Infoblox Trinzic Appliances with NIOS v6.3 (Models: IB-810, IB-820, IB-1400, IB-1410, IB-1420, IB-2200, IB-2210, IB-2220, IB-4000 and IB-4010)

Security Target

Document Version: 1.0

Prepared For:
Infoblox
4750 Patrick Henry Drive
Santa Clara, CA 95054

Prepared By:
CSC
7231 Parkway Drive
Hanover, MD 21076

This page is intentionally blank.

Table of Contents

List of Table

1 Security Target Introduction

This Chapter presents Security Target (ST) identification information and an overview of the ST. An ST contains the information technology (IT) security requirements of an identified Target of Evaluation (TOE) and specifies the functional and assurance security measures offered by that TOE to meet stated requirements. An ST principally defines:

a) A security problem expressed as a set of assumptions about the security aspects of the environment, a list of threats that the product is intended to counter, and any known rules with which the product must comply (Chapter 3, TOE Security Environment).

b) A set of security objectives and a set of security requirements to address the security problem (Chapters 4, 5 and 6, Security Objectives, Extended Components Definition, and IT Security Requirements, respectively).

c) The IT security functions provided by the TOE that meet the set of requirements (Chapter 7, TOE Summary Specification).

The structure and content of this ST comply with the requirements specified in the Common Criteria (CC), Part 1, Annex A.

1.1 ST and TOE Identification

This section provides information needed to identify and control this ST and its associated TOE. This ST targets Evaluation Assurance Level (EAL) 2 augmented with ALC_FLR.2 and ALC_DVS.1.

ST Title:	Infoblox Trinzic Appliances with NIOS v6.3 (Models: IB-810, IB-820, IB-1400, IB-1410, IB-1420, IB-2200, IB-2210, IB-2220, IB-4000 andIB-4010) Security Target
ST Version:	1.0
Revision Number:	20
Publication Date:	25 September 2012
Authors:	CSC Security Testing and Certification Laboratories, Infoblox
TOE Identification:	Infoblox Trinzic Appliances with NIOS v6.3.15 (Models: IB-810, IB-820, IB-1400, IB-1410, IB-1420, IB-2200, IB-2210, IB-2220, IB-4000 and IB-4010)
ST Evaluator:	CSC Security Testing and Certification Laboratories
Keywords:	network device, secure DNS, DHCP, IP Address Management (IPAM),

1.2 TOE Overview

The Infoblox Trinzic Appliances with NIOS v6.3 (hereafter referred to as Infoblox Trinzic Appliances or the TOE) are a family of network appliances which consolidate the delivery and management of core IP network services historically provided by multiple general purpose operating systems and servers (core IP network services include DNS, DHCP, IPAM, FTP, TFTP, and HTTP). The NIOS operating system is a hardened version of the Fedora Linux distribution optimized for security and network perfomance. The appliance models are differentiated by performance, capacity and availabilty to support various deployment scenarios such as a branch-office or large enterprise.

The TOE provides the following major security features:

- **Secure management.** Administrators manage the TOE via a TLS protected web GUI or via the CLI console port. The TOE implements role based access control, password based authentication and auditing of management functions. Communication with the TOE's API interface is protected by TLS.
- **High availability.** The TOE enforces quotas on exhaustible resources thereby preventing failover due to resource exhaustion.
- **Trusted updates.** The TOE uses digital signatures to verify updates prior to installation.
- **Self protection.** The TOE performs self-test at startup to verify the integrity of hardware components and the cryptographic module.
- **Secure DNS.** The TOE employs secure DNS protocols to verify and authenticate DNS updates.
- **Secure Grid.** The TOE uses an SSL/TLS VPN to protect commincation between itself and other TOE instances when deployed in a grid.

An example enterprise deployment of the TOE is shown in Figure 1 below (**Note:** the TOE does not include software running on Cisco devices).

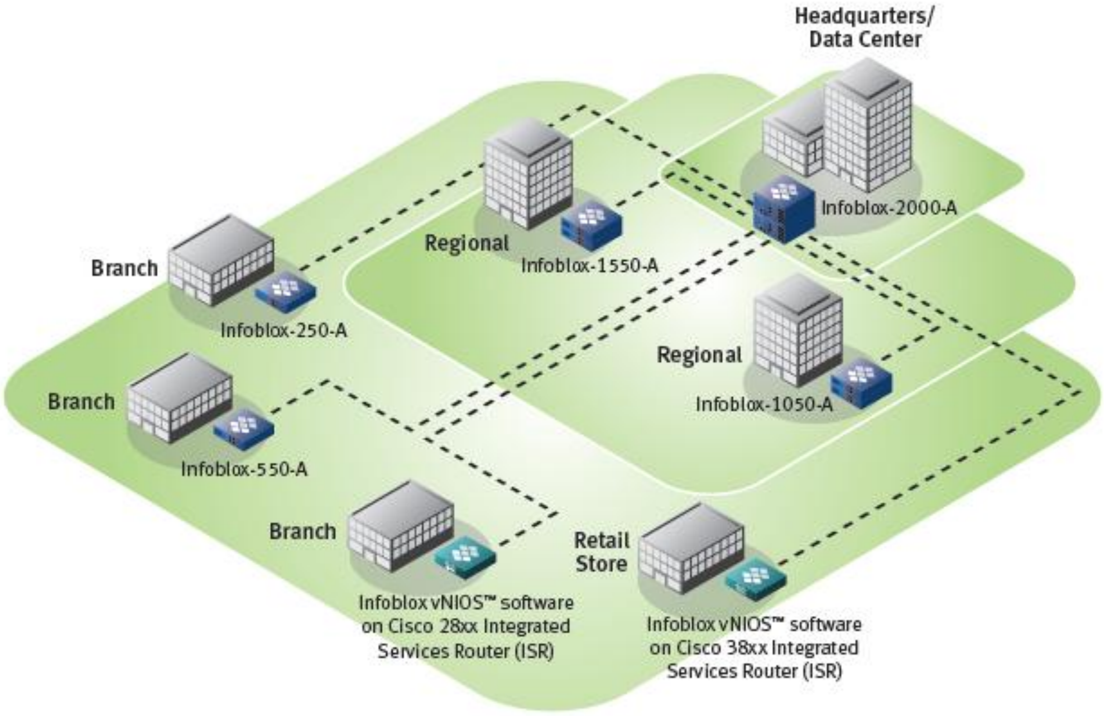

Figure 1: Example TOE Deployment

The Infoblox Trinzic Appliances within the scope of evaluation are shown in Table 1.

Table 1: TOE Models

Infoblox Model	CPU	Memory	Disk	Operating System
IB-810	Intel Pentium 9650	2gb	300gb	NIOS 6.3.15
IB-820	Intel Pentium 9650	2gb	300gb	NIOS 6.3.15
IB-1400	Intel Xeon X3450	8gb	2 x 600gb – RAID-1	NIOS 6.3.15
IB-1410	Intel Xeon X3450	8gb	300gb	NIOS 6.3.15
IB-1420	Intel Xeon X3450	8gb	300gb	NIOS 6.3.15
IB-2200	Intel Xeon 5620	24gb	4 x 600gb – RAID-10	NIOS 6.3.15
IB-2210	Intel Xeon 5620	24gb	4 x 600gb – RAID-10	NIOS 6.3.15
IB-2220	Intel Xeon 5620	24gb	4 x 600gb – RAID-10	NIOS 6.3.15
IB-4000	Intel Xeon 5650	6 x 4G	8 x hot plug 2.5-inch SAS	NIOS 6.3.15
IB-4010	Intel Xeon 5650	6 x 4G	4 x hot plug 2.5-inch SAS	NIOS 6.3.15

1.2.1 TOE Type

The TOE is a network appliance which provides core network services including DNS, DHCP, IPAM, FTP, TFTP, and HTTP.

