithms and key sizes supported by the TOE for signature services for the claimed cryptographic protocols. [AGD] Section 3.8 specifies how to configure the IPsec protocol to use the selected cryptographic algorithm and key sizes for signature services. [AGD] section 3.10 specifies how to configure the TLS client protocol to use the selected cryptographic algorithms and key sizes for signature services. The TOE only supports Elliptic-curve (P-384) for the Administrative GUI TLS server and no further configuration is required.

3.2.5.3 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

112 For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

ECDSA FIPS 186-4 Signature Verification Test

113 For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024-bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values.

RSA Signature Algorithm Tests

Signature Generation Test

114 The evaluator generates or obtains 10 messages for each modulus size/SHA combination supported by the TOE. The TOE generates and returns the corresponding signatures.

115 The evaluator shall verify the correctness of the TOE's signature using a trusted reference implementation of the signature verification algorithm and the associated public keys to verify the signatures.

Signature Verification Test

116 For each modulus size/hash algorithm selected, the evaluator generates a modulus and three associated key pairs, (*d*, *e*). Each private key *d* is used to sign six pseudorandom messages each of 1024 bits using a trusted reference implementation of the signature generation algorithm. Some of the public keys, *e*, messages, or signatures are altered so that signature verification should fail. For both the set of original messages and the set of altered messages: the modulus, hash algorithm, public key *e* values, messages, and signatures are forwarded to the TOE, which then attempts to verify the signatures and returns the verification results.

117 The evaluator verifies that the TOE confirms correct signatures on the original messages and detects the errors introduced in the altered messages.

Findings:	The vendor uses the CAVP certificates C1159 and C1104 for ECDSA and RSA signature generation and verification. These are described in [ST] Table 4.

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

3.2.6.1 TSS

118 The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

Findings:	[ST] Section 6.3.5 – The TSS identifies hash functions that work with other TSF cryptographic functions such as SHA-1, SHA-256, and SHA-384. These hash functions are used for SigGen and SigVer operations. The TSS clearly documents the association of the hash functions with other TSF cryptographic functions.

3.2.6.2 Guidance Documentation

119 The evaluator checks the AGD documents to determine that any configuration that is required to configure the required hash sizes is present.

Findings:	[AGD] Section 3.8 specifies how the TOE can be configured for IPsec using each of the supported hash functions. [AGD] Section 3.10 specifies how to configure the hash functions for TLS client connections for remote syslog. [AGD] Section 3.11 specifies that the TLS Server only uses one ciphersuite and the hash function is configured by default.

3.2.6.3 Tests

120 The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF only hashes messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs.

121 The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP.

Short Messages Test - Bit-oriented Mode

122 The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Short Messages Test - Byte-oriented Mode

123 The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Bit-oriented Mode

124 The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the ith message is m + 99*i, where $1 \leq i \leq m$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Byte-oriented Mode

125 The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the ith message is m + 8*99*i, where $1 \leq i \leq m/8$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Test

126 This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.

Findings:	The vendor uses the CAVP certificates C1159 and C1104 for SHA-1, SHA-256 and SHA-384 hashes. These are described in [ST] Table 4.

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

3.2.7.1 TSS

127 The evaluator shall examine the TSS to ensure that it specifies the following values used by the HMAC function: key length, hash function used, block size, and output MAC length used.

Findings:	[ST] Section 6.3.6 – The TSS identifies all values used by the HMAC function. The values used by the TOE are as follows: The TOE uses HMAC-SHA-256 and HMAC-SHA-384 TLS KDF and TLS message authentication, and TLS client connections to the syslog server with key sizes [256, 384] bits, [512, 1024] bit block size, and [256, 384] bit message digest size. The administrative GUI uses HMAC-SHA-384 with a key size of 384 bits. The TOE uses HMAC-SHA-1, HMAC-SHA-256, and HMAC-SHA-384 for IKE/IPsec, with key sizes [160, 256, 384] bits, [512, 512, 1024] bit block size, and [160, 256, 384] bit message digest size.

3.2.7.2 Guidance Documentation

128 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the values used by the HMAC function: key length, hash function used, block size, and output MAC length used defined in the Security Target supported by the TOE for keyed hash function.

Findings:	[AGD] Section 3.8 specifies how the TOE can be configured for IPsec using each of the supported keyed hash functions. [AGD] Section 3.10 specifies that the administrator can configure the TLS ciphers for the TLS client. As per section 3.11 of the [AGD], this indirectly will make use of the appropriate keyed-hash function. [AGD] Section 3.11 states a single ciphersuite, TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384, is supported for the TLS/HTTPS server. Note [AGD] section 3.11 also states that keyed hash functions are not configured independently and are selected in conjunction with the desired ciphersuite.

3.2.7.3 Tests

129 For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and message data using a known good implementation.

Findings:	The vendor uses the CAVP certificates C1159 and C1104 for HMAC-SHA-256 and HMAC-SHA-384 keyed hashes. These are described in [ST] Table 4.

3.2.8 FCS_RBG_EXT.1 Extended: Cryptographic Operation (Random Bit Generation)

130 Documentation shall be produced—and the evaluator shall perform the activities—in accordance with Appendix D of [NDcPP].

3.2.8.1 TSS

131 The evaluator shall examine the TSS to determine that it specifies the DRBG type, identifies the entropy source(s) seeding the DRBG, and state the assumed or calculated min-entropy supplied either separately by each source or the min-entropy contained in the combined seed value.

Findings:	[ST] Section 6.3.10 specifies that the TOE implements a SP800-90A compliant Counter DRBG (AES) which is seeded with 256 bits of entropy from a software-based noise source.

3.2.8.2 Guidance Documentation

132 The evaluator shall confirm that the guidance documentation contains appropriate instructions for configuring the RNG functionality.

Findings:	[AGD] Section 3.11 specifies that the RNG functionality provided by the TOE does not require additional configuration.

3.2.8.3 Tests

133 The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable, the evaluator shall perform 15 trials for each configuration.

134 If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. “generate one block of random bits” means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP800-90A).

135 If the RNG does not have prediction resistance, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) reseed, (4) generate a second block of random bits (5) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to reseed. The final value is additional input to the second generate call.

136 The following paragraphs contain more information on some of the input values to be generated/selected by the evaluator.

Entropy input: the length of the entropy input value must equal the seed length.

Nonce: If a nonce is supported (CTR_DRBG with no Derivation Function does not use a nonce), the nonce bit length is one-half the seed length.

Personalization string: The length of the personalization string must be <= seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is support, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.

Additional input: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.

Findings:	The vendor uses the CAVP certificates C1159 and C1104 for the DRBG. This is described in [ST] Table 4.

3.3 Identification and Authentication (FIA)

3.3.1 FIA_AFL.1 Authentication Failure Management

3.3.1.1 TSS

137 The evaluator shall examine the TSS to determine that it contains a description, for each supported method for remote administrative actions, of how successive unsuccessful authentication attempts are detected and tracked. The TSS shall also describe the method by which the remote administrator is prevented from successfully logging on to the TOE, and the actions necessary to restore this ability.

Findings:	[ST] Section 6.6.1 describes that the TOE supports remote admin actions via HTTPS. This section then goes on to describe that the TOE tracks the number of successive failed authentication attempts for each user account. Upon meeting this threshold, the TOE locks the account in question for an administrator configured period of time. During this time, entering a correct password for the locked account will result in authentication failure. A successful authentication will reset the failed logon counter to zero.

138 The evaluator shall examine the TSS to confirm that the TOE ensures that authentication failures by remote administrators cannot lead to a situation where no administrator access is available, either permanently or temporarily (e.g. by providing local logon which is not subject to blocking).

Findings:	[ST] Section 6.6.1 describes that administrator account lockout is not applicable to local access.

3.3.1.2 Guidance Documentation

139 The evaluator shall examine the guidance documentation to ensure that instructions for configuring the number of successive unsuccessful authentication attempts and time period (if implemented) are provided, and that the process of allowing the remote administrator to once again successfully log on is described for each “action” specified (if that option is chosen). If different actions or mechanisms are implemented depending on the secure protocol employed (e.g., TLS vs. SSH), all must be described.

Findings:	[AGD] Section 3.4 – The AGD describes how to modify the lockout functions on the TOE as well as the threshold for the lockout. The admin will regain access depending on the length of time specified in the lockout settings.

140 The evaluator shall examine the guidance documentation to confirm that it describes, and identifies the importance of, any actions that are required in order to ensure that administrator access will always be maintained, even if remote administration is made

permanently or temporarily unavailable due to blocking of accounts as a result of FIA_AFL.1.

Findings:	[AGD] Sections 3.3.3.3 states that the Administrator accounts are not subject to account locking when using the local interface.

3.3.1.3 Tests

141 The evaluator shall perform the following tests for each method by which remote administrators access the TOE (e.g. any passwords entered as part of establishing the connection protocol or the remote administrator application):

a) Test 1: The evaluator shall use the operational guidance to configure the number of successive unsuccessful authentication attempts allowed by the TOE (and, if the time period selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall also use the operational guidance to configure the time period after which access is re-enabled). The evaluator shall test that once the authentication attempts limit is reached, authentication attempts with valid credentials are no longer successful.

High-Level Test Description
The evaluator shall set a lockout policy as well as a lockout interval. The evaluator shall attempt to login to a user account with invalid credentials past the set threshold of allowed login attempts before lockout.
Findings: PASS – The evaluator confirmed that the TOE properly denies access to a user after the configured threshold of invalid login attempts is met.

b) Test 2: After reaching the limit for unsuccessful authentication attempts as in Test 1 above, the evaluator shall proceed as follows.

1. If the administrator action selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall confirm by testing that following the operational guidance and performing each action specified in the ST to re-enable the remote administrator's access results in successful access (when using valid credentials for that administrator).

2. If the time period selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall wait for just less than the time period configured in Test 1 and show that an authorisation attempt using valid credentials does not result in successful access. The evaluator shall then wait until just after the time period configured in Test 1 and show that an authorisation attempt using valid credentials results in successful access.

High-Level Test Description
After the lockout interval has been reached the evaluator shall log into the same account with valid credentials.
Findings: PASS – The evaluator configured the TOE to lockout for 60 seconds after 3 failed login attempts and attempted to login to the TOE with the incorrect credentials 3 times. The evaluator confirmed that the TOE did not allow the user to login and locked the user out after 3 attempts. The evaluator then attempted to log into the TOE with the correct credentials before the 60 second threshold and confirmed that the TOE did not allow the login due to the lockout. The evaluator then confirmed that the TOE allows the user to login to the TOE after the configured time elapses.

3.3.2 FIA_PMG_EXT.1 Password Management

3.3.2.1 TSS

142 The evaluator shall examine the TSS to determine that it contains the lists of the supported special character(s) and minimum and maximum number of charters supported for administrator passwords.

Findings:	[ST] Section 6.6.1 – The TSS identifies the following supported characters for passwords on the TOE. The minimum password may be configured from 8 to 40 characters, that incorporate a combination of lowercase letters, uppercase letters, numbers, and special characters (“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”).

3.3.2.2 Guidance Documentation

143 The evaluator shall examine the guidance documentation to determine that it:

a) identifies the characters that may be used in passwords and provides guidance to security administrators on the composition of strong passwords, and

b) provides instructions on setting the minimum password length and describes the valid minimum password lengths supported.

Findings:	[AGD] Section 3.6 provides instructions for how to configure the minimum password length for both the GSS and Cube as well as the acceptable range for minimum password length. Section 3.6 also states the password composition characters which are consistent with the FIA_PMG_EXT.1.1 in the [ST].

3.3.2.3 Tests

144 The evaluator shall perform the following tests.

a) Test 1: The evaluator shall compose passwords that meet the requirements in some way. For each password, the evaluator shall verify that the TOE supports the password. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that all characters, and a minimum length listed in the requirement are supported and justify the subset of those characters chosen for testing.

High-Level Test Description
The evaluator shall define a minimum password length and create a new user with the minimum password length. The evaluator shall change the minimum password length and edit the user’s password to satisfy the new password requirement.
Findings: PASS – The evaluator confirmed that the TOE enforces the password policy configured by the administrator.

b) Test 2: The evaluator shall compose passwords that do not meet the requirements in some way. For each password, the evaluator shall verify that the TOE does not support the password. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that the TOE enforces the allowed characters and the minimum length

listed in the requirement and justify the subset of those characters chosen for testing.

High-Level Test Description
The evaluator shall provide an invalid password to the TOE that did not meet the minimum length requirement and verify it is not accepted due to the invalid parameters.
Findings: PASS – The evaluator confirmed that the TOE rejected a password that did not meet the minimum length requirement.

3.3.3 FIA_UIA_EXT.1 User Identification and Authentication

3.3.3.1 TSS

145 The evaluator shall examine the TSS to determine that it describes the logon process for each logon method (local, remote (HTTPS, SSH, etc.)) supported for the product. This description shall contain information pertaining to the credentials allowed/used, any protocol transactions that take place, and what constitutes a "successful logon".

Findings:	[ST] Section 6.6.1 – The TSS identifies the login method as username and password through a GUI using the HTTPS protocol.

146 The evaluator shall examine the TSS to determine that it describes which actions are allowed before user identification and authentication. The description shall cover authentication and identification for local and remote TOE administration.

Findings:	[ST] Section 6.6.1 – The TSS describes that the TOE provides the warning banner as described in FTA_TAB.1 prior to authentication for local and remote connections. For local connections only, the GoSilent Cube also allows a factory reset function prior to login. Local access in available to the GoSilent Server via an HTTPS session through the MGMT interface. Remote administrator access is available to the GoSilent Server via HTTPS from GoSilent Cube client systems. Local access is available to the GoSilent Cube via the 'Device' Ethernet interface. Remote administration of the Cube is performed via configuration on the GoSilent Server. The GoSilent Cube downloads the configuration settings when it establishes an IPsec tunnel to the GoSilent Server.

147 For distributed TOEs the evaluator shall examine that the TSS details how Security Administrators are authenticated and identified by all TOE components. If not all TOE components support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the TSS shall describe how the overall TOE functionality is split between TOE components including how it is ensured that no unauthorized access to any TOE component can occur.

Findings:	[ST] Section 6.6.1 – The TSS identifies all login methods for each TOE component part of the evaluated configuration. Each TOE component can be logged into remotely and locally.

148 For distributed TOEs, the evaluator shall examine the TSS to determine that it describes for each TOE component which actions are allowed before user identification and authentication. The description shall cover authentication and identification for local and remote TOE administration. For each TOE component that does not support authentication of Security Administrators according to

FIA_UIA_EXT.1 and FIA_UAU_EXT.2 the TSS shall describe any unauthenticated services/services that are supported by the component.

Findings:	[ST] Section 6.6.1 – The TSS describes the method of login for both components and that there are no actions available before user identification and authentication. The description covers both local and remote login method for all TOE components.

3.3.3.2 Guidance Documentation

149 The evaluator shall examine the guidance documentation to determine that any necessary preparatory steps (e.g., establishing credential material such as pre-shared keys, tunnels, certificates, etc.) to logging in are described. For each supported the login method, the evaluator shall ensure the guidance documentation provides clear instructions for successfully logging on. If configuration is necessary to ensure the services provided before login are limited, the evaluator shall determine that the guidance documentation provides sufficient instruction on limiting the allowed services.

Findings:	[AGD] Section 3.1 – The AGD describes the interfaces used to access both the GoSilent Server and the GoSilent Cube Administrative GUI. The guidance provides clear instructions on how to access the TOE such as which port number to use and what interface to interact with.

3.3.3.3 Tests

150 The evaluator shall perform the following tests for each method by which administrators access the TOE (local and remote), as well as for each type of credential supported by the login method:

a) Test 1: The evaluator shall use the guidance documentation to configure the appropriate credential supported for the login method. For that credential/login method, the evaluator shall show that providing correct I&A information results in the ability to access the system, while providing incorrect information results in denial of access.

High-Level Test Description
Login to the TOE using valid credentials and verify the user has access to the TOE. Following this, attempt to login with an invalid password and verify this attempt fails.
Findings: PASS – The evaluator confirmed that the TOE allows a valid user with a valid password to login. When this same valid user uses an invalid password, the user is not allowed access to the TOE.

b) Test 2: The evaluator shall configure the services allowed (if any) according to the guidance documentation, and then determine the services available to an external remote entity. The evaluator shall determine that the list of services available is limited to those specified in the requirement.

High-Level Test Description
Verify there are no services available to a remote user without prior authentication. Note this is N/A to the GoSilent Cube since it only allows local administration.
Findings: PASS – The evaluator confirmed that the TOE does not allow any services to the user without prior authentication.

c) Test 3: For local access, the evaluator shall determine what services are available to a local administrator prior to logging in, and make sure this list is consistent with the requirement.

High-Level Test Description
Verify there are no services available to a local user without prior authentication for the GoSilent Server. Verify that the only service available to a local user without prior authentication is Factory Reset for GoSilent Cube.
Findings: PASS – The evaluator confirmed that the TOE does not allow any services to a local user without prior authentication for the GoSilent Server. The evaluator also confirmed that the only function available without prior authentication on the GoSilent Cube is the Factory Reset option.

d) Test 4: For distributed TOEs where not all TOE components support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the evaluator shall test that the components authenticate Security Administrators as described in the TSS.

Test Not Applicable
N/A—All components that are a part of the TOE adhere to the requirements found in FIA_UIA_EXT.1 and FIA_UAU_EXT.2.

3.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

151 Evaluation Activities for this requirement are covered under those for FIA_UIA_EXT.1. If other authentication mechanisms are specified, the evaluator shall include those methods in the activities for FIA_UIA_EXT.1.

3.3.5 FIA_UAU.7 Protected Authentication Feedback

3.3.5.1 TSS

152 None.

3.3.5.2 Guidance Documentation

153 The evaluator shall examine the guidance documentation to determine that any necessary preparatory steps to ensure authentication data is not revealed while entering for each local login allowed.

Findings:	There are no preparatory steps needed to ensure authentication data is not revealed while entering login information locally. Information is obscured by default.

3.3.5.3 Tests

154 The evaluator shall perform the following test for each method of local login allowed:

a) Test 1: The evaluator shall locally authenticate to the TOE. While making this attempt, the evaluator shall verify that at most obscured feedback is provided while entering the authentication information.

High-Level Test Description
The evaluator attempts to login to the TOE locally. While logging into the TOE, the evaluator verifies there is obscured feedback when entering authentication information. These same steps were performed for both TOE components.
Findings: PASS – The evaluator confirmed that there is obscured feedback when entering login credentials on the TOE components.

3.4 Security management (FMT)

3.4.1 General requirements for distributed TOEs

3.4.1.1 TSS

155 For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Findings:	[ST] Section 6.7 – The TSS identifies all TOE components and how every function is related to security management. All relevant aspects of each TOE components are covered by FMT_SFRs.

3.4.1.2 Guidance Documentation

156 For distributed TOEs it is required to verify the Guidance Documentation to describe management of each TOE component. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Findings:	[AGD] Section 3.1 – The AGD describes how each TOE component is managed by an administrator on separate interfaces.

3.4.1.3 Tests

157 Tests defined to verify the correct implementation of security management functions shall be performed for every TOE component. For security management functions that are implemented centrally, sampling should be applied when defining the evaluator's tests (ensuring that all components are covered by the sample).

High-Level Test Description
The evaluator shall confirm that each TOE component can be managed via all claimed management interfaces and that each management function can be exercised on the TOE.
Findings: PASS – The evaluator confirmed that each TOE component can be managed through the claimed management interfaces in [AGD] section 3.1 when completing the remaining Security Management (FMT) test activities. The evaluator also confirmed throughout all test activities that all management functions could be exercised on the TOE.

3.4.2 FMT_MOF.1/ManualUpdate

3.4.2.1 TSS

158 For distributed TOEs see chapter 2.4.1.1. There are no specific requirements for non-distributed TOEs.

Findings:	This activity refers to [NDcPP-SD] section 2.4.1.1. Refer to [AAR] section 3.4.1.1 for coverage.

3.4.2.2 Guidance Documentation

159 The evaluator shall examine the guidance documentation to determine that any necessary steps to perform manual update are described. The guidance documentation shall also provide warnings regarding functions that may cease to operate during the update (if applicable).

Findings:	[AGD] Section 2.2 provides instructions for performing a manual update by connecting to the TOE component and downloading the update via HTTPS.

160 For distributed TOEs the guidance documentation shall describe all steps how to update all TOE components. This shall contain description of the order in which components need to be updated if the order is relevant to the update process. The guidance documentation shall also provide warnings regarding functions of TOE components and the overall TOE that may cease to operate during the update (if applicable).

Findings:	[AGD] Section 2.2 provides instructions for performing a manual update by connecting to the TOE component and downloading the update via HTTPS. Section 2.2 also states that prior to the installation of a software update, the Cube will tear down any active tunnels before the update is applied to ensure the integrity of the update process. Once the update process has completed successfully on the Cube, the tunnel is reestablished.

3.4.2.3 Tests

161 The evaluator shall try to perform the update using a legitimate update image without prior authentication as Security Administrator (either by authentication as a user with no administrator privileges or without user authentication at all – depending on the configuration of the TOE). The attempt to update the TOE shall fail.

162 The evaluator shall try to perform the update with prior authentication as Security Administrator using a legitimate update image. This attempt should be successful. This test case should be covered by the tests for FPT_TUD_EXT.1 already.

High-Level Test Description
Attempt to update the TOE as an underprivileged user and verify this fails. Login as an administrator following this and verify that the admin can perform the update function.
Findings: PASS - The evaluator confirmed that the TOE does not allow a user without administrative privileges to perform update functions. The evaluator then logged on as a user with administrative privileges and confirmed that they were able to perform an update.

3.4.3 FMT_MTD.1/CoreData Management of TSF Data

3.4.3.1 TSS

163 The evaluator shall examine the TSS to determine that, for each administrative function identified in the guidance documentation; those that are accessible through an interface prior to administrator log-in are identified. For each of these functions, the evaluator shall also confirm that the TSS details how the ability to manipulate the TSF data through these interfaces is disallowed for non-administrative users.

Findings:	[ST] Section 6.7.1 – The TSS does not identify any administrative functions that are available prior to administrator login.

164 If TOE supports handling of X.509v3 certificates and implements a trust store, the evaluator shall examine the TSS to determine that it contains sufficient information to describe how the ability to manage the TOE's trust store is restricted.

Findings:	[ST] Section 6.7.1 – The TSS identifies the TOE's trust store as only being able to be managed by a logged in administrator.

3.4.3.2 Guidance Documentation

165 The evaluator shall review the guidance documentation to determine that each of the TSF-data-manipulating functions implemented in response to the requirements of the cPP is identified, and that configuration information is provided to ensure that only administrators have access to the functions.

Findings:	[AGD] Section 3.1 states how TOE administration is performed through each of the supported administrative interfaces. The evaluator confirmed that each TSF-data-manipulating function is implemented in conjunction with FMT_SMF.1.

166 If the TOE supports handling of X.509v3 certificates and provides a trust store, the evaluator shall review the guidance documentation to determine that it provides sufficient information for the administrator to configure and maintain the trust store in a secure way. If the TOE supports loading of CA certificates, the evaluator shall review the guidance documentation to determine that it provides sufficient information for the administrator to securely load CA certificates into the trust store. The evaluator shall also review the guidance documentation to determine that it explains how to designate a CA certificate a trust anchor.

Findings:	[AGD] Section 3.12 provides instructions for the administrator to configure and maintain the trust store in a secure way. The TOE supports loading of CA certificates and section 3.12 provides sufficient information for the administrator to securely load CA certificates into the trust store how to designate trust anchors.

3.4.3.3 Tests

167 No separate testing for FMT_MTD.1/CoreData is required unless one of the management functions has not already been exercised under any other SFR.

3.4.4 FMT_SMF.1 Specification of Management Functions

168 The security management functions for FMT_SMF.1 are distributed throughout the cPP and are included as part of the requirements in FTA_SSL_EXT.1, FTA_SSL.3, FTA_TAB.1, FMT_MOF.1/ManualUpdate, FMT_MOF.1/AutoUpdate (if included in the ST), FIA_AFL.1, FIA_X509_EXT.2.2 (if included in the ST), FPT_TUD_EXT.1.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-

configurable action), FMT_MOF.1/Services, and FMT_MOF.1/Functions (for all of these SFRs that are included in the ST), FMT_MTD, FPT_TST_EXT, and any cryptographic management functions specified in the reference standards. Compliance to these requirements satisfies compliance with FMT_SMF.1.

3.4.4.1 TSS (containing also requirements on Guidance Documentation and Tests)

169 The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).

Findings:	[ST] Section 6.7.1 – The evaluator examined the TSS, Guidance Documentation and the TOE and confirmed that all available management functions on the TOE and all TOE components specified in FMT_SMF.1 were identified and the TSS details which security management functions are available through the specified GoSilent Server and GoSilent Cube interface, locally and remotely.

170 The evaluator shall examine the TSS and Guidance Documentation to verify they both describe the local administrative interface. The evaluator shall ensure the Guidance Documentation includes appropriate warnings for the administrator to ensure the interface is local.

Findings:	[ST] Section 6.7.1 – The TSS identifies a local interface and that the interface is supported on both TOE components. The AGD also describes the interfaces used by the TOE in section 3.1.

171 For distributed TOEs with the option 'ability to configure the interaction between TOE components' the evaluator shall examine that the ways to configure the interaction between TOE components is detailed in the TSS and Guidance Documentation. The evaluator shall check that the TOE behaviour observed during testing of the configured SFRs is as described in the TSS and Guidance Documentation.

Findings:	[ST] Section 6.7.1 – Due to the nature of the TOE being distributed, the administrator can configure the interaction between the components. The TSS in the ST describes how these components interact with each other and what configurable options are available to the administrator. [AGD] Section 2.2 states how the Cube can be manually registered on the GoSilent Server. [AGD] Section 3.8 also provides the VPN configuration to allow the components to communicate.

3.4.4.2 Guidance Documentation

172 See section 2.4.4.1.

3.4.4.3 Tests

173 The evaluator tests management functions as part of testing the SFRs identified in section 2.4.4. No separate testing for FMT_SMF.1 is required unless one of the management functions in FMT_SMF.1.1 has not already been exercised under any other SFR.

Note
The evaluator tests management functions as part of testing the SFRs throughout the [DTR]. No separate testing for FMT_SMF.1 is required unless one of the management functions in

Note

FMT_SMF.1.1 has not already been exercised under any other SFR. The following list maps the tested management functions to the SFRs:

- Ability to administer the TOE locally and remotely
  - FTA_SSL_EXT.1
  - FTA_SSL.3
  - FTA_SSL.4
- Ability to configure the access banner
  - FTA_TAB.1
- Ability to configure the session inactivity time before session termination or locking
  - FTA_SSL.3
  - FTA_SSL_EXT.1
- Ability to update the TOE, and to verify the updates using [digital signature] capability prior to installing those updates
  - FPT_TUD_EXT.1
- Ability to configure the authentication failure parameters for FIA_AFL.1
  - FIA_AFL.1
- Ability to start and stop services
  - FMT_MOF.1/Services
- Ability to modify the behaviour of the transmission of audit data to an external IT entity
  - FTP_ITC.1
  - FIA_X509_EXT.1/Rev
  - FCS_TLSC_EXT.1
- Ability to manage the cryptographic keys
  - FIA_X509_EXT.1/Rev
  - FIA_X509_EXT.1/ITT
- Ability to configure the cryptographic functionality
  - FIA_X509_EXT.1/Rev
  - FIA_X509_EXT.1/ITT
  - FCS_TLSC_EXT.1
  - FCS_TLSS_EXT.1
  - FCS_IPSEC_EXT.1
- Ability to configure the lifetime for IPsec SAs
  - FCS_IPSEC_EXT.1.7
  - FCS_IPSEC_EXT.1.8
- Ability to configure the interaction between TOE components
  - FCO_CPC_EXT.1
  - FIA_X509_EXT.1/ITT
  - FCS_IPSEC_EXT.1
  - FPT_ITT.1
- Ability to set the time which is used for time-stamps
  - FPT_STM_EXT.1
- Ability to configure the reference identifier for the peer
  - FCS_TLSC_EXT.1.2
  - FCS_IPSEC_EXT.1.13
  - FCS_IPSEC_EXT.1.14
- Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors
  - FIA_X509_EXT.1/Rev
  - FIA_X509_EXT.1/ITT
- Ability to import X.509v3 certificates to the TOE's trust store
  - FIA_X509_EXT.1/Rev
  - FIA_X509_EXT.1/ITT

3.4.5 FMT_SMR.2 Restrictions on security roles

3.4.5.1 TSS

174 The evaluator shall examine the TSS to determine that it details the TOE supported roles and any restrictions of the roles involving administration of the TOE.

Findings:	[ST] Section 6.7.1 – The TSS identifies the only role as security administrator on the TOE. The TSS also details what kind of abilities this role has.

3.4.5.2 Guidance Documentation

175 The evaluator shall review the guidance documentation to ensure that it contains instructions for administering the TOE both locally and remotely, including any configuration that needs to be performed on the client for remote administration.

Findings:	[AGD] Section 3.1 describes in detail the ways to access both TOE components locally and remotely.

3.4.5.3 Tests

176 In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of this cPP be tested; for instance, if the TOE can be administered through a local hardware interface; SSH; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team's test activities.

Note	There are no explicit test activities and therefore none are recorded here. All interfaces are tested throughout the test activities. Refer to testing for FTA_SSL.3, FCS_TLSS_EXT.1 and FIA_UIA_EXT.1.

3.5 Protection of the TSF (FPT)

3.5.1 FPT_SKP_EXT.1 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)

3.5.1.1 TSS

177 The evaluator shall examine the TSS to determine that it details how any pre-shared keys, symmetric keys, and private keys are stored and that they are unable to be viewed through an interface designed specifically for that purpose, as outlined in the application note. If these values are not stored in plaintext, the TSS shall describe how they are protected/obscured.

Findings:	[ST] Section 6.8.1 states that the TOE does not provide access to any administrative interface that allow access to view plaintext passwords, pre-shared keys, symmetric keys or private keys. All administrative password hashes, pre-shared keys, symmetric keys, and private keys are stored in non-volatile memory at non-fixed locations that are not accessible by administrative users.

3.5.2 FPT_APW_EXT.1 Protection of Administrator Passwords

3.5.2.1 TSS

178 The evaluator shall examine the TSS to determine that it details all authentication data that are subject to this requirement, and the method used to obscure the plaintext password data when stored. The TSS shall also detail passwords are stored in such a way that they are unable to be viewed through an interface designed specifically for that purpose, as outlined in the application note.

Findings:	[ST] Section 6.8.1 states that the TOE does not provide access to any administrative interface that allows access to view plaintext passwords, pre-shared keys, symmetric keys or private keys. Additionally the TOE stores administrative passwords as SHA-256 hashes.

3.5.3 FPT_TST_EXT.1 TSF testing

3.5.3.1 TSS

179 The evaluator shall examine the TSS to ensure that it details the self-tests that are run by the TSF; this description should include an outline of what the tests are actually doing (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly.

Findings:	[ST] Section 6.8.2 – The TSS describes in detail how the self-tests are run by the TSF. The integrity is tested in the firmware of the TOE.The noise source health test performs a statistical assessment of 1000 samples. DRBG randomness test compares the current and previously generated blocks to see if they are the same. Identical blocks indicate a failure of the generation function and produce an error and an error return value. The TSS then asserts why these tests are sufficient to make sure the TSF is operating correctly.

180 For distributed TOEs the evaluator shall examine the TSS to ensure that it details which TOE component performs which self-tests and when these self-tests are run.

Findings:	[ST] Section 6.8.2 – The power-on self-tests are performed on each component of the distributed TOE.

3.5.3.2 Guidance Documentation

181 The evaluator shall also ensure that the guidance documentation describes the possible errors that may result from such tests, and actions the administrator should take in response; these possible errors shall correspond to those described in the TSS.

Findings:	[AGD] Section 3.7 describes the firmware integrity test and statistical assessment of the entropy source which execute on boot. If any of the tests fail the TOE component will not enter an operational state. Upon failure, the GSS will display an error message. Upon failure of the Cube, the onboard LED will indicate an error. This section states that diagnostic information on the GSS and the behavior of the LED's on the Cube will identify the specific failure. If any component fails, the administrator should contact ID Technologies for support.

182 For distributed TOEs the evaluator shall ensure that the guidance documentation describes how to determine from an error message returned which TOE component has failed the self-test.