1.2.2 Required Non-TOE Hardware, Software, and Firmware

The TOE incorporates all hardware, software and firmware of the appliances listed in Table 1. Depending on the administrator defined configuration, the TOE may require the following services to be present in the environment:

- Active Directory when the TOE is configured to use an external authentication source
- NTP server when the TOE is configured to use an NTP server
- Kerberos server where GSS-TSIG or external authentication is enabled

1.3 TOE Description

This section provides context for the TOE evaluation by identifying the logical and physical scope of the TOE, as well as its evaluated configuration.

1.3.1 Physical Scope of the TOE

The physical scope of the TOE comprises all hardware, software and firmware of the appliances listed in Table 1 and the guidance documents listed as below:

- Infoblox Administrator Guide (6.3)
- Infoblox CLI Guide (NIOS 6.3)
- Infoblox API Documentation (NIOS 6.3)
- Infoblox CSV Import Reference (NIOS 6.3)
- Infoblox Installation Guide for the Trinzic 800 Appliances
- Infoblox Installation Guide for the Trinzic 1400 Appliances
- Infoblox Installation Guide for the Trinzic 2200 Appliances
- Infoblox Installation Guide for the IB-4010 Appliance
- Infoblox Installation Guide for the Trinzic Reporting 1400 Appliance
- Infoblox Installation Guide for the Trinzic Reporting 2200 Appliance

- Infoblox Installation Guide for the Trinzic Reporting 4000 Appliance
- NIOS 6.3.15 Release Notes
- Infoblox Safety Guide

Figure 2 below depicts the typical physical aspects of the Infoblox Trinzic Appliances.

Figure 2: Infoblox Trinzic Appliance

1.3.2 Logical Scope of the TOE

The TOE logical boundary is comprised of the following security functions:

- TSF_TOE_COMM
- TSF_TRUSTED_UPDATES
- TSF_AUDIT
- TSF_TOE_ACCESS
- TSF_RESOURCE_EXHAUSTION
- TSF_USER_DATA_DISCLOSURE
- TSF_SELF_TEST

1.3.2.1 TSF_TOE_COMM

The TOE communicates with other network devices, as well as administrators, over the network. The critical communication paths and their related protection mechanisms are as follows:

- **Grid communication.** The TOE may be configured to communicate with other TOE instances in a grid. This communication is protected via an SSL/TLS VPN.
- **Remote Administration.** Remote administrators configure the TOE via a web based GUI that is protected using TLS/HTTPS. **Note:** In the evaluated configuration, CLI access (which is used for diagnostic purposes) is restricted to the local serial port (refer to section 1.3.3).
- **Application Programming Interface.** The TOE provides a Perl API to assist integration of the Infoblox device into network environments. The API is protected using TLS/HTTPS. The Perl API provides interfaces to DHCP, DNS, Grid and IPAM services.
- **DNS.** The TOE implements DNSSEC, TSIG (Transaction SIGnature) and GSS-TSIG (Generic Security Service Algorithm for Secret Key Transaction) to verify DNS updates between itself and other trusted IT products. These options must be configured according to the evaluated configuration (refer to section 1.3.3).

1.3.2.2 TSF_TRUSTED_UPDATES

The TOE provides security administrators with the ability to query the current version of the TOE firmware/software and perform updates. The TOE verifies RSA digital signatures associated with TOE updates. The certificate used for validation is stored in a protected file on the appliance.

1.3.2.3 TSF_AUDIT

The TOE generates audit records associated with use of the administrative functions. Audit records may be stored locally or on a Syslog server. The TOE deletes the oldest records if the audit trail exceeds a defined maximum. A local time source supports reliable time stamps for the audit function.

1.3.2.4 TSF_TOE_ACCESS

The TOE provides administrative access via a console port (local) and HTTPS (remote). The TOE provides a password-based logon mechanism for local and remote access and enforces a defined password complexity and expiration policy. The TOE optionally supports authentication against an Active Directory server.

The TOE enforces Role Based Access Control (RBAC), session timeouts and displays an advisory banner at login.

1.3.2.5 TSF_RESOURCE_EXHAUSTION

The TOE enforces maximum quotas on log files, uploaded files and number of simultaneous GUI and API sessions.

1.3.2.6 TSF_USER_DATA_DISCLOSURE

The TOE ensures that any previous information content of a resource is made unavailable upon the allocation of the resource to all objects by clearing the residual information before network packets are sent from the TOE.

1.3.2.7 TSF_SELF_TEST

The TOE implements self-test, during initial startup, to verify the basic hardware components that the TOE is relient upon, and also verify the integrity of the cryptographic module. The TOE will log to sys log (internal system log) when the test runs on power-up. In the event of failure, the TOE is not permitted to use cryptography services and the appliance will start in a broken state where the only permitted access is via the physical serial port. For appliances with an LCD panel on the front, they will display a failure message.

1.3.3 Evaluated Configuration

In the evaluated configuration the TOE is deployed as described in the guidance documents which are delivered with the TOE. The TOE is evaluated using the following configuration settings:

- DNSSEC is enabled (zone policy configured according to user needs)
- TSIG is configured for dynamic DNS updates from ISC DHCP servers and DNS clients (if applicable to the environment)
- GSS-TSIG is configured for dynamic DNS updates from Microsoft DHCP servers and DNS servers and clients (if applicable to the environment)
- bloxTools is disabled
- SSH is disabled (CLI access is performed via the local consol port)
- RADIUS authentication is disabled
- TACACS+ authentication is disabled
- Secure Copy (SCP) is disabled / not used

2 Conformance Claims

This section describes the conformance claims of this Security Target.

2.1 Common Criteria Conformance Claims

The Security Target is based upon:

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; Version 3.1, Revision 3, CCMB-2009-07-001;
- Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components; Version 3.1, Revision 3, CCMB-2009-07-002;
- Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Components; Version 3.1, Revision 3, CCMB-2009-07-003.

This Security Target claims the following CC conformance:

- Part 2 extended
- Part 3 conformant, augmented with *Security Requirements for Network Devices* Assurance Activities (refer to section 2.2)
- Evaluation Assurance Level (EAL) 2 augmented with ALC_FLR.2 and ALC_DVS.1

2.2 Protection Profile Conformance Claims

This Security Target **does not** claim conformance to any Protection Profile (PP) however it incorporates the security functional requirements (SFRs) from the *Security Requirements for Network Devices, Version 1.0, December 10, 2010*, except for FAU_STG_EXT.1 and FAU_STG_EXT.3.

Note that as SFRs from the *Security Requirements for Network Devices* PP are included in the ST, the additional assurance activities, which are identified by “Assurance Activity” in Sections 4.2 and Annex C of *Security Requirements for Network Devices* PP shall be applicable to the evaluation of the TOE, in addition to the assurance activities required for EAL2+ (ALC_FLR.2 and ALC_DVS.1).

3 Security Problem Definition

The Security Problem Definition describes assumptions about the operational environment in which the TOE is intended to be used and represents the conditions for the secure operation of the TOE.

3.1 Assumptions

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

Table 2: Assumptions for the TOE

Assumption Name	Assumption Definition
A.NO_GENERAL_PURPOSE	It is assumed that there are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.
A.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the environment.
A.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.
A.ACCESS	External authentication entities will be properly configured and operate correctly

3.2 Threats

This security problem definition addresses threats posed by four categories of threat agents:

a) Persons who are not permitted to use the TOE who may attempt to use the TOE

b) Persons who are authorized to use the TOE who may attempt to use TOE functions for which they are not authorized.

c) Persons who are authorized to use the TOE who may attempt to access data in ways for which they not authorized.

d) Persons who unintentionally cause a software malfunction that may expose the TOE to unanticipated threats.

The threats and policies defined in this Security Target address the threats posed by these threat agents.

3.2.1 Threats Addressed by the TOE

This section describes the threats that are addressed by the TOE.

Table 3: Threats Addressed by the TOE

Threat	Description
T.ADMIN_ERROR	An administrator may unintentionally install or configure the TOE incorrectly, resulting in ineffective security mechanisms.

Threat	Description
T.RESOURCE_EXHAUSTION	A process or user may deny access to TOE services by exhausting critical resources on the TOE.
T.TSF_FAILURE	Security mechanisms of the TOE may fail, leading to a compromise of the TSF.
T.UNDETECTED_ACTIONS	Malicious remote users or external IT entities may take actions that adversely affect the security of the TOE. These actions may remain undetected and thus their effects cannot be effectively mitigated.
T.UNAUTHORIZED_ACCESS	A user may gain unauthorized access to the TOE data and TOE executable code. A malicious user, process, or external IT entity may masquerade as an authorized entity in order to gain unauthorized access to data or TOE resources. A malicious user, process, or external IT entity may misrepresent itself as the TOE to obtain identification and authentication data.
T.UNAUTHORIZED_UPDATE	A malicious party attempts to supply the end user with an update to the product that may compromise the security features of the TOE.
T.USER_DATA_REUSE	User data may be inadvertently sent to a destination not intended by the original sender.

3.2.2 Threats addressed by the IT Environment

There are no threats addressed by the IT Environment.

3.3 Organizational Security Policies

This section describes the Organizational Security Policies (OSPs) that apply to the TOE. OSPs are used to provide a basis for security objectives that are commonly desired by TOE Owners in this operational environment, but for which it is not practical to universally define the assets being protected or the threats to those assets.

Table 4: Organizational Security Policies for the TOE

Name	Definition
P.ACCESS_BANNER	The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE.

4 Security Objectives

The purpose of the security objectives is to detail the planned response to a security problem or threat. Threats can be directed against the TOE, against the security environment, or both; therefore, the CC identifies two categories of security objectives:

- Security objectives for the TOE, and
- Security objectives for the environment.

4.1 Security Objectives for the TOE

This section describes the security objectives that the TOE shall fulfill.

Table 5: Security Objectives for the TOE

Objective	Definition
O.PROTECTED_COMMUNICATIONS	The TOE will provide protected communication channels for administrators, other parts of a distributed TOE, and authorized IT entities.
O.VERIFIABLE_UPDATES	The TOE will provide the capability to help ensure that any updates to the TOE can be verified by the administrator to be unaltered and (optionally) from a trusted source.
O.SYSTEM_MONITORING	The TOE will provide the capability to generate audit data and **store** those **audit** data locally.
O.DISPLAY_BANNER	The TOE will display an advisory warning regarding use of the TOE.
O.TOE_ADMINISTRATION	The TOE will provide mechanisms to ensure that only administrators are able to log in and configure the TOE, and provide protections for logged-in administrators.
O.RESIDUAL_INFORMATION_CLEARING	The TOE will ensure that any data contained in a protected resource is not available when the resource is reallocated.
O.RESOURCE_AVAILABILITY	The TOE shall provide mechanisms that mitigate user attempts to exhaust TOE resources (e.g., persistent storage).
O.SESSION_LOCK	The TOE shall provide mechanisms that mitigate the risk of unattended sessions being hijacked.
O.TSF_SELF_TEST	The TOE will provide the capability to test some subset of its security functionality to ensure it is operating properly.

4.2 Security Objectives for the Operational Environment

This section describes the security objectives that must be fulfilled by the operational environment of the TOE.

Table 6: Security Objectives for the Operational Environment

Objective	Definition
OE.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.

Objective	Definition
OE.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.
OE.TRUSTED_REMOTE_AUTH	The Operational environment will provide mechanisms that control a user's logical access to the TOE and to explicitly deny access to specific users when appropriate.
OE.USER_AUTHENTICATION	The IT environment shall provide support for user identification and authentication when the TOE is configured to use external authentication mechanisims

4.3 Rationale

This section demonstrates that each threat, organizational security policy, and assumption are mitigated by at least one security objective for the TOE, and that those security objectives counter the threats, enforce the policies, and uphold the assumptions.

Table 7: Completeness of Security Objectives

	Objectives													
Assumptions, Threats, Policies	**O.PROTECTED_COMMUNICATIONS**	**O.VERIFIABLE_UPDATES**	**O.SYSTEM_MONITORING**	**O.DISPLAY_BANNER**	**O.TOE_ADMINISTRATION**	**O.RESIDUAL_INFORMATION_CLEARING**	**O.RESOURCE_AVAILABILITY**	**O.SESSION_LOCK**	**O.TSF_SELF_TEST**	**OE.NO_GENERAL_PURPOSE**	**OE.PHYSICAL**	**OE.TRUSTED_ADMIN**	**OE.TRUSTED_REMOTE_AUTH**	**OE.USER_AUTHENTICATION**
A.NO_GENERAL_PURPOSE										X				
A.PHYSICAL											X			
A.TRUSTED_ADMIN												X		
A.ACCESS													X	X
T.ADMIN_ERROR					X									
T.RESOURCE_EXHAUSTION							X							
T.TSF_FAILURE									X					
T.UNDETECTED_ACTIONS			X											
T.UNAUTHORIZED_ACCESS	X			X	X			X						
T.UNAUTHORIZED_UPDATE		X												

	Objectives													
Assumptions, Threats, Policies	O.PROTECTED_COMMUNICATIONS	O.VERIFIABLE_UPDATES	O.SYSTEM_MONITORING	O.DISPLAY_BANNER	O.TOE_ADMINISTRATION	O.RESIDUAL_INFORMATION_CLEARING	O.RESOURCE_AVAILABILITY	O.SESSION_LOCK	O.TSF_SELF_TEST	OE.NO_GENERAL_PURPOSE	OE.PHYSICAL	OE.TRUSTED_ADMIN	OE.TRUSTED_REMOTE_AUTH	OE.USER_AUTHENTICATION
T.USER_DATA_REUSE						X								
P.ACCESS_BANNER				X										

Table 8: Sufficiency of Security Objectives

Assumptions, Threats, Policies	Summary	Objectives and rationale
A.NO_GENERAL_PURPOSE	General purpose capabilities on the TOE.	OE.NO_GENERAL_PURPOSE ensures that the TOE has no general purpose computing capabilities installed or in use on the TOE.
A.PHYSICAL	Physical protection of the TOE.	OE.PHYSICAL ensures that the TOE is protected from physical modification or attack.
A.TRUSTED_ADMIN	Admins are trustworthy.	OE.TRUSTED_ADMIN ensures that TOE administrators are trustworthy and follow all guidance.
A.ACCESS	External authentication entities will be properly configured and operate correctly	OE.TRUSTED_REMOTE_AUTH The Operational environment will provide mechanisms that control a user's logical access to the TOE and to explicitly deny access to specific users when appropriate.
	External authentication entities will be properly configured and operate correctly	OE.USER_AUTHENTICATION The IT environment shall provide support for user identification and authentication when the TOE is configured to use external authentication mechanisims
T.ADMIN_ERROR	An admin could install or configure the TOE incorrectly.	O.TOE_ADMINISTRATION ensures that the TOE only provides administrative capabilities to authenticated TOE Administrators.
T.RESOURCE_EXHAUSTION	A process or user could exhaust TOE resources.	O.RESOURCE_AVAILABILITY ensures that users are not able to exhaust TOE resources.