Findings:	[AGD] Section 3.7 states that upon failure of the self-tests, the GSS will display an error message and the Cube's onboard LEDs will indicate an error.

3.5.3.3 Tests

183 It is expected that at least the following tests are performed:

a) Verification of the integrity of the firmware and executable software of the TOE

b) Verification of the correct operation of the cryptographic functions necessary to fulfil any of the SFRs.

184 Although formal compliance is not mandated, the self-tests performed should aim for a level of confidence comparable to:

a) [FIPS 140-2], chap. 4.9.1, Software/firmware integrity test for the verification of the integrity of the firmware and executable software. Note that the testing is not restricted to the cryptographic functions of the TOE.

b) [FIPS 140-2], chap. 4.9.1, Cryptographic algorithm test for the verification of the correct operation of cryptographic functions. Alternatively, national requirements of any CCRA member state for the security evaluation of cryptographic functions should be considered as appropriate.

185 The evaluator shall either verify that the self-tests described above are carried out during initial start-up or that the developer has justified any deviation from this.

High-Level Test Description
Reset the TOE and witness that the startup includes an indicator that self-tests were executed and passed permitting the device to operate.
Findings: PASS – The evaluator confirmed that the TOE runs, passes, and logs the self-checks during initialization.

186 For distributed TOEs the evaluator shall perform testing of self-tests on all TOE components according to the description in the TSS about which self-test are performed by which component.

High-Level Test Description
Reset the TOE and witness that the startup includes an indicator that self-tests were executed and passed permitting the device to operate.
Findings: PASS – The evaluator confirmed in conjunction with Test 1 that the TOE runs, passes, and logs the self-checks during initialization on all TOE components.

3.5.4 FPT_TUD_EXT.1 Trusted Update

3.5.4.1 TSS

187 The evaluator shall verify that the TSS describe how to query the currently active version. If a trusted update can be installed on the TOE with a delayed activation, the

TSS needs to describe how and when the inactive version becomes active. The evaluator shall verify this description.

Findings:	[ST] Section 6.8.4 – The TSS describes how a user can query the currently active version running on the TOE through the administrative GUI.

188 The evaluator shall verify that the TSS describes all TSF software update mechanisms for updating the system firmware and software (for simplicity the term 'software' will be used in the following although the requirements apply to firmware and software). The evaluator shall verify that the description includes a digital signature verification of the software before installation and that installation fails if the verification fails. Alternatively, an approach using a published hash can be used. In this case the TSS shall detail this mechanism instead of the digital signature verification mechanism. The evaluator shall verify that the TSS describes the method by which the digital signature or published hash is verified to include how the candidate updates are obtained, the processing associated with verifying the digital signature or published hash of the update, and the actions that take place for both successful and unsuccessful signature verification or published hash verification.

Findings:	[ST] Section 6.8.4 – The TSS describes in detail how the update mechanism on the TOE functions. The TSS description identifies the method of update checking as signature based. If the signature verification fails, the TOE does not update the image. Upon successful verification and installation of the update, the GSS will automatically reboot and the update takes effect. For both components, if the update is unsuccessful the TOE will not proceed with the installation and the currently installed software will continue executing.

189 If the options ‘support automatic checking for updates’ or ‘support automatic updates’ are chosen from the selection in FPT_TUD_EXT.1.2, the evaluator shall verify that the TSS explains what actions are involved in automatic checking or automatic updating by the TOE, respectively.

Findings:	The TOE does not support automatic updates or automatic checking for updates.

190 For distributed TOEs, the evaluator shall examine the TSS to ensure that it describes how all TOE components are updated, that it describes all mechanisms that support continuous proper functioning of the TOE during update (when applying updates separately to individual TOE components) and how verification of the signature or checksum is performed for each TOE component. Alternatively, this description can be provided in the guidance documentation. In that case the evaluator should examine the guidance documentation instead.

Findings:	[ST] Section 6.8.4 – The TSS describes how the different components of the TOE perform updates. Updates are downloaded from a specified URI using HTTPS. Administrators can manually load the update file on the GoSilent Server. An option is provided for administrators to download GoSilent Cube updates to itself from the ID Technologies update server. For both components, if the update is unsuccessful the TOE will not proceed with the installation and the currently installed software will continue executing. To ensure integrity of the GoSilent Cube during the software update, the Cube will tear down any active tunnels during the update process. Once the update is successful, it will reestablish the tunnel with the GoSilent Server.

191 If a published hash is used to protect the trusted update mechanism, then the evaluator shall verify that the trusted update mechanism does involve an active authorization step of the Security Administrator, and that download of the published hash value, hash comparison and update is not a fully automated process involving no active authorization by the Security Administrator. In particular, authentication as Security Administration according to FMT_MOF.1/ManualUpdate needs to be part of the update process when using published hashes.

Findings:	The TOE does not support published hash to protect the trusted update mechanism.

3.5.4.2 Guidance Documentation

192 The evaluator shall verify that the guidance documentation describes how to query the currently active version. If a trusted update can be installed on the TOE with a delayed activation, the guidance documentation needs to describe how to query the loaded but inactive version.

Findings:	[AGD] Section 2.2 describes how to query the current version of the TOE. The TOE does not support delayed activation.

193 The evaluator shall verify that the guidance documentation describes how the verification of the authenticity of the update is performed (digital signature verification or verification of published hash). The description shall include the procedures for successful and unsuccessful verification. The description shall correspond to the description in the TSS.

Findings:	[AGD] Section 2.2 states that the update is verified by the TOE via digital signature prior to installation. This section also states that in the event of a signature check or package upload failure, both TOE components will halt installation and continue executing the current version. Section 2.2 also states that the GoSilent Cube will tear down any active tunnels before an update is successfully applied. Once the update completes, the GoSilent Cube reestablishes the tunnel. Upon successful update of the GoSilent Server, the server will reboot itself and the update to take effect.

194 If a published hash is used to protect the trusted update mechanism, the evaluator shall verify that the guidance documentation describes how the Security Administrator can obtain authentic published hash values for the updates.

Findings:	N/A – The TOE only supports digital signature.

195 For distributed TOEs the evaluator shall verify that the guidance documentation describes how the versions of individual TOE components are determined for FPT_TUD_EXT.1, how all TOE components are updated, and the error conditions that may arise from checking or applying the update (e.g. failure of signature verification, or exceeding available storage space) along with appropriate recovery actions. . The guidance documentation only has to describe the procedures relevant for the Security Administrator; it does not need to give information about the internal communication that takes place when applying updates.

Findings:	[AGD] Section 2.2 describes how to query the current version of each TOE component and how the TOE components are updated. This section also states that in the event of a signature check or package upload failure, both TOE components will halt installation and continue executing the current version. In the event of subsequent failures, ID Technologies should be contacted for support.

196 If this was information was not provided in the TSS: For distributed TOEs, the evaluator shall examine the Guidance Documentation to ensure that it describes how all TOE components are updated, that it describes all mechanisms that support continuous proper functioning of the TOE during update (when applying updates separately to individual TOE components) and how verification of the signature or checksum is performed for each TOE component.

Findings:	[AGD] Section 2.2 describes how both components of the TOE are updated. This section also notes that the Cube will tear down any active tunnels prior to applying

	the update and once the update has completed, the Cube will re-establish the tunnel. Once the update on the GSS successfully installs, the server must be restarted.

197 If this was information was not provided in the TSS: If the ST author indicates that a certificate-based mechanism is used for software update digital signature verification, the evaluator shall verify that the Guidance Documentation contains a description of how the certificates are contained on the device. The evaluator also ensures that the Guidance Documentation describes how the certificates are installed/updated/selected, if necessary.

Findings:	N/A – The TOE does not support certificate-based updates.

3.5.4.3 Tests

198 The evaluator shall perform the following tests:

a) Test 1: The evaluator performs the version verification activity to determine the current version of the product. If a trusted update can be installed on the TOE with a delayed activation, the evaluator shall also query the most recently installed version (for this test the TOE shall be in a state where these two versions match). The evaluator obtains a legitimate update using procedures described in the guidance documentation and verifies that it is successfully installed on the TOE. For some TOEs loading the update onto the TOE and activation of the update are separate steps ('activation' could be performed e.g. by a distinct activation step or by rebooting the device). In that case the evaluator verifies after loading the update onto the TOE but before activation of the update that the current version of the product did not change but the most recently installed version has changed to the new product version. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update and that current version of the product and most recently installed version match again.

High-Level Test Description
Perform an update on the TOE and verify that after the update occurs, the version changes to the version seen in the update file.
Findings: PASS – The evaluator confirmed that the TOE properly updates after being provided with a valid update file.

b) Test 2 [conditional]: If the TOE itself verifies a digital signature to authorize the installation of an image to update the TOE the following test shall be performed (otherwise the test shall be omitted). The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. The evaluator obtains or produces illegitimate updates as defined below and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

1) A modified version (e.g. using a hex editor) of a legitimately signed update

2) An image that has not been signed

3) An image signed with an invalid signature (e.g. by using a different key as expected for creating the signature or by manual modification of a legitimate signature)

4) If the TOE allows a delayed activation of updates the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs depending on the point in time when an attempted update is rejected. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

High-Level Test Description
Provide an update file to the TOE that contains an invalid signature, unsigned image, and modified image. Verify that the TOE rejects all of these update files.
Findings: PASS – The evaluator confirmed that the TOE rejects all invalid update attempts.

c) Test 3 [conditional]: If the TOE itself verifies a hash value over an image against a published hash value (i.e. reference value) that has been imported to the TOE from outside such that the TOE itself authorizes the installation of an image to update the TOE, the following test shall be performed (otherwise the test shall be omitted. If the published hash is provided to the TOE by the Security Administrator and the verification of the hash value over the update file(s) against the published hash is performed by the TOE, then the evaluator shall perform the following tests. The evaluator first confirms that no update is pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.

1) The evaluator obtains or produces an illegitimate update such that the hash of the update does not match the published hash. The evaluator provides the published hash value to the TOE and calculates the hash of the update either on the TOE itself (if that functionality is provided by the TOE), or else outside the TOE. The evaluator confirms that the hash values are different, and attempts to install the update on the TOE, verifying that this fails because of the difference in hash values (and that the failure is logged). Depending on the implementation of the TOE, the TOE might not allow the Security Administrator to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE

2) The evaluator uses a legitimate update and tries to perform verification of the hash value without providing the published hash value to the TOE. The evaluator confirms that this attempt fails. The evaluator confirms that this attempt fails. Depending on the implementation of the TOE it might not be possible to attempt the verification of the hash value without providing a hash value to the TOE, e.g. if the hash value needs to be handed over to the TOE as a parameter in a command line message and the syntax check of the command prevents the execution of the command without providing

a hash value. In that case the mechanism that prevents the execution of this check shall be tested accordingly, e.g. that the syntax check rejects the command without providing a hash value, and the rejection of the attempt is regarded as sufficient verification of the correct behaviour of the TOE in failing to verify the hash. The evaluator then attempts to install the update on the TOE (in spite of the unsuccessful hash verification) and confirms that this fails. Depending on the implementation of the TOE, the TOE might not allow to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE

3) If the TOE allows delayed activation of updates, the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs. Depending on the point in time when the attempted update is rejected, the most recently installed version might or might not be updated. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

Test Not Applicable
Findings: N/A—The TOE does not support published hash for trusted update.

199 If the verification of the hash value over the update file(s) against the published hash is not performed by the TOE, Test 3 shall be skipped.

200 The evaluator shall perform Test 1, Test 2 and Test 3 (if applicable) for all methods supported (manual updates, automatic checking for updates, automatic updates).

Test Not Applicable
Findings: N/A—The TOE only supports manual updates which are tested in Test 1 and Test 2 above. Test 3 is not applicable.

201 For distributed TOEs the evaluator shall perform Test 1, Test 2 and Test 3 (if applicable) for all TOE components.

High-Level Test Description
Test 1 and Test 2 are applied to all TOE components. Test 3 is not applicable.
Findings: PASS – The evaluator performed testing for Test 1 and Test 2 above on all TOE components.

3.5.5 FPT_STM_EXT.1 Reliable Time Stamps

3.5.5.1 TSS

NIAP TD0632

202 The evaluator shall examine the TSS to ensure that it lists each security function that makes use of time, and that it provides a description of how the time is maintained and considered reliable in the context of each of the time related functions.

Findings:	[ST] Section 6.8.5 states that the GoSilent Server and GoSilent Cube maintain a system clock to provide date and time information for reliable timestamping for audit logs and certificate expiration. The time is considered reliable since it can only be manually set by authorized administrators. The GoSilent Server also allows the time to be set via the underlying virtualization system. Section 6.8.5 also states that time information obtained from the underlying virtualization platform is considered to be reliable because access to the virtualization platform, including the time setting function, is restricted to authorized administrators of the platform only.

203 If "obtain time from the underlying virtualization system" is selected, the evaluator shall examine the TSS to ensure that it identifies the VS interface the TOE uses to obtain time. If there is a delay between updates to the time on the VS and updating the time on the TOE, the TSS shall identify the maximum possible delay.

Findings:	[ST] Section 6.8.5 describes "When a VM is started, ESXi provides the VM with the ESXi's internal time value. Upon startup, the GSS sets its internal clock to that time from ESXi. Subsequently, the GSS internal clock is used and subsequent synchronization with the ESXi clock is not performed. There is a delay in updating the clock until the next GoSilent Server restart."

3.5.5.2 Guidance Documentation

NIAP TD0632

204 The evaluator examines the guidance documentation to ensure it instructs the administrator how to set the time. If the TOE supports the use of an NTP server, the guidance documentation instructs how a communication path is established between the TOE and the NTP server, and any configuration of the NTP client on the TOE to support this communication.

Findings:	[AGD] Section 3.2 – The AGD provides a description on how to configure the date/time manually on the GoSilent Cube and GoSilent Server. This section also states that the GoSilent Server receives time information from the virtualization platform.

205 If the TOE supports obtaining time from the underlying VS, the evaluator shall verify the Guidance Documentation specifies any configuration steps necessary. If no configuration is necessary, no statement is necessary in the Guidance Documentation. If there is a delay between updates to the time on the VS and updating the time on the TOE, the evaluator shall ensure the Guidance Documentation informs the administrator of the maximum possible delay.

Findings:	[AGD] Section 3.2 – The AGD does not detail any steps necessary to configure the TOE to obtain time from the underlying VS. The GoSlient Server sets its internal clock from the underlying VS upon startup. The maximum possible delay is the next restart.

3.5.5.3 Tests

206 The evaluator shall perform the following tests:

a) Test 1: If the TOE supports direct setting of the time by the Security Administrator then the evaluator uses the guidance documentation to set the time. The evaluator shall then use an available interface to observe that the time was set correctly.

High-Level Test Description
The evaluator shall enable the manually setting of the UTC date and time and confirm it is set correctly.
Findings: PASS – The evaluator confirmed that the TOE accepts the time changes.

b) Test 2: If the TOE supports the use of an NTP server; the evaluator shall use the guidance documentation to configure the NTP client on the TOE and set up a communication path with the NTP server. The evaluator will observe that the NTP server has set the time to what is expected. If the TOE supports multiple protocols for establishing a connection with the NTP server, the evaluator shall perform this test using each supported protocol claimed in the guidance documentation.

Test Not Applicable
Findings: N/A – The TOE does not support NTP servers

NIAP TD0632

c) Test 3: [conditional] If the TOE obtains time from the underlying VS, the evaluator shall record the time on the TOE, modify the time on the underlying VS, and verify the modified time is reflected by the TOE. If there is a delay between the setting the time on the VS and when the time is reflected on the TOE, the evaluator shall ensure this delay is consistent with the TSS and Guidance.

High-Level Test Description
The evaluator will change the time to a time that does not match the time used by the underlying virtual system. The evaluator will then change the time on the underlying virtual system to be provided to the TOE.
Findings: Pass – The evaluator confirmed that the GoSilent Server can successfully retrieve time from the underlying VS. Due to the GoSilent Cube not running on an underlying VS, this test is N/A for that TOE component.

207 If the audit component of the TOE consists of several parts with independent time information, then the evaluator shall verify that the time information between the different parts are either synchronized or that it is possible for all audit information to relate the time information of the different part to one base information unambiguously.

Test Not Applicable
Findings: Each TOE component consists of its own independent time source. The administrator is expected to ensure that the time is consistent between all TOE components. The evaluator confirmed in conjunction with FAU_GEN.1 that each auditable event that the individual TOE component appends its current time information to every audit event. This ensures that the audit information can be traced to the time source unambiguously.

3.6 TOE Access (FTA)

3.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

3.6.1.1 TSS

208 The evaluator shall examine the TSS to determine that it details whether local administrative session locking or termination is supported and the related inactivity time period settings.

Findings:	[ST] Section 6.9.1 – The TSS describes that there is an inactivity timeout period that can be configured for both local and remote sessions. After the time period elapses, the user will then have to log back into their respective interface.

3.6.1.2 Guidance Documentation

209 The evaluator shall confirm that the guidance documentation states whether local administrative session locking or termination is supported and instructions for configuring the inactivity time period.

Findings:	[AGD] Section 3.4.1 – The AGD details where to find the setting to configure the inactivity period for termination of local administration sessions for the GoSilent Server and GoSilent Cube.

3.6.1.3 Tests

210 The evaluator shall perform the following test:

a) Test 1: The evaluator follows the guidance documentation to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator establishes a local interactive session with the TOE. The evaluator then observes that the session is either locked or terminated after the configured time period. If locking was selected from the component, the evaluator then ensures that re-authentication is needed when trying to unlock the session.

High-Level Test Description
The administrator logs into the TOE via local connection and verifies that the user is auto logged out after configured timeout intervals.
Findings: PASS – The evaluator confirmed that TOE automatically logs out inactive users after the configured time elapses.

3.6.2 FTA_SSL.3 TSF-initiated Termination

3.6.2.1 TSS

211 The evaluator shall examine the TSS to determine that it details the administrative remote session termination and the related inactivity time period.

Findings:	[ST] Section 6.9.1 details that once the administrator configured period of inactivity is reached, the session is automatically terminated and re-authentication is required to gain access to the TOE functionality.

3.6.2.2 Guidance Documentation

212 The evaluator shall confirm that the guidance documentation includes instructions for configuring the inactivity time period for remote administrative session termination.

Findings:	[AGD] Section 3.4.1 – The AGD details where to find the setting to configure the inactivity period for termination of remote administration sessions for the GSS. The GoSilent Cube only supports local administration.

3.6.2.3 Tests

213 For each method of remote administration, the evaluator shall perform the following test:

a) Test 1: The evaluator follows the guidance documentation to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator establishes a remote interactive session with the TOE. The evaluator then observes that the session is terminated after the configured time period.

High-Level Test Description
The administrator logs into the TOE via remote connection and verifies that the user is auto logged out after configured timeout intervals.
Findings: PASS – The evaluator confirmed that the user was logged out due to inactivity on all components.

3.6.3 FTA_SSL.4 User-initiated Termination

3.6.3.1 TSS

214 The evaluator shall examine the TSS to determine that it details how the local and remote administrative sessions are terminated.

Findings:	[ST] Section 6.9.1 – The TSS describes how the local and remote admin sessions are terminated by logging out.

3.6.3.2 Guidance Documentation

215 The evaluator shall confirm that the guidance documentation states how to terminate a local or remote interactive session.

Findings:	[AGD] Section 3.1 states that administrative sessions are terminated by manual log out via the ‘Logout’ option in the GUI.

3.6.3.3 Tests

216 For each method of remote administration, the evaluator shall perform the following tests:

a) Test 1: The evaluator initiates an interactive local session with the TOE. The evaluator then follows the guidance documentation to exit or log off the session and observes that the session has been terminated.

High-Level Test Description
The evaluator logs into the TOE from a local session. The evaluator then logs off from the TOE and verifies the session has been terminated. These same steps were then performed on both TOE components.
Findings: PASS – The evaluator confirms that the TOE successfully logs out local users when they click “logout”.

b) Test 2: The evaluator initiates an interactive remote session with the TOE. The evaluator then follows the guidance documentation to exit or log off the session and observes that the session has been terminated.

High-Level Test Description
The evaluator logs into the TOE from a remote session. The evaluator then logs off from the TOE and verifies the session was terminated. These same steps were then performed on both TOE components.
Findings: PASS – The evaluator confirmed that the TOE successfully logs out remote user when they click “logout”.

3.6.4 FTA_TAB.1 Default TOE Access Banners

3.6.4.1 TSS

217 The evaluator shall check the TSS to ensure that it details each administrative method of access (local and remote) available to the Security Administrator (e.g., serial port, SSH, HTTPS). The evaluator shall check the TSS to ensure that all administrative methods of access available to the Security Administrator are listed and that the TSS states that the TOE is displaying an advisory notice and a consent warning message for each administrative method of access. The advisory notice and the consent warning message might be different for different administrative methods of access, and might be configured during initial configuration (e.g. via configuration file).

Findings:	[ST] Section 6.9.2 states that the TOE displays an administrator configurable message to local and remote users via the GUI prior to login. The Cube also displays an administrator configurable message prior to local login via the Cube GUI. This message is displayed as a consent banner to the administrator warning that proceeding with authentication is consenting to the terms of use of the TOE. The user is then prompted to enter their username and password.

3.6.4.2 Guidance Documentation

218 The evaluator shall check the guidance documentation to ensure that it describes how to configure the banner message.

Findings:	[AGD] Section 3.5 – The AGD provides a description on how to configure the login banner and the login banner text.

3.6.4.3 Tests

219 The evaluator shall also perform the following test:

a) Test 1: The evaluator follows the guidance documentation to configure a notice and consent warning message. The evaluator shall then, for each method of access specified in the TSS, establish a session with the TOE. The evaluator shall verify that the notice and consent warning message is displayed in each instance.

High-Level Test Description
The evaluator configured a warning message on the GoSilent Server and GoSilent Cube. The evaluator then verified that when this setting is configured, the TOE logs the configuration action. The evaluator then verifies that the banner is shown when logging into the Go Silent Server and GoSilent Cube.
Findings: PASS – The evaluator confirmed that the TOE successfully displays a banner after being configured by a security administrator. This was confirmed for local and remote administration of the GoSilent Server and local administration of the GoSilent Cube.

3.7 Trusted path/channels (FTP)

3.7.1 FTP_ITC.1 Inter-TSF trusted channel

3.7.1.1 TSS

220 The evaluator shall examine the TSS to determine that, for all communications with authorized IT entities identified in the requirement, each secure communication mechanism is identified in terms of the allowed protocols for that IT entity, whether the TOE acts as a server or a client, and the method of assured identification of the non-TSF endpoint. The evaluator shall also confirm that all secure communication mechanisms are described in sufficient detail to allow the evaluator to match them to the cryptographic protocol Security Functional Requirements listed in the ST.

Findings:	[ST] Section 6.10.1 – The TSS identifies all secure channels used for connections on the TOE including the connection between distributed components. GoSilent Server and GoSilent Cube initiate TLS connections to a syslog server. GoSilent Cube initiates the IPsec connection between itself and the GoSilent Server. [ST] Section 6.6.2 – The TSS describes that the TOE uses X.509 certificates to authenticate the peers in the IPsec connection between GoSilent Server and GoSilent Cube devices, and to authenticate the identity of the syslog server in a TLS connection.

3.7.1.2 Guidance Documentation

221 The evaluator shall confirm that the guidance documentation contains instructions for establishing the allowed protocols with each authorized IT entity, and that it contains recovery instructions should a connection be unintentionally broken.

Findings:	[AGD] Section 3.10 discusses how a trusted communication channel can be configured on the TOE to a remote audit server using TLS. Section 3.10 states that if the connection is unintentionally broken, each TOE component will continue to log locally until the connection is re-established. Each TOE component attempts to re-establish the TLS channel automatically every five minutes with no administrator intervention required.

3.7.1.3 Tests

222 The developer shall provide to the evaluator application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement. This information should be sufficiently detailed to allow the evaluator to determine the application layer timeout settings for each cryptographic protocol. There is no expectation that this information must be recorded in any public-facing document or report.

223 The evaluator shall perform the following tests:

a) Test 1: The evaluators shall ensure that communications using each protocol with each authorized IT entity is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

High-Level Test Description
Establish a secure TLS session to the syslog server from the GoSilent Server and GoSilent Cube, respectively, and ensure the TLS communications are successful.
Findings: PASS – This test is tested as part of FCS_TLSC_EXT.1.1 Test 1.

b) Test 2: For each protocol that the TOE can initiate as defined in the requirement, the evaluator shall follow the guidance documentation to ensure that in fact the communication channel can be initiated from the TOE.

High-Level Test Description
Ensure that communication is initiated by the TOE for TLS syslog connections and traffic is not sent in plaintext.
Findings: PASS – This test is tested as part of FCS_TLSC_EXT.1.1 Test 1.

c) Test 3: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

High-Level Test Description
Ensure that communication is initiated by the TOE for TLS syslog connections and traffic is not sent in plaintext.
Findings: PASS – The TOE maintains trusted channels to the syslog server, which are set up as per the evaluated configuration. It is constantly tested throughout the evaluation. FCS_TLSC_EXT.1 Test 1 shows the communication between the TOE and syslog server using a TLS protected channel.

d) Test 4: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route to the external IT entities.

The evaluator shall, for each instance where the TOE acts as a client utilizing a secure communication mechanism with a distinct IT entity, physically interrupt the connection of that IT entity for the following durations: i) a duration that exceeds the TOE's application layer timeout setting, ii) a duration shorter than the application layer timeout but of sufficient length to interrupt the network link layer.

The evaluator shall ensure that, when the physical connectivity is restored, communications are appropriately protected and no TSF data is sent in plaintext.

In the case where the TOE is able to detect when the cable is removed from the device, another physical network device (e.g. a core switch) shall be used to interrupt the connection between the TOE and the distinct IT entity. The interruption shall not be performed at the virtual node (e.g. virtual switch) and must be physical in nature.

High-Level Test Description
The evaluator will test two different lengths of time for disconnect using the TLS protocol and verify communication is appropriately restored after interruption.
Findings: PASS – The evaluator confirmed that the TOE restores secure communications properly after interruption.

224 Further assurance activities are associated with the specific protocols.

225 For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of external secure channels to TOE components in the Security Target.

High-Level Test Description
Tests 1, 2, 3 and 4 are applied to all TOE components.
Findings: PASS – Refer to above. Test steps were performed on all TOE components.

226 The developer shall provide to the evaluator application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement. This information should be sufficiently detailed to allow the evaluator to determine the application layer timeout settings for each cryptographic protocol. There is no expectation that this information must be recorded in any public- facing document or report.

3.7.2 FTP_TRP.1/Admin Trusted Path

3.7.2.1 TSS

227 The evaluator shall examine the TSS to determine that the methods of remote TOE administration are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

Findings:	[ST] Section 6.10.2 – The TSS identifies that HTTPS is used as the secure protocol to connect the administrator to the Administrative GUI.

3.7.2.2 Guidance Documentation

228 The evaluator shall confirm that the guidance documentation contains instructions for establishing the remote administrative sessions for each supported method.

Findings:	[AGD] Section 3.1 – The AGD provides a description on how to establish a remote admin session to the TOE for both the GoSilent Server and the GoSilent Cube.

3.7.2.3 Tests

229 The evaluator shall perform the following tests:

a) Test 1: The evaluators shall ensure that communications using each specified (in the guidance documentation) remote administration method is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

High-Level Test Description
The evaluator establishes a secure path with the TOE using HTTPS and traffic is not sent in plaintext.
Findings: PASS – This is tested throughout the course of the evaluation. Refer to evidence found in FCS_TLSS_EXT.1 Test 1. The administration method is constantly tested throughout the evaluation via FTA_SSL.4, FCS_TLSS_EXT.1 and FTA_SSL.3 testing.

b) Test 2: The evaluator shall ensure, for each communication channel, the channel data is not sent in plaintext.

High-Level Test Description
The evaluator establishes a secure path with the TOE using HTTPS and traffic is not sent in plaintext.
Findings: PASS – This is tested throughout the course of the evaluation. Refer to evidence found in FCS_TLSS_EXT.1 Test 1. The administration method is constantly tested throughout the evaluation via FTA_SSL.4, FCS_TLSS_EXT.1 and FTA_SSL.3 testing.

230 Further assurance activities are associated with the specific protocols.

231 For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of trusted paths to TOE components in the Security Target.

High-Level Test Description
The evaluator applies test 1 and 2 above to all TOE components.
Findings: PASS – Tests 1 and 2 above were performed on all TOE components.

4 Evaluation Activities for VPN Gateway PP-Module

4.1 Security Audit (FAU)

4.1.1 FAU_GEN.1/VPN Audit Data Generation (VPN Gateway)

4.1.1.1 TSS

232 The evaluator shall examine the TSS to verify that it describes the audit mechanisms that the TOE uses to generate audit records for VPN gateway behavior. If any audit mechanisms the TSF uses for this are not used to generate audit records for events defined by FAU_GEN.1 in the Base-PP, the evaluator shall ensure that any VPN gateway-specific audit mechanisms also meet the relevant functional claims from the Base-PP. For example, FAU_STG_EXT.1 requires all audit records to be transmitted to the OE over a trusted channel. This includes the audit records that are required by FAU_GEN.1/VPN. Therefore, if the TOE has an audit mechanism that is only used for VPN gateway functionality, the evaluator shall ensure that the VPN gateway related audit records meet this requirement, even if the mechanism used to generate these audit records does not apply to any of the auditable events defined in the Base-PP.

Findings:	[ST] Section 6.1.1 – The TSS describes in detail how the TOE logs events and what information is provided with each audit record that is generated. All audit records are stored on the TOE or sent to an audit server using a secure protocol as defined in FTP_ITC.1.

4.1.1.2 Guidance

233 The evaluator shall examine the operational guidance to verify that it identifies all security-relevant auditable events claimed in the ST and includes sample records of each event type. If the TOE uses multiple audit mechanisms to generate different sets of records, the evaluator shall verify that the operational guidance identifies the audit records that are associated with each of the mechanisms such that the source of each audit record type is clear.

Findings:	[AGD] Section 4.2 – The operational guidance identifies all security-relevant auditable events claimed in the ST and includes sample records of each event type in Table 5. The operational guidance identifies the audit records that are associated with each of the TOE component mechanisms such that the source of each audit record type is clear.

4.1.1.3 Tests

234 The evaluator shall test the audit functionality by performing actions that trigger each of the claimed audit events and verifying that the audit records are accurate and that their format is consistent with what is specified in the operational guidance. The evaluator may generate these audit events as a consequence of performing other tests that would cause these events to be generated.

High-Level Test Description
The evaluator performed the testing in conjunction with the testing of the security mechanisms directly. The evaluator confirmed that the audit records are accurate and their format is consistent with what is specified in the operational guidance.
Findings: PASS – The evaluator confirmed all audit events were generated by the TOE in conjunction with FAU_GEN.1.

4.2 Cryptographic Support (FCS)

4.2.1 FCS_CKM.1/IKE Cryptographic Key Generation (for IKE Peer Authentication)

4.2.1.1 TSS

235 The evaluator shall check to ensure that the TSS describes how the key-pairs are generated.

Findings:	[ST] Section 6.3.1 – The TSS describes how the TOE generates ECDSA P-256 and P-384 Elliptic Curve keys as specified in FIPS Pub 186-4 “Digital Signature Standard (DSS)” Appendix B.4. In addition, the TOE also generates keys with FFC schemes using safe-prime groups that meet NIST SP 800-56A Revision 3 and per RFC3526 for DH14. ECDSA key pairs for X509 certificates can also be generated through the TOE GUI where the option of keys using P-256 or P-384 curves can be selected.

236 In order to show that the TSF implementation complies with FIPS PUB 186-4, the evaluator shall ensure that the TSS contains the following information:

- The TSS shall list all sections of Appendix B to which the TOE complies.
- For each applicable section listed in the TSS, for all statements that are not “shall” (that is, “shall not”, “should”, and “should not”), if the TOE implements such options it shall be described in the TSS. If the included functionality is indicated as “shall not” or “should not” in the standard, the TSS shall provide a rationale for why this will not adversely affect the security policy implemented by the TOE;
- For each applicable section of Appendix B, any omission of functionality related to “shall” or “should” statements shall be described;

Findings:	[ST] Section 6.3.1 states that the TOE complies with Appendix section B.4 and implements all “shall” and “should” statements and does not implement any “shall not” and “should not” statements.