Assumptions, Threats, Policies	Summary	Objectives and rationale
T.TSF_FAILURE	The TSF may fail.	O.TSF_SELF_TEST ensures that the TOE can test its security functionality to ensure proper operation.
T.UNDETECTED_ACTIONS	User actions may go undetected.	O.SYSTEM_MONITORING ensures that the TOE provides an audit capability that sends event data to an external device so that user actions cannot go undetected.
T.UNAUTHORIZED_ACCESS	Users may gain unauthorized access to the TOE.	O.TOE_ADMINISTRATION ensures that the TOE only provides administrative capabilities to authenticated TOE Administrators.
		O.DISPLAY_BANNER ensures that users are informed of appropriate system usage and warned of trespass penalties before logging in.
		O.PROTECTED_COMMUNICATIONS ensures that the TOE provides protected communication channels for administrators, other parts of a distributed TOE, and authorized IT entities.
		O.SESSION_LOCK ensures that the TOE provides mechanisms that mitigate the risk of unattended sessions being hijacked.
T.UNAUTHORIZED_UPDATE	The product may accept unauthorized updates.	O.VERIFIABLE_UPDATES ensures that the TOE administrator can verify updates are unaltered before they are applied.
T.USER_DATA_REUSE	User data may be exposed.	O.RESIDUAL_INFORMATION_CLEARING ensures that user data cannot be recovered from resources that are reallocated.
P.ACCESS_BANNER	The TOE should warn users before login.	O.DISPLAY_BANNER ensures that users are informed of appropriate system usage and warned of trespass penalties before logging in.

5 Extended Components Definition

The Extended Components Definition describes components for security objectives which cannot be translated or could only be translated with great difficulty to existing requirements. The extended components defined in this section are replicated directly from *Security Requirements for Network Devices, Version 1.0, December 10, 2010*.

5.1 Extended TOE Security Functional Components

This section defines fourteen extended SFRs met by the TOE.

5.1.1 Class FCS: Cryptographic Support

The extended families in this class address the requirements of cryptographic key zeroization, random bit generation, communication protection, TLS and HTTPS.

5.1.1.1 FCS_CKM_EXT

Family Behaviour

The family “FCS_CKM_EXT” defines requirement of zeroization of all plaintext secret and private cryptographic keys and CSPs.

Component leveling

FCS_CKM_EXT.4 defines the requirement of zeroization of all plaintext secret and private cryptographic keys and CSPs. FCS_CKM_EXT.4 is modeled after FCS_CKM.4, and is replicated from *Security Requirements for Network Devices, Version 1.0, dated December 10, 2010*.

5.1.1.1.1 FCS_CKM_EXT.4 Cryptographic Key Zeroization

Management

There are no management activities foreseen.

Audit

Failure on invoking functionality.

Hierarchical to: No other components.

Dependencies: [FDP_ITC.1 Import of user data without security attributes, or

FDP_ITC.2 Import of user data with security attributes, or

FCS_CKM.1 Cryptographic key generation]

FCS_CKM_EXT.4.1 The TSF shall zeroize all plaintext secret and private cryptographic keys and CSPs when no longer required.

5.1.1.2 FCS_RBG_(EXT)

The family “FCS_RBG_(EXT)” defines requirement of random bit generation.

Component leveling

FCS_RBG_(EXT).1 defines the requirement of random bit generation. It is replicated from *Security Requirements for Network Devices, Version 1.0, dated December 10, 2010*.

5.1.1.2.1 FCS_RBG_(EXT).1 Extended: Cryptographic Operation (Random Bit Generation)

Management

There are no management activities foreseen.

Audit

Failure of the randomization process.

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FCS_RBG_(EXT).1.1	The TSF shall perform all random bit generation (RBG) services in accordance with [selection, choose one of: NIST Special Publication 800-90 using [selection: Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES) , Dual_EC_DRBG (any)]; FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using AES] seeded by an entropy source that accumulates entropy from at least one independent TSF-hardware-based noise sources.
FCS_RBG_(EXT).1.2	The deterministic RBG shall be seeded with a minimum of [selection, choose one of: 128 bits, 256 bits] of entropy at least equal to the greatest bit length of the keys and authorization factors that it will generate.

5.1.1.3 FCS_COMM_PROT_EXT

Family Behaviour

The family “FCS_COMM_PROT_EXT” defines requirement of protection of communications.

Component leveling

FCS_COMM_PROT_EXT.1 defines the requirement of protection of communications using IPsec, or SSH, or TLS/HTTPS. This SFR is replicated from *Security Requirements for Network Devices, Version 1.0, dated December 10, 2010*.

5.1.1.3.1 FCS_COMM_PROT_EXT.1 Communications Protection

Management

There are no management activities foreseen.

Audit

There are no auditable events foreseen.

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_COMM_PROT_EXT.1.1 The TSF shall protect communications using [selection: IPsec, SSH] and [selection: TLS/HTTPS, no other protocol].

5.1.1.4 FCS_TLS_EXT

Family Behaviour

The family "FCS_TLS_EXT" defines requirement of communications protection using TLS.

Component leveling

FCS_COMM_PROT_EXT.1 defines the requirement of communication protection using TLS. This SFR is replicated from *Security Requirements for Network Devices, Version 1.0, dated December 10, 2010*.

5.1.1.4.1 FCS_TLS_EXT.1 Explicit: TLS

Management

There are no management activities foreseen.

Audit

Failure to establish a TLS Session, and estaliblishment/termination of a TLS session.

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_TLS_EXT.1.1 The TSF shall implement one or more of the following protocols [selection: TLS 1.0 (RFC 2246), TLS 1.1 (RFC 4346), TLS 1.2 (RFC 5246)] supporting the following ciphersuites:

Mandatory Ciphersuites:

TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA

Optional Ciphersuites:

[selection:
None
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256

TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384].

5.1.1.5 FCS_HTTPS_EXT

Family Behaviour

The family "FCS_HTTPS_EXT" defines requirement of protection of communications using HTTPS.

Component leveling

FCS_HTTPS_EXT.1 defines the requirement of protection of communications using HTTPS. This SFR is replicated from *Security Requirements for Network Devices, Version 1.0, dated December 10, 2010*.

5.1.1.5.1 FCS_HTTPS_EXT.1 Explicit: HTTPS

Management

There are no management activities foreseen.

Audit

Failure to establish a HTTPS Session, and estaliblishment/termination of a HTTPS session.

Hierarchical to: No other components.

Dependencies: No dependencies.

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

FCS_HTTPS_EXT.1.2 The TSF shall implement the HTTPS using TLS as specified in FCS_TLS_EXT.1.

5.1.2 Class FIA: Identification and Authentication

The extended families in this class address the requirements of password management, user identification and authentication, and password-based authentication mechanism. The extended components of "FIA_UIA_EXT.1" and "FIA_UAU_EXT.5" are modeled after FIA_UIA.1, FIA_UAU.1, and FIA_UAU.5. All the extended SFRs in this class are replicated from *Security Requirements for Network Devices, Version 1.0, dated December 10, 2010*.

5.1.2.1 FIA_PMG_EXT

Family Behaviour

The family "FIA_PMG_EXT" defines requirement of password management.

Component leveling

FIA_PMG_EXT.1 defines the requirement of password management.

5.1.2.1.1 FIA_PMG_EXT.1 Password Management

Management

There are no management activities foreseen.

Audit

There are no auditable events foreseen.

Hierarchical to: No other components.

Dependencies: No dependencies.

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

1. *Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and special characters (that include: "!", "@", "#", "$", "%", "^", "&", "*", "(", and ")");*
2. *Minimum password length shall settable by the Security Administrator, and support passwords of 8 characters or greater;*
3. *Passwords composition rules specifying the types and number of required characters that comprise the password shall be settable by the Security Administrator.*
4. *Passwords shall have a maximum lifetime, configurable by the Security Administrator.*
5. *New passwords must contain a minimum of 4 character changes from the previous password.*

5.1.2.2 FIA_UIA_EXT

Family Behaviour

The family "FIA_UIA_EXT" defines requirement of user identification and authentication.

Component leveling

FIA_UIA_EXT.1 defines the requirement of user identification and authentication.

5.1.2.2.1 FIA_UIA_EXT.1 User Identification and Authentication

Management

Configuration of the list of TOE services available before an entity is identified and authenticated.

Audit

All use of the identification and authentication mechanism.

Hierarchical to:	No other components.
Dependencies:	No dependencies.

FIA_UIA_EXT.1.1 The TSF shall allow [selection: [assignment: *list of TOE-provided services*], *no services*] on behalf of the user to be performed before the user is identified and authenticated.

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

5.1.2.3 FIA_UAU_EXT

Family Behaviour

The family “FIA_UAU_EXT” defines requirement of password-based authentication mechanism and other authentication mechanisms.

Component leveling

FIA_UAU_EXT.5 defines the requirement of password-based authentication mechanism and other authentication mechanisms.

5.1.2.3.1 FIA_UAU_EXT.5 Extended: Password-based Authenticaiton Mechanism

Management

There are no management activities foreseen.

Audit

All use of the identification and authentication mechanisms.

Hierarchical to:	No other components.
Dependencies:	No dependencies.

FIA_UAU_EXT.5.1 The TSF shall provide a local password-based authentication mechanism, [selection: [assignment: *other authentication mechanism(s)*], none] to perform user authentication.

FIA_UAU_EXT.5.2 The TSF shall ensure that users with expired passwords are [selection: required to create a new password after correctly entering the expired password, locked out until their password is reset by an administrator].

5.1.3 Class FPT: Protection of the TSF

The extended families in this class address the requirements of management of TSF Data, trusted update and TSF testing. The extended SFRs in this class are replicated from *Security Requirements for Network Devices, Version 1.0, dated December 10, 2010*.

5.1.3.1 FPT_PTD

Family Behaviour

The family “FPT_PTD” defines requirement of management of TSF Data.

Component leveling

FPT_PTD.1 defines the requirement of TSF Data

5.1.3.1.1 FPT_PTD.1(1) Management of TSF Data (for reading of authentication data)

Management

There are no management activities foreseen.

Audit

There are no auditable events foreseen.

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_PTD.1.1(1) Refinement: The TSF shall **prevent** *reading of the plaintext passwords*.

Application Note: The refinement was indicated in the original PP, but the ST author cannot find any indication of the SFR which the refinement operation is based on.

5.1.3.1.2 FPT_PTD.1(2) Management of TSF Data (for reading of all symmetric keys)

Management

There are no management activities foreseen.

Audit

There are no auditable events foreseen.

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_PTD.1.1(1) Refinement: The TSF shall **prevent** *reading of the pre-shared keys, sysmmetric key, and private keys*.

Application Note: *The refinement was indicated in the original PP, but the ST author cannot find any indication of the SFR which the refinement operation is based on.*

5.1.3.2 FPT_TUD _(EXT)

Family Behaviour

The family “FPT_TUD_(EXT)” defines requirement of trusted update.

Component leveling

FPT_TUD_(EXT).1 defines the requirement of trusted update.

5.1.3.2.1 FPT_TUD_(EXT).1 Extended: Trusted Update

Management

There are no management activities foreseen.

Audit

Initiation of update.

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_TUD_(EXT).1.1 The TSF shall provide security administrators the ability to query the current version of the TOE firmware/software.

FPT_TUD_(EXT).1.2 The TSF shall provide security administrators the ability to initiate updates to TOE firmware/software.

FPT_TUD_(EXT).1.3 The TSF shall provide a means to verify firmware/software updates to the TOE using a [selection: digital signature mechanism, published hash] prior to installing those updates.

5.1.3.3 FPT_TST_EXT

Family Behaviour

The family “FPT_TST_EXT” defines requirement of TST testing.

Component leveling

FPT_TST_EXT.1 defines the requirement of TST testing.

5.1.3.3.1 FPT_TST_EXT.1 TSF Testing

Management

There are no management activities foreseen.

Audit

Indication that TSF self-test was completed.

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FPT_TST_EXT.1.1	The TSF shall run a suite of self tests during initial start-up (on power on) to demonstrate the correct operation of the TSF.

5.1.4 Class FTA: TOE Access

The extended families in this class address the requirements of TSF-initiated session locking. The extended SFR in this class is replicated from *Security Requirements for Network Devices, Version 1.0, dated December 10, 2010*.

5.1.4.1 FTA_SSL_EXT

Family Behaviour

The family “FTA_SSL_EXT” defines requirement of TSF-initiated session locking.

Component leveling

FTA_SSL_EXT.1 defines the requirement of TSF-initiated session locking.

5.1.4.1.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

Management

There are no management activities foreseen.

Audit

Any attempts at unlocking of any interactive session.

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 Timing of authentication.

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions, [selection:

- lock the session – disable any activity of the user’s data access/display devices other than unlocking the session, and requiring that the administrator re-authenticate to the TSF prior to unlocking the session;
- terminate the session]

after a Security Administrator-specified time period of inactivity.

5.2 Rationale for Extended Security Functional Requirements

The TOE contains the following explicitly stated security functional requirements:

- FCS_CKM_EXT.4
- FCS_RBG_(EXT).1
- FCS_COMM_PROT_EXT.1
- FCS_TLS_EXT.1
- FCS_HTTPS_EXT.1
- FIA_PMG_EXT.1
- FIA_UIA_EXT.1
- FIA_UAU_EXT.5
- FPT_PTD.1
- FPT_TUD_(EXT).1
- FPT_TST_EXT.1
- FTA_SSL_EXT.1

The FCS_CKM_EXT.4 is explicitly stated because the TOE is required to support special key destruction method, i.e. key zeroization. The existing SFR of FCS_CKM.4 of CC Part 2 cannot meet this requirement.

FCS_RBG_(EXT).1 is explicitly stated because the TOE is required to support special random bit generation methods. The existing SFR in FCS class of CC Part 2 cannot meet this requirement.

FCS_COMM_PROT_EXT.1, FCS_TLS_EXT.1 and FCS_HTTPS_EXT.1 are explicitly stated because the TOE is required to support communications protection using IPsec, or SSH, or TLS/HTTPS. The existing SFR in FCS class and FDP class of CC Part cannot meet the requirements. Meanwhile, the protection is provided mostly by cryptographic functionality. Hence, the extended components are categorized as part of FCS class.

FIA_PMG_EXT.1, FIA_UIA_EXT.1, and FIA_UAU_EXT.5 are explicitly stated because the TOE is required to support special password-based authentication mechanism(s). The existing SFRs in FIA class of CC Part 2 cannot meet the requirements.