237 Any TOE-specific extensions, processing that is not included in the Appendices, or alternative implementations allowed by the Appendices that may impact the security requirements the TOE is to enforce shall be described.

Findings:	[ST] Section 6.3.1 states that the TOE also generates keys with FFC schemes using safe-prime groups that meet NIST SP 800-56A Revision 3 and per RFC3526 for DH14.

4.2.1.2 Guidance

238 The evaluator shall check that the operational guidance describes how the key generation functionality is invoked, and describes the inputs and outputs associated

with the process for each signature scheme supported. The evaluator shall also check that guidance is provided regarding the format and location of the output of the key generation process.

Findings:	[AGD] Section 3.11 specifies that the TOE performs cryptographic key generation services for IKE peer authentication using ECDSA P-256 and P-384 Elliptic curve keys and FFC schemes using safe-prime groups per RFC3526 for DH group 14. [AGD] Section 3.8 provides instructions for invoking the algorithms that support the key generation functionality. [AGD] Section 3.12 provides instructions for generating ECDSA keys for x509 certificates used in IPsec authentication. When generating certificate signing requests, the option is given to display the CSR information on screen to copy or can be downloaded in PEM format. Keys generated during this process are stored on the TOE in non-volatile memory and are not accessible by any interface.

4.2.1.3 Test

239 ***For FFC Schemes using "safe-prime" groups:***

240 Testing for FFC Schemes using safe-prime groups is done as part of testing in FCS_CKM.2.

241 ***For all other selections:***

242 The evaluator shall perform the corresponding tests for FCS_CKM.1 specified in the NDcPP SD, based on the selections chosen for this SFR. If IKE key generation is implemented by a different algorithm than the NDcPP key generation function, the evaluator shall ensure this testing is performed using the correct implementation**.**

Findings:	FFC EC key generation is covered by the following CAVP certificate all of which claim KAS ECC Component for NIST curves P-256 and P-384: C1159 and C1104. These claims are consistent with FCS_CKM.1 and FCS_CKM.2 in the [ST] Diffie-Hellman group 14 is covered by protocol testing in FCS_IPSEC_EXT.1.11 Test 1.

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

243 There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to require the ST author to make certain selections, but these selections are all part of the original definition of the SFR so no new behavior is defined by the PP-Module.

4.2.3 FCS_IPSEC_EXT.1 Ipsec Protocol

4.2.3.1 TSS

244 All existing activities regarding "Pre-shared keys" apply to all selections including pre-shared keys. If any selection with "Pre-shared keys" is included, the evaluator shall check to ensure that the TSS describes how the selection works in conjunction with the authentication of Ipsec connections.

Findings:	The TOE does not claim use of pre-shared keys in IPsec connections.

4.2.3.2 Guidance

245 If any selection with “Pre-shared Keys” is selected, the evaluator shall check that the operational guidance describes any configuration necessary to enable any selected authentication mechanisms.

Findings:	The TOE does not claim use of pre-shared keys in Ipsec connections.

4.2.3.3 Tests

246 There are no additional testing activities.

4.3 Security management (FMT)

4.3.1 FMT_SMF.1/VPN Specification of Management Functions

4.3.1.1 TSS

247 The evaluator shall examine the TSS to confirm that all management functions specified in FMT_SMF.1/VPN are provided by the TOE. As with FMT_SMF.1 in the Base-PP, the evaluator shall ensure that the TSS identifies what logical interfaces are used to perform these functions and that this includes a description of the local administrative interface.

Findings:	[ST] Section 6.7.1 – The TSS identifies all available management functions on all the TOE components. The TSS also describes how VPN management functions are controlled by the administrator on the TOE.

4.3.1.2 Guidance

248 The evaluator shall examine the operational guidance to confirm that all management functions specified in FMT_SMF.1/VPN are provided by the TOE. As with FMT_SMF.1 in the Base-PP, the evaluator shall ensure that the operational guidance identifies what logical interfaces are used to perform these functions and that this includes a description of the local administrative interface.

Findings:	[AGD] Section 3.1 states how TOE administration is performed through each of the supported administrative interfaces. [AGD] Section 3.3 states how packet filtering rules are defined, how they can be associated with network interfaces and if any packet matches any rule, the rule action will be applied to that packet.

4.3.1.3 Tests

249 The evaluator tests management functions as part of performing other test EAs. No separate testing for FMT_SMF.1/VPN is required unless one of the management functions in FMT_SMF.1.1/VPN has not already been exercised under any other SFR.

High-Level Test Description
The evaluator confirms that all management functions are exercised in conjunction with FMT_SMF.1.1/VPN testing.
Findings: PASS – The evaluator confirmed that all management functions within FMT_SMF.1.1/VPN are satisfied as part of testing in VPNGW SFRs.

4.4 Packet Filtering (FPF)

4.4.1 FPF_RUL_EXT.1 Rules for Packet Filtering

4.4.1.1 FPF_RUL_EXT.1.1

4.4.1.1.1 TSS

250 The evaluator shall verify that the TSS provide a description of the TOE's initialization and startup process, which clearly indicates where processing of network packets begins to take place, and provides a discussion that supports the assertion that packets cannot flow during this process.

Findings:	[ST] Section 6.5.1 – The TSS provides details on how network processing is handled during the boot sequence to ensure the packet filtering rules cannot be bypassed. The boot sequence for the TOE includes: - BIOS hardware and memory checks - Loading and initialization of the OS - Self-tests including firmware integrity tests are executed - The init utility is started (mounts file systems, sets up network cards, and generally starts all the processes that usually are run on the system at startup) - Daemon programs such as Internet Service Daemon (INETD), Syslogd are started; Routing and forwarding tables are initialized - Application daemons are loaded, enabling access to the GoSilent Server management GUI - Physical interfaces are active Only when the interfaces are fully brought up successfully will the packets start to be received and sent. The TSS also identifies how the TOE applies traffic policies to safeguard against packets flowing through the TOE without any policies active.

251 The evaluator shall verify that the TSS also includes a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describes the safeguards that would prevent packets flowing through the TOE without applying the ruleset in the event of a component failure. This could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets.

Findings:	[ST] Section 6.5.1 states in the event of a component failure, such as the firewall failing to initialize, the TOE will not enter a state where network traffic is able to flow. If a component fails during operation, the TOE will enter into a non-operational state and reboot. In situations where the TOE receives packets faster than they can be processed (flooding), the TOE silently discards the excess packets. An audit record is generated when flooding is detected, but not for each packet.

4.4.1.1.2 Guidance

252 The operational guidance associated with this requirement is assessed in the subsequent test EAs.

4.4.1.1.3 Tests

253 The evaluator shall perform the following tests:

254 Test 1: The evaluator shall attempt to get network traffic to flow through the TOE while the TOE is being initialized. A steady flow of network packets that would otherwise be denied by the ruleset should be sourced and directed to a host. The evaluator shall use a packet sniffer to verify none of the generated network traffic is permitted through the TOE during initialization.

High-Level Test Description
The evaluator confirms that no traffic is permitted through the TOE during initialization.
Findings: PASS – The evaluator performed this test case in conjunction with FFW_RUL_EXT.1.1 Test 1.

255 Test 2: The evaluator shall attempt to get network traffic to flow through the TOE while the TOE is being initialized. A steady flow of network packets that would be permitted by the ruleset should be sourced and directed to a host. The evaluator shall use a packet sniffer to verify none of the generated network traffic is permitted through the TOE during initialization and is only permitted once initialization is complete.

256 Note: The remaining testing associated with application of the ruleset is addressed in the subsequent test EAs.

High-Level Test Description
The evaluator confirms that no traffic is permitted through the TOE during initialization.
Findings: PASS – The evaluator performed this test case in conjunction with FFW_RUL_EXT.1.1 Test 2.

4.4.1.2 FPF_RUL_EXT.1.2

257 There are no Eas specified for this element. Definition of packet filtering policy, association of operations with packet filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

4.4.1.3 FPF_RUL_EXT.1.3

258 There are no Eas specified for this element. Definition of packet filtering policy, association of operations with packet filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

4.4.1.4 FPF_RUL_EXT.1.4

4.4.1.4.1 TSS

NIAP TD0683

259 The evaluator shall verify that the TSS describes a packet filtering policy that can use the following fields for each identified protocol, and that the RFCs identified for each protocol are supported:

- Ipv4 (RFC 791)
  - source address
  - destination Address

- protocol
- Ipv6 (RFC 8200)
  - source Address
  - destination Address
  - next Header (Protocol)
- TCP (RFC 793)
  - source Port
  - destination Port
- UDP (RFC 768)
  - source Port
  - destination Port

260 The evaluator shall verify that the TSS describes how conformance with the identified RFCs has been determined by the TOE developer (e.g., third party interoperability testing, protocol compliance testing).

Findings:	[ST] Section 6.5.1 states conformance to the IPv4 (RFC 791), IPv6 (RFC 8200), TCP (RFC 793), and UDP (768) protocols. This section also states that the conformance to the RFCs has been determined (by the developer) through testing and development.

261 The evaluator shall verify that each rule can identify the following actions: permit, discard, and log.

Findings:	[ST] Section 6.5.1 states that the TOE allows permit, deny, and log operations to be associated with rules.

262 The evaluator shall verify that the TSS identifies all interface types subject to the packet filtering policy and explains how rules are associated with distinct network interfaces. Where interfaces can be grouped into a common interface type (e.g., where the same internal logical path is used, perhaps where a common device driver is used), they can be treated collectively as a distinct network interface.

Findings:	[ST] Section 6.5.1 states that the TOE performs stateful packet filtering on all packets received from or being sent to the External Network, MGMT interface or WAN interface (excepting IPsec traffic) of the GoSilent Server.

4.4.1.4.2 Guidance

NIAP TD0683

263 The evaluators shall verify that the operational guidance identifies the following protocols as being supported and the following attributes as being configurable within Packet filtering rules for the associated protocols:

- Ipv4 (RFC 791)
  - source address
  - destination Address
  - protocol
- Ipv6 (RFC 8200)
  - source Address
  - destination Address
  - next Header (Protocol)
- TCP (RFC 793)
  - source Port
  - destination Port
- UDP (RFC 768)
  - source Port

- destination Port

264 The evaluator shall verify that the operational guidance indicates that each rule can identify the following actions: permit, discard, and log.

265 The evaluator shall verify that the operational guidance explains how rules are associated with distinct network interfaces.

266 The guidance may describe the other protocols contained within the ST (e.g., Ipsec, IKE, potentially HTTPS, SSH, and TLS) that are processed by the TOE. The evaluator shall ensure that it is made clear what protocols were not considered as part of the TOE evaluation.

Findings:	[AGD] Section 3.3 states that stateful packet filtering rules can be configured via iptables by selecting "Custom Mode" in the Administrative GUI for each TOE component. The packet filtering rules can be configured with the following actions: accept (permit/allow), reject (discard/deny), ignore, or to pass the packet on to other rules for more processing. Rules can be associated to specific interfaces by defining the interface name in the rule. The TOE permits filtering rules for incoming or outgoing traffic based on: IPv4 (RFC 791): Source Address, Destination Address, Protocol IPv6 (RFC 8200): Source Address, Destination Address, Next Header (Protocol) TCP (RFC 793): Source Port, Destination Port UDP (RFC 768): Source Port, Destination Port

4.4.1.4.3 Tests

267 The evaluator shall perform the following tests:

268 Test 1: The evaluator shall use the instructions in the operational guidance to test that packet filter rules can be created that permit, discard, and log packets for each of the following attributes:

- IPv4
  - Source address
  - Destination Address
  - Protocol
- IPv6
  - Source Address
  - Destination Address
  - Next Header (Protocol)
- TCP
  - Source Port
  - Destination Port
- UDP
  - Source Port
  - Destination Port

High-Level Test Description
The evaluator confirms that the TOE can filter packets based on the stated attributes.
Findings: PASS – The evaluator performed this test case in conjunction FPF_RUL_EXT.1.6 tests 1 – 10.

269 Test 2: The evaluator shall repeat Test 1 above for each distinct network interface type supported by the TOE to ensure that packet filtering rules can be defined for all supported types.

Test Not Applicable
Findings: N/A – Only one network interface is supported.

270 Note that these test activities should be performed in conjunction with those of FPF_RUL_EXT.1.6 where the effectiveness of the rules is tested; here the evaluator is just ensuring the guidance is sufficient and the TOE supports the administrator creating a ruleset based on the above attributes. The test activities for FPF_RUL_EXT.1.6 define combinations of protocols and attributes required to be tested. If those combinations are configured manually, that will fulfill the objective of these test activities, but if those combinations are configured otherwise (e.g., using automation), these test activities may be necessary in order to ensure the guidance is correct and the full range of configurations can be achieved by a TOE administrator.

4.4.1.5 FPF_RUL_EXT.1.5

4.4.1.5.1 TSS

271 The evaluator shall verify that the TSS describes the algorithm applied to incoming packets, including the processing of default rules, determination of whether a packet is part of an established session, and application of administrator defined and ordered ruleset.

Findings: [ST] Section 6.5.1 – The TSS outlines the algorithm that is applied to incoming packets to the TOE. Once the interfaces are brought up, they will start to receive and send packets based on the current configuration (or not receive or send any packets if they have not been previously configured).

The TOE applies a uniform policy to the traffic flows to and from all GoSilent Cube users. By default, no traffic is allowed to flow from GoSilent Cube users to the External Network, no traffic is allowed to flow from the External Network to GoSilent Cube users, and no traffic is allowed to flow between GoSilent Server interfaces.

The TOE maintains a session table which tracks all known TCP and UDP sessions based on the information in incoming packets. Specifically, the lookup is based on an exact match of the following network packet attributes:

- TCP
  - i) Source and Destination IP address
  - ii) Source and Destination Port
  - iii) Protocol Flags (e.g. SYN, ACK, RST, and FIN flags),
  - iv) Sequence numbers
- UDP
  - i) Source and Destination IP address
  - ii) Source and Destination Port

	Traffic for known sessions is permitted to flow. For traffic not associated with known sessions, rules within information flow policies are processed in an administrator-defined order to determine if the traffic should be forwarded. By default, the TOE behavior is to deny packets when there is no rule match.

4.4.1.5.2 Guidance

272 The evaluator shall verify that the operational guidance describes how the order of packet filtering rules is determined and provides the necessary instructions so that an administrator can configure the order of rule processing.

Findings:	[AGD] Section 3.3 states that packet filtering rules are set by tables that contain chains which contain individual rules. If a packet matches any rule, then the action associated with that rule is applied to the packet.

4.4.1.5.3 Tests

273 The evaluator shall perform the following tests:

274 Test 1: The evaluator shall devise two equal packet filtering rules with alternate operations – permit and discard. The rules should then be deployed in two distinct orders and in each case the evaluator shall ensure that the first rule is enforced in both cases by generating applicable packets and using packet capture and logs for confirmation.

High-Level Test Description
The evaluator configures a packet filtering rule with an identical accept rule before a drop rule. The evaluator then sends a packet matching this ruleset and verifies the packet is accepted and logged due to the accept rule being before the drop rule. The evaluator then flips the rules so the drop rule is before the accept rule and then verifies that when another packet is sent, it is dropped and logged by the TOE.
Findings: PASS – The evaluator confirmed that the TOE enforces the firewall rules based on the order configured.

275 Test 2: The evaluator shall repeat the procedure above, except that the two rules should be devised where one is a subset of the other (specific address vs. a network segment). Again, the evaluator should test both orders to ensure that the first is enforced regardless of the specificity of the rule.

High-Level Test Description
The evaluator configures a packet filtering rule with a specific address drop rule before an accept network segment rule. The evaluator then sends a packet matching the specific address and verifies that the packet is dropped and logged by the TOE. The evaluator then flips the rules so the network segment accept rule is before the specific address rule. The evaluator then verifies that the same packet is then accepted due to the accept rule being before the drop rule.
Findings: PASS – The evaluator confirmed that the TOE enforces the firewall rules based on the order configured.

4.4.1.6 FPF_RUL_EXT.1.6

4.4.1.6.1 TSS

276 The evaluator shall verify that the TSS describes the process for applying packet filtering rules and also that the behavior (either by default, or as configured by the administrator) is to discard packets when there is no rule match. The evaluator shall verify the TSS describes when the IPv4 and IPv6 protocols supported by the TOE differ from the full list provided in the RFC Values for IPv4 and IPv6 table.

Findings:	[ST] Section 6.5.1 states that traffic for known sessions is permitted to flow. For traffic not associated with known sessions, rules within information flow policies are processed in an administrator-defined order to determine if the traffic should be forwarded. By default, the TOE behavior is to deny packets when there is no rule match. The TSS does not specify any instances where the IPv4 and IPv6 protocols supported by the TOE differ from the full list provided in the RFC Values for IPv4 and IPv6 table.

4.4.1.6.2 Guidance

277 The evaluator shall verify that the operational guidance describes the behavior if no rules or special conditions apply to the network traffic. If the behavior is configurable, the evaluator shall verify that the operational guidance provides the appropriate instructions to configure the behavior to discard packets with no matching rules. The evaluator shall verify that the operational guidance describes the range of IPv4 and IPv6 protocols supported by the TOE.

Findings:	[AGD] Section 3.3 states that if there is no matching rule for the traffic exists, the traffic is denied. The TOE permits filtering rules for incoming or outgoing traffic based on: IPv4 (RFC 791): Source Address, Destination Address, Protocol IPv6 (RFC 8200): Source Address, Destination Address, Next Header (Protocol) TCP (RFC 793): Source Port, Destination Port UDP (RFC 768): Source Port, Destination Port

4.4.1.6.3 Tests

278 The evaluator shall perform the following tests:

279 Test 1: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
The evaluator configured a ruleset to accept specific IPv4 Transport Layer Protocols for a source address, destination address, wildcard source address and wildcard destination address. After the evaluator configures the ruleset, packets are generated matching the ruleset and then passed through the TOE. Due to the rules being set to accept the traffic, all traffic is accepted and logged.
Findings: PASS – The evaluator confirmed that the TOE enforces all rules properly.

280 Test 2: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv4 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

High-Level Test Description
The evaluator configured a ruleset to accept all traffic but discard IPv4 Transport Layer Protocols for a source address, destination address, wildcard source address and wildcard destination address. After the evaluator configures the ruleset, packets are generated matching the ruleset and then passed through the TOE. Due to the rules being set to accept all traffic but discard Transport Layer Protocols, the respective traffic is dropped, and all other traffic is accepted.
Findings: PASS – The evaluator confirmed that the TOE enforces all rules properly.

281 Test 3: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall configure the TOE to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
The evaluator configures the TOE to permit, drop and log each IPv4 transport layer protocol for a source address, destination address, wildcard source address and wildcard destination address and sends packets through the TOE matching none of these rules. The evaluator confirms that the TOE does not allow the traffic to forward through the TOE.
Findings: PASS – The evaluator confirmed that the TOE drops and logs all traffic not matching any of the rules.

282 Test 4: The evaluator shall configure the TOE to permit and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
The evaluator configures the TOE to permit and log each IPv6 transport layer protocol for a source address, destination address, wildcard source address and wildcard destination address.
Findings: PASS – The evaluator confirmed that the TOE enforces and logs all rules properly.

283 Test 5: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

High-Level Test Description
The evaluator configured a ruleset to drop and log specific IPv6 Transport Layer Protocols. After the evaluator configures the ruleset, packets are generated matching the ruleset and then sent to the TOE. Due to the rules being set to drop the traffic, all traffic is dropped and logged.
Findings: PASS – The evaluator confirmed that the TOE enforces and logs all rules properly.

284 Test 6: The evaluator shall configure the TOE to permit and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall configure the TOE to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are dropped (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
The evaluator configures the TOE to permit, drop and log each IPv6 transport layer protocol for a source address, destination address, wildcard source address and wildcard destination address and sends packets through the TOE matching none of these rules. The evaluator confirms that the TOE does not allow the traffic to forward through the TOE.
Findings: PASS – The evaluator confirmed that the TOE drops and logs all traffic not matching any of the rules.

285 Test 7: The evaluator shall configure the TOE to permit and log protocol 6 (TCP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the

configured source and destination TCP ports in order to ensure that they are permitted (i.e., by capturing the packets after passing through the TOE) and logged.

High-Level Test Description
Configure permit rules on the TOE for specific ports and verify that the rules are enforced (TCP).
Findings: PASS – The evaluator confirmed that the TOE enforces all rules properly.

286 Test 8: The evaluator shall configure the TOE to discard and log protocol 6 (TCP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination TCP ports in order to ensure that they are denied (i.e., by capturing no applicable packets passing through the TOE) and logged.

High-Level Test Description
Configure drop rules on the TOE for specific ports and verify that the rules are enforced (TCP).
Findings: PASS – The evaluator confirmed that the TOE enforces all rules properly.

287 Test 9: The evaluator shall configure the TOE to permit and log protocol 17 (UDP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination UDP ports in order to ensure that they are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Here the evaluator ensures that the UDP port 500 (IKE) is included in the set of tests.

High-Level Test Description
Configure accept rules on the TOE for specific source port, specific destination port, and specific source and specific destination port and send traffic matching these rules to verify that the rules are enforced (UDP).
Findings: PASS – The evaluator confirmed that the TOE enforces and logs all rules properly.

288 Test 10: The evaluator shall configure the TOE to discard and log protocol 17 (UDP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination UDP ports in order to ensure that they are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Again, the evaluator ensures that UDP port 500 is included in the set of tests.

High-Level Test Description
Configure drop rules on the TOE for specific source port, specific destination port, and specific source and specific destination port and send traffic matching these rules to verify that the rules are enforced (UDP).
Findings: PASS – The evaluator confirmed that the TOE enforces and logs all rules properly.

289 The following table identifies the RFC defined values for the protocol fields for IPv4 and IPv6 to be used in configuring and otherwise testing Packet Filtering rule definition and enforcement:

Protocol	**Defined Attributes**

IPv4	• Transport Layer Protocol 1 - Internet Control Message • Transport Layer Protocol 2 - Internet Group Management • Transport Layer Protocol 3 - Gateway-to-Gateway • Transport Layer Protocol 4 - IP in IP (encapsulation) • Transport Layer Protocol 5 - Stream • Transport Layer Protocol 6 - Transmission Control • Transport Layer Protocol 7 - UCL • Transport Layer Protocol 8 - Exterior Gateway Protocol • Transport Layer Protocol 9 - any private interior gateway • Transport Layer Protocol 10 - BBN RCC Monitoring • Transport Layer Protocol 11 - Network Voice Protocol • Transport Layer Protocol 12 - PUP • Transport Layer Protocol 13 - ARGUS • Transport Layer Protocol 14 - EMCON • Transport Layer Protocol 15 - Cross Net Debugger • Transport Layer Protocol 16 - Chaos • Transport Layer Protocol 17 - User Datagram • Transport Layer Protocol 18 - Multiplexing • Transport Layer Protocol 19 - DCN Measurement Subsystems • Transport Layer Protocol 20 - Host Monitoring • Transport Layer Protocol 21 - Packet Radio Measurement • Transport Layer Protocol 22 - XEROX NS IDP • Transport Layer Protocol 23 - Trunk-1 • Transport Layer Protocol 24 - Trunk-2 • Transport Layer Protocol 25 - Leaf-1 • Transport Layer Protocol 26 - Leaf-2 • Transport Layer Protocol 27 - Reliable Data Protocol • Transport Layer Protocol 28 - Internet Reliable Transaction • Transport Layer Protocol 29 - ISO Transport Protocol Class 4 • Transport Layer Protocol 30 - Bulk Data Transfer Protocol • Transport Layer Protocol 31 - MFE Network Services Protocol • Transport Layer Protocol 32 - MERIT Internodal Protocol • Transport Layer Protocol 33 - Sequential Exchange Protocol • Transport Layer Protocol 34 - Third Party Connect Protocol • Transport Layer Protocol 35 - Inter-Domain Policy Routing Protocol • Transport Layer Protocol 36 - XTP • Transport Layer Protocol 37 - Datagram Delivery Protocol • Transport Layer Protocol 38 - IDPR Control Message Transport Protocol • Transport Layer Protocol 39 - TP++ Transport Protocol • Transport Layer Protocol 40 - IL Transport Protocol • Transport Layer Protocol 41 - Simple Internet Protocol • Transport Layer Protocol 42 - Source Demand Routing Protocol • Transport Layer Protocol 43 - SIP Source Route • Transport Layer Protocol 44 - SIP Fragment • Transport Layer Protocol 45 - Inter-Domain Routing Protocol • Transport Layer Protocol 46 - Reservation Protocol • Transport Layer Protocol 47 - General Routing Encapsulation • Transport Layer Protocol 48 - Mobile Host Routing Protocol • Transport Layer Protocol 49 - BNA • Transport Layer Protocol 50 - SIPP Encap Security Payload • Transport Layer Protocol 51 - SIPP Authentication Header

	• Transport Layer Protocol 52 - Integrated Net Layer Security TUBA • Transport Layer Protocol 53 - IP with Encryption • Transport Layer Protocol 54 - NBMA Next Hop Resolution Protocol • Transport Layer Protocol 61 - Any host internal protocol • Transport Layer Protocol 62 - CFTP • Transport Layer Protocol 63 - Any local network • Transport Layer Protocol 64 - SATNET and Backroom EXPAK • Transport Layer Protocol 65 - Kryptolan • Transport Layer Protocol 66 - MIT Remote Virtual Disk Protocol • Transport Layer Protocol 67 - Internet Pluribus Packet Core • Transport Layer Protocol 68 - Any distributed file system • Transport Layer Protocol 69 - SATNET Monitoring • Transport Layer Protocol 70 - VISA Protocol • Transport Layer Protocol 71 - Internet Packet Core Utility • Transport Layer Protocol 72 - Computer Protocol Network Executive • Transport Layer Protocol 73 - Computer Protocol Heart Beat • Transport Layer Protocol 74 - Wang Span Network • Transport Layer Protocol 75 - Packet Video Protocol • Transport Layer Protocol 76 - Backroom SATNET Monitoring • Transport Layer Protocol 77 - SUN ND PROTOCOL-Temporary • Transport Layer Protocol 78 - WIDEBAND Monitoring • Transport Layer Protocol 79 - WIDEBAND EXPAK • Transport Layer Protocol 80 - ISO Internet Protocol • Transport Layer Protocol 81 - VMTP • Transport Layer Protocol 82 - SECURE-VMTP • Transport Layer Protocol 83 - VINES • Transport Layer Protocol 84 - TTP • Transport Layer Protocol 85 - NSFNET-IGP • Transport Layer Protocol 86 - Dissimilar Gateway Protocol • Transport Layer Protocol 87 - TCF • Transport Layer Protocol 88 - IGRP • Transport Layer Protocol 89 - OSPFIGP • Transport Layer Protocol 90 - Sprite RPC Protocol • Transport Layer Protocol 91 - Locus Address Resolution Protocol • Transport Layer Protocol 92 - Multicast Transport Protocol • Transport Layer Protocol 93 - AX.25 Frames • Transport Layer Protocol 94 - IP-within-IP Encapsulation Protocol • Transport Layer Protocol 95 - Mobile Internetworking Control Protocol • Transport Layer Protocol 96 - Semaphore Communications Security Protocol • Transport Layer Protocol 97 - Ethernet-within-IP Encapsulation • Transport Layer Protocol 98 - Encapsulation Header • Transport Layer Protocol 99 - Any private encryption scheme • Transport Layer Protocol 100 - GMTP
IPv6	• Transport Layer Protocol 1 - Internet Control Message • Transport Layer Protocol 2 - Internet Group Management • Transport Layer Protocol 3 - Gateway-to-Gateway

	• Transport Layer Protocol 4 - IPv4 encapsulation • Transport Layer Protocol 5 - Stream • Transport Layer Protocol 6 - Transmission Control • Transport Layer Protocol 7 - CBT • Transport Layer Protocol 8 - Exterior Gateway Protocol • Transport Layer Protocol 9 - any private interior gateway • Transport Layer Protocol 10 - BBN RCC Monitoring • Transport Layer Protocol 11 - Network Voice Protocol • Transport Layer Protocol 12 - PUP • Transport Layer Protocol 13 - ARGUS • Transport Layer Protocol 14 - EMCON • Transport Layer Protocol 15 - Cross Net Debugger • Transport Layer Protocol 16 - Chaos • Transport Layer Protocol 17 - User Datagram • Transport Layer Protocol 18 - Multiplexing • Transport Layer Protocol 19 - DCN Measurement Subsystems • Transport Layer Protocol 20 - Host Monitoring • Transport Layer Protocol 21 - Packet Radio Measurement • Transport Layer Protocol 22 - XEROX NS IDP • Transport Layer Protocol 23 - Trunk-1 • Transport Layer Protocol 24 - Trunk-2 • Transport Layer Protocol 25 - Leaf-1 • Transport Layer Protocol 26 - Leaf-2 • Transport Layer Protocol 27 - Reliable Data Protocol • Transport Layer Protocol 28 - Internet Reliable Transaction • Transport Layer Protocol 29 - Transport Protocol Class 4 • Transport Layer Protocol 30 - Bulk Data Transfer Protocol • Transport Layer Protocol 31 - MFE Network Services Protocol • Transport Layer Protocol 32 - MERIT Internodal Protocol • Transport Layer Protocol 33 - Datagram Congestion Control Protocol • Transport Layer Protocol 34 - Third Party Connect Protocol • Transport Layer Protocol 35 - Inter-Domain Policy Routing Protocol • Transport Layer Protocol 36 - XTP • Transport Layer Protocol 37 - Datagram Delivery Protocol • Transport Layer Protocol 38 - IDPR Control Message Transport Protocol • Transport Layer Protocol 39 - TP++ Transport Protocol • Transport Layer Protocol 40 - IL Transport Protocol • Transport Layer Protocol 41 - IPv6 encapsulation • Transport Layer Protocol 42 - Source Demand Routing Protocol • Transport Layer Protocol 43 - Intentionally blank • Transport Layer Protocol 44 - Intentionally blank • Transport Layer Protocol 45 - Inter-Domain Routing Protocol • Transport Layer Protocol 46 - Reservation Protocol • Transport Layer Protocol 47 - General Routing Encapsulation • Transport Layer Protocol 48 - Dynamic Source Routing Protocol • Transport Layer Protocol 49 - BNA • Transport Layer Protocol 50 - Intentionally Blank • Transport Layer Protocol 51 - Intentionally Blank • Transport Layer Protocol 52 - Integrated Net Layer Security • Transport Layer Protocol 53 - IP with Encryption

	• Transport Layer Protocol 54 - NBMA Address Resolution Protocol • Transport Layer Protocol 55 - Mobility • Transport Layer Protocol 56 - Transport Layer Security Protocol using Kryptonet key management • Transport Layer Protocol 57 - SKIP • Transport Layer Protocol 58 - ICMP for IPv6 • Transport Layer Protocol 59 - No Next Header for IPv6 • Transport Layer Protocol 60 - Intentionally Blank • Transport Layer Protocol 61 - Any host internal protocol • Transport Layer Protocol 62 - CFTP • Transport Layer Protocol 63 - Any local network • Transport Layer Protocol 64 - SATNET and Backroom EXPAK • Transport Layer Protocol 65 - Kryptolan • Transport Layer Protocol 66 - MIT Remote Virtual Disk Protocol • Transport Layer Protocol 67 - Internet Pluribus Packet Core • Transport Layer Protocol 68 - Any distributed file system • Transport Layer Protocol 69 - SATNET Monitoring • Transport Layer Protocol 70 - VISA Protocol • Transport Layer Protocol 71 - Internet Packet Core Utility • Transport Layer Protocol 72 - Computer Protocol Network Executive • Transport Layer Protocol 73 - Computer Protocol Heart Beat • Transport Layer Protocol 74 - Wang Span Network • Transport Layer Protocol 75 - Packet Video Protocol • Transport Layer Protocol 76 - Backroom SATNET Monitoring • Transport Layer Protocol 77 - SUN ND PROTOCOL-Temporary • Transport Layer Protocol 78 - WIDEBAND Monitoring • Transport Layer Protocol 79 - WIDEBAND EXPAK • Transport Layer Protocol 80 - ISO Internet Protocol • Transport Layer Protocol 81 - VMTP • Transport Layer Protocol 82 - SECURE-VMTP • Transport Layer Protocol 83 - VINES • Transport Layer Protocol 84 - TTP • Transport Layer Protocol 85 - Internet Protocol Traffic Manager • Transport Layer Protocol 86 - NSFNET-IGP • Transport Layer Protocol 87 - Dissimilar Gateway Protocol • Transport Layer Protocol 88 - TCF • Transport Layer Protocol 89 - EIGRP • Transport Layer Protocol 90 - OSPFIGP • Transport Layer Protocol 91 - Sprite RPC Protocol • Transport Layer Protocol 92 - Locus Address Resolution Protocol • Transport Layer Protocol 93 - Multicast Transport Protocol • Transport Layer Protocol 94 - AX.25 Frames • Transport Layer Protocol 95 - IP-within-IP Encapsulation Protocol • Transport Layer Protocol 96 - Mobile Internetworking Control Pro. • Transport Layer Protocol 97 - Semaphore Communications Sec. Pro. • Transport Layer Protocol 98 - Ethernet-within-IP Encapsulation

	• Transport Layer Protocol 99 - Encapsulation Header • Transport Layer Protocol 100 - GMTP • Transport Layer Protocol 101 - Ipsilon Flow Management Protocol • Transport Layer Protocol 102 - PNNI over IP • Transport Layer Protocol 103 - Protocol Independent Multicast • Transport Layer Protocol 104 - ARIS • Transport Layer Protocol 105 - SCPS Transport Layer Protocol • Transport Layer Protocol 106 – QNX • Transport Layer Protocol 107 - Active Networks • Transport Layer Protocol 108 - Payload Compression Protocol • Transport Layer Protocol 109 - Sitara Networks Protocol • Transport Layer Protocol 110 - Compaq Peer Protocol • Transport Layer Protocol 111 - IPX in IP • Transport Layer Protocol 112 - Virtual Router Redundancy Protocol • Transport Layer Protocol 113 - PGM Reliable Transport Protocol • Transport Layer Protocol 114 - Any 0-hop protocol • Transport Layer Protocol 115 - Layer Two Tunneling Protocol • Transport Layer Protocol 116 - D-II Data Exchange (DDX) • Transport Layer Protocol 117 - Interactive Agent Transfer Protocol • Transport Layer Protocol 118 - Schedule Transfer Protocol • Transport Layer Protocol 119 - SpectraLink Radio Protocol • Transport Layer Protocol 120 - UTI • Transport Layer Protocol 121 - Simple Message Protocol • Transport Layer Protocol 122 - SM • Transport Layer Protocol 123 - Performance Transparency Protocol • Transport Layer Protocol 124 - ISIS over IPv4 • Transport Layer Protocol 125 - FIRE • Transport Layer Protocol 126 - Combat Radio Transport Protocol • Transport Layer Protocol 127 - Combat Radio User Datagram • Transport Layer Protocol 128 - SSCOPMCE • Transport Layer Protocol 129 - IPLT • Transport Layer Protocol 130 - Secure Packet Shield • Transport Layer Protocol 131 - Private IP Encapsulation within IP • Transport Layer Protocol 132 - Stream Control Transmission Protocol • Transport Layer Protocol 133 - Fibre Channel • Transport Layer Protocol 134 - RSVP-E2E-IGNORE • Transport Layer Protocol 135 - Mobility Header • Transport Layer Protocol 136 - UDPLite • Transport Layer Protocol 137 - MPLS-in-IP • Transport Layer Protocol 138 - MANET Protocols • Transport Layer Protocol 139 - Host Identity Protocol • Transport Layer Protocol 140 - Shim6 Protocol • Transport Layer Protocol 141 - Wrapped Encapsulating Security Payload • Transport Layer Protocol 142 - Robust Header Compression

4.5 Protection of the TSF (FPT)

4.5.1 FPT_FLS.1/SelfTest Failure with Preservation of Secure State (Self-test Failures)

4.5.1.1 TSS

290 The evaluator shall ensure the TSS describes how the TOE ensures a shutdown upon a self-test failure, a failed integrity check of the TSF executable image, or a failed health test of the noise source. If there are instances when a shut-down does not occur, (e.g., a failure is deemed non- security relevant), the evaluator shall ensure that those cases are identified and a rationale is provided that supports the classification and justifies why the TOE's ability to enforce its security policies is not affected in any such instance.