FPT_PTD.1, FPT_TUD_(EXT).1, and FPT_TST_EXT.1 are explicitly stated because the TOE is required to support special methods of protection of TSF data, trusted update, and TSF testing. The existing SFRs in FPT class of CC Part 2 cannot meet the requirements.

FTA_SSL_EXT.1 is explicitly stated because the TOE is required to support special methods of TSF-initiated session locking. The existing SFR in FTA class of CC Part 2 cannot meet this requirement.

6 Security Requirements

This section defines the IT security requirements that shall be satisfied by the TOE or its environment. The CC divides TOE security requirements into two categories:

- Security functional requirements (SFRs) (such as, identification and authentication, security management, and user data protection) that the TOE and the supporting evidence need to satisfy to meet the security objectives of the TOE.

- Security assurance requirements (SARs) that provide grounds for confidence that the TOE and its supporting IT environment meet its security objectives (e.g., configuration management, testing, and vulnerability assessment).

These requirements are discussed separately within the following subsections.

6.1 Conventions

All operations performed on the Security Functional Requirements or the Security Assurance Requirements need to be identified. For this purpose the following conventions shall be used.

- Assignments will be written in *italic text*.
- Selections will be written in underlined text.
- Refinements will be written **bold** and/or ~~strikethrough~~ text.
- Assignment within a Selection: Indicated with *italicized and underlined text.*
- Iteration: Indicated by appending the iteration number in parenthesis, e.g., (1), (2), (3).
- Explicitly stated SFRs are identified by having a label 'EXT' after the requirement name for TOE SFRs.

6.2 Security Functional Requirements for the TOE

The TOE satisfies all the SFRs in the *Security Requirements for Network Devices* PP except for FAU_STG_EXT.1 and FAU_STG_EXT.3 which are replaced with FAU_STG.1 and FAU_STG.3 as the TOE is capable of storing the audit log locally.

Note that the *Security Requirements for Network Devices* PP contains instructions to ST authors for writing an ST, and instructions to evaluators for the purposes of testing, interspersed among the SFRs in the Security Requirements section. For easy reference, the additional assurance activities in the PP have been duplicated in the ST.

The SFRs satisified by the TOE are delineated in the table below. The rest of this section contains a description of each component and any related dependencies.

Table 9: TOE Security Functional Requirements and Auditable Security Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	
FAU_GEN.2	None.	
FAU_STG.1	None.	
FAU_STG.3	Actions taken due to exceeding of a threshold	No additional information.
FCS_CKM.1	Failure on invoking functionality.	No additional information.
FCS_CKM_EXT.4	Failure on invoking functionality.	No additional information.
FCS_COP.1(1)	Failure on invoking functionality.	No additional information.
FCS_COP.1(2)	Failure on invoking functionality.	No additional information.

Requirement	Auditable Events	Additional Audit Record Contents
FCS_COP.1(3)	Failure on invoking functionality.	No additional information.
FCS_COP.1(4)	Failure on invoking functionality.	No additional information.
FCS_COMM_PROT_EXT.1	None.	
FCS_TLS_EXT.1	Failure to establish a TLS Session Establishment/Termination of a TLS session	Reason for failure Non-TOE endpoint of connection (IP address) for both successes and failures.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session Establishment/Termination of a HTTPS session	Reason for failure Non-TOE endpoint of connection (IP address) for both successes and failures.
FCS_RBG_EXT.1	Failure of the randomization process.	No additional information.
FDP_RIP.2	None	
FIA_PMG_EXT.1	None.	
FIA_UIA_EXT.1	All use of the identification and authentication mechanism.	Provided user identity, origin of the attempt (e.g., IP address)
FIA_UAU_EXT.5	All use of the authentication mechanism	Origin of the attempt (e.g., IP address)
FIA_UAU.6	Attempt to re-authenticate	Origin of the attempt (e.g., IP address)
FIA_UAU.7	None.	
FMT_MTD.1	None.	
FMT_SMR.1	None.	
FPT_ITT.1(1)	None.	
FPT_ITT.1(2)	None.	
FPT_PTD.1(1)	None.	
FPT_PTD.1(2)	None.	
FPT_RPL.1	Detected replay attacks.	Origin of the attempt (e.g., IP address)
FPT_STM.1	Changes to the time.	The old and new values for the time. Origin of the attempt (e.g., IP address)
FPT_TST_EXT.1	Indication that TSF self-test was completed.	Any additional information generated by the tests beyond “success” or “failure”.
FPT_TUD_EXT.1	Initiation of update	No additional information.
FRU_RSA.1	Maximum quota being exceeded.	Resource identifier.
FTA_SSL_EXT.1	Any attempts at unlocking of an interactive session	No additional information.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	No additional information.
FTA_TAB.1	None.	
FTP_ITC.1(1)	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_ITC.1(2)	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_ITC.1(3)	Failure of verification of the signed DNS messages	Identification of the initiator and target of failed attempt of verification

Requirement	Auditable Events	Additional Audit Record Contents
FTP_TRP.1(1)	Initiation of the trusted channel. Termination of the trusted channel. Failures of the trusted path functions.	Identification of the claimed user identity.
FTP_TRP.1(2)	Initiation of the trusted channel. Termination of the trusted channel. Failures of the trusted path functions.	Identification of the claimed user identity.

6.2.1 Security Audit (FAU)

6.2.1.1 FAU_GEN.1 Audit data generation

Hierarchical to: No other components.

Dependencies: FPT_STM.1 Reliable time stamps

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

a) Start-up and shutdown of the audit functions;
b) All auditable events for the basic level of audit; and
c) All administrative actions;
d) *Specifically defined auditable events listed in Table 9*.

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

a) Date and time of the event, type of event, subject identity (if applicable),and the outcome (success or failure) of the event; and
b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, *information specified in column three of the Table 9*.

Assurance Activity:

The evaluator shall check the administrative guide and ensure that it lists all of the auditable events and provides a format for audit records. Each audit record format type must be covered, along with a brief description of each field. The evaluator shall check to make sure that every audit event type mandated by the PP is described and that the description of the fields contains the information required in FAU_GEN1.2, and the additional information specified in Table 9.

The evaluator shall also make a determination of the administrative actions that are relevant in the context of this PP. The evaluator shall examine the administrative guide 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 PP. The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are security relevant with respect to this PP. The evaluator may perform this activity as part of the activities associated with ensuring the AGD_OPE guidance satisfies the requirements.

The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records for the following events: the establishment and termination of channels, detection of a replay attack, and administrative actions. The evaluator shall test that the establishment and termination of a channel is performed for each of the cryptographic protocols contained in the PP (i.e., IPsec, SSH, TLS, HTTPS). 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 administrative guide, and that the fields in each audit record have the proper entries.*

Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly. For example, testing to ensure the TOE can detect replay attempts will more than likely be done to demonstrate that requirement FPT_RPL.1 is satisfied. Another example is that testing performed to ensure that the administrative guidance provided is correct verifies that AGD_OPE.1 is satisfied and should address the invocation of the administrative actions that are needed to verify the audit records are generated as expected.

6.2.1.2 FAU_GEN.2 User identity association

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation

FIA_UID.1 Timing of identification

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

6.2.1.3 FAU_STG.1 Protected audit trail storage

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation

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

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

6.2.1.4 FAU_STG.3 Action in case of possible audit data loss

Hierarchical to: No other components.

Dependencies: FAU_STG.1 Protected audit trail storage

FAU_STG.3.1 The TSF shall *delete the oldest audit file* if the audit trail exceeds *maximum of 10 audit log files*.

6.2.2 Cryptographic Support (FCS)

6.2.2.1 FCS_CKM.1 Cryptographic Key Generation (for asymmetric keys)

Hierarchical to: No other components.

Dependencies: FCS_COP.1 Cryptographic operation

FCS_CKM_EXT.4 Cryptographic key zeroization

FCS_CKM.1.1 **Refinement:** The TSF shall generate **asymmetric** cryptographic keys **in accordance with a domain parameter generator and a random number generator and a prime number generator** that meet the following:

a) All cases: (i.e., any of the above)

- **ANSI X9.80 (3 January 2000), "Prime Number Generation, Primality Testing, and Primality Certificates" using random integers with deterministic tests, or constructive generation methods**
- **Generated key strength shall be equivalent to, or greater than, a symmetric key strength of 112 bits using conservative estimates.**

Application Note:

The generated key strength of 2048-bit DSA and rDSA keys need to be equivalent to, or greater than, a symmetric key strength of 112 bits. See NIST Special Publication 800-57, "Recommendation for Key Management" for information about equivalent key strengths.

b) Case: For domain parameters used in finite field-based key establishment schemes1

- **NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"**

c) Case: For domain parameters used in RSA-based key establishment schemes

- **NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography"**

Application Note: Although ANSI X9.31 is a standard intended for digital signatures, it is being used here for its coverage of the generation of RSA parameters.

Assurance Activity:

The evaluator shall use the domain parameter generation and key pair generation portions of "The FIPS 186-3 Digital Signature Algorithm Validation System (DSAVS)", "The FIPS 186-3 Elliptic Curve Digital Signature Algorithm Validation System (ECDSA2VS)", and "The RSA Validation System (RSAVS)" as a guide in testing the requirement above. This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test.

6.2.2.2 FCS_CKM_EXT.4 Cryptographic Key Zeroization

Hierarchical to: No other components.

[1] For example, "classic" Diffie-Hellman-based scheme.

Dependencies: FDP_ITC.1 Import of user data without security attributes, or

FDP_ITC.2 Import of user data with security attributes, or

FCS_CKM.1 Cryptographic key generation

FCS_CKM_EXT.4.1 The TSF shall zeroize all plaintext secret and private cryptographic keys and CSPs when no longer required.

Application Note: "Cryptographic Critical Security Parameters" are defined in FIPS 140-2 as "security-related information (e.g., secret and private cryptographic keys, and authentication data such as passwords and PINs) whose disclosure or modification can compromise the security of a cryptographic module."

The zeroization indicated above applies to each intermediate storage area for plaintext key/cryptographic critical security parameter (i.e., any storage, such as memory buffers, that is included in the path of such data) upon the transfer of the key/cryptographic critical security parameter to another location.

Assurance Activity:

The evaluator shall check to ensure the TSS describes each of the secret keys (keys used for symmetric encryption), private keys, and CSPs used to generate key; when they are zeroized (for example, immediately after use, on system shutdown, etc.); and the type of zeroization procedure that is performed (overwrite with zeros, overwrite three times with random pattern, etc.). If different types of memory are used to store the materials to be protected, the evaluator shall check to ensure that the TSS describes the zeroization procedure in terms of the memory in which the data are stored (for example, "secret keys stored on flash are zeroized by overwriting once with zeros, while secret keys stored on the internal hard drive are zeroized by overwriting three times with a random pattern that is changed before each write").

6.2.2.3 FCS_COP.1(1) Cryptographic Operation (for data encryption/decryption)

Hierarchical to: No other components.

Dependencies: FCS_CKM.1 Cryptographic key generation

FCS_CKM_EXT.4 Cryptographic key zeroization

FCS_COP.1.1(1) **Refinement:** The TSF shall perform *encryption and decryption* in accordance with a specified cryptographic algorithm *AES operating in **CBC mode*** and cryptographic key sizes *128-bits, 256-bits, and* ***192-bits*** that meets the following:

- FIPS PUB 197, "Advanced Encryption Standard (AES)"
- NIST SP 800-38B

Assurance Activity:

The evaluator shall use tests appropriate to the modes selected in the above requirement from "The Advanced Encryption Standard Algorithm Validation Suite (AESAVS)", "The XTS-AES Validation System (XTSVS)", The CMAC Validation System (CMACVS)", "The Counter with Cipher Block Chaining-Message Authentication Code (CCM) Validation System (CCMVS)", and "The Galois/Counter Mode (GCM) and GMAC Validation System (GCMVS)" (these documents *are available from http://csrc.nist.gov/groups/STM/cavp/index.html) as a guide in testing the requirement above. This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test.*

6.2.2.4 FCS_COP.1(2) Cryptographic Operation (for cryptographic signature)

Hierarchical to: No other components.

Dependencies: FCS_CKM.1 Cryptographic key generation

FCS_CKM_EXT.4 Cryptographic key zeroization

FCS_COP.1.1(2) **Refinement:** The TSF shall perform *cryptographic signature services* in accordance with***:***

(2) RSA Digital Signature Algorithm (rDSA) with a key size (modulus) of 2048 bits or greater

that meets the following:

Case: RSA Digital Signature Algorithm

- FIPS PUB 186-3, “Digital Signature Standard”

Assurance Activity:

The evaluator shall use the signature generation and signature verification portions of "The Digital Signature Algorithm Validation System” (DSAVS or DSA2VS), "The Elliptic Curve Digital Signature Algorithm Validation System” (ECDSAVS or ECDSA2VS), and "The RSA Validation System” (RSAVS) as a guide in testing the requirement above. The Validation System used shall comply with the conformance standard identified in the ST (i.e. FIPS PUB 186-2 or FIPS PUB 186-3). This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test.

6.2.2.5 FCS_COP.1(3) Cryptographic Operation (for cryptographic hashing)

Hierarchical to: No other components.

Dependencies: FCS_CKM.1 Cryptographic key generation

FCS_CKM_EXT.4 Cryptographic key zeroization

FCS_COP.1.1(3) **Refinement:** The TSF shall perform *cryptographic hashing services* in accordance with a specified cryptographic algorithm *SHA-1, SHA-256, SHA-384, and SHA-512* **and message digest sizes** *160, 256, 384, and 512 bits* that meet the following: *FIPS Pub 180-2, “Secure Hash Standard.”*

Assurance Activity:

The evaluator shall use "The Secure Hash Algorithm Validation System (SHAVS)" as a guide in testing the requirement above. This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test.

6.2.2.6 FCS_COP.1(4) Cryptographic Operation (for keyed-hash message authentication)

Hierarchical to: No other components.

Dependencies:	FCS_CKM.1 Cryptographic key generation FCS_CKM_EXT.4 Cryptographic key zeroization
FCS_COP.1.1(4)	**Refinement:** The TSF shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm **HMAC-*SHA-1, SHA-256, SHA-384, SHA-512***, **key size *128, 256 and 512 bits***, **and message digest sizes** ***160, 256, 384, 512 bits*** that meet the following: *FIPS Pub 198, "The Keyed-Hash Message Authentication Code", FIPS Pub 180-2, "Secure Hash Standard."*

Assurance Activity:

The evaluator shall use "The Keyed-Hash Message Authentication Code (HMAC) Validation System (HMACVS)" as a guide in testing the requirement above. This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test.

6.2.2.7 Extended: Cryptographic Operation (Random Bit Generation) (FCS_RBG_(EXT))

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FCS_RBG_(EXT).1.1	The TSF shall perform all random bit generation (RBG) services in accordance with FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using AES seeded by an entropy source that accumulates entropy from at least one independent TSF-hardware-based noise sources.
FCS_RBG_(EXT).1.2	The deterministic RBG shall be seeded with a minimum 256 bits of entropy at least equal to the greatest bit length of the keys and authorization factors that it will generate.