Findings:	[ST] Section 6.8.2 provides details of the power-on tests and how the TOE functions after a self-test fails. If any test fails, the TOE does not enter an operational state and network interfaces are not activated.

4.5.1.2 Guidance

291 The evaluator shall verify that the operational guidance provides information on the self-test failures that can cause the TOE to shut down and how to diagnose the specific failure that has occurred, including possible remediation steps if available.

Findings:	[AGD] Section 3.7 describes the firmware integrity tests which should be configured and execute on boot. If any of the tests fail the TOE component will not enter an operational state. Upon failure, the GSS will display an error message. Upon failure of the Cube, the onboard LEDs will indicate an error. For the rectification of error states, diagnostic information displayed in relevant error messages on GSS and behaviour of the LEDs on Cube should be referenced.

4.5.1.3 Tests

292 There are no test EAs for this SFR.

4.5.2 FPT_TST_EXT.3 Self-Test with Defined Methods

4.5.2.1 TSS

293 The evaluator shall verify that the TSS describes the method used to perform self-testing on the TSF executable code, and that this method is consistent with what is described in the SFR.

Findings:	[ST] Section 6.8.2 – The TSS describes the integrity of the firmware is tested using a stored signature that covers all of the executable code. This is consistent with what is described in the SFR.

4.5.2.2 Guidance

294 There are no guidance EAs for this SFR.

4.5.2.3 Tests

295 There are no test EAs for this SFR.

4.6 Trusted Path/Channels (FTP)

4.6.1 FTP_ITC.1/VPN Inter-TSF Trusted Channel (VPN Communications)

4.6.1.1 TSS

296 The EAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

4.6.1.2 Guidance

297 The EAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

4.6.1.3 Tests

298 The EAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications. Additional testing for IPsec is covered in FCS_IPSEC_EXT.1.

5 Evaluation Activities for Stateful Traffic Filter Firewalls PP-Module

5.1 Security Audit (FAU)

5.1.1 FAU_GEN.1 Audit Data Generation

5.1.1.1 TSS

299 No additional Evaluation Activities are specified.

Findings:	N/A -- No additional Evaluation Activities are specified.

5.1.1.2 Guidance Documentation

300 In addition to the Evaluation Activities specified in the Supporting Document for the Base-PP, the evaluator shall check the guidance documentation to ensure that it describes the audit records specified in Table 2 of the PP-Module in addition to those required by the Base-PP. If the optional SFR FFW_RUL_EXT.2 is claimed by the TOE, the evaluator shall also check the guidance documentation to ensure that it describes the relevant audit record specified in Table 3 of the PP-Module.

Findings:	[AGD] Section 4.2 – Table 5 describes the audit records specified in Table 2 of the PP-Module in addition to those required by the Base-PP. The optional SFR FFW_RUL_EXT.2 is not claimed.

5.1.1.3 Tests

301 In addition to the Evaluation Activities specified in the Supporting Document for the Base-PP, the evaluator shall perform tests to demonstrate that audit records are generated for the auditable events as specified in Table 2 of the PP-Module and, if the optional SFR FFW_RUL_EXT.2 is claimed by the TOE, Table 3.

High-Level Test Description
Ensure that the TOE displays an audit record for each of the auditable events defined for this requirement.
Findings: PASS—The evaluator confirmed that all audit records are generated by the TOE in conjunction with FAU_GEN.1.

5.2 User Data Protection (FDP)

5.2.1 FDP_RIP.2 Full Residual Information Protection

5.2.1.1 TSS

302 "Resources" in the context of this requirement are network packets being sent through (as opposed to "to", as is the case when a security administrator connects to the TOE) the TOE. The concern is that once a network packet is sent, the buffer or memory area used by the packet still contains data from that packet, and that if that buffer is re-used, those data might remain and make their way into a new packet. The evaluator shall check to ensure that the TSS describes packet processing to the extent that they can determine that no data will be reused when processing network packets. The evaluator shall ensure that this description at a minimum describes how the previous data are zeroized/overwritten, and at what point in the buffer processing this occurs.

Findings:	[ST] Section 6.4.1 – The TSS details how resources made available to the packet when traversing is zeroized when the packet is released. The TSS also describes that no residual information from previous streams can traverse through the TOE.

5.3 Firewall (FFW)

5.3.1 FFW_RUL_EXT.1 Stateful Traffic Filtering

5.3.1.1 TSS

303 The evaluator shall verify that the TSS provides a description of the TOE's initialization/startup process, which clearly indicates where processing of network packets begins to take place, and provides a discussion that supports the assertion that packets cannot flow during this process.

Findings:	[ST] Section 6.5.1 – The TSS provides details on how network processing is handled during the boot sequence to ensure the packet filtering rules cannot be bypassed. The boot sequence for the TOE includes: - BIOS hardware and memory checks - Loading and initialization of the OS - Self-tests including firmware integrity tests are executed - The init utility is started (mounts file systems, sets up network cards, and generally starts all the processes that usually are run on the system at startup) - Daemon programs such as Internet Service Daemon (INETD), Syslogd are started; Routing and forwarding tables are initialized - Application daemons are loaded, enabling access to the GoSilent Server management GUI - Physical interfaces are active Only when the interfaces are fully brought up successfully will the packets start to be received and sent. The TSS also identifies how the TOE applies traffic policies to safeguard against packets flowing through the TOE without any policies active.

304 The evaluator shall verify that the TSS also include a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describe the safeguards that would prevent packets flowing

through the TOE without applying the ruleset in the event of a component failure. This could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets. The description shall also include a description how the TOE behaves in the situation where the traffic exceeds the amount of traffic the TOE can handle and how it is ensured that also in this condition stateful traffic filtering rules are still applied so that traffic does not pass that shouldn't pass according to the specified rules.

Findings:	[ST] Section 6.5.1 states in the event of a component failure, such as the firewall failing to initialize, the TOE will not enter a state where network traffic is able to flow. If a component fails during operation, the TOE will enter into a non-operational state and reboot. In situations where the TOE receives packets faster than they can be processed (flooding), the TOE silently discards the excess packets. An audit record is generated when flooding is detected, but not for each packet.

5.3.1.2 Guidance Documentation

305 The guidance documentation associated with this requirement is assessed in the subsequent test evaluation activities.

Findings:	Requirement is assessed in subsequent test activities.

5.3.1.3 Tests

a) Test 1: The evaluator shall attempt to get network traffic to flow through the TOE while the TOE is being initialized. A steady flow of network packets that would otherwise be denied by the ruleset should be sourced and be directed at a host. The evaluator shall verify using a packet sniffer that none of the generated network traffic is permitted through the firewall during initialization.

High-Level Test Description
The evaluator configures a filter on the TOE to drop traffic from a source address. The evaluator then sends traffic from this source address while the TOE is being rebooted. After the TOE reboots, the evaluator verifies that no traffic was allowed to flow through it.
Findings: PASS – The evaluator confirmed that the TOE enforces the rules properly.

b) Test 2: The evaluator shall attempt to get network traffic to flow through the TOE while the TOE is being initialized. A steady flow of network packets that would be permitted by the ruleset should be sourced and be directed at a host. The evaluator shall verify using a packet sniffer that none of the generated network traffic is permitted through the firewall during initialization and is only permitted once initialization is complete.

High-Level Test Description
The evaluator for DTR test case FFW_RUL_EXT.1.1 Test 2 configures a filter on the TOE to accept traffic from a source address. The evaluator then sends traffic from the source address while the TOE is being rebooted. After the TOE reboots, the evaluator verifies that the traffic was allowed to flow through the TOE except for when it was initializing.

High-Level Test Description
Findings: PASS – The evaluator confirmed that the TOE does not allow traffic to flow while it is initializing.

306 Note: The remaining testing associated with application of the ruleset is addressed in the subsequent test evaluation activities.

5.3.2 FFW_RUL_EXT.1.2/FFW_RUL_EXT.1.3/FFW_RUL_EXT.1.4

5.3.2.1 TSS

307 The evaluator shall verify that the TSS describes a stateful packet filtering policy and the following attributes are identified as being configurable within stateful traffic filtering rules for the associated protocols:

- ICMPv4
  - Type
  - Code
- ICMPv6
  - Type
  - Code
- IPv4
  - Source Address
  - Destination Address
  - Transport Layer Protocol
- IPv6
  - Source Address
  - Destination Address
  - Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields
- TCP
  - Source Port
  - Destination Port
- UDP
  - Source Port
  - Destination Port

308 The evaluator shall verify that each rule can identify the following actions: permit or drop with the option to log the operation. The evaluator shall verify that the TSS identifies all interface types subject to the stateful packet filtering policy and explains how rules are associated with distinct network interfaces.

Findings: [ST] Section 6.5.1 states by default, the TOE allows administrators to define traffic filtering rules based on a distinct interface and the following network protocol fields:

- ICMPv4 (Type, Code)
- ICMPv6 (Type, Code)
- IPv4 (Source address, Destination address, Transport Layer Protocol)
- IPv6 (Source address, Destination address, Transport Layer Protocol)
- TCP (Source port, Destination port)

	• UDP (Source port, Destination port) This section also states that the TOE allows permit, deny, and log operations to be associated with rules and these rules can be assigned to distinct network interfaces.

5.3.2.2 Guidance Documentation

309 The evaluators shall verify that the guidance documentation identifies the following attributes as being configurable within stateful traffic filtering rules for the associated protocols:

- ICMPv4
  - Type
  - Code
- ICMPv6
  - Type
  - Code
- IPv4
  - Source Address
  - Destination Address
  - Transport Layer Protocol
- IPv6
  - Source Address
  - Destination Address
  - Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields
- TCP
  - Source Port
  - Destination Port
- UDP
  - Source Port
  - Destination Port

310 The evaluator shall verify that the guidance documentation indicates that each rule can identify the following actions: permit, drop, and log.

311 The evaluator shall verify that the guidance documentation explains how rules are associated with distinct network interfaces.

Findings:	[AGD] Section 3.3 states that packet filtering rules can be configured with the following actions: accept (permit/allow), reject (discard/deny), ignore, or to pass the packet on to other rules for more processing. Rules can be associated to specific interfaces by defining the interface name in the rule. The TOE permits filtering rules for incoming or outgoing traffic based on: IPv4 (RFC 791): Source Address, Destination Address, Protocol IPv6 (RFC 8200): Source Address, Destination Address, Next Header (Protocol) TCP (RFC 793): Source Port, Destination Port UDP (RFC 768): Source Port, Destination Port

5.3.2.3 Tests

a) Test 1: The evaluator shall use the instructions in the guidance documentation to test that stateful packet filter firewall rules can be created that permit, drop, and log packets for each of the following attributes

- ICMPv4
  - Type
  - Code
- ICMPv6
  - Type
  - Code
- IPv4
  - Source Address
  - Destination Address
  - Transport Layer Protocol
- IPv6
  - Source Address
  - Destination Address
  - Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields
- TCP
  - Source Port
  - Destination Port
- UDP
  - Source Port
  - Destination Port

High-Level Test Description
Configure a firewall ruleset to enforce all the attributes identified.
Findings: PASS – The evaluator confirmed that the TOE enforces the rules properly. IPv4, IPv6, TCP and UDP are all tested within FPF_RUL_EXT.1.5, FPF_RUL_EXT.1.6.

b) Test 2: Repeat the test evaluation activity above to ensure that stateful traffic filtering rules can be defined for each distinct network interface type supported by the TOE.

Test Not Applicable
Findings: N/A – The TOE only claims one network interface type

312 Note that these test activities should be performed in conjunction with those of FFW_RUL_EXT.1.9 where the effectiveness of the rules is tested. The test activities for FFW_RUL_EXT.1.9 define the protocol/attribute combinations required to be tested. If those combinations are configured manually, that will fulfil the objective of these test activities, but if those combinations are configured otherwise (e.g., using automation), these test activities may be necessary in order to ensure the guidance is correct and the full range of configurations can be achieved by a TOE administrator.

5.3.3 FFW_RUL_EXT.1.5

5.3.3.1 TSS

313 The evaluator shall verify that the TSS identifies the protocols that support stateful session handling. The TSS shall identify TCP, UDP, and, if selected by the ST author, also ICMP.

Findings:	[ST] Section 6.5.1 states that the TOE performs stateful network traffic filtering on network packets using the network traffic protocols and network fields specified in FFW_RUL_EXT.1.5. FFW_RUL_EXT.1.5 selection TCP and UDP. ICMP is not selected.

314 The evaluator shall verify that the TSS describes how stateful sessions are established (including handshake processing) and maintained.

Findings:	[ST] Section 6.5.1 The TOE maintains a session table which tracks all known TCP and UDP sessions based on the information in incoming packets. For TCP sessions, the TOE removes existing traffic flows from the list of established sessions based on the completion of the session. An exchange of FIN flags indicates the end of data transmission to finish a TCP connection, at which point the session is terminated and the session is removed from the table. TCP session are also removed after a configurable session inactivity timeout threshold. By default, this threshold is 432000 seconds. Once this timeout period has been reached, the TOE will then terminate the connection and remove it from the table. New packets that match the session will start a new session. UDP sessions are removed after the session inactivity timeout threshold. Once this threshold has been reached, the TOE terminates the connection and removes it from the table. New packets that match the session will start a new session.

315 The evaluator shall verify that for TCP, the TSS identifies and describes the use of the following attributes in session determination: source and destination addresses, source and destination ports, sequence number, and individual flags

Findings:	[ST] Section 6.5.1 states that the TOE maintains a session table which tracks all known TCP sessions. Specifically, the lookup is based on an exact match of the following network packet attributes: • TCP i) Source and Destination IP address ii) Source and Destination Port iii) Protocol Flags (e.g. SYN, ACK, RST, and FIN flags), iv) Sequence numbers

316 The evaluator shall verify that for UDP, the TSS identifies and describes the following attributes in session determination: source and destination addresses, source and destination ports.

Findings:	[ST] Section 6.5.1 states that the TOE maintains a session table which tracks all known UDP sessions. Specifically, the lookup is based on an exact match of the following network packet attributes: • UDP i) Source and Destination IP address ii) Source and Destination Port

317 The evaluator shall verify that for ICMP (if selected), the TSS identifies and describes the following attributes in session determination: source and destination addresses, other attributes chosen in FFW_RUL_EXT.1.5.

Findings:	The [ST] does not selection ICMP for FFW_RUL_EXT.1.5.

318 The evaluator shall verify that the TSS describes how established stateful sessions are removed. The TSS shall describe how connections are removed for each protocol based on normal completion and/or timeout conditions. The TSS shall also indicate when session removal becomes effective (e.g., before the next packet that might match the session is processed).

Findings:	[ST] Section 6.5.1 states that the TOE maintains a session table which tracks all known TCP and UDP sessions based on the information in incoming packets. For TCP sessions, the TOE removes existing traffic flows from the list of established sessions based on the completion of the session. An exchange of FIN flags indicates the end of data transmission to finish a TCP connection, at which point the session is terminated and the session is removed from the table. TCP session are also removed after a configurable session inactivity timeout threshold. By default, this threshold is 432000 seconds. Once this timeout period has been reached, the TOE will then terminate the connection and remove it from the table. New packets that match the session will start a new session. UDP sessions are removed after the configured session inactivity timeout threshold is reached. The default timeouts are: - Generic IP Timeout: 600 - TCP Connection Timeout: 432000 - UDP Protocol Timeout: 30 - UDP Stream Timeout: 180

5.3.3.2 Guidance Documentation

319 The evaluator shall verify that the guidance documentation describes stateful session behaviours. For example, a TOE might not log packets that are permitted as part of an existing session.

Findings:	[AGD] Section 3.3 states that all traffic that correlates to an existing session is permitted and any traffic not related to a known session is subject to processing rules that apply flow policies in an administrator-defined order.

5.3.3.3 Tests

320 The following tests shall be run using IPv4 and IPv6.

a) Test 1: The evaluator shall configure the TOE to permit and log TCP traffic. The evaluator shall initiate a TCP session. While the TCP session is being established, the evaluator shall introduce session establishment packets with incorrect flags to determine that the altered traffic is not accepted as part of the session (i.e., a log event is generated to show the ruleset was applied). After a TCP session is successfully established, the evaluator shall alter each of the session determining attributes (source and destination addresses, source and destination ports, sequence number, flags) one at a time in order to verify that the altered packets are not accepted as part of the established session.

High-Level Test Description
The evaluator configured a traffic rule to permit and log TCP traffic. The evaluator then initiated a TCP connection with the TOE acting as a firewall between both endpoints. While this TCP session is open, the evaluator introduces invalid TCP session packets and verifies that these packets are dropped and logged by the TOE.
Findings: Pass – The evaluator confirmed that the TOE does not accept the invalid TCP session packets.

b) Test 2: The evaluator shall terminate the TCP session established per Test 1 as described in the TSS. The evaluator shall then immediately send a packet matching the former session definition in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

High-Level Test Description
The evaluator configured the TOE to accept and log TCP traffic. The evaluator then opened a TCP connection with the TOE acting as a firewall between both endpoints. The evaluator then terminated this TCP connection and sent a packet that matched the former session and verified that this packet was dropped and logged by the TOE.
Findings: Pass – The evaluator confirmed that the TOE enforces all rules properly.

c) Test 3: The evaluator shall expire (i.e., reach timeout) the TCP session established per Test 1 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

High-Level Test Description
The evaluator configured the TOE to permit and log TCP traffic. The evaluator then opened a TCP connection with the TOE acting as a firewall between both endpoints. The evaluator let the TCP session expire and then sent packets matching the expired session through the TOE. The evaluator then verified that the packets were dropped and logged.
Findings: Pass – The evaluator confirmed that the TOE does not accept the expired TCP packets.

d) Test 4: The evaluator shall configure the TOE to permit and log UDP traffic. The evaluator shall establish a UDP session. Once a UDP session is established, the evaluator shall alter each of the session determining attributes (source and destination addresses, source and destination ports) one at a time in order to verify that the altered packets are not accepted as part of the established session.

High-Level Test Description
The evaluator configured the TOE to permit and log UDP traffic. The evaluator then opened a UDP connection with the TOE acting as a firewall between both endpoints. The evaluator then terminated this UDP connection and sent a packet that matched the former session and verified that this packet was not accepted as part of the previous session by the TOE.
Findings: Pass – The evaluator confirmed that the TOE does not accept UDP packets matching the former session as part of the former session.

e) Test 5: The evaluator shall expire (i.e., reach timeout) the UDP session established per Test 4 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

High-Level Test Description
The evaluator configured the TOE to permit and log UDP traffic. The evaluator then opened a UDP connection with the TOE acting as a firewall between both endpoints. The evaluator let the UDP session expire and then sent packets matching the expired session through the TOE. The evaluator then verified that the packets were not accepted as part of the previous session by the TOE.
Findings: Pass – The evaluator confirmed that the TOE does not accept the expired UDP packets as part of the previous session.

f) Test 6: If ICMP is selected, the evaluator shall configure the TOE to permit and log ICMP traffic. The evaluator shall establish a session for ICMP as defined in the TSS. Once an ICMP session is established, the evaluator shall alter each of the session determining attributes (source and destination addresses, other attributes chosen in FFW_RUL_EXT.1.5) one at a time in order to verify that the altered packets are not accepted as part of the established session.

Test Not Applicable
Findings: N/A - This test requirement is not applicable as ICMP is not part of the selection made in FFW_RUL_EXT.1.5.

g) Test 7: If applicable, the evaluator shall terminate the ICMP session established per Test 6 as described in the TSS. The evaluator shall then immediately send a packet matching the former session definition in order to ensure it is not forwarded through the TOE without being subject to the ruleset

Test Not Applicable
Findings: N/A - This test requirement is not applicable as ICMP is not part of the selection made in FFW_RUL_EXT.1.5.

h) Test 8: The evaluator shall expire (i.e., reach timeout) the ICMP session established per Test 6 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test Not Applicable
Findings: N/A - This test requirement is not applicable as ICMP is not part of the selection made in FFW_RUL_EXT.1.5.

5.3.4 FFW_RUL_EXT.1.6

5.3.4.1 TSS

321 The evaluator shall verify that the TSS identifies the following as packets that will be automatically dropped and are counted or logged:

a) Packets which are invalid fragments, including a description of what constitutes an invalid fragment
b) Fragments that cannot be completely re-assembled
c) Packets where the source address is defined as being on a broadcast network
d) Packets where the source address is defined as being on a multicast network
e) Packets where the source address is defined as being a loopback address
f) The TSF shall reject and be capable of logging network packets where the source or destination address of the network packet is defined as being unspecified (i.e. 0.0.0.0) or an address “reserved for future use” (i.e. 240.0.0.0/4) as specified in RFC 5735 for IPv4;
g) The TSF shall reject and be capable of logging network packets where the source or destination address of the network packet is defined as an “unspecified address” or an address “reserved for future definition and use” (i.e. unicast addresses not in this address range: 2000::/3) as specified in RFC 3513 for IPv6;
h) Packets with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified
i) Other packets defined in FFW_RUL_EXT.1.6 (if any)

Findings:	[ST] Section 6.5.1 identifies the following factors that decide if a packet is automatically dropped or counted and logged:

Item a:

- invalid fragments (counted by TSF);
- packets are checked for validity. “Invalid fragments” are those that violate these rules:
  - i) No overlap
  - ii) The total fragments in one packet should not be more than 62 pieces
  - iii) The total length of merged fragments should not larger than 64k
  - iv) All fragments in one packet should arrive in 2 seconds
  - v) The total queued fragments has limitation, depending on the platform
  - vi) The total number of concurrent fragment processing for different packet has limitations depending on platform

Item b:

- fragmented IP packets which cannot be re-assembled completely (counted by TSF);

Item c:

- where the source address is defined as being on a broadcast network;

Item d:

- where the source address is defined as being on a multicast network;

Item e:

- where the source address is defined as being a loopback address;

Item f:

- where the source or destination address is defined as being an address “reserved for future use” as specified in RFC 5735 for IPv4;

Item g:

- where the source or destination address is defined as an “unspecified address” or an address “reserved for future definition and use” as specified in RFC 3513 for IPv6;

Item h:

- with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified;

Item i:

FFW_RUL_EXT.1.6 selects “no other rules”

5.3.4.2 Guidance Documentation

322 The evaluator shall verify that the guidance documentation describes packets that are discarded and potentially logged by default. If applicable protocols are identified,

their descriptions need to be consistent with the TSS. If logging is configurable, the evaluator shall verify that applicable instructions are provided to configure auditing of automatically rejected packets.

Findings:	[AGD] Section 3.3.2 describes packets that are discarded ad potentially logged by default consistent with the TSS. [AGD] Section 3.9 provides instructions for configuring the auditing of automatically rejected packets.

5.3.4.3 Tests

323 Both IPv4 and IPv6 shall be tested for items a), b), c), d), and e) of the SFR element. Both IPv4 and IPv6 shall be tested for item i) unless the rule definition is specific to IPv4 or IPv6. Note: f), g), and h) are specific to IPv4 or IPv6 and shall be tested accordingly.

a) Test 1: The evaluator shall test each of the conditions for automatic packet rejection in turn. In each case, the TOE should be configured to allow all network traffic and the evaluator shall generate a packet or packet fragment that is to be rejected. The evaluator shall use packet captures to ensure that the unallowable packet or packet fragment is not passed through the TOE.

High-Level Test Description
Generate packets that match the items in a-h from the ST and verify that the TOE rejects these invalid packets.
Findings: Pass – The evaluator confirmed that the TOE rejects all invalid packets.

b) Test 2: For each of the cases above, the evaluator shall use any applicable guidance to enable dropped packet logging or counting. In each case above, the evaluator shall ensure that the rejected packet or packet fragment was recorded (either logged or an appropriate counter incremented).

High-Level Test Description
Verify a counter is incremented when invalid packets are detected by the TOE.
Findings: Pass – The evaluator confirmed that the TOE rejects all invalid packets. This was performed in conjunction with Test 1.

5.3.5 FFW_RUL_EXT.1.7

5.3.5.1 TSS

324 The evaluator shall verify that the TSS explains how the following traffic can be dropped and counted or logged:

a) Packets where the source address is equal to the address of the network interface where the network packet was received
b) Packets where the source or destination address of the network packet is a link-local address

c) Packets where the source address does not belong to the networks associated with the network interface where the network packet was received, including a description of how the TOE determines whether a source address belongs to a network associated with a given network interface

Findings:	[ST] Section 6.5.1 identifies the following factors that decide if a packet is automatically dropped or counted and logged: Item a: • where the source address of the network packet is equal to the address of the network interface where the network packet was received Item b: • where the source or destination address of the network packet is a link-local address Item c: • where the source address of the network packet does not belong to the networks associated with the network interface where the network packet was received

5.3.5.2 Guidance Documentation

325 The evaluator shall verify that the guidance documentation describes how the TOE can be configured to implement the required rules. If logging is configurable, the evaluator shall verify that applicable instructions are provided to configure auditing of automatically rejected packets.

Findings:	[AGD] Section 3.3 specifies that the TOE's firewall functions use iptables built on the Netfilter framework. The AGD describes how to configure the firewall settings to apply custom rules and lists all the parameters able to be processed by the TOE to implement the required rules. [AGD] section 3.9 provides the instructions for configuring the logging of firewall actions.

5.3.5.3 Tests

326 The following tests shall be run using IPv4 and IPv6.

a) Test 1: The evaluator shall configure the TOE to drop and log network traffic where the source address of the packet matches that of the TOE network interface upon which the traffic was received. The evaluator shall generate suitable network traffic to match the configured rule and verify that the traffic is dropped and a log message generated.

High-Level Test Description
Configure the TOE to drop and log IPv4 and IPv6 packets containing a source IP address matching the IP address of the TOE's receiving interface. Send traffic through the TOE with a source address of the receiving interface.

High-Level Test Description
Findings: Pass – The evaluator confirmed that the TOE rejects and logs the invalid packets due to the invalid source address.

b) Test 2: The evaluator shall configure the TOE to drop and log network traffic where the source IP address of the packet fails to match the network reachability information of the interface to which it is targeted, e.g. if the TOE believes that network 192.168.1.0/24 is reachable through interface 2, network traffic with a source address from the 192.168.1.0/24 network should be generated and sent to an interface other than interface 2. The evaluator shall verify that the network traffic is dropped and a log message generated.

High-Level Test Description
Configure the TOE to drop and log IPv4 and IPv6 packets containing a source IP address matching that is not reachable by the TOE's receiving interface.Send traffic through the TOE that is sourced from a subnet that is unreachable from the interface.
Findings: Pass – The evaluator confirmed that the TOE rejects and logs the packet due to the invalid address.

5.3.6 FFW_RUL_EXT.1.8

5.3.6.1 TSS

NIAP TD0545

327 The evaluator shall verify that the TSS describes the algorithm applied to incoming packets, including the processing of default rules, determination of whether a packet is part of an established session, and application of administrator defined and ordered ruleset.

Findings: [ST] Section 6.5.1 – The TSS outlines the algorithm that is applied to incoming packets to the TOE. Once the interfaces are brought up, they will start to receive and send packets based on the current configuration (or not receive or send any packets if they have not been previously configured).

The TOE applies a uniform policy to the traffic flows to and from all GoSilent Cube users. By default, no traffic is allowed to flow from GoSilent Cube users to the External Network, no traffic is allowed to flow from the External Network to GoSilent Cube users, and no traffic is allowed to flow between GoSilent Server interfaces.

The TOE maintains a session table which tracks all known TCP and UDP sessions based on the information in incoming packets. Specifically, the lookup is based on an exact match of the following network packet attributes:

- TCP

  i) Source and Destination IP address

  ii) Source and Destination Port

  iii) Protocol Flags (e.g. SYN, ACK, RST, and FIN flags),

iv) Sequence numbers

- UDP

  i) Source and Destination IP address

  ii) Source and Destination Port

Traffic for known sessions is permitted to flow. For traffic not associated with known sessions, rules within information flow policies are processed in an administrator-defined order to determine if the traffic should be forwarded. By default, the TOE behavior is to deny packets when there is no rule match.

328 If the TOE implements a mechanism that ensures that no conflicting rules can be configured, the TSS shall describe the underlying mechanism.

Findings:	The TSS does not identify a mechanism that ensures no conflicting rules can be configured.

5.3.6.2 Guidance Documentation

329 The evaluator shall verify that the guidance documentation describes how the order of stateful traffic filtering rules is determined and provides the necessary instructions so that an administrator can configure the order of rule processing.

Findings:	[AGD] Section 3.3 states that rules are processed in the administrator defined order. “Rules can be inserted into the ruleset via GUI. Rules are associated to specific interfaces by defining the interface name in the rule. The GUI displays the order of the rules. If no rule matching the traffic exists, the traffic is denied.”

5.3.6.3 Tests

NIAP TD0545

a) Test 1: If the TOE implements a mechanism that ensures that no conflicting rules can be configured, the evaluator shall try to configure two conflicting rules and verify that the TOE rejects the conflicting rule(s). It is important to verify that the mechanism is implemented in the TOE but not in the non-TOE environment. If the TOE does not implement a mechanism that ensures that no conflicting rules can be configured, the evaluator shall devise two equal stateful traffic filtering rules with alternate operations – permit and drop. The rules should then be deployed in two distinct orders and in each case the evaluator shall ensure that the first rule is enforced in both cases by generating applicable packets and using packet capture and logs for confirmation.

High-Level Test Description
Create rules on the TOE with a distinct order and verify that the first rule is always enforced first. The evidence and findings can be found in FPF_RUL_EXT.1.5 Test 1
Findings: Pass – The evaluator confirmed that the TOE performs firewall filtering in the proper order in conjunction with FPF_RUL_EXT.1.5 Test 1.

b) Test 2: The evaluator shall repeat the procedure above, except that the two rules should be devised where one is a subset of the other (e.g., a specific address vs. a network segment). Again, the evaluator should test both

orders to ensure that the first is enforced regardless of the specificity of the rule.

High-Level Test Description
Create rules on the TOE with a distinct order and subset. The evaluator should also verify that the first rule is always enforces first. The evidence and findings can be found in FPF_RUL_EXT.1.5 Test 1
Findings: Pass – The evaluator confirmed that the TOE performs firewall filtering in the proper order in conjunction with FPF_FUL_EXT.1.5 Test 1.

5.3.7 FFW_RUL_EXT.1.9

5.3.7.1 TSS

330 The evaluator shall verify that the TSS describes the process for applying stateful traffic filtering rules and also that the behavior (either by default, or as configured by the administrator) is to deny packets when there is no rule match unless another required conditions allows the network traffic (i.e., FFW_RUL_EXT.1.5 or FFW_RUL_EXT.2.1).

Findings:	[ST] Section 6.5.1 states that traffic for known sessions is permitted to flow. For traffic not associated with known sessions, rules within information flow policies are processed in an administrator-defined order to determine if the traffic should be forwarded. By default, the TOE behavior is to deny packets when there is no rule match.

5.3.7.2 Guidance Documentation

331 The evaluator shall verify that the guidance documentation describes the behavior if no rules or special conditions apply to the network traffic. If the behavior is configurable, the evaluator shall verify that the guidance documentation provides the appropriate instructions to configure the behavior to deny packets with no matching rules.

Findings:	[AGD] Section 3.3 states that if no rule matching the traffic exists, the traffic is denied.

5.3.7.3 Tests

332 For each attribute in FFW_RUL_EXT.1.2, the evaluator shall construct a test to demonstrate that the TOE can correctly compare the attribute from the packet header to the ruleset, and shall demonstrate both the permit and deny for each case. It shall also be verified that a packet is dropped if no matching rule can be identified for the packet. The evaluator shall check the log in each case to confirm that the relevant rule was applied. The evaluator shall record a packet capture for each test to demonstrate the correct TOE behaviour.

High-Level Test Description
This test has been covered by the testing found in FPF_RUL_EXT.1.6 for both IPv4 and IPv6.

High-Level Test Description
Findings: Pass – The evaluator confirmed that the TOE enforces all rules properly in conjunction with FPF_RUL_EXT.1.6.

5.3.8 FFW_RUL_EXT.1.10

5.3.8.1 TSS

333 The evaluator shall verify that the TSS describes how the TOE tracks and maintains information relating to the number of half-open TCP connections. The TSS should identify how the TOE behaves when the administratively defined limit is reached and should describe under what circumstances stale half-open connections are removed (e.g. after a timer expires).

Findings:	[ST] Section 6.5.1 states that the TOE tracks the number of half-open TCP connections. When the configured limit for half-open TCP connections is exceeded, TCP SYN are discarded until the number drops below the configured limit. An audit record is generated if the firewall rule is configured to log these events. Half-open TCP connections are automatically discarded after a 60 second timeout period.

5.3.8.2 Guidance Documentation

334 The evaluator shall verify that the guidance documentation describes the behaviour of imposing TCP half-open connection limits and its default state if unconfigured. The evaluator shall verify that the guidance clearly indicates the conditions under which new connections will be dropped e.g. per-destination or per-client.

Findings:	[AGD] Section 3.3 states that the TOE tracks the number of half-open sessions to an administrator defined threshold. When threshold is exceeded, the TOE will discard TCP SYN packets and log the event. The default behaviour of the TOE is to discard half-open TCP connections after 60 seconds.

5.3.8.3 Tests

335 The following tests shall be run using IPv4 and IPv6.

a) Test 1: The evaluator shall define a TCP half-open connection limit on the TOE. The evaluator shall generate TCP SYN requests to pass through the TOE to the target system using a randomised source IP address and common destination IP address. The number of SYN requests should exceed the TCP half-open threshold defined on the TOE. TCP SYN-ACK messages should not be acknowledged. The evaluator shall verify through packet capture that once the defined TCP half-open threshold has been reached, subsequent TCP SYN packets are not transmitted to the target system. The evaluator shall verify that when the configured threshold is reached that, depending upon the selection, either a log entry is generated or a counter is incremented.

High-Level Test Description
The evaluator configured a ruleset to limit the half open TCP connections. The evaluator then sent modified traffic through the TOE until the threshold is met. Once the threshold is met, the TOE drops all the remaining traffic and it is logged.
Findings: Pass – The evaluator confirmed that the TOE enforces all rules regarding TCP half-open connections properly.

5.4 Security Management (FMT)

5.4.1 FMT_SMF.1/FFW Specification of Management Functions

5.4.1.1 TSS

336 The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).

Findings:	This has been addressed in the FMT_SMF.1 evaluation activities for the Base-PP. See section 2.4.4.1.

337 The evaluator shall examine the TSS and Guidance Documentation to verify they both describe the local administrative interface. The evaluator shall ensure the Guidance Documentation includes appropriate warnings for the administrator to ensure the interface is local.

Findings:	This has been addressed in the FMT_SMF.1 evaluation activities for the Base-PP. See section 2.4.4.1.

338 For distributed TOEs with the option 'ability to configure the interaction between TOE components' the evaluator shall examine that the ways to configure the interaction between TOE components is detailed in the TSS and Guidance Documentation. The evaluator shall check that the TOE behaviour observed during testing of the configured SFRs is as described in the TSS and Guidance Documentation.

Findings:	This has been addressed in the FMT_SMF.1 evaluation activities for the Base-PP. See section 2.4.4.1.

5.4.1.2 Guidance Documentation

339 See section 2.4.4.1.

5.4.1.3 Tests

340 The evaluator tests management functions as part of testing the SFRs identified in section 2.4.4. No separate testing for FMT_SMF.1 is required unless one of the management functions in FMT_SMF.1.1 has not already been exercised under any other SFR.

High-Level Test Description
The evaluator confirms that the management functions for FMT_SMF.1/FFW are covered with testing of the other SFRs.

High-Level Test Description
Findings: Pass – The evaluator confirmed that the FMT_SMF.1/FFW is performed throughout all FFW_RUL_EXT.1 and FPF_RUL_EXT.1 testing when the administrator sets firewall rules.

6 Evaluation Activities for Optional Requirements

6.1 Identification and Authentication (FIA)

6.1.1 FIA_X509_EXT.1/ITT X.509 Certificate Validation

6.1.1.1 TSS

341 The evaluator shall examine the TSS to ensure it describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). If selected, the TSS shall describe how certificate revocation checking is performed. It is not sufficient to verify the status of a X.509 certificate only when it's loaded onto the device.

Findings:	[ST] Section 6.6.2 – The TSS provides a description on when validity checks take place for certificates during authentication. The TSS also identifies what kind of checks take place including the checking of items such as extendedKeyUsage, expiration, etc. The TSS also identifies the revocation method the TOE uses for certificates.

6.1.1.2 Guidance Documentation

342 The evaluator shall also ensure that the guidance documentation describes where the check of validity of the certificates takes place, describes any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) and describe how certificate revocation checking is performed.

Findings:	[AGD] Section 3.12 describes the TOE's handling of X509 certificates. Section 3.12 also states that the validity, revocation, extendedKeyUsage, certificate chain and reference identifier checks of the certificates are done by the TOE. Revocation checks are performed using OCSP. Certificate validity checks are performed on certificate upload and during connections using certificates.

6.1.1.3 Tests

343 The evaluator shall demonstrate that checking the validity of a certificate is performed when a certificate is used in an authentication step. It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the device. The evaluator shall perform the following tests for FIA_X509_EXT.1.1/ITT. These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols.:

a) Test 1a: The evaluator shall present the TOE with a valid chain of certificates (terminating in a trusted CA certificate) as needed to validate the leaf certificate to be used in the function and shall use this chain to demonstrate that the function succeeds. Test 1a shall be designed in a way that the chain can be 'broken' in Test 1b by either being able to remove the trust anchor from the TOEs trust store, or by setting up the trust store in a way that at least one intermediate CA certificate needs to be provided, together with the leaf certificate from outside the TOE, to complete the chain (e.g. by storing only the root CA certificate in the trust store).

High-Level Test Description
Present the TOE with a valid chain of certificates and then attempt an IPsec connection to verify the chain can properly be used.
Findings: PASS – The evaluator confirmed that the TOE accepted a secure connection when a valid chain of certificates were used.

Test 1b: The evaluator shall then 'break' the chain used in Test 1a by either removing the trust anchor in the TOE's trust store used to terminate the chain, or by removing one of the intermediate CA certificates (provided together with the leaf certificate in Test 1a) to complete the chain. The evaluator shall show that an attempt to validate this broken chain fails.

High-Level Test Description
Remove a CA from the chain in test 1a and verify the connection attempt fails due to improper validation.
Findings: PASS – The evaluator confirmed that the TOE denies a connection when the chain can't be verified.

b) Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.

High-Level Test Description
Provide an expired certificate to the TOE during a VPN connection and verify the connection is unsuccessful.
Findings: PASS – The evaluator confirmed that the TOE denies a connection when it detects an expired certificate.

c) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates-–conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the TOE certificate and revocation of the TOE intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails. No testing is required if no revocation method is selected. Revocation checking is only applied to certificates that are not designated as trust anchors. Therefore, the revoked certificate(s) used for testing shall not be a trust anchor.

High-Level Test Description
Attempt a connection with unrevoked certificate using OCSP and verify the connection succeeds. Revoke the end entity certificate and then verify the connection fails due to the revoked certificate. After this, revoke an intermediate CA and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects connection when revoked certificates are provided. When those certificates are unrevoked, the TOE accepts the connection.

d) Test 4: If OCSP is selected, the evaluator shall configure the OCSP server or use a man-in-the-middle tool to present a certificate that does not have the OCSP

signing purpose and verify that validation of the OCSP response fails. If CRL is selected, the evaluator shall configure the CA to sign a CRL with a certificate that does not have the cRLsign key usage bit set and verify that validation of the CRL fails.

High-Level Test Description
Provide an OCSP response to the TOE that is signed by a certificate not containing the OCSP sign EKU. Verify when the TOE receives this OCSP response, it is ignored, and the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects OCSP responses that do not contain the OCSP sign EKU in the signing certificate.

e) Test 5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)

High-Level Test Description
Modify a byte in any of the first eight bytes of the certificate and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection attempt after the certificate had a byte in the first eight bytes modified.

f) Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

High-Level Test Description
Modify a byte in any of the last eight bytes of the certificate and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection attempt after the certificate had a byte in the last eight bytes modified.

g) Test 7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The hash of the certificate will not validate.)

High-Level Test Description
Modify the public key of the certificate and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection attempt after the certificate had the public key modified.

NIAP TD0527 (REVISED 1 December 2020)

344 The following tests are run when a minimum certificate path length of three certificates is implemented.

345 Test 8: (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). The evaluator shall conduct the following tests:

346 Test 8a: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC

Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a valid chain of EC certificates (terminating in a trusted CA certificate), where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

Test Not Applicable
Findings: N/A - The TOE does not process CA certificates presented in the certificate message.

347 Test 8b: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a chain of EC certificates (terminating in a trusted CA certificate), where the intermediate certificate in the certificate chain uses an explicit format version of the Elliptic Curve parameters in the public key information field, and is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

Test Not Applicable
Findings: N/A – The TOE does not process CA certificates presented in the certificate message.

348 Test 8c: The evaluator shall establish a subordinate CA certificate, where the elliptic curve parameters are specified as a named curve, that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is accepted into the TOE's trust store. The evaluator shall then establish a subordinate CA certificate that uses an explicit format version of the elliptic curve parameters, and that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is rejected, and not added to the TOE's trust store.

High-Level Test Description
Testing for this test case can be found in FIA_X509_EXT.1/Rev Test 8c.
Findings: PASS—The evaluator confirmed in conjunction with FIA_X509_EXT.1/Rev Test 8c that the TOE accepts subordinate CA certificates with a named curve and rejects subordinate certificates with explicit curve parameters.

6.1.2 FIA_X509_EXT.1.2/ITT

349 The evaluator shall perform the following tests for FIA_X509_EXT.1.2/ITT. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/ITT. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. Where the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) then the associated extendedKeyUsage rule testing may be omitted

350 The goal of the following tests is to verify that the TOE accepts a certificate as a CA certificate only if it has been marked as a CA certificate by using basicConstraints

with the CA flag set to True (and implicitly tests that the TOE correctly parses the basicConstraints extension as part of X509v3 certificate chain validation).

351 For each of the following tests the evaluator shall create a chain of at least two certificates: a self-signed root CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

a) Test 1: The evaluator shall ensure that one CA in the chain does not contain the basicConstraints extension. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate without the basicConstraints extension to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).

High-Level Test Description
Testing for this test case can be found in FIA_X509_EXT.1.2/Rev Test 1.
Findings: PASS—The evaluator confirmed that the TOE rejects CA certificates that do not contain the basicConstraints extension in conjunction with FIA_X509_EXT.1.2/Rev Test 1.

b) Test 2: The evaluator shall ensure that at least one of the CA certificates in the chain has a basicConstraints extension in which the CA flag is set to FALSE. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate with the CA flag set to FALSE to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).

High-Level Test Description
Testing for this test case can be found in FIA_X509_EXT.1.2/Rev Test 2.
Findings: PASS—The evaluator confirmed that the TOE rejects CA certificates with the basicConstraints extension set to FALSE in conjunction with FIA_X509_EXT.1.2/Rev Test 1.

6.2 Protection of the TSF (FPT)

6.2.1 FPT_ITT.1 Basic internal TSF data transfer protection

352 If the TOE is not a distributed TOE, then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

6.2.1.1 TSS

353 The evaluator shall examine the TSS to determine that, for all communications between components of a distributed TOE, each communications mechanism is identified in terms of the allowed protocols for that IT entity. The evaluator shall also confirm that all protocols listed in the TSS for these inter-component communications are specified and included in the requirements in the ST.

Findings:	[ST] Section 6.8.3 – The TSS identifies the communication mechanism used between TOE components to be IPsec. This protocol is included in the requirements of the ST.

6.2.1.2 Guidance Documentation

354 The evaluator shall confirm that the guidance documentation contains instructions for establishing the relevant allowed communication channels and protocols between each pair of authorized TOE components, and that it contains recovery instructions should a connection be unintentionally broken.

Findings:	[AGD] Section 3.8 describes the IPsec parameters that can be configured for communication between the GSS and the Cube. Additionally, this section describes the configurable number of retries to re-establish a connection in the event of a failure or the connection is unintentionally broken. Once the configured threshold is met, the Cube can be restarted to initiate the connection again. Section 3.8.2 also describes the configuration needed on the GoSilent Cube to add the Virtual Server profile to connect to the GoSilent Server. Section 2.2 describes how the GoSilent Cube is registered with the GoSilent Server to enable communication via the 'VPN Clients' tab on the GoSilent Server.

6.2.1.3 Tests

355 The evaluator shall perform the following tests:

a) Test 1: The evaluator shall ensure that communications using each protocol between each pair of authorized TOE components is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

High-Level Test Description
Testing for this test case can be found throughout the FIA_X509_EXT.1/ITT and FCS_IPSEC_EXT.1 test cases.
Findings: PASS—The evaluator confirmed that the communications for each TOE component is successful in conjunction with FIA_X509_EXT.1/ITT and FCS_IPSEC_EXT.1.

b) Test 2: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

High-Level Test Description
Testing for this test case can be found throughout the FIA_X509_EXT.1/ITT and FCS_IPSEC_EXT.1 test cases.
Findings: PASS—The evaluator confirmed that the communications for each TOE component is not sent in plaintext in conjunction with FIA_X509_EXT.1/ITT and FCS_IPSEC_EXT.1.

c) Test 3: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route between distributed components.

The evaluator shall ensure that, for each different pair of nonequivalent component types, the connection is physically interrupted for the following durations: i) a duration that exceeds the TOE's application layer timeout setting, ii) a duration that is shorter than the application layer timeout but is of sufficient length to interrupt the network link layer.

The evaluator shall ensure that when physical connectivity is restored, either communications are appropriately protected, or the secure channel is terminated

and the registration process (as described in the FTP_TRP.1/Join) re-initiated, with the TOE generating adequate warnings to alert the Security Administrator.

In the case that the TOE is able to detect when the cable is removed from the device, another physical network device (e.g. a core switch) shall be used to interrupt the connection between the components.

The interruption shall not be performed at the virtual node (e.g. virtual switch) and must be physical in nature.

356 Further assurance activities are associated with the specific protocols.

High-Level Test Description
Successfully bring up a VPN connection and verify that when communication is interrupted, it is properly restored when brought back online. Repeat this test for short (network link) and long (application layer) interruptions.
Findings: PASS – The evaluator confirmed that the TOE properly restores secure communication after the short and long interruptions. In the event of the short (network link) interruption, the application continued with the existing IPsec Security Association when the connection was restored. For the long (application layer) interruption, the TOE initiated a new IPsec Security Association when the connection was restored.

6.3 Communication (FCO)

6.3.1 FCO_CPC_EXT.1 Component Registration Channel Definition

357 If the TOE is not a distributed TOE, then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below. In carrying out these activities the evaluator shall determine answers to the following questions based on a combination of documentation analysis and testing (possibly also using input from carrying out the Evaluation Activities for the relevant registration channel, such as FTP_TRP.1/Join), and shall report the answers.

a) What stops[3] a component from successfully communicating with TOE components (in a way that enables it to participate as part of the TOE) before it has properly authenticated and joined the TOE?

b) What is the enablement step? (Describe what interface it uses, with a reference to the relevant section and step in the operational guidance).

[3] The intent of the phrasing “what stops…” as opposed to “what secures…” is for the evaluator to pursue the answer to its lowest level of dependency, i.e. a level at which the security can clearly be seen to depend on things that are under appropriate control. For example, a channel may be protected by a public key that is provided to the relying party in a self-signed certificate. This enables cryptographic mechanisms to be applied to provide authentication (and therefore invites an answer that “the check on the public key certificate secures…”), but does not ultimately stop an attacker from apparently authenticating because the attacker can produce their own self-signed certificate. The question “what stops an unauthorised component from successfully communicating…” focuses attention on what an attacker needs to do, and therefore pushes the answer down to the level of whether a self-signed certificate could be produced by an attacker. Similarly, a well-known key, or a key that is common to a type of device rather than an individual device, may be used in a confidentiality mechanism but does not provide confidentiality because an attacker can find the well-known key or obtain his own instance of a device containing the non-unique key.

1) What stops anybody other than a Security Administrator from carrying out this step?

2) How does the Security Administrator know that they are enabling the intended component to join? (Identification of the joiner might be part of the enablement action itself or might be part of secure channel establishment, but it must prevent unintended joining of components)

c) What stops a component successfully joining if the Security Administrator has not carried out the enablement step; or, equivalently, how does the TOE ensure that an action by an authentic Security Administrator is required before a component can successfully join?

d) What stops a component from carrying out the registration process over a different, insecure channel?

e) If the FTP_TRP.1/Join channel type is selected in FCO_CPC_EXT.1.2 then how do the registration process and its secure channel ensure that the data is protected from disclosure and provides detection of modification?

f) Where the registration channel does not rely on protection of the registration environment, does the registration channel provide a sufficient level of protection (especially with regard to confidentiality) for the data that passes over it?

g) Where the registration channel is subsequently used for normal internal communication between TOE components (i.e. after the joiner has completed registration), do any of the authentication or encryption features of the registration channel result in use of a channel that has weaker protection than the normal FPT_ITT.1 requirements for such a channel?

h) What is the disablement step? (Describe what interface it uses, with a reference to the relevant section and step in the operational guidance).

i) What stops a component successfully communicating with other TOE components if the Security Administrator has carried out the disablement step?

6.3.1.1 TSS

358 (Note: paragraph 253 lists questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities.)

359 The evaluator shall examine the TSS to confirm that it:

a) Describes the method by which a Security Administrator enables and disables communications between pairs of TOE components.

b) Describes the relevant details according to the type of channel in the main selection made in FCO_CPC_EXT.1.2:

- First type: the TSS identifies the relevant SFR iteration that specifies the channel used

- Second type: the TSS (with support from the operational guidance if selected in FTP_TRP.1.3/Join) describes details of the channel and the mechanisms that it uses (and describes how the process ensures that the key is unique to the pair of components) – see also the Evaluation Activities for FTP_TRP.1/Join.

360 The evaluator shall confirm that if any aspects of the registration channel are identified as not meeting FTP_ITC.1 or FPT_ITT.1, then the ST has also selected the FTP_TRP.1/Join option in the main selection in FCO_CPC_EXT.1.2.

Findings:	[ST] Section 6.2.1 describes how the GoSilent Cube is registered by an administrator on the GoSilent Server with the X.509 certificates for each component communicated out of band and imported by the administrator. This section also describes that the GoSilent Server will not establish an IPsec tunnel with a GoSilent Cube if the enablement process has not been completed. This section also states that a cube can be suspended from within the GSS GUI, whereby subsequent connections from the cube are rejected (the cube is disabled). This section also states that the channel described in FCO_CPC_EXT.1 is used to finalize the registration process during the first connection of a Cube to GSS.

6.3.1.2 Guidance Documentation

361 (Note: paragraph 253 lists questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities.)

362 The evaluator shall examine the guidance documentation to confirm that it contains instructions for enabling and disabling communications with any individual component of a distributed TOE. The evaluator shall confirm that the method of disabling is such that all other components can be prevented from communicating with the component that is being removed from the TOE (preventing the remaining components from either attempting to initiate communications to the disabled component, or from responding to communications from the disabled component).

Findings:	[AGD] Section 2.2 describes how to enable communications between the GoSilent Server and the GoSilent Cube via the 'VPN Clients' tab on the GSS. This section also describes how to disable communications using the 'Suspend' option on the GSS. The TOE does not support any other communications between TOE components.

363 The evaluator shall examine the guidance documentation to confirm that it includes recovery instructions should a connection be unintentionally broken during the registration process.

Findings:	[AGD] Section 2.2 covers the process to enable and disable communications between the GoSilent Server and the GoSilent Cube. Additionally, this section describes the process to check the network connections for the GoSilent Server and GoSilent Cube and the 'Remove' option along with a reboot of the GoSilent Cube in the event of a failure or the connection is unintentionally broken.

364 If the TOE uses a registration channel for registering components to the TOE (i.e. where the ST author uses the FTP_ITC.1/FPT_ITT.1 or FTP_TRP.1/Join channel types in the main selection for FCO_CPC_EXT.1.2) then the evaluator shall examine the Preparative Procedures to confirm that they:

a) describe the security characteristics of the registration channel (e.g. the protocol, keys and authentication data on which it is based) and shall highlight any aspects which do not meet the requirements for a steadystate inter-component channel (as in FTP_ITC.1 or FPT_ITT.1)

b) identify any dependencies between the configuration of the registration channel and the security of the subsequent inter-component communications (e.g. where AES-256 inter-component communications depend on transmitting 256 bit keys between components and therefore rely on the registration channel being configured to use an equivalent key length)

c) identify any aspects of the channel can be modified by the operational environment in order to improve the channel security and shall describe how this modification can be achieved (e.g. generating a new key pair, or replacing a default public key certificate).

Findings:	[AGD] Section 3.8 specifies the IPsec protocol parameters to be configured for the GSS and the GoSilent Cubes to communicate. These parameters include the protocol mode, cipher suite, security association lifetimes, child association lifetimes and OCSP checking. By enabling OCSP checking, the TOE will use the operational environment to check the revocation status of certificates, thus improving security. [AGD] Section 3.12 also specifies how to enable CN and O checking of certificates for both the GSS and the Cubes.

365 As background for the examination of the registration channel description, it is noted that the requirements above are intended to ensure that administrators can make an accurate judgement of any risks that arise from the default registration process. Examples would be the use of self-signed certificates (i.e. certificates that are not chained to an external or local Certification Authority), manufacturer-issued certificates (where control over aspects such as revocation, or which devices are issued with recognised certificates, is outside the control of the operational environment), use of generic/non-unique keys (e.g. where the same key is present on more than one instance of a device), or well-known keys (i.e. where the confidentiality of the keys is not intended to be strongly protected – note that this need not mean there is a positive action or intention to publicise the keys).

366 In the case of a distributed TOE for which the ST author uses the FTP_TRP.1/Join channel type in the main selection for FCO_CPC_EXT.1.2 and the TOE relies on the operational environment to provide security for some aspects of the registration channel security then there are additional requirements on the Preparative Procedures as described in section 3.4.1.2.

Findings:	N/A—The TOE does not use the FTP_TRP.1/Join channel type.

6.3.1.3 Tests

367 (Note: paragraph 253 lists questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities.)

368 The evaluator shall carry out the following tests:

a) Test 1.1: the evaluator shall confirm that an IT entity that is not currently a member of the distributed TOE cannot communicate with any component of the TOE until the non-member entity is enabled by a Security Administrator for each of the non-equivalent TOE components[4] that it is required to communicate with

[4] An 'equivalent TOE component' is a type of distributed TOE component that exhibits the same security characteristics, behaviour and role in the TSF as some other TOE component. In principle a distributed TOE could operate with only one instance of each equivalent TOE component, although the minimum configuration of the distributed TOE may include more than one instance (see discussion of the minimum configuration of a distributed TOE, in section A.9). In practice a deployment of the TOE may include more than one instance of some equivalent TOE components for practical reasons, such as performance or the need to have separate instances for separate subnets or VLANs.

(non-equivalent TOE components are as defined in the minimum configuration for the distributed TOE)

High-Level Test Description
Verify when an IT entity is not enabled to communicate with the TOE, there is no communication between the devices.
Findings: PASS – The evaluator confirmed that non-enabled TOE components cannot communicate with each other until they are enabled.

b) Test 1.2: the evaluator shall confirm that after enablement, an IT entity can communicate only with the components that it has been enabled for. This includes testing that the enabled communication is successful for the enabled component pair, and that communication remains unsuccessful with any other component for which communication has not been explicitly enabled

Some TOEs may set up the registration channel before the enablement step is carried out, but in such a case the channel must not allow communications until after the enablement step has been completed.

High-Level Test Description
Enable the devices to communicate with each other and verify that the connection is successful.
Findings: PASS – The evaluator confirmed that after a TOE component is enabled, that TOE component can successfully communicate with the GoSilent Server.

369 The evaluator shall repeat Tests 1.1 and 1.2 for each different type of enablement process that can be used in the TOE.

c) Test 2: The evaluator shall separately disable each TOE component in turn and ensure that the other TOE components cannot then communicate with the disabled component, whether by attempting to initiate communications with the disabled component or by responding to communication attempts from the disabled component.

High-Level Test Description
Disable an already enabled TOE component and verify that the communications are not allowed after disabled.
Findings: PASS – The evaluator confirmed that after a TOE component is disabled, the TOE component cannot communicate with the GoSilent Server anymore.

d) Test 3: The evaluator shall carry out the following tests according to those that apply to the values of the main (outer) selection made in the ST for FCO_CPC_EXT.1.2.

1) If the ST uses the first type of communication channel in the selection in FCO_CPC_EXT.1.2 then the evaluator tests the channel via the Evaluation Activities for FTP_ITC.1 or FPT_ITT.1 according to the second selection – the evaluator shall ensure that the test coverage for these SFRs includes their use in the registration process.

2) If the ST uses the second type of communication channel in the selection in FCO_CPC_EXT.1.2 then the evaluator tests the channel via the Evaluation Activities for FTP_TRP.1/Join.

3) If the ST uses the ‘no channel’ selection, then no test is required.

High-Level Test Description
Testing for this test case can be found throughout the FPT_ITT.1 test cases.
Findings: PASS—The evaluator confirmed that the TOE components communicate over IPsec after the registration process is completed in conjunction with FPT_ITT.1.

e) Test 4: The evaluator shall perform one of the following tests, according to the TOE characteristics identified in its TSS and operational guidance:

1) If the registration channel is not subsequently used for intercomponent communication, and in all cases where the second selection in FCO_CPC_EXT.1.2 is made (i.e. using FTP_TRP.1/Join) then the evaluator shall confirm that the registration channel can no longer be used after the registration process has completed, by attempting to use the channel to communicate with each of the endpoints after registration has completed

2) If the registration channel is subsequently used for intercomponent communication then the evaluator shall confirm that any aspects identified in the operational guidance as necessary to meet the requirements for a steady-state intercomponent channel (as in FTP_ITC.1 or FPT_ITT.1) can indeed be carried out (e.g. there might be a requirement to replace the default key pair and/or public key certificate).

High-Level Test Description
Testing for this test case can be found throughout the FPT_ITT.1 test cases.
Findings: PASS—The evaluator confirmed that the TOE components communicate over IPsec after the registration process is completed in conjunction with FPT_ITT.1.

f) Test 5: For each aspect of the security of the registration channel that operational guidance states can be modified by the operational environment in order to improve the channel security (cf. AGD_PRE.1 refinement item 2 in (cf. the requirements on Preparative Procedures in 3.5.1.2), the evaluator shall confirm, by following the procedure described in the operational guidance, that this modification can be successfully carried out.

High-Level Test Description
Follow the Guidance provided to enable the registration of a TOE component using OCSP as idenfied in the TSS.
Findings: PASS – The FCO_CPC_EXT.1 Test 1.1 and Test 1.2 show the components must be configured through the enablement process to communicate. FIA_X509_EXT.1/ITT Test 3 shows the TOE successfully using OCSP to check the revocation status of the certificates before accepting the connection.

7 Evaluation Activities for Selection-Based Requirements

7.1 Security Audit (FAU)

7.1.1 FAU_GEN_EXT.1 Security Audit Data Generation for Distributed TOE Components

370 For distributed TOEs, the requirements on TSS, Guidance Documentation and Tests regarding FAU_GEN_EXT.1 are already covered by the corresponding requirements for FAU_GEN.1.

7.1.2 FAU_STG_EXT.4 Protected Local audit event storage for distributed TOEs & FAU_STG_EXT.5 Protected Remote audit event storage for Distributed TOEs

7.1.2.1 TSS

371 The evaluator examines the TSS to confirm that it describes which TOE components store their security audit events locally and which send their security audit events to other TOE components for local storage. For the latter, the target TOE component(s) which store security audit events for other TOE components shall be identified. For every sending TOE component the corresponding receiving TOE component(s) need to be identified. For every transfer of audit information between TOE components it shall be described how the data is secured during transfer according to FTP_ITC.1 or FPT_ITT.1.

Findings:	[ST] Section 6.1.2 – The TSS identifies that the TOE stores audit event records on the respective components on which they were generated. The TSS also specifies that each component can send their audit logs to an external server using protocols found in FTP_ITC.1.

372 For each TOE component which does not store audit events locally by itself, the evaluator confirms that the TSS describes how the audit information is buffered before sending to another TOE component for local storage.

Findings:	[ST] Section 6.1.2 – The TSS states that audit data is stored locally on each TOE component.

7.1.2.2 Guidance Documentation

373 The evaluator shall examine the guidance documentation to ensure that it describes how the link between different TOE components is established if audit data is exchanged between TOE components for local storage. The guidance documentation shall describe all possible configuration options for local storage of audit data and provide all instructions how to perform the related configuration of the TOE components.

Findings:	[AGD] Section 3.10 states that each TOE component stores its own audit records locally, up to 50MB of audit data. There are no further configurations for local audit data and the TOE does not share audit data between components.

374 The evaluator shall also ensure that the guidance documentation describes for every TOE component which does not store audit information locally how audit information is buffered before transmission to other TOE components.

Findings:	N/A – All components of the TOE store local audit information

7.1.2.3 Tests

375 For at least one of each type of distributed TOE components (sensors, central nodes, etc.), the following tests shall be performed using distributed TOEs.

376 Test 1: For each type of TOE component, the evaluator shall perform a representative subset of auditable actions and ensure that these actions cause the generation of appropriately formed audit records. Generation of such records can be observed directly on the distributed TOE component (if there is appropriate interface), or indirectly after transmission to a central location.

High-Level Test Description
This requirement is performed in conjunction with FAU_GEN.1. All TOE components produce the required audit events for the auditable actions.
Findings: PASS. The evaluator confirmed in conjunction with FAU_GEN.1 that all required audit events are produced by the expected component.

377 Test 2: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to the external audit server (as specified in FTP_ITC.1), the evaluator shall configure a trusted channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

High-Level Test Description
This requirement is performed in conjunction with FCS_TLSC_EXT.1 Test 1.
Findings: PASS. The evaluator confirmed that each TOE component transmits its audit records to the external syslog server via TLS in conjunction with FCS_TLSC_EXT.1 Test 1.

378 Test 3: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to another TOE component (as specified in FTP_ITT.1 or FTP_ITC.1, respectively), the evaluator shall configure a secure channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

379 While performing these tests, the evaluator shall verify that the TOE behaviour observed during testing is consistent with the descriptions provided in the TSS and the Guidance Documentation. Depending on the TOE configuration, there might be a large number of different possible configurations. In such cases, it is acceptable to

perform subset testing, accompanied by an equivalency argument describing the evaluator's sampling methodology.

High-Level Test Description
This requirement is performed in conjunction with FCS_TLSC_EXT.1 Test 1.
Findings: PASS. The evaluator confirmed that each TOE component transmits its audit records to the external syslog server via TLS in conjunction with FCS_TLSC_EXT.1 Test 1. All transmission of audit events are included in the protected TLS channel. No audit events are transmitted in the clear.

7.2 Cryptographic Support (FCS)

7.2.1 FCS_HTTPS_EXT.1 HTTPS Protocol

7.2.1.1 TSS

380 The evaluator shall examine the TSS and determine that enough detail is provided to explain how the implementation complies with RFC 2818.

Findings:	[ST] Section 6.3.9 – The TSS describes that the TOE acts as a TLS/HTTPS server to provide a web GUI to administrators. Key establishment if performed using ECDH P-384 keys and the connection is always initiated by the remote end. The evaluator determined that the TSS provides enough detail to explain how the implementation complies with RFC 2818.

7.2.1.2 Guidance Documentation

381 The evaluator shall examine the guidance documentation to verify it instructs the Administrator how to configure TOE for use as an HTTPS client or HTTPS server.

Findings:	[AGD] Section 3.3.3.1 – The AGD provides instructions on how to configure the TOE to provide an Administrative GUI interface over HTTPS to administrators operating as GoSilent Cube clients.

7.2.1.3 Tests

382 This test is now performed as part of FIA_X509_EXT.1/Rev testing.

383 Tests are performed in conjunction with the TLS evaluation activities.

384 If the TOE is an HTTPS client or an HTTPS server utilizing X.509 client authentication, then the certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1.

7.2.2 FCS_IPSEC_EXT.1 IPsec Protocol

7.2.2.1 TSS

FCS_IPSEC_EXT.1.1

385 The evaluator shall examine the TSS and determine that it describes what takes place when a packet is processed by the TOE, e.g., the algorithm used to process the packet. The TSS describes how the SPD is implemented and the rules for processing both inbound and outbound packets in terms of the IPsec policy. The TSS describes the rules that are available and the resulting actions available after matching a rule.

The TSS describes how those rules and actions form the SPD in terms of the BYPASS (e.g., no encryption), DISCARD (e.g., drop the packet), and PROTECT (e.g., encrypt the packet) actions defined in RFC 4301.

386 As noted in section 4.4.1 of RFC 4301, the processing of entries in the SPD is non-trivial and the evaluator shall determine that the description in the TSS is sufficient to determine which rules will be applied given the rule structure implemented by the TOE. For example, if the TOE allows specification of ranges, conditional rules, etc., the evaluator shall determine that the description of rule processing (for both inbound and outbound packets) is sufficient to determine the action that will be applied, especially in the case where two different rules may apply. This description shall cover both the initial packets (that is, no SA is established on the interface or for that particular packet) as well as packets that are part of an established SA.

Findings:	[ST] Section 6.3.7 states that the GSS has a nominal SPD that denies all traffic. Once an IPsec tunnel is established, an SPD entry is created to permit all traffic from the IP address of the GoSilent Cube. Once the IPsec tunnel is up, the TOE only allows traffic through the IPsec tunnel.

FCS_IPSEC_EXT.1.3

387 The evaluator checks the TSS to ensure it states that the VPN can be established to operate in transport mode and/or tunnel mode (as identified in FCS_IPSEC_EXT.1.3).

Findings:	[ST] Section 6.3.7 - The TSS identifies tunnel mode as the only supported mode for VPN traffic.

FCS_IPSEC_EXT.1.4

388 The evaluator shall examine the TSS to verify that the selected algorithms are implemented. In addition, the evaluator ensures that the SHA-based HMAC algorithm conforms to the algorithms specified in FCS_COP.1/KeyedHash Cryptographic Operations (for keyed-hash message authentication) and if the SHA-based HMAC function truncated output is utilized it must also be described.

Findings:	[ST] Section 6.3.7 – The TSS provides a list of supported algorithms implemented by the TOE. The list of SHA-based HMAC algorithms conform to the algorithms specified in FCS_COP.1/KeyedHash.

FCS_IPSEC_EXT.1.5

389 The evaluator shall examine the TSS to verify that IKEv1 and/or IKEv2 are implemented.

390 For IKEv1 implementations, the evaluator shall examine the TSS to ensure that, in the description of the IPsec protocol, it states that aggressive mode is not used for IKEv1 Phase 1 exchanges, and that only main mode is used. It may be that this is a configurable option.

Findings:	[ST] Section 6.3.7 – The TSS identifies IKEv2 as the only supported key exchange method.

FCS_IPSEC_EXT.1.6

391 The evaluator shall ensure the TSS identifies the algorithms used for encrypting the IKEv1 and/or IKEv2 payload, and that the algorithms chosen in the selection of the requirement are included in the TSS discussion.

Findings:	[ST] Section 6.3.7 – The TSS identifies AES in both CBC and GCM modes as well as 128 and 256 bit keys for use with IKEv2.

FCS_IPSEC_EXT.1.7

392 The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 1 SA lifetime and/or the IKEv2 SA lifetime. The evaluator shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.

Findings:	[ST] Section 6.3.7 – The TSS identifies only IKEv2 support and SA lifetimes to be configurable based on a time period between 1 and 24 hours. This selection corresponds to the selection in FCS_IPSEC_EXT.1.5.

FCS_IPSEC_EXT.1.8

393 The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 2 SA lifetime and/or the IKEv2 Child SA lifetime. The evaluator shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.

Findings:	[ST] Section 6.3.7 – The TSS specifies only IKEv2 support and Child SA lifetimes to be configurable based on a time period between 1 and 8 hours. This selection corresponds to the selection in FCS_IPSEC_EXT.1.5.

FCS_IPSEC_EXT.1.9

394 The evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating "x". The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of "x" meets the stipulations in the requirement.

Findings:	[ST] Section 6.3.7 – The TSS identifies that the random secret value “x” is generated by the DRBG and the length of “x” is twice the security strength of the associated DH group. The random number generated meets the requirement in this PP and the length of “x” meets the stipulations of the requirement.

FCS_IPSEC_EXT.1.10

395 If the first selection is chosen, the evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	[ST] Section 6.3.7 – The TSS describes that the nonce is generated by the DRBG and is twice the security strength of the associated DH group. The random number generated meets the requirements in this PP and the length of the nonces meet the stipulations in the requirement.

396 If the second selection is chosen, the evaluator shall check to ensure that, for each PRF hash supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	N/A – The second selection is not chosen

FCS_IPSEC_EXT.1.11

397 The evaluator shall check to ensure that the DH groups specified in the requirement are listed as being supported in the TSS. If there is more than one DH group supported, the evaluator checks to ensure the TSS describes how a particular DH group is specified/negotiated with a peer.

Findings:	[ST] Section 6.3.7 – The TSS identifies all DH groups supported by the TOE. [ST] Section 6.3.7 includes a footnote that describes how DH groups are specified.

FCS_IPSEC_EXT.1.12

398 The evaluator shall check that the TSS describes the potential strengths (in terms of the number of bits in the symmetric key) of the algorithms that are allowed for the IKE and ESP exchanges. The TSS shall also describe the checks that are done when negotiating IKEv1 Phase 2 and/or IKEv2 CHILD_SA suites to ensure that the strength (in terms of the number of bits of key in the symmetric algorithm) of the negotiated algorithm is less than or equal to that of the IKE SA this is protecting the negotiation.

Findings:	[ST] Section 6.3.7 – The TSS identifies that IKEv2 and ESP exchanges support AES in CBC and GCM mode using 128- or 256-bit keys. During the IKEv2 CHILD_SA process, the TOE performs checks to ensure that encryption strengths for the selected algorithm are less than or equal to that of the IKEv2 SA.

FCS_IPSEC_EXT.1.13

399 The evaluator ensures that the TSS identifies RSA and/or ECDSA as being used to perform peer authentication. The description must be consistent with the algorithms as specified in FCS_COP.1/SigGen Cryptographic Operations (for cryptographic signature).

Findings:	[ST] Section 6.3.7 – The TSS identifies ECDSA as being used to perform peer authentication. This is consistent with the algorithms specified in FCS_COP.1/SigGen, [ST] Section 6.3.4.

400 If pre-shared keys are chosen in the selection, the evaluator shall check to ensure that the TSS describes how pre-shared keys are established and used in authentication of IPsec connections. The description in the TSS shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

Findings:	Pre-shared keys are not used for peer authentication.

FCS_IPSEC_EXT.1.14

401 The evaluator shall ensure that the TSS describes how the TOE compares the peer's presented identifier to the reference identifier. This description shall include which field(s) of the certificate are used as the presented identifier (DN, Common Name, or SAN). If the TOE simultaneously supports the same identifier type in the CN and SAN, the TSS shall describe how the TOE prioritizes the comparisons (e.g. the result of comparison if CN matches but SAN does not). If the location (e.g. CN or SAN) of non-DN identifier types must explicitly be configured as part of the reference identifier, the TSS shall state this. If the ST author assigned an additional identifier type, the TSS description shall also include a description of that type and the method by which that type is compared to the peer's presented certificate, including what field(s) are compared and which fields take precedence in the comparison.

Findings:	[ST] Section 6.3.7 states that Distinguished Names (DN) are used as reference and presented identifiers.

7.2.2.2 Guidance Documentation

FCS_IPSEC_EXT.1.1

402 The evaluator shall examine the guidance documentation to verify it instructs the Administrator how to construct entries into the SPD that specify a rule for processing a packet. The description includes all three cases – a rule that ensures packets are encrypted/decrypted, dropped, and flow through the TOE without being encrypted. The evaluator shall determine that the description in the guidance documentation is consistent with the description in the TSS, and that the level of detail in the guidance documentation is sufficient to allow the administrator to set up the SPD in an unambiguous fashion. This includes a discussion of how ordering of rules impacts the processing of an IP packet.

Findings:	[AGD] section 3.3.2 states the TOE enforces a default ruleset that encrypts all traffic between the GoSilent Cube and GSS. Modification of these rules is strongly discouraged and outside of the scope of the evaluated configuration. The evaluator confirmed this is consistent with the description in the TSS. Traffic coming from the Cube is protected with IPsec by default to the GSS TOE component, and passing through the GSS is subject to the stateful traffic filter firewall rules which are described in [AGD] section 3.3. The evaluator confirmed that [AGD] section 3.3 describes how to construct entries into the SPD to PROTECT, DROP and BYPASS and is sufficient to allow the administrator to set up the SPD in an unambiguous fashion. The TOE enforces a default ruleset that encrypts [PROTECT] all traffic between the GoSilent Cube and GoSilent Server (GSS). Modification of these default rules, or otherwise bypass encryption between the Cube and GSS is strongly discouraged and outside the scope of the evaluated configuration.

FCS_IPSEC_EXT.1.3

403 The evaluator shall confirm that the guidance documentation contains instructions on how to configure the connection in each mode selected.

Findings:	[AGD] Section 3.8 specifies the TOE configuration parameters for IPsec. The TOE only supports tunnel mode; thus no further configuration is needed.

FCS_IPSEC_EXT.1.4

404 The evaluator checks the guidance documentation to ensure it provides instructions on how to configure the TOE to use the algorithms selected.

Findings:	[AGD] Section 3.8 specifies the algorithms supported by the TOE and how to configure them.

FCS_IPSEC_EXT.1.5

405 The evaluator shall check the guidance documentation to ensure it instructs the administrator how to configure the TOE to use IKEv1 and/or IKEv2 (as selected), and how to configure the TOE to perform NAT traversal (if selected).

Findings:	The TOE only supports IKEv2. This is made clear in the [AGD] section 3.8 to configure the TOE to use 'IKEv2_Certificates' for 'IPsec Protocol Mode'. Section 3.3.1 states that the NAT table can be configured to handle the processing of NAT related traffic.

406 If the IKEv1 Phase 1 mode requires configuration of the TOE prior to its operation, the evaluator shall check the guidance documentation to ensure that instructions for this configuration are contained within that guidance.

Findings:	N/A – The TOE does not support IKEv1.

FCS_IPSEC_EXT.1.6

407 The evaluator ensures that the guidance documentation describes the configuration of all selected algorithms in the requirement.

Findings:	[AGD] Section 3.8 specifies the algorithms supported by the TOE and how to configure them.

FCS_IPSEC_EXT.1.7

NIAP TD0633

408 The evaluator shall verify that the values for SA lifetimes can be configured and that the instructions for doing so are located in the guidance documentation. If time-based limits are supported, configuring the limit may lead to a rekey no later than the specified limit. For some implementations, it may be necessary, though, to configure the TOE with a lower time value to ensure a rekey is performed before the maximum SA lifetime of 24 hours is exceeded (e.g. configure a time value of 23h 45min to ensure the actual rekey is performed no later than 24h). The evaluator shall verify that the guidance documentation allows the Administrator to configure the Phase 1 SA value of 24 hours or provides sufficient instruction about the time value to configure to ensure the rekey is performed no later than the maximum SA lifetime of 24 hours. It is not permitted to configure a value of 24 hours if that leads to an actual rekey after more than 24hours. Currently there are no values mandated for the number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

Findings:	The TOE only supports time-based limits. [AGD] Section 3.8 specifies how this can be configured between 1 and 24 hours.

FCS_IPSEC_EXT.1.8

NIAP TD0633

409 The evaluator shall verify that the values for SA lifetimes can be configured and that the instructions for doing so are located in the guidance documentation. If time-based limits are supported, configuring the limit may lead to a rekey no later than the specified limit. For some implementations, it may be necessary, though, to configure the TOE with a lower time value to ensure a rekey is performed before the maximum SA lifetime of 8 hours is exceeded (e.g. configure a time value of 7h 45min to ensure the actual rekey is performed no later than 8h). The evaluator shall verify that the guidance documentation allows the Administrator to configure the Phase 2 SA value of 8 hours or provides sufficient instruction about the time value to configure to ensure the rekey is performed no later than the maximum SA lifetime of 8 hours. It is not permitted to configure a value of 8 hours if that leads to an actual rekey after more than 8hours. Currently there are no values mandated for the number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

Findings:	The TOE only supports time-based limits. [AGD] Section 3.8 specifies how this can be configured between 1 and 8 hours.

FCS_IPSEC_EXT.1.11

410 The evaluator ensures that the guidance documentation describes the configuration of all algorithms selected in the requirement.

Findings:	[AGD] Section 3.8 specifies the algorithms supported by the TOE and how to configure them.

FCS_IPSEC_EXT.1.13

411 The evaluator ensures the guidance documentation describes how to set up the TOE to use certificates with RSA and/or ECDSA signatures and public keys.

Findings:	The TOE only supports ECDSA certificates for IPsec. [AGD] Section 3.12 describes the certificate configuration parameters that need to be set on the TOE.

412 The evaluator shall check that the guidance documentation describes how pre-shared keys are to be generated and established. The description in the guidance documentation shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

Findings:	N/A – The TOE does not support pre-shared keys

413 The evaluator will ensure that the guidance documentation describes how to configure the TOE to connect to a trusted CA, and ensure a valid certificate for that CA is loaded into the TOE and marked “trusted”.

Findings:	[AGD] Section 3.12 discussed how trusted X509 certificates are imported by an authorized administrator to the TOE for each component and notes the communication with the TOE should be performed over a secure channel.

FCS_IPSEC_EXT.1.14

414 The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not, and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE does not guarantee unique identifiers, the evaluator shall ensure that the operational guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

Findings:	[AGD] Section 3.12 discusses CN and O fields as a function of a DN for reference identifier checking and how to configure the TOE to check them. Only DN reference identifiers are supported in the evaluated configuration for IPsec trusted channels.

7.2.2.3 Tests

FCS_IPSEC_EXT.1.1

415 The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

a) Test 1: The evaluator shall configure the SPD such that there is a rule for dropping a packet, encrypting a packet, and allowing a packet to flow in plaintext. The selectors used in the construction of the rule shall be different such that the evaluator can generate a packet and send packets to the gateway with the appropriate fields (fields that are used by the rule - e.g., the IP addresses, TCP/UDP ports) in the packet header. The evaluator performs both positive and negative test cases for each type of rule (e.g. a packet that matches the rule and

another that does not match the rule). The evaluator observes via the audit trail, and packet captures that the TOE exhibited the expected behaviour: appropriate packets were dropped, allowed to flow without modification, encrypted by the IPsec implementation.

High-Level Test Description
Configure SPD rules on the TOE to drop (DISCARD), encrypt (PROTECT) and allow a packet to flow in plaintext (BYPASS).
Findings: PASS – The evaluator confirmed that the TOE enforces all SPD rules properly.

b) Test 2: The evaluator shall devise several tests that cover a variety of scenarios for packet processing. As with Test 1, the evaluator ensures both positive and negative test cases are constructed. These scenarios must exercise the range of possibilities for SPD entries and processing modes as outlined in the TSS and guidance documentation. Potential areas to cover include rules with overlapping ranges and conflicting entries, inbound and outbound packets, and packets that establish SAs as well as packets that belong to established SAs. The evaluator shall verify, via the audit trail and packet captures, for each scenario that the expected behavior is exhibited, and is consistent with both the TSS and the guidance documentation.

High-Level Test Description
Configure SPD rules that cover a variety of scenarios for packet processing.
Findings: PASS – The evaluator confirmed that the TOE enforces all SPD rules properly.

FCS_IPSEC_EXT.1.2

416 The assurance activity for this element is performed in conjunction with the activities for FCS_IPSEC_EXT.1.1.

417 The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

418 The evaluator shall configure the SPD such that there is a rule for dropping a packet, encrypting a packet, and allowing a packet to flow in plaintext. The evaluator may use the SPD that was created for verification of FCS_IPSEC_EXT.1.1. The evaluator shall construct a network packet that matches the rule to allow the packet to flow in plaintext and send that packet. The evaluator should observe that the network packet is passed to the proper destination interface with no modification. The evaluator shall then modify a field in the packet header; such that it no longer matches the evaluator-created entries (there may be a “TOE created” final entry that discards packets that do not match any previous entries). The evaluator sends the packet, and observes that the packet was dropped.

High-Level Test Description
This testing is covered by FCS_IPSEC_EXT.1.1 Tests 1 & 2
Findings: PASS— The evaluator confirmed that the TOE enforces all SPD rules properly in conjunction with FCS_IPSEC_EXT.1.1 Tests 1 & 2.

FCS_IPSEC_EXT.1.3

419 The evaluator shall perform the following test(s) based on the selections chosen:

a) Test 1: If tunnel mode is selected, the evaluator uses the guidance documentation to configure the TOE to operate in tunnel mode and also configures a VPN peer to operate in tunnel mode. The evaluator configures the TOE and the VPN peer to use any of the allowable cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator shall then initiate a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the tunnel mode.

High-Level Test Description
Initiate a VPN connection and verify the connection is using tunnel mode.
Findings: PASS – The evaluator confirmed that the TOE uses tunnel mode for IPsec communication.

b) Test 2: If transport mode is selected, the evaluator uses the guidance documentation to configure the TOE to operate in transport mode and also configures a VPN peer to operate in transport mode. The evaluator configures the TOE and the VPN peer to use any of the allowed cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator then initiates a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the transport mode.

Test Not Applicable
Findings: N/A – Transport mode is not supported or claimed by the TOE.

FCS_IPSEC_EXT.1.4

420 The evaluator shall configure the TOE as indicated in the guidance documentation configuring the TOE to use each of the supported algorithms, attempt to establish a connection using ESP, and verify that the attempt succeeds.

High-Level Test Description
Verify all claimed ESP algorithms are supported for VPN connections on the TOE.
Findings: PASS – The evaluator confirmed that the TOE can connect using all claimed ESP algorithms.

FCS_IPSEC_EXT.1.5

421 Tests are performed in conjunction with the other IPsec evaluation activities.

a) Test 1: If IKEv1 is selected, the evaluator shall configure the TOE as indicated in the guidance documentation, and attempt to establish a connection using an IKEv1 Phase 1 connection in aggressive mode. This attempt should fail. The evaluator should then show that main mode exchanges are supported.

Test Not Applicable
Findings: N/A – The TOE does not support IKEv1 for IPSEC connections.

b) Test 2: If NAT traversal is selected within the IKEv2 selection, the evaluator shall configure the TOE so that it will perform NAT traversal processing as described in the TSS and RFC 5996, section 2.23. The evaluator shall initiate an IPsec connection and determine that the NAT is successfully traversed.

High-Level Test Description
Verify that the IPsec connection performs NAT traversal.
Findings: PASS – The evaluator confirmed that the TOE properly traverses NAT.

FCS_IPSEC_EXT.1.6

422 The evaluator shall configure the TOE to use the ciphersuite under test to encrypt the IKEv1 and/or IKEv2 payload and establish a connection with a peer device, which is configured to only accept the payload encrypted using the indicated ciphersuite. The evaluator will confirm the algorithm was that used in the negotiation.

High-Level Test Description
Verify all claimed IKE algorithms are supported for VPN connections on the TOE.
Findings: PASS – The evaluator confirmed that the TOE properly connects using all claimed IKE algorithms.

FCS_IPSEC_EXT.1.7

423 When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC “A difference between IKEv1 and IKEv2 is that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered.”

424 Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a) Test 1: If ‘number of bytes’ is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime in terms of the number of bytes allowed following the guidance documentation. The evaluator shall configure a test peer with a byte lifetime that exceeds the lifetime of the TOE. The evaluator shall establish an SA between the TOE and the test peer, and determine that once the allowed number of bytes through this SA is exceeded, a new SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 1 negotiation.

Test Not Applicable
Findings: N/A – “Number of bytes” is not supported by the TOE.

b) Test 2: If 'length of time' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime no later than 24 hours for the Phase 1 SA following the guidance documentation. The evaluator shall configure a test peer with a Phase 1 SA lifetime that exceeds the Phase 1 SA lifetime on the TOE.

High-Level Test Description
Verify that before 24 hours has elapsed, the IPsec connection rekeys properly.
Findings: PASS – The evaluator confirmed that the TOE properly rekeys before 24 hours.

FCS_IPSEC_EXT.1.8

425 When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC "A difference between IKEv1 and IKEv2 is that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered."

426 Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a) Test 1: If 'number of bytes' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime in terms of the number of bytes allowed following the guidance documentation. The evaluator shall configure a test peer with a byte lifetime that exceeds the lifetime of the TOE. The evaluator shall establish an SA between the TOE and the test peer, and determine that once the allowed number of bytes through this SA is exceeded, a new SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 2 negotiation.

Test Not Applicable
Findings: N/A – "Number of bytes" is not supported by the TOE.

NIAP TD0633

b) Test 2: If 'length of time' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime no later than 8 hours for the Phase 2 SA following the guidance documentation. The evaluator shall configure a test peer with a Phase 2 SA lifetime that exceeds the Phase 2 SA lifetime on the TOE.

High-Level Test Description
Verify that before 8 hours has elapsed, the IPsec connection issues new child SAs
Findings: PASS – The evaluator confirmed that the TOE creates new Child SAs before 8 hours.

FCS_IPSEC_EXT.1.10

427 Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a) Test 1: If the first selection is chosen, the evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	[ST] Section 6.3.7 states that the size of the random secret value “x” and nonce used for key establishment, as generated by the DRBG, is at least twice the security strength of that associated with the negotiated DH group and therefore meets the stipulations of the requirement.

b) Test 2: If the second selection is chosen, the evaluator shall check to ensure that, for each PRF hash supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	N/A—the second selection is not chosen in [ST].

FCS_IPSEC_EXT.1.11

428 For each supported DH group, the evaluator shall test to ensure that all supported IKE protocols can be successfully completed using that particular DH group.

High-Level Test Description
Verify the TOE supports all claimed DH groups.
Findings: PASS – The evaluator confirmed that the TOE connects successfully with all claimed DH groups.

FCS_IPSEC_EXT.1.12

429 The evaluator simply follows the guidance to configure the TOE to perform the following tests.

a) Test 1: This test shall be performed for each version of IKE supported. The evaluator shall successfully negotiate an IPsec connection using each of the supported algorithms and hash functions identified in the requirements.

High-Level Test Description
Refer to evidence found in FCS_IPSEC_EXT.1.6 Test 1 for IKE algorithms. Refer to FCS_IPSEC_EXT.1.4 for ESP algorithms. IKEv2 is the only supported key exchange method
Findings: PASS. The evaluator confirmed that the TOE can be successfully configured and negotiate each claimed algorithm and hash function.

b) Test 2: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an SA for ESP that selects an encryption algorithm with more strength than that being used for the IKE SA (i.e., symmetric algorithm with a key size larger than that being used for the IKE SA). Such attempts should fail.

High-Level Test Description
Note the TOE UI does not permit the TOE to be configured with an ESP algorithm of greater strength than IKE. Instead, the TOE is always configured to use an equivalent strength for ESP and IKE. Attempt a connection with an ESP algorithm stronger than the IKE algorithm and verify the connection fails.
Findings: PASS – The evaluator attempted a connection to the TOE from an IPsec endpoint that attempts to use an ESP algorithm is stronger than the IKE algorithm and confirmed that the TOE rejected the connection.

c) Test 3: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an IKE SA using an algorithm that is not one of the supported algorithms and hash functions identified in the requirements. Such an attempt should fail.

High-Level Test Description
Attempt a VPN connection to the TOE with invalid IKE algorithm and verify it fails
Findings: PASS – The TOE rejects a connection when an unsupported IKE SA algorithm is provided

d) Test 4: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an SA for ESP (assumes the proper parameters where used to establish the IKE SA) that selects an encryption algorithm that is not identified in FCS_IPSEC_EXT.1.4. Such an attempt should fail.

High-Level Test Description
Attempt a VPN connection to the TOE with an invalid ESP algorithm.
Findings: PASS – The evaluator confirmed that the TOE rejects a connection when an invalid ESP algorithm is detected.

FCS_IPSEC_EXT.1.13

430 For efficiency sake, the testing that is performed may be combined with the testing for FIA_X509_EXT.1, FIA_X509_EXT.2 (for IPsec connections), and FCS_IPSEC_EXT.1.1.

FCS_IPSEC_EXT.1.14

431 For each the context of the tests below, a valid certificate is a certificate that passes FIA_X509_EXT.1 validation checks but does not necessarily contain an authorized subject.

432 The evaluator shall perform the following tests:

- Test 1: (conditional) For each CN/identifier type combination selected, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the field in the peer's presented certificate and shall verify that the IKE authentication succeeds. If the TOE prioritizes CN checking over SAN (through explicit configuration of the field when specifying the

reference identifier or prioritization rules), the evaluator shall also configure the SAN so it contains an incorrect identifier of the correct type (e.g. the reference identifier on the TOE is example.com, the CN=example.com, and the SAN:FQDN=otherdomain.com) and verify that IKE authentication succeeds.

Test Not Applicable
Findings: N/A – The [ST] only selects DN identifier types.

- Test 2: (conditional) For each SAN/identifier type combination selected, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the field in the peer's presented certificate and shall verify that the IKE authentication succeeds. If the TOE prioritizes SAN checking over CN (through explicit specification of the field when specifying the reference identifier or prioritization rules), the evaluator shall also configure the CN so it contains an incorrect identifier formatted to be the same type (e.g. the reference identifier on the TOE is DNS-ID; identify certificate has an identifier in SAN with correct DNS-ID, CN with incorrect DNS-ID (and not a different type of identifier)) and verify that IKE authentication succeeds.

Test Not Applicable
Findings: N/A – The [ST] only selects DN identifier types.

- Test 3: (conditional) For each CN/identifier type combination selected, the evaluator shall:

e) Create a valid certificate with the CN so it contains the valid identifier followed by '\0'. If the TOE prioritizes CN checking over SAN (through explicit specification of the field when specifying the reference identifier or prioritization rules) for the same identifier type, the evaluator shall configure the SAN so it matches the reference identifier.

Test Not Applicable
Findings: N/A – The [ST] only selects DN identifier types.

f) Configure the peer's reference identifier on the TOE (per the administrative guidance) to match the CN without the '\0' and verify that IKE authentication fails.

Test Not Applicable
Findings: N/A – The [ST] only selects DN identifier types.

- Test 4: (conditional) For each SAN/identifier type combination selected, the evaluator shall:

a) Create a valid certificate with an incorrect identifier in the SAN. The evaluator shall configure a string representation of the correct identifier in the DN. If the TOE prioritizes CN checking over SAN (through explicit specification of the field when specifying the reference identifier or prioritization rules) for the same identifier type, the addition/modification shall be to any non-CN field of the DN. Otherwise, the addition/modification shall be to the CN.

Test Not Applicable
Findings: N/A – The [ST] only selects DN identifier types.

b) Configure the peer's reference identifier on the TOE (per the administrative guidance) to match the correct identifier (expected in the SAN) and verify that IKE authentication fails.

Test Not Applicable
Findings: N/A – The [ST] only selects DN identifier types.

- Test 5: (conditional) If the TOE supports DN identifier types, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the subject DN in the peer's presented certificate and shall verify that the IKE authentication succeeds.

High-Level Test Description
Configure the DN on the TOE and provide a certificate with the CN specified on the TOE. Verify that the connection succeeds.
Findings: PASS – The evaluator confirmed that the TOE connects successfully when a certificate with the proper DN is offered.

- Test 6: (conditional) If the TOE supports DN identifier types, to demonstrate a bit-wise comparison of the DN, the evaluator shall create the following valid certificates and verify that the IKE authentication fails when each certificate is presented to the TOE:

a) Duplicate the CN field, so the otherwise authorized DN contains two identical CNs.

High-Level Test Description
Duplicate the CN field of a certificate and provide that certificate to the TOE when performing a VPN connection. Verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection with a duplicated CN.

b) Append '\0' to a non-CN field of an otherwise authorized DN.

High-Level Test Description
Append a \0 to a non-CN field in an otherwise valid DN and verify that the connection fails due to an invalid certificate.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection with a certificate that contains a \0 in the 'O' field(non-CN field) of an otherwise valid DN.

7.2.3 FCS_TLSC_EXT.1 Extended: TLS Client Protocol without mutual authentication

7.2.3.1 TSS

FCS_TLSC_EXT.1.1

433 The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified include those listed for this component.

Findings:	[ST] Section 6.3.8 – The TSS identifies the ciphersuites supported by the TOE and are consistent with those listed for this component.

FCS_TLSC_EXT.1.2

434 The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the administrator/application configured reference identifier, including which types of reference identifiers are supported (e.g. application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported.

Findings:	[ST] Section 6.3.8 – The TSS describes how reference identifiers are used to establish a secure TLS connection on the TOE.

435 Note that where a TLS channel is being used between components of a distributed TOE for FPT_ITT.1, the requirements to have the reference identifier established by the user are relaxed and the identifier may also be established through a "Gatekeeper" discovery process. The TSS should describe the discovery process and highlight how the reference identifier is supplied to the "joining" component. Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1 and the ST author selected attributes from RFC 5280, the evaluator shall ensure the TSS describes which attribute type, or combination of attributes types, are used by the client to match the presented identifier with the configured identifier. The evaluator shall ensure the TSS presents an argument how the attribute type, or combination of attribute types, uniquely identify the remote TOE component; and the evaluator shall verify the attribute type, or combination of attribute types, is sufficient to support unique identification of the maximum supported number of TOE components.

Findings:	[ST] Section 6.3.8 – TLS is not used as a protocol between distributed TOE components.

436 If IP addresses are supported in the CN as reference identifiers, the evaluator shall ensure that the TSS describes the TOE's conversion of the text representation of the IP address in the CN to a binary representation of the IP address in network byte order. The evaluator shall also ensure that the TSS describes whether canonical format (RFC 5952 for IPv6, RFC 3986 for IPv4) is enforced.

Findings:	[ST] Section 6.3.8 – The TSS identifies IP addresses as a supported reference identifier. The TSS also describes how the TOE treats the CN as a string value and compares this against the configured syslog server string value. The TOE does not support any wildcards.

FCS_TLSC_EXT.1.4

437 The evaluator shall verify that TSS describes the Supported Elliptic Curves Extension and whether the required behaviour is performed by default or may be configured.

Findings:	[ST] Section 6.3.8 – The TSS states the TOE always presents the Supported Elliptic Curves Extension indicating support for P-384 in the Client Hello.

7.2.3.2 Guidance Documentation

FCS_TLSC_EXT.1.1

438 The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS.

Findings:	[AGD] Section 3.10 discusses how the TLS parameters are configured on the TOE to connect the TOE to an external syslog server. This description is consistent with the FCS_TLSC_EXT.1 TSS description in the [ST].

FCS_TLSC_EXT.1.2

439 The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE includes support for IP addresses, the evaluator shall ensure that the operational guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

Findings:	[AGD] Section 3.12 discusses that the TOE supports DNS SAN, IP SAN, and CN fields for reference identifier checking. The TOE automatically detects the presence of these fields and describes the order in which they are checked.

440 Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1, the SFR selects attributes from RFC 5280, and FCO_CPC_EXT.1.2 selects “no channel”; the evaluator shall verify the guidance provides instructions for establishing unique reference identifiers based on RFC5280 attributes.

Findings:	N/A – The TOE does not support TLS for Inter-TOE communication

FCS_TLSC_EXT.1.4

441 If the TSS indicates that the Supported Elliptic Curves/Supported Groups Extension must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the Supported Elliptic Curves/Supported Groups Extension.

Findings:	N/A – The TSS does not indicate that the Supported Elliptic Curves/Supported Groups Extension must be configured to meet the requirement.

7.2.3.3 Tests

442 **NIAP TD0670**

FCS_TLSC_EXT.1.1

443 Test 1: The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an HTTPS session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

High-Level Test Description
Establish a secure syslog connection using TLS and all claimed encryption algorithms.
Findings: PASS – The evaluator confirmed that the TOE accepts all syslog connection using valid claimed algorithms.

444 Test 2: The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field, and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.

High-Level Test Description
Provide a server certificate to the TOE during a TLS handshake that does not contain the ServerAuth EKU bit set.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection when an invalid server certificate is presented to it.

445 Test 3: The evaluator shall send a server certificate in the TLS connection that does not match the server-selected ciphersuite (for example, send an ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite). The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.

High-Level Test Description
Provide a server certificate to the TOE that does not match the server-selected ciphersuite and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection when the server certificate does not match the server-selected ciphersuite.

446 Test 4: The evaluator shall perform the following 'negative tests':

a) The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the client denies the connection.

High-Level Test Description
Offer a TLS_NULL cipher from the TLS server and verify the TOE does not connect.

High-Level Test Description
Findings: PASS – The evaluator confirmed that TOE rejects a connection with the TLS_NULL cipher.

b) Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator shall verify that the client rejects the connection after receiving the Server Hello.

High-Level Test Description
Provide a cipher on the server not listed in the TOE's client hello and verify the connection fails.
Findings: PASS – The evaluator confirmed that theTOE rejects the connection with an invalid cipher not supported by the TOE.

c) [conditional]: If the TOE presents the Supported Elliptic Curves/Supported Groups Extension the evaluator shall configure the server to perform an ECDHE or DHE key exchange in the TLS connection using a non-supported curve/group (for example P-192) and shall verify that the TOE disconnects after receiving the server's Key Exchange handshake message.

High-Level Test Description
Provide an unsupported curve to the TOE using an ECDHE cipher and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection when an unsupported curve is detected.

447 Test 5: The evaluator performs the following modifications to the traffic:

a) Change the TLS version selected by the server in the Server Hello to a non-supported TLS version and verify that the client rejects the connection.

High-Level Test Description
Provide invalid TLS versions from the TLS server to the TOE and verify that the TOE fails to connect with all unclaimed/unsupported protocol versions.
Findings: PASS – The evaluator confirmed that the TOE rejects all invalid TLS versions when attempting to connect to the syslog server.

b) [conditional]: If using DHE or ECDH, modify the signature block in the Server's Key Exchange handshake message, and verify that the handshake does not finished successfully, and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.

High-Level Test Description
Modify the server key exchange message during the TLS handshake and verify the TOE does not connect and no application data flows.
Findings: PASS – The evaluator confirmed that no application data flowed from the TOE after the server's key exchange message was modified.

448 Test 6: The evaluator performs the following 'scrambled message tests':

a) Modify a byte in the Server Finished handshake message and verify that the handshake does not finish successfully and no application data flows.

High-Level Test Description
Modify a byte in the server finished handshake message and verify that the connection fails.
Findings: PASS – The evaluator confirmed that no application data flowed from the TOE after a byte was modified in the server finished handshake message.

b) Send a garbled message from the server after the server has issued the ChangeCipherSpec message and verify that the handshake does not finish successfully and no application data flows.

High-Level Test Description
Send a garbled message from the server after the server has issued the ChangeCipherSpec and verify the connection fails.
Findings: PASS – The evaluator confirmed that no application data flowed from the TOE after the garbled message was received.

c) Modify at least one byte in the server's nonce in the Server Hello handshake message and verify that the client rejects the Server Key Exchange handshake message (if using a DHE or ECDHE ciphersuite) or that the server denies the client's Finished handshake message.

High-Level Test Description
Modify a byte in the server's nonce in the server hello handshake message and verify the connection fails.
Findings: PASS – The evaluator confirmed that no application data flowed from the TOE after the modification of the server's nonce in the server hello handshake message.

FCS_TLSC_EXT.1.2

449 Note that the following tests are marked conditional and are applicable under the following conditions:

a) For TLS-based trusted channel communications according to FTP_ITC.1 where RFC 6125 is selected, tests 1-6 are applicable.

or

b) For TLS-based trusted path communications according to FTP_TRP where RFC 6125 is selected, tests 1-6 are applicable

or

c) For TLS-based trusted path communications according to FPT_ITT.1 where RFC 6125 is selected, tests 1-6 are applicable. Where RFC 5280 is selected, only test 7 is applicable.

Note that for some tests additional conditions apply.

450 IP addresses are binary values that must be converted to a textual representation when presented in the CN of a certificate. When testing IP addresses in the CN, the evaluator shall follow the following formatting rules:

- IPv4: The CN contains a single address that is represented a 32-bit numeric address (IPv4) is written in decimal as four numbers that range from 0-255 separated by periods as specified in RFC 3986.

- IPv6: The CN contains a single IPv6 address that is represented as eight colon separated groups of four lowercase hexadecimal digits, each group representing 16 bits as specified in RFC 4291. Note: Shortened addresses, suppressed zeros, and embedded IPv4 addresses are not tested.

451 The evaluator shall configure the reference identifier per the AGD guidance and perform the following tests during a TLS connection:

a) Test 1 [conditional]: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each identifier type (e.g. IPv4, IPv6, FQDN) supported in the CN. When testing IPv4 or IPv6 addresses, the evaluator shall modify a single decimal or hexadecimal digit in the CN.

   Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 1.

High-Level Test Description
Present a server certificate to the TOE that contains a CN that does not match the reference identifier and no SAN.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection due to the CN of the certificate not matching the reference ID on the TOE.

b) Test 2 [conditional]: The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type (e.g. IPv4, IPv6, FQDN, URI). When testing IPv4 or IPv6 addresses, the evaluator shall modify a single decimal or hexadecimal digit in the SAN.

High-Level Test Description
Provide a server certificate to the TOE that does not contain a proper SAN extension but valid CN. Verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection due to the SAN being invalid.

c) Test 3 [conditional]: If the TOE does not mandate the presence of the SAN extension, the evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds. The evaluator shall repeat this test for each identifier type (e.g. IPv4, IPv6, FQDN) supported in the CN. If the TOE does mandate the presence of the SAN extension, this Test shall be omitted.

High-Level Test Description
Provide a certificate to the TOE with a valid CN and no SAN extension and verify the connection succeeds. Perform this test for IPv4 and FQDN identifiers.
Findings: PASS – The evaluator confirmed that the TOE accepts the connection when there is a missing SAN but valid CN for both IPv4 and FQDN identifiers.

d) Test 4 [conditional]: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds. The evaluator shall repeat this test for each supported SAN type (e.g. IPv4, IPv6, FQDN, SRV).

High-Level Test Description
Provide a certificate to the TOE that has an invalid CN and a valid SAN and verify the connection succeeds. Perform this test for IPv4 and FQDN identifiers.
Findings: PASS – The evaluator confirmed that the TOE accepts the connection when a valid SAN is detected even though an invalid CN was also seen for both IPv4 and FQDN identifiers.

e) Test 5 [conditional]: The evaluator shall perform the following wildcard tests with each supported type of reference identifier that includes a DNS name (i.e. CN-ID with DNS, DNS-ID, SRV-ID, URI-ID):

1) [conditional]: The evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo.*.example.com) and verify that the connection fails.

High-Level Test Description
Provide a server certificate to the TOE with a wildcard not in the left-most label and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection due to wildcards not being supported by the TOE.

2) [conditional]: The evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds, if wildcards are supported, or fails if wildcards are not supported. The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g.

bar.foo.example.com) and verify that the connection fails. (Remark: Support for wildcards was always intended to be optional. It is sufficient to state that the TOE does not support wildcards and observe rejected connection attempts to satisfy corresponding assurance activities.)

High-Level Test Description
Provide a server certificate with an invalid wildcard CN to the TOE and verify all connection are rejected (wildcards are not supported by the TOE).
Findings: PASS - The evaluator confirmed that the TOE rejects the connection due to wildcards not being supported by the TOE.

NIAP TD0634

452 Objective: The objective of this test is to ensure the TOE is able to differentiate between IP address identifiers that are not allowed to contain wildcards and other types of identifiers that may contain wildcards.

a) Test 6 [conditional]: If IP address identifiers supported in the SAN or CN, the evaluator shall present a server certificate that contains a CN that matches the reference identifier, except one of the groups has been replaced with a wildcard asterisk (*) (e.g. CN=*.168.0.1 when connecting to 192.168.0.1, CN=2001:0DB8:0000:0000:0008:0800:200C:* when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A). The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6).

   This negative test corresponds to the following section of the Application Note 64/105: "The exception being, the use of wildcards is not supported when using IP address as the reference identifier."

   Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 6.

High-Level Test Description
Provide a server certificate with a wildcard within the IP address and verify the connection fails.
Findings: PASS - The evaluator confirmed that the TOE rejects the connection due to wildcards not being supported by the TOE.

453 Test 7 [conditional]: If the secure channel is used for FPT_ITT, and RFC 5280 is selected, the evaluator shall perform the following tests. Note, when multiple attribute types are selected in the SFR (e.g. when multiple attribute types are combined to form the unique identifier), the evaluator modifies each attribute type in accordance with the matching criteria described in the TSS (e.g. creating a mismatch of one attribute type at a time while other attribute types contain values that will match a portion of the reference identifier):

   1) The evaluator shall present a server certificate that does not contain an identifier in the Subject (DN) attribute type(s) that matches the reference identifier. The evaluator shall verify that the connection fails.

2) The evaluator shall present a server certificate that contains a valid identifier as an attribute type other than the expected attribute type (e.g. if the TOE is configured to expect id-at-serialNumber=correct_identifier, the certificate could instead include id-at-name=correct_identifier), and does not contain the SAN extension. The evaluator shall verify that the connection fails. Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass this test.

3) The evaluator shall present a server certificate that contains a Subject attribute type that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds.

4) The evaluator shall confirm that all use of wildcards results in connection failure regardless of whether the wildcards are used in the left or right side of the presented identifier. (Remark: Use of wildcards is not addressed within RFC 5280.)

Test Not Applicable
Findings: N/A--The [ST] does not claim FPT_ITT.1 with RFC 5280.

454 **FCS_TLSC_EXT.1.3**

455 The evaluator shall demonstrate that using an invalid certificate results in the function failing as follows:

456 Test 1: Using the administrative guidance, the evaluator shall load a CA certificate or certificates needed to validate the presented certificate used to authenticate an external entity and demonstrate that the function succeeds and a trusted channel can be established.

High-Level Test Description
This test case is performed as part of FIA_X509_EXT.1.1/Rev Test 1a.
Findings: PASS—The evaluator confirmed that a connection using a valid certificate chain succeeds in conjunction with FIA_X509_EXT.1.1/Rev Test 1a.

457 Test 2: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

High-Level Test Description
This test case is performed as part of FIA_X509_EXT.1.1/Rev Test 1b.
Findings: PASS—The evaluator confirmed that a connection without valid certificate chain fails in conjunction with FIA_X509_EXT.1.1/Rev Test 1b.

458 Test 3 [conditional]: The purpose of this test to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

Test Not Applicable
Findings: N/A--There is no override mechanism for certificates on the TOE.

FCS_TLSC_EXT.1.4

459 Test 1 [conditional]: If the TOE presents the Supported Elliptic Curves/Supported Groups Extension, the evaluator shall configure the server to perform ECDHE or DHE (as applicable) key exchange using each of the TOE's supported curves and/or groups. The evaluator shall verify that the TOE successfully connects to the server.

High-Level Test Description
Perform a TLSC connection from the TOE to the syslog server using all claimed elliptic curve groups.
Findings: PASS – The evaluator confirmed that the TOE successfully connects with all claimed curves.

7.2.4 FCS_TLSS_EXT.1 Extended: TLS Server Protocol without mutual authentication

7.2.4.1 TSS

FCS_TLSS_EXT.1.1

460 The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified are identical to those listed for this component.

Findings:	[ST] Section 6.3.9 – The TSS identifies which ciphersuites are supported for TLSS connections on the TOE and are identical to those listed for this component.

FCS_TLSS_EXT.1.2

461 The evaluator shall verify that the TSS contains a description of how the TOE technically prevents the use of old SSL and TLS versions.

Findings:	[ST] Section 6.3.9 – The TSS identifies how old TLS versions are terminated due to them not being supported by the TSF.

FCS_TLSS_EXT.1.3

NIAP TD0635

462 If using ECDHE and/or DHE ciphers, the evaluator shall verify that the TSS lists all EC Diffie-Hellman curves and/or Diffie-Hellman groups used in the key establishment by the TOE when acting as a TLS Server. For example, if the TOE supports TLS_DHE_RSA_WITH_AES_128_CBC_SHA cipher and Diffie-Hellman parameters with size 2048 bits, then list Diffie-Hellman Group 14.

Findings:	[ST] Section 6.3.9 – The TSS identifies the key agreement parameters of the key exchange message due to ECDHE being supported by the TOE

FCS_TLSS_EXT.1.4

463 The evaluator shall verify that the TSS describes if session resumption based on session IDs is supported (RFC 4346 and/or RFC 5246) and/or if session resumption based on session tickets is supported (RFC 5077).

464 If session tickets are supported, the evaluator shall verify that the TSS describes that the session tickets are encrypted using symmetric algorithms consistent with FCS_COP.1/DataEncryption. The evaluator shall verify that the TSS identifies the key lengths and algorithms used to protect session tickets.

Findings:	[ST] Section 6.3.9 – The TSS identifies session resumption as not being supported by the TOE.

465 If session tickets are supported, the evaluator shall verify that the TSS describes that session tickets adhere to the structural format provided in section 4 of RFC 5077 and if not, a justification shall be given of the actual session ticket format.

Findings:	[ST] Section 6.3.9 – The TSS identifies session tickets as not being supported by the TOE.

NIAP TD0569

466 If the TOE claims a (D)TLS server capable of session resumption (as a single context, or across multiple contexts), the evaluator verifies that the TSS describes how session resumption operates (i.e. what would trigger a full handshake, e.g. checking session status, checking Session ID, etc.). If multiple contexts are used the TSS describes how session resumption is coordinated across those contexts. In case session establishment and session resumption are always using a separate context, the TSS shall describe how the contexts interact with respect to session resumption (in particular regarding the session ID). It is acceptable for sessions established in one context to be resumable in another context.

Findings:	N/A **–** The TOE does not support session resumption and therefore there is no support for multiple contexts.

7.2.4.2 Guidance Documentation

FCS_TLSS_EXT.1.1

467 The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the

TSS (for instance, the set of ciphersuites advertised by the TOE may have to be restricted to meet the requirements).

Findings:	[AGD] Section 3.11 discusses the TLS parameters supported by the TOE which are not configurable. This description is consistent with the FCS_TLSS_EXT.1 TSS description in the [ST].

FCS_TLSS_EXT.1.2

468 The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Findings:	[AGD] Section 3.11 discusses the TLS parameters supported by the TOE which are not configurable. This description is consistent with the FCS_TLSC_EXT.1 TSS description in the [ST].

FCS_TLSS_EXT.1.3

469 The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Findings:	[AGD] Section 3.11 discusses the TLS parameters supported by the TOE which are not configurable. This description is consistent with the FCS_TLSC_EXT.1 TSS description in the [ST].

NIAP TD0569

FCS_TLSS_EXT.1.4

470 The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Findings:	N/A – The TOE does not claim session resumption.

7.2.4.3 Tests

FCS_TLSS_EXT.1.1

471 Test 1: The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an HTTPS session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

High-Level Test Description
Perform a secure TLS connection to the TOE TLS server using all claimed ciphers.
Findings: PASS – The evaluator confirmed that the TOE successfully connects with all claimed ciphers.

472 Test 2: The evaluator shall send a Client Hello to the server with a list of ciphersuites that does not contain any of the ciphersuites in the server's ST and verify that the server denies the connection. Additionally, the evaluator shall send a Client Hello to the server containing only the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the server denies the connection.

High-Level Test Description
Attempt a TLS client connection to the TOE with a NULL cipher and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection attempt using a NULL cipher.

473 Test 3: The evaluator shall perform the following modifications to the traffic:

a) Modify a byte in the Client Finished handshake message, and verify that the server rejects the connection and does not send any application data.

High-Level Test Description
Modify a byte in the client finished handshake message and verify the server rejects the connection.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection after the modified in the client finished handshake message.

b) (Test Intent: The intent of this test is to ensure that the server's TLS implementation immediately makes use of the key exchange and authentication algorithms to: a) Correctly encrypt (D)TLS Finished message and b) Encrypt every (D)TLS message after session keys are negotiated.)

The evaluator shall use one of the claimed ciphersuites to complete a successful handshake and observe transmission of properly encrypted application data. The evaluator shall verify that no Alert with alert level Fatal (2) messages were sent.

The evaluator shall verify that the Finished message (Content type hexadecimal 16 and handshake message type hexadecimal 14) is sent immediately after the server's ChangeCipherSpec (Content type hexadecimal 14) message. The evaluator shall examine the Finished message (encrypted example in hexadecimal of a TLS record containing a Finished message, 16 03 03 00 40 11 22 33 44 55...) and confirm that it does not contain unencrypted data (unencrypted example in hexadecimal of a TLS record containing a Finished message, 16 03 03 00 40 14 00 00 0c...), by verifying that the first byte of the encrypted Finished message does not equal hexadecimal 14 for at least one of three test messages. There is a chance that an encrypted Finished message contains a hexadecimal value of '14' at the position where a plaintext Finished message would contain the message type code '14'. If the observed Finished message contains a hexadecimal value of '14' at the position where the plaintext Finished message would contain the message type code, the test shall be repeated three times in total. In case the value of '14' can be observed in all three tests it can be assumed that the Finished message has indeed been sent in plaintext and the test has to be regarded as 'failed'. Otherwise it has to be assumed that the observation of the value '14' has been due to chance and that the Finished message has indeed been sent encrypted. In that latter case the test shall be regarded as 'passed'.

High-Level Test Description
The evaluator should verify that the Finished message does not contain unencrypted data.
Findings: PASS – The evaluator confirmed that the Finished message does not contain unencrypted data.

FCS_TLSS_EXT.1.2

474 The evaluator shall send a Client Hello requesting a connection for all mandatory and selected protocol versions in the SFR (e.g. by enumeration of protocol versions in a test client) and verify that the server denies the connection for each attempt.

High-Level Test Description
Attempt to connect to the TOE using invalid TLS protocols and verify the TOE rejects all these attempts.
Findings: PASS – The evaluator confirmed that the TOE rejects all connection attempts using invalid TLS protocols.

FCS_TLSS_EXT.1.3

475 Test 1: [conditional] If ECDHE ciphersuites are supported:

a) The evaluator shall repeat this test for each supported elliptic curve. The evaluator shall attempt a connection using a supported ECDHE ciphersuite and a single supported elliptic curve specified in the Elliptic Curves Extension. The Evaluator shall verify (though a packet capture or instrumented client) that the TOE selects the same curve in the Server Key Exchange message and successfully establishes the connection.

High-Level Test Description
Connect to the TOE on port 443 using all supported curves.
Findings: PASS – The evaluator confirmed that the TOE successfully connects using all supported curves.

b) The evaluator shall attempt a connection using a supported ECDHE ciphersuite and a single unsupported elliptic curve (e.g. secp192r1 (0x13)) specified in RFC4492, chap. 5.1.1. The evaluator shall verify that the TOE does not send a Server Hello message and the connection is not successfully established.

High-Level Test Description
Connect to the TOE using a valid ECDHE cipher but invalid curve and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection attempt using an invalid curve.

476 Test 2: [conditional] If DHE ciphersuites are supported, the evaluator shall repeat the following test for each supported parameter size. If any configuration is necessary, the evaluator shall configure the TOE to use a supported Diffie-Hellman parameter size. The evaluator shall attempt a connection using a supported DHE ciphersuite. The evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a Server Key Exchange Message where p Length is consistent with the message are the ones configured Diffie-Hellman parameter size(s).

Test Not Applicable
Findings: N/A – DHE ciphersuites are not supported by the TOE.

477 Test 3: [conditional] If RSA key establishment ciphersuites are supported, the evaluator shall repeat this test for each RSA key establishment key size. If any

configuration is necessary, the evaluator shall configure the TOE to perform RSA key establishment using a supported key size (e.g. by loading a certificate with the appropriate key size). The evaluator shall attempt a connection using a supported RSA key establishment ciphersuite. The evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a certificate whose modulus is consistent with the configured RSA key size.

Test Not Applicable
Findings: N/A – RSA ciphersuites are not supported by the TOE.

FCS_TLSS_EXT.1.4

478 *Test Objective: To demonstrate that the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption).*

479 Test 1 [conditional]: If the TOE does not support session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2) or session tickets according to RFC5077, the evaluator shall perform the following test:

a) The client sends a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.
b) The client verifies the server does not send a NewSessionTicket handshake message (at any point in the handshake).
c) The client verifies the Server Hello message contains a zero-length session identifier or passes the following steps:
    Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.
d) The client completes the TLS handshake and captures the SessionID from the ServerHello.
e) The client sends a ClientHello containing the SessionID captured in step d). This can be done by keeping the TLS session in step d) open or start a new TLS session using the SessionID captured in step d).
f) The client verifies the TOE (1) implicitly rejects the SessionID by sending a ServerHello containing a different SessionID and by performing a full handshake (as shown in Figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

NIAP TD0569

480 Remark: If multiple contexts are supported for session resumption, the session ID or session ticket may be obtained in one context for resumption in another context. It is possible that one or more contexts may only permit the construction of sessions to be reused in other contexts but not actually permit resumption themselves. For contexts which do not permit resumption, the evaluator is required to verify this behaviour subject to the description provided in the TSS. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

High-Level Test Description
Due to the TOE not supporting either session ID's or session tickets, the session ID field should be empty and no Session ticket should be sent by the TOE.
Findings: PASS – The evaluator confirmed that the TOE sends empty session ID and no session ticket due to not supporting them.

481 Test 2 [conditional]: If the TOE supports session resumption using session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2), the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall conduct a successful handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages (as shown in Figure 2 of RFC 4346 or RFC 5246).
b) The evaluator shall initiate a handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then, within the same handshake, generate or force an unencrypted fatal Alert message immediately before the client would otherwise send its ChangeCipherSpec message thereby disrupting the handshake. The evaluator shall then initiate a new Client Hello using the previously captured session ID, and verify that the server (1) implicitly rejects the session ID by sending a ServerHello containing a different SessionID and performing a full handshake (as shown in figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

NIAP TD0569

482 Remark: If multiple contexts are supported for session resumption, for each of the above test cases, the session ID may be obtained in one context for resumption in another context. There is no requirement that the session ID be obtained and replayed within the same context subject to the description provided in the TSS. All contexts that can reuse a session ID constructed in another context must be tested. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

Test Not Applicable
Findings: N/A – The TOE does not support session resumption using session IDs.

483 Test 3 [conditional]: If the TOE supports session tickets according to RFC5077, the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

NIAP TD0556

a) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with an abbreviated handshake described in section 3.1 of RFC 5077 and illustrated with an example in figure 2. Of particular note: if the server successfully verifies the client's ticket, then it may renew the ticket by including a NewSessionTicket handshake message after the ServerHello in the abbreviated handshake (which is shown in figure 2). This is not required, however as further clarified in section 3.3 of RFC 5077.

b) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator will then modify the session ticket and send it as part of a new Client Hello message. The evaluator shall confirm that the TOE either (1) implicitly rejects the session

ticket by performing a full handshake (as shown in figure 3 or 4 of RFC 5077), or (2) terminates the connection in some way that prevents the flow of application data.

NIAP TD0569

484 Remark: If multiple contexts are supported for session resumption, for each of the above test cases, the session ticket may be obtained in one context for resumption in another context. There is no requirement that the session ticket be obtained and replayed within the same context subject to the description provided in the TSS. All contexts that can reuse a session ticket constructed in another context must be tested. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

Test Not Applicable
Findings: N/A – The TOE does not support session resumption tickets.

485

7.3 Identification and Authentication (FIA)

7.3.1 FIA_X509_EXT.1/Rev X.509 Certificate Validation

7.3.1.1 TSS

486 The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected). It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected).

Findings:	[ST] Section 6.6.2 - The TSS identifies how the TOE checks the validity of certificates presented to it by checking certain fields such as extendedKeyUsage.

487 The TSS shall describe when revocation checking is performed and on what certificates. If the revocation checking during authentication is handled differently depending on whether a full certificate chain or only a leaf certificate is being presented, any differences must be summarized in the TSS section and explained in the Guidance.

Findings:	[ST] Section 6.6.2 – The TSS describes that the TOE performs CRL revocation checking for TLS. Certificate revocation is checked when the certificate is initially presented to the TOE. Revocation checking is done for the entire certificate chain.

7.3.1.2 Guidance Documentation

488 The evaluator shall also ensure that the guidance documentation describes where the check of validity of the certificates takes place, describes any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) and describes how certificate revocation checking is performed and on which certificate.

Findings:	[AGD] Section 3.12 describes the TOE's handling of X509 certificates. Section 3.12 also states that the validity, revocation, extendedKeyUsage, certificate chain and

	reference identifier checks of the certificates are done by the TOE. Revocation checking is performed using CRL for TLS. Certificate validity checks are performed on certificate upload and during IPsec and TLS connections using certificates.

7.3.1.3 Tests

489 The evaluator shall demonstrate that checking the validity of a certificate is performed when a certificate is used in an authentication step or when performing trusted updates (if FPT_TUD_EXT.2 is selected). It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the TOE. It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected). The evaluator shall perform the following tests for FIA_X509_EXT.1.1/Rev. These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols:

Test 1a: The evaluator shall present the TOE with a valid chain of certificates (terminating in a trusted CA certificate) as needed to validate the leaf certificate to be used in the function, and shall use this chain to demonstrate that the function succeeds. . Test 1a shall be designed in a way that the chain can be 'broken' in Test 1b by either being able to remove the trust anchor from the TOEs trust store, or by setting up the trust store in a way that at least one intermediate CA certificate needs to be provided, together with the leaf certificate from outside the TOE, to complete the chain (e.g. by storing only the root CA certificate in the trust store) **High-Level Test Description**
Install a chain of trusted CA's and verify the connection is successful (Root > Intermediate > Leaf Certificate).
Findings: PASS – The evaluator confirmed that the TOE successfully connects when a full chain of certificates can be validated.

Test 1b: The evaluator shall then 'break' the chain used in Test 1a by either removing the trust anchor in the TOE's trust store used to terminate the chain, or by removing one of the intermediate CA certificates (provided together with the leaf certificate in Test 1a) to complete the chain. The evaluator shall show that an attempt to validate this broken chain fails.

High-Level Test Description
Delete the trusted CA in the chain installed in Test 1a and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection when the CA chain is broken.

a) Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.

High-Level Test Description
Provide an expired certificate to the TOE and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection attempt using an expired certificate.

b) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates-–conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the peer certificate and revocation of the peer intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails. Revocation checking is only applied to certificates that are not designated as trust anchors. Therefore, the revoked certificate(s) used for testing shall not be a trust anchor.

High-Level Test Description
The purpose of this test is for the TOE to verify the validity of certificates using CRLs. The TOE will retrieve either good or revoked statuses of these certificates from the CRL distribution points.
Findings: PASS – The evaluator confirmed that the TOE successfully connects when all certificates are valid. When a certificate is revoked, the TOE rejects the connection.

c) Test 4: If OCSP is selected, the evaluator shall configure the OCSP server or use a man-in-the-middle tool to present a certificate that does not have the OCSP signing purpose and verify that validation of the OCSP response fails. If CRL is selected, the evaluator shall configure the CA to sign a CRL with a certificate that does not have the cRLsign key usage bit set, and verify that validation of the CRL fails.

High-Level Test Description
Provide an invalidly signed CRL that is signed by a CA without the CRL sign bit and verify the connection is rejected.
Findings: PASS – The evaluator confirmed that the TOE rejects any certificate status received from a CRL signed by a CA without the CRL sign bit. The TOE then terminates the connection due to there being no override mechanism.

d) Test 5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)

High-Level Test Description
Modify a byte in the first 8 bytes of the certificate and provide it to the TOE during a TLS connection; The connection should be rejected due to the invalid certificate.
Findings: PASS – The evaluator confirmed that the TOE rejects a connection attempt when a certificate is modified in the first eight bytes.

e) Test 6: The evaluator shall modify any byte in the certificate signatureValue field (see RFC5280 Sec. 4.1.1.3), which is normally the last field in the certificate, and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

High-Level Test Description
Modify a byte in the signatureValue field and verify the connection fails when the TOE attempts to connect to the server.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection attempt when a certificate is modified in the signatureValue field.

f) Test 7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The hash of the certificate will not validate.)

High-Level Test Description
Modify the public key of the certificate and verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects the connection attempt when the public key of the certificate is modified.

NIAP TD0527 (REVISED 1 December 2020)

490 The following tests are run when a minimum certificate path length of three certificates is implemented.

491 Test 8: (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). The evaluator shall conduct the following tests:

492 Test 8a: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a valid chain of EC certificates (terminating in a trusted CA certificate), where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

Test Not Applicable
Findings: N/A – The TOE does not process CA certificates presented in the certificate message.

493 Test 8b: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a chain of EC certificates (terminating in a trusted CA certificate), where the intermediate certificate in the certificate chain uses an explicit format version of the Elliptic Curve parameters in the public key information field, and is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

Test Not Applicable
Findings: N/A – The TOE does not process CA certificates presented in the certificate message.

494 Test 8c: The evaluator shall establish a subordinate CA certificate, where the elliptic curve parameters are specified as a named curve, that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is accepted into the TOE's trust store. The evaluator shall then establish a subordinate CA certificate that uses an explicit format version of the elliptic curve parameters, and that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is rejected, and not added to the TOE's trust store.

High-Level Test Description
Establish a chain with an EC root and subordinate CA with an elliptic curve and verify this action is successful. The evaluator then should upload a CA certificate that uses an explicit format version and verify that it is rejected.
Findings: PASS – The evaluator confirmed that the TOE allows an EC root with a subordinate CA that has elliptic curve parameters to be uploaded successfully. Immediately following this, the TOE rejects an upload attempt of a subordinate CA with explicit parameters set.

495 The evaluator shall perform the following tests for FIA_X509_EXT.1.2/Rev. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/Rev. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. Where the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) then the associated extendedKeyUsage rule testing may be omitted.

496 The goal of the following tests is to verify that the TOE accepts a certificate as a CA certificate only if it has been marked as a CA certificate by using basicConstraints with the CA flag set to True (and implicitly tests that the TOE correctly parses the basicConstraints extension as part of X509v3 certificate chain validation).

497 For each of the following tests the evaluator shall create a chain of at least three certificates: a self-signed root CA certificate, an intermediate CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

a) Test 1: The evaluator shall ensure that at least one of the CAs in the chain does not contain the basicConstraints extension. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate without the basicConstraints extension to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).

High-Level Test Description
Upload a CA certificate with the missing basicConstraints extension.
Findings: PASS – The evaluator confirmed that the TOE rejects this upload attempt due to a missing basicConstraints extension.

b) Test 2: The evaluator shall ensure that at least one of the CA certificates in the chain has a basicConstraints extension in which the CA flag is set to FALSE. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate with the CA flag set to FALSE to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).

High-Level Test Description
Upload a CA certificate that has the basicConstraints extension set to False.
Findings: PASS – The evaluator confirmed that the TOE rejects this upload attempt due to a False basicConstraints CA flag.

498 The evaluator shall repeat these tests for each distinct use of certificates. Thus, for example, use of certificates for TLS connection is distinct from use of certificates for trusted updates so both of these uses would be tested. But there is no need to repeat the tests for each separate TLS channel in FTP_ITC.1 and FTP_TRP.1/Admin (unless the channels use separate implementations of TLS).

High-Level Test Description
The evaluator repeats these tests for all channels.
Findings: PASS – Certificates are used for TLS and IPsec channels. FIA_X509_EXT.1/Rev covers testing of certificates in conjunction with TLS. FIA_X509_EXT.1/ITT covers testing of certificates in conjunction with IPsec.

7.3.2 FIA_X509_EXT.1/Rev X.509 Certificate Validation (MOD VPNGW)

499 There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to make it mandatory because of the TOE's required support for IPsec.

7.3.3 FIA_X509_EXT.2 X.509 Certificate Authentication

7.3.3.1 TSS

500 The evaluator shall check the TSS to ensure that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.

Findings:	[ST] Section 6.6.2 – The TSS identifies how the TOE chooses which certificates to use when initially presented to the TOE. The Cube determines the certificate to be used during configuration of the server profile and the GSS supports distinct certificates for IPsec and the administrator GUI where the usage is determined upon import. When receiving a certificate, GSS will verify it against its own trust store, and check the revocation status via OCSP (for IPsec) and CRL (for TLS).

501 The evaluator shall examine the TSS to confirm that it describes the behaviour of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. The evaluator shall verify that any distinctions between trusted channels are described. If the requirement that the

administrator is able to specify the default action, then the evaluator shall ensure that the guidance documentation contains instructions on how this configuration action is performed.

Findings:	[ST] Section 6.6.2 – The TSS describes that the TOE does not allow a connection to continue if the certificate validity cannot be verified.

7.3.3.2 Guidance Documentation

502 The evaluator shall also ensure that the guidance documentation describes the configuration required in the operating environment so the TOE can use the certificates. The guidance documentation shall also include any required configuration on the TOE to use the certificates. The guidance document shall also describe the steps for the Security Administrator to follow if the connection cannot be established during the validity check of a certificate used in establishing a trusted channel.

Findings:	[AGD] Section 3.12 describes the steps needed to import and configure certificates to be used for TLS and IPsec connections. When importing a certificate onto GSS, options for “IKEv2” or “WebDashboard” are provided for the administrator to define the certificates usage. If the certificate’s validity check fails, relevant logs will be generated. The administrator should reference the log messages to determine the reason for failure.

7.3.3.3 Tests

503 The evaluator shall perform the following test for each trusted channel:

504 The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the guidance documentation to determine that all supported administrator-configurable options behave in their documented manner.

High-Level Test Description
Attempt to validate a certificate via OCSP and CDP when the CDP and OCSP responder are offline.
Findings: PASS—The evaluator confirmed that when a TLS connection using a CDP or an IPsec connection using OCSP is attempted and the CDP or OCSP responder is offline, the connection is denied.

7.3.4 FIA_X509_EXT.2 X.509 Certificate Authentication (MOD VPNGW)

505 There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to support its use for IPsec at a minimum. The evaluator shall ensure that all evaluation of this SFR is performed against its use in IPsec communications as well as any other supported usage.

7.3.5 FIA_X509_EXT.3 Extended: X509 Certificate Requests

7.3.5.1 TSS

506 If the ST author selects "device-specific information", the evaluator shall verify that the TSS contains a description of the device-specific fields used in certificate requests.

Findings:	Device-specific information is not selected

7.3.5.2 Guidance Documentation

507 The evaluator shall check to ensure that the guidance documentation contains instructions on requesting certificates from a CA, including generation of a Certificate Request. If the ST author selects "Common Name", "Organization", "Organizational Unit", or "Country", the evaluator shall ensure that this guidance includes instructions for establishing these fields before creating the Certification Request.

Findings:	[AGD] Section 3.12 states that certificate signing requests can be generated through the Administrative GUI. This section contains separate instructions to use the 'Create CSR' option on the Cube and GSS. The ST author selected "Common Name" and [AGD] Section 3.12 provides instructions for establishing this field before creating the Certification Request.

7.3.5.3 Tests

508 The evaluator shall perform the following tests:

a) Test 1: The evaluator shall use the guidance documentation to cause the TOE to generate a Certification Request. The evaluator shall capture the generated message and ensure that it conforms to the format specified. The evaluator shall confirm that the Certification Request provides the public key and other required information, including any necessary user-input information.

High-Level Test Description
Generate a certificate request on the TOE and verify the captured request conforms to the format specified in the guidance.
Findings: PASS – The evaluator confirmed that the CSR generated on the TOE provides the CN and public key.

b) Test 2: The evaluator shall demonstrate that validating a response message to a Certification Request without a valid certification path results in the function failing. The evaluator shall then load a certificate or certificates as trusted CAs needed to validate the certificate response message, and demonstrate that the function succeeds.

High-Level Test Description
Attempt to validate a signed certificate request without a valid certificate path and verify this function fails. Upload the CA and then attempt to validate again and verify the function succeeds.
Findings: PASS – The evaluator confirmed that the TOE rejects the signed certificate request when it can't be validated by the TOE. After the trusted CA was installed onto the TOE, the signed certificate request was successfully validated.

7.3.6 FIA_X509_EXT.3 X.509 Certificate Requests (MOD VPNGW)

509 There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to make it mandatory because of the TOE's required support for IPsec.

7.4 Security management (FMT)

7.4.1 FMT_MOF.1/Services Management of security functions behaviour

7.4.1.1 TSS

510 For distributed TOEs see chapter 2.4.1.1.

Findings:	This activity refers to [NDcPP-SD] section 2.4.1.1. Refer to [AAR] section 3.4.1.1 for coverage.

511 For non-distributed TOEs, the evaluator shall ensure the TSS lists the services the Security Administrator is able to start and stop and how that how that operation is performed.

Findings:	The TOE is distributed.

7.4.1.2 Guidance Documentation

512 For distributed TOEs see chapter 2.4.1.2.

Findings:	This activity refers to [NDcPP-SD] section 2.4.1.2. Refer to [AAR] section 3.4.1.2 for coverage.

513 For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation describes how the TSS lists the services the Security Administrator is able to start and stop and how that how that operation is performed.

Findings:	The TOE is distributed.

7.4.1.3 Tests

514 The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) without prior authentication as Security Administrator (either by authenticating as a user with no administrator privileges, if possible, or without prior authentication at all). The attempt to enable/disable this service/these services should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to enable/disable this service/these services can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

High-Level Test Description
Attempt to disable a service without prior authentication as an administrator. Attempt to modify the crypto functions without prior authentication as an administrator.

High-Level Test Description
Findings: PASS – The evaluator confirmed that the TOE does not allow disabling the logging service or modifying the cryptographic functions without administrator authentication.

515 The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) with prior authentication as Security Administrator. The attempt to enable/disable this service/these services should be successful.

High-Level Test Description
Attempt to enable and disable at least one service with prior authentication as a security administrator.
Findings: PASS – The evaluator confirmed that the TOE allows the administrator to enable and disable services after successful authentication.

7.4.2 FMT_MTD.1/CryptoKeys Management of TSF Data

7.4.2.1 TSS

516 For distributed TOEs see chapter 2.4.1.1.

Findings:	This activity refers to [NDcPP-SD] section 2.4.1.1. Refer to [AAR] section 3.4.1.1 for coverage.

517 For non-distributed TOEs, the evaluator shall ensure the TSS lists the keys the Security Administrator is able to manage to include the options available (e.g. generating keys, importing keys, modifying keys or deleting keys) and how that how those operations are performed.

Findings:	The TOE is distributed.

7.4.2.2 Guidance Documentation

518 For distributed TOEs see chapter 2.4.1.2.

Findings:	This activity refers to [NDcPP-SD] section 2.4.1.2. Refer to [AAR] section 3.4.1.2 for coverage.

519 For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation lists the keys the Security Administrator is able to manage to include the options available (e.g. generating keys, importing keys, modifying keys or deleting keys) and how that how those operations are performed.

Findings:	The TOE is distributed

7.4.2.3 Tests

520 The evaluator shall try to perform at least one of the related actions (modify, delete, generate/import) without prior authentication as Security Administrator (either by authentication as a non-administrative user, if supported, or without authentication at all). Attempts to perform related actions without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to

the point where the attempt to manage cryptographic keys can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

High-Level Test Description
Refer to the findings found in FMT_MOF.1/Services to view what occurs when a user attempts to modify anything on the TOE without prior authentication
Findings: PASS--The evaluator confirmed that the TOE does not allow the cryptographic functions to be modified without administrator authentication in conjunction with FMT_MOF.1/Services testing.

521 The evaluator shall try to perform at least one of the related actions with prior authentication as Security Administrator. This attempt should be successful.

High-Level Test Description
This is performed in conjunction with FIA_X509_EXT.3.
Findings: PASS—The evaluator confirmed that the administrator can generate and import cryptographic key in conjunction with FIA_X509_EXT.3 testing.

7.4.3 FMT_MTD.1/CryptoKeys Management of TSF Data (MOD VPNGW)

522 There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to make it mandatory and to state that it applies specifically to the keys and certificates used for VPN operation. The evaluator shall perform the Evaluation Activities as written for this SFR as applicable to the VPN cryptographic data.

8 Evaluation Activities for Security Assurance Requirements

8.1 ASE: Security Target

8.1.1 General ASE

523 When evaluating a Security Target, the evaluator performs the work units as presented in the CEM. In addition, the evaluator ensures the content of the TSS in the ST satisfies the EAs specified in Section 2 (Evaluation Activities for SFRs).

Findings:	See above sections.

524 For distributed TOEs only the SFRs classified as 'all' have to be fulfilled by all TOE parts. The SFRs classified as 'One' or 'Feature Dependent' only have to be fulfilled by either one or some TOE parts, respectively. To make sure that the distributed TOE as a whole fulfills all the SFRs the following actions for ASE_TSS.1 have to be performed as part of ASE_TSS.1.1E.

ASE_TSS.1 element	**Evaluator Action**
ASE_TSS.1.1C	The evaluator shall examine the TSS to determine that it is clear which TOE components contribute to each SFR or how the components combine to meet each SFR. The evaluator shall verify the sufficiency to fulfil the related SFRs. This includes checking that the TOE as a whole fully covers all SFRs and that all functionality that is required to be audited is in fact audited regardless of the component that carries it out.

Findings:	See above sections.

8.2 ADV: Development

8.2.1 Basic Functional Specification (ADV_FSP.1)

525 The EAs for this assurance component focus on understanding the interfaces (e.g., application programming interfaces, command line interfaces, graphical user interfaces, network interfaces) described in the AGD documentation, and possibly identified in the TOE Summary Specification (TSS) in response to the SFRs. Specific evaluator actions to be performed against this documentation are identified (where relevant) for each SFR in Section 2, and in EAs for AGD, ATE and AVA SARs in other parts of Section 5.

526 The EAs presented in this section address the CEM work units ADV_FSP.1- 1, ADV_FSP.1-2, ADV_FSP.1-3, and ADV_FSP.1-5.

527 The EAs are reworded for clarity and interpret the CEM work units such that they will result in more objective and repeatable actions by the evaluator. The EAs in this SD are intended to ensure the evaluators are consistently performing equivalent actions.

528 The documents to be examined for this assurance component in an evaluation are therefore the Security Target, AGD documentation, and any required supplementary information required by the cPP: no additional "functional specification" documentation is necessary to satisfy the EAs. The interfaces that need to be evaluated are also identified by reference to the EAs listed for each SFR and are expected to be identified in the context of the Security Target, AGD documentation, and any required supplementary information defined in the cPP rather than as a separate list specifically for the purposes of CC evaluation. The direct identification of documentation requirements and their assessment as part of the EAs for each SFR also means that the tracing required in ADV_FSP.1.2D (work units ADV_FSP.1-4, ADV_FSP.1-6 and ADV_FSP.1-7) is treated as implicit and no separate mapping information is required for this element.

8.2.1.1 Evaluation Activity:

529 *The evaluator shall examine the interface documentation to ensure it describes the purpose and method of use for each TSFI that is identified as being security relevant.*

530 In this context, TSFI are deemed security relevant if they are used by the administrator to configure the TOE, or to perform other administrative functions (e.g. audit review or performing updates). Additionally, those interfaces that are identified in the ST, or guidance documentation, as adhering to the security policies (as presented in the SFRs), are also considered security relevant. The intent is that these interfaces will be adequately tested and having an understanding of how these interfaces are used in the TOE is necessary to ensure proper test coverage is applied.

531 The set of TSFI that are provided as evaluation evidence are contained in the Administrative Guidance and User Guidance.

Findings:	From section 7.2.1 of the NDcPP : "For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation." The [ST] and the AGD comprise the functional specification. If the test in [SD] cannot be completed because the [ST] or the AGD are incomplete, then the functional specification is not complete, and observations are required. During the evaluator's use of the product and its interface (the Administrative GUI), there were no areas that were deficient.

8.2.1.2 Evaluation Activity

532 *The evaluator shall check the interface documentation to ensure it identifies and describes the parameters for each TSFI that is identified as being security relevant.*

Findings:	See comments in the previous work unit.

8.2.1.3 Evaluation Activity:

533 *The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.*

534 The evaluator uses the provided documentation and first identifies, and then examines a representative set of interfaces to perform the EAs presented in Section 2, including the EAs associated with testing of the interfaces.

535 It should be noted that there may be some SFRs that do not have an interface that is explicitly "mapped" to invoke the desired functionality. For example, generating a random bit string, destroying a cryptographic key that is no longer needed, or the TSF failing to a secure state, are capabilities that may be specified in SFRs, but are not invoked by an interface.

536 However, if the evaluator is unable to perform some other required EA because there is insufficient design and interface information, then the evaluator is entitled to conclude that an adequate functional specification has not been provided, and hence that the verdict for the ADV_FSP.1 assurance component is a 'fail'.

Findings:	See comments in the previous work unit.

8.3 AGD: Guidance Documents

537 It is not necessary for a TOE to provide separate documentation to meet the individual requirements of AGD_OPE and AGD_PRE. Although the EAs in this section are described under the traditionally separate AGD families, the mapping between the documentation provided by the developer and AGD_OPE and AGD_PRE requirements may be many-to-many, as long as all requirements are met in documentation that is delivered to Security Administrators and users (as appropriate) as part of the TOE.

538 Note that additional Evaluation Activities for the guidance documentation in the case of a distributed TOE are defined in section A.9.1.1. (in the NDcPP-SD)

8.3.1 Operational User Guidance (AGD_OPE.1)

539 The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR.

540 In addition, the evaluator performs the EAs specified below.

8.3.1.1 Evaluation Activity:

541 *The evaluator shall ensure the Operational guidance documentation is distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.*

Findings:	The documentation is available for public download from the NIAP website as described in [ST] Section 2.4.2.

8.3.1.2 Evaluation Activity

542 *The evaluator shall ensure that the Operational guidance is provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.*

Findings:	There is only one operational environment claimed in the [ST]. All TOE platforms claimed in [ST] are covered by the operational guidance. This is evidenced by the platform equivalency.

8.3.1.3 Evaluation Activity

543 *The evaluator shall ensure that the Operational guidance contains instructions for configuring any cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.*

Findings:	There are no other cryptographic engines used by the TOE. The [AGD] Section 3.11 contains instructions for configuring the cryptographic engine associated with the evaluated configuration.

8.3.1.4 Evaluation Activity

544 *The evaluator shall ensure the Operational guidance makes it clear to an administrator which security functionality and interfaces have been assessed and tested by the EAs.*

Findings:	The [AGD] document covers configuration of the in-scope functionality where additional configuration might be required.

8.3.1.5 Evaluation Activity

545 In addition the evaluator shall ensure that the following requirements are also met.

a) The guidance documentation shall contain instructions for configuring any cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

NIAP TD0536

b) The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps:

5) Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).

6) Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes instructions that describe at least one method of validating the hash/digital signature.

c) The TOE will likely contain security functionality that does not fall in the scope of evaluation under this cPP. The guidance documentation shall make it clear to an administrator which security functionality is covered by the Evaluation Activities.

Findings:	The [AGD] Section 3.11 contains instructions for configuring the cryptographic engine associated with the evaluated configuration. [AGD] section 2.2 provides instructions for the download and verification of the TOE updates. The [AGD] document covers configuration of the in-scope functionality where additional configuration might be required. Section 1.3.4 clearly identifies functionality that is not included in the evaluated configuration.

8.3.2 Preparative Procedures (AGD_PRE.1)

546 The evaluator performs the CEM work units associated with the AGD_PRE.1 SAR. Specific requirements and EAs on the preparative documentation are identified (and where relevant are captured in the Guidance Documentation portions of the EAs) in the individual EAs for each SFR.

547 Preparative procedures are distributed to Security Administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that Security Administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

548 In addition, the evaluator performs the EAs specified below.

8.3.2.1 Evaluation Activity:

549 *The evaluator shall examine the Preparative procedures to ensure they include a description of how the Security Administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target).*

550 The documentation should be in an informal style and should be written with sufficient detail and explanation that they can be understood and used by the target audience (which will typically include IT staff who have general IT experience but not necessarily experience with the TOE product itself).

Findings:	[AGD] section 3 provides instructions for configuration of the Operational Environment. The evaluator determined it was written with sufficient detail and explanation to be understood and used by the target audience.

8.3.2.2 Evaluation Activity

551 *The evaluator shall examine the Preparative procedures to ensure they are provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.*

Findings:	The evaluator examined the [AGD] and ensured that the Preparative procedures are provided for every platform and Operational Environment that the product supports.

8.3.2.3 Evaluation Activity

552 *The evaluator shall examine the preparative procedures to ensure they include instructions to successfully install the TSF in each Operational Environment.*

Findings:	See previous work unit.

8.3.2.4 Evaluation Activity

553 *The evaluator shall examine the preparative procedures to ensure they include instructions to manage the security of the TSF as a product and as a component of the larger operational environment.*

Findings:	The guidance documentation provides extensive information on managing the security of the TOE as an individual product. Additional best practice guidance provided within those documents help instill a culture of secure manageability within a larger operational environment.

8.3.2.5 Evaluation Activity

554 In addition the evaluator shall ensure that the following requirements are also met.

555 The preparative procedures must:

a) include instructions to provide a protected administrative capability; and

b) identify TOE passwords that have default values associated with them and instructions shall be provided for how these can be changed.

Findings:	The [AGD] section 3 specifies the secure installation of the TOE to provide a protected interface. No passwords have been identified that have default values.

8.4 ALC: Life-cycle Support

8.4.1 Labelling of the TOE (ALC_CMC.1)

556 When evaluating that the TOE has been provided and is labelled with a unique reference, the evaluator performs the work units as presented in the CEM.

Findings:	Please refer to associated ETR document for ALC_CMC.1 work units.

8.4.2 TOE CM coverage (ALC_CMS.1)

557 When evaluating the developer's coverage of the TOE in their CM system, the evaluator performs the work units as presented in the CEM.

Findings:	Please refer to associated ETR document for ALC_CMS.1 work units.

8.5 ATE: Tests

8.5.1 Independent Testing – Conformance (ATE_IND.1)

558 The focus of the testing is to confirm that the requirements specified in the SFRs are being met. Additionally, testing is performed to confirm the functionality described in the TSS, as well as the dependencies on the Operational guidance documentation is accurate.

559 The evaluator performs the CEM work units associated with the ATE_IND.1 SAR. Specific testing requirements and EAs are captured for each SFR in Sections 2, 3 and 4.

560 The evaluator should consult Appendix 709 when determining the appropriate strategy for testing multiple variations or models of the TOE that may be under evaluation.

561 Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in section A.9.3.1.

Findings:	Please refer to associated ETR document for ATE_IND.1 work units. Section 8.7 below addresses additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE.

8.6 Vulnerability Assessment

8.6.1 Vulnerability Survey (AVA_VAN.1)

562 While vulnerability analysis is inherently a subjective activity, a minimum level of analysis can be defined and some measure of objectivity and repeatability (or at least comparability) can be imposed on the vulnerability analysis process. In order to achieve such objectivity and repeatability it is important that the evaluator follows a set of well-defined activities and documents their findings so others can follow their arguments and come to the same conclusions as the evaluator. While this does not guarantee that different evaluation facilities will identify exactly the same type of vulnerabilities or come to exactly the same conclusions, the approach defines the minimum level of analysis and the scope of that analysis and provides Certification Bodies a measure of assurance that the minimum level of analysis is being performed by the evaluation facilities

563 In order to meet these goals some refinement of the AVA_VAN.1 CEM work units is needed. The following table indicates, for each work unit in AVA_VAN.1, whether the CEM work unit is to be performed as written, or if it has been clarified by an Evaluation Activity. If clarification has been provided, a reference to this clarification is provided in the table.

564 Because of the level of detail required for the evaluation activities, the bulk of the instructions are contained in Appendix A, while an “outline” of the assurance activity is provided below.

8.6.1.1 Evaluation Activity (Documentation):

565 In addition to the activities specified by the CEM in accordance with Table 2, the evaluator shall perform the following activities.

566 *The evaluator shall examine the documentation outlined below provided by the developer to confirm that it contains all required information. This documentation is in addition to the documentation already required to be supplied in response to the EAs listed previously.*

NIAP TD0547

567 The developer shall provide documentation identifying the list of software and hardware components that compose the TOE. Hardware components should identify

at a minimum the processors used by the TOE. Software components include applications, the operating system and other major components that are independently identifiable and reusable (outside of the TOE), for example a web server, protocol or cryptographic libraries, (independently identifiable and reusable components are not limited to the list provided in the example). This additional documentation is merely a list of the name and version number of the components and will be used by the evaluators in formulating vulnerability hypotheses during their analysis.

Findings:	The evaluator collected this information from the developer which was used to feed into the Type 1 Flaw Hypotheses search (below).

568 If the TOE is a distributed TOE then the developer shall provide:

a) documentation describing the allocation of requirements between distributed TOE components as in [NDcPP, 3.4]

b) a mapping of the auditable events recorded by each distributed TOE component as in [NDcPP, 6.3.3]

c) additional information in the Preparative Procedures as identified in the refinement of AGD_PRE.1 in additional information in the Preparative Procedures as identified in 3.4.1.2 and 3.5.1.2.

8.6.1.2 Evaluation Activity:

569 The evaluator formulates hypotheses in accordance with process defined in Appendix A. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.3. The evaluator shall perform vulnerability analysis in accordance with Appendix A.2. The results of the analysis shall be documented in the report according to Appendix A.3.

Findings:	The following sources of public vulnerabilities were considered in formulating the specific list of flaws to be investigated by the evaluators, as well as to reference in directing the evaluators to perform key-word searches during the evaluation of the TOE. Hypothesis sources for public vulnerabilities were: NIST National Vulnerabilities Database (can be used to access CVE and US-CERT databases identified below): https://web.nvd.nist.gov/view/vuln/search Common Vulnerabilities and Exposures: https://cve.mitre.org/cve/search_cve_list.html US-CERT: http://www.kb.cert.org/vuls/html/search Tenable Network Security: https://www.tenable.com/cve Tipping Point Zero Day Initiative: https://www.zerodayinitiative.com/advisories Offensive Security Exploit Database: https://www.exploit-db.com/ Rapid7 Vulnerability Database: https://www.rapid7.com/db/vulnerabilities Type 1 Hypothesis searches were last conducted on December 2, 2022 and included the following search terms: Openssl

	Strongswan Syslog-ng GoSilent GoSilent Cube GoSilent Server IPSec TLS Linux kernel Intel Xeon E3-1270 v5 AllWinner H5 ARM v8 Cortex-A53 GSC-100 GSC-120 The evaluation team determined that no residual vulnerabilities exist based on these searches that are exploitable by attackers with Basic Attack Potential. There are no type-2 hypotheses identified for the NDcPP. The evaluation team developed Type 3 flaw hypotheses in accordance with Sections A.1.3, A.1.4, and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential. The evaluation team developed Type 4 flaw hypotheses in accordance with Sections A.1.3, A.1.4, and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

8.7 Evaluating additional components for a distributed TOE

570 In the case of a distributed TOE the Security Target will identify an evaluated configuration that consists of a number of separate components chosen by the ST author, which collectively satisfy the requirements of the cPP. This evaluated configuration need not be the minimum set of components that could possibly meet the cPP (e.g. if the TOE is intended for large enterprise deployments then the evaluated configuration might include some redundancy in components in order to support expected connectivity and loads), but because this is the main configuration referred to in the ST and the evaluation, it is treated in this section as the minimum configuration of interest and is referred to here as the 'minimum configuration' as well as the 'evaluated configuration'.

571 In addition to the minimum configuration above, the ST may also identify (at the author's discretion, and subject to verification as described in this section) which TOE components can have instances added to an operational configuration without affecting the validity of the CC certification. The ST description may include

constraints on how such components are added, including required and/or prohibited configurations of the components.

572 Extra instances of a TOE component must have the same hardware and software as the original component included in the evaluated configuration.

573 It is noted that undesirable configurations may be possible in the operational deployment of a TOE – such as allowing a TOE component to be managed from separate and potentially conflicting administration domains. However, the definition of 'undesirable' and of the risks involved in such cases will be specific to each operational environment and is therefore not treated as part of the evaluation. Correct and appropriate configuration of this sort remains a matter for expert network planning and design in the operational environment.

8.7.1 Evaluator Activities for Assessing the ST

8.7.1.1 TSS

574 The evaluator shall examine the TSS to identify any extra instances of TOE components allowed in the ST and shall examine the description of how the additional components maintain the SFRs to confirm that it is consistent with the role that the component plays in the evaluated configuration. For example: the secure channels used by the extra component for intra-TOE communications (FPT_ITT) and external communications (FTP_ITC) must be consistent, the audit information generated by the extra component must be maintained, and the management of the extra component must be consistent with that used for the original instance of the component in the minimum configuration.

Findings:	[ST] Section 2.2.1 describes that the TOE is deployed in a distributed configuration with the GoSilent Server providing central management of the GoSilent Server and GoSilent Cube devices as well acting as a central peer for all IPsec connection from the GoSilent Cube devices. All extra instances of GoSilent Cubes are identical to the original instance. All GoSilent Cube devices connect to the GoSilent Server using the IPsec protocol as claimed in FPT_ITT.1 and communicate with the syslog server over TLS as claimed in FPT_ITC.1. The extra instances of GoSilent Cubes maintain the audit requirements of the original instance and are centrally managed by the GoSilent Server which is consistent with the original instance of the component in the minimum configuration. There are no unclaimed communication channels (FTP_ITT, FTP_ITC) identified on any extra instances as per [ST] Section 6.5.1, "By default, no traffic is allowed to flow from GoSilent Cube users to the External Network, no traffic is allowed to flow from the External Network to GoSilent Cube users, and no traffic is allowed to flow between GoSilent Server interfaces."

8.7.2 Evaluator Activities for Assessing the Guidance Documentation

8.7.2.1 Guidance Documentation

575 The evaluator shall examine the description of the extra instances of TOE components in the guidance documentation to confirm that they are consistent with those identified as allowed in the ST. This includes confirmation that the result of applying the guidance documentation to configure the extra component will leave the TOE in a state such that the claims for SFR support in each component are as described in the ST and therefore that all SFRs continue to be met when the extra components are present.

576 The evaluator shall examine the secure communications described for the extra components to confirm that they are the same as described for the components in the minimum configuration (additional connections between allowed extra components and the components in the minimum configuration are allowed of course).

Findings:	There are no differences between the extra instances of the GoSilent Cube devices. The evaluator confirmed that they are consistent with those identified in the ST. The evaluator confirmed that the result of applying the guidance documentation to configure the extra components will leave the TOE in state such that all SFRs continue to be met. The evaluator examined the secure communications described for the extra components and confirmed that they are the same as described for the components in the minimum configuration. There were no additional connections identified between the extra components that were not claimed in the minimum configuration.

8.7.3 Evaluator Activities for Testing the TOE

8.7.3.1 Tests

577 The evaluator tests the TOE in the minimum configuration as defined in the ST (and the guidance documentation).

578 If the description of the use of extra components in the ST and guidance documentation identifies any difference in the SFRs allocated to a component, or the scope of the SFRs involved (e.g. if different selections apply to different instances of the component) then the evaluator tests these additional SFR cases that were not included in the minimum configuration.

Findings:	Table 20 in [ST] section 7.4 states the SFR distribution between components. All components were tested for all relevant SFRs. The GoSilent Cube deviates in the SFR selections is identified for FIA_UIA_EXT.1 where the GoSilent Cube only allows local access and has access to the Factory Reset function prior to login. This testing is performed with FIA_UIA_EXT.1 Test 3.

579 In addition, the evaluator tests the following aspects for each extra component that is identified as allowed in the distributed TOE:

580 Communications: the evaluator follows the guidance documentation to confirm, by testing, that any additional connections introduced with the extra component and not present in the minimum configuration are consistent with the requirements stated in the ST (e.g. with regard to protocols and ciphersuites used). An example of such an additional connection would be if a single instance of the component is present in the minimum configuration and adding a duplicate component then introduces an extra communication between the two instances. Another example might be if the use of the additional components necessitated the use of a connection to an external authentication server instead of using locally stored credentials.

High-Level Test Description
Repeat FCO_CPC_EXT.1 Test 1.2, this time adding a 2nd GoSilent Cube to the environment. Verify the cube communicates with the GoSilent Server. Next, attempt to connect the secondary GoSilent Cube to the first Go Silent Cube. This connection should fail (GoSilent Cubes can only communicate with GoSilent Servers).

High-Level Test Description
Findings: PASS – The evaluator confirmed that the GoSilent Cube only communicates with the GoSilent Server. The GoSilent Cube did respond to communication attempts from another GoSilent Cubes.

581 Audit: the evaluator confirms that the audit records from different instances of a component can be distinguished so that it is clear which instance generated the record.

Findings:	Each component of the TOE stores audit logs locally and transmits logs to remote syslog. Log messages include a hostname field which uniquely identifies which component generated the event.

582 Management: if the extra component manages other components in the distributed TOE then the evaluator shall follow the guidance documentation to confirm that management via the extra component uses the same roles and role holders for administrators as for the component in the minimum configuration.

Findings:	This is not applicable. The GoSilent Cube does not manage other components of the distributed TOE.