Assurance Activity:

The evaluator shall review the TSS section to determine the version number of the product containing the RBG(s) used in the TOE. The evaluator shall also confirm that the TSS describes the hardware-based noise source from which entropy is gathered, and confirm the location of this noise source. The evaluator will further verify that all of the underlying functions and parameters used in the RBG are listed in the TSS.

The evaluator shall verify that the TSS contains a description of the RBG model, including the method for obtaining entropy input, as well as identifying the entropy source(s) used and how much entropy is produced by each entropy source. The evaluator shall also ensure that the TSS describes known modes of entropy source failure. Finally, the evaluator shall ensure that the TSS contains a description of the RBG outputs in terms of the independence of the output and variance with time and/or environmental conditions.

The evaluator shall also perform the following tests, depending on the standard to which the RBG conforms.

Implementations Conforming to FIPS 140-2, Annex C

The reference for the tests contained in this section is The Random Number Generator Validation System (RNGVS) [RNGVS]. The evaluator shall conduct the following two tests. Note that the "expected values" are produced by a reference implementation of the algorithm that is known to be correct. Proof of correctness is left to each Scheme.

The evaluator shall perform a Variable Seed Test. The evaluator shall provide a set of 128 (Seed, DT) pairs to the TSF RBG function, each 128 bits. The evaluator shall also provide a key (of the length appropriate to the AES algorithm) that is constant for all 128 (Seed, DT) pairs. The DT value is incremented by 1 for each set. The seed values shall have no repeats within the set. The evaluator ensures that the values returned by the TSF match the expected values.

The evaluator shall perform a Monte Carlo Test. For this test, they supply an initial Seed and DT value to the TSF RBG function; each of these is 128 bits. The evaluator shall also provide a key (of the length appropriate to the AES algorithm) that is constant throughout the test. The evaluator then invokes the TSF RBG 10,000 times, with the DT value being incremented by 1 on each iteration, and the new seed for the subsequent iteration produced as specified in NIST-Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithms, Section 3. The evaluator ensures that the 10,000th value produced matches the expected value.

Implementations Conforming to NIST Special Publication 800-90

The evaluator shall perform 15 trials for the RBG implementation. If the RBG is configurable, the evaluator shall perform 15 trials for each configuration. The evaluator shall also confirm that the operational guidance contains appropriate instructions for configuring the RBG functionality.

If the RBG 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 SP 800-90).

If the RBG 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.

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 df 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.

6.2.2.8 FCS_COMM_PROT_EXT.1 Communications Protection

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_COMM_PROT_EXT.1.1 The TSF shall protect communications using **none,**~~ and~~ TLS/HTTPS, **TLS VPN, DNSSEC, TSIG and GSS-TSIG**.

Application Note: TLS/HTTPS is used for web GUI access, TLS VPN is used for grid communication between TOE instances, DNSSEC, TSIG and GSS-TSIG are used for DNS traffic.

6.2.2.9 FCS_TLS_EXT.1 Explicit: TLS

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_TLS_EXT.1.1 The TSF shall implement the TLS 1.0 (RFC 2246) supporting the following ciphersuites:

Mandatory Ciphersuites:

TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA

Optional Ciphersuites:

None.

Assurance Activity:

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that optional characteristics (e.g., extensions supported, client authentication supported) are specified, and the ciphersuites supported are specified as well. The evaluator shall check the TSS to ensure that the ciphersuites specified are identical to those listed for this component. The evaluator shall also check the operational guidance 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). The evaluator shall also perform the following test:

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 a HTTPS session. It is sufficient to observe (on the wire) 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 in an attempt to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

6.2.2.10 FCS_HTTPS_EXT.1 Explicit: HTTPS

Hierarchical to: No other components.

Dependencies: No dependencies.

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

FCS_HTTPS_EXT.1.2 The TSF shall implement the HTTPS using TLS as specified in FCS_TLS_EXT.1.

Assurance Activity: The evaluator shall check the TSS to ensure that it is clear on how HTTPS uses TLS to establish an administrative session, focusing on any client authentication required by the TLS protocol vs. security administrator authentication which may be done at a different level of the processing stack. Testing for this activity is done as part of the TLS testing; this may result in additional testing if the TLS tests are done at the TLS protocol level.

6.2.3 User Data Protection (FDP)

6.2.3.1 FDP_RIP.2 Full residual information protection

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_RIP.2.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the allocation of the resource to all objects.

Assurance Activity:

"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.

6.2.4 Identification and Authentication (FIA)

6.2.4.1 FIA_PMG_EXT.1 Password Management

Hierarchical to: No other components.

Dependencies: No dependencies.

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

1. *Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and special characters (that include: "!", "@", "#", "$", "%", "^", "&", "*", "(", and ")");*

2. *Minimum password length shall settable by the Security Administrator, and support passwords of 8 characters or greater;*

3. *Passwords composition rules specifying the types and number of required characters that comprise the password shall be settable by the Security Administrator.*

Application Note: The intent of this caveat is that the Security Administrator is able to specify, for example, that passwords contain at least 1 upper case letter, 1 lower case letter, 1 numeric character, and 1 special character, and the TOE enforces this restriction. "Types" refers to all of the types listed in item 1 in this element.

4. *Passwords shall have a maximum lifetime, configurable by the Security Administrator.*

5. *New passwords must contain a minimum of 4 character changes from the previous password.*

Application Note: Note that it is not necessary to store a plaintext version of the password in order to determine that at least 4 characters have changed, since FIA_UAU.6 requires re-authentication when changing the password.

"Administrative passwords" refers to passwords used by administrators at the local console or over protocols that support passwords, such as SSH and HTTPS.

Assurance Activity:

The evaluator shall examine the operational guidance to determine that it provides guidance to security administrators on the composition of strong passwords, and that it provides instructions on setting the minimum password length; the formulation and specification of password composition rules and how to configure these for the TOE; and how to configure the maximum lifetime for a password. The evaluator shall also perform the following tests. Note that one or more of these tests can be performed with a single test case.

Test 1: The evaluator shall configure the TOE with different password composition rules, as specified in the requirement. The evaluator shall then, for each set of rules, compose passwords that either meet the requirements, or fail to meet the requirements, in some way. For each password, the evaluator shall verify that the composition rules are enforced. While the evaluator is not required (nor is it feasible) to test all possible composition rules, the evaluator shall ensure that all characters, rule characteristics, and a minimum length listed in the requirement are supported, and justify the subset of those characters chosen for testing.

Test 2: The evaluator shall ensure that the operational guidance contains instructions on setting the maximum password lifetime. The evaluator shall then configure this lifetime to several values, and ensure that it is enforced for each of those values.

Test 3: The evaluator shall test that a minimum of 4 character changes from previous passwords is enforced. This shall be done for more than one password.

6.2.4.2 FIA_UIA_EXT.1 User identification and authentication

Hierarchical to:	No other components.
Dependencies:	No dependencies.

FIA_UIA_EXT.1.1 The TSF shall allow no services on behalf of the user to be performed before the user is identified and authenticated.

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

Application Note: This requirement applies to users (administrators) of services available from the TOE directly, and not services available by connecting through the TOE. Authentication can be password-based through the local console or through a protocol that supports passwords (such as SSH), or be certificate based (SSH, TLS).

6.2.4.3 FIA_UAU_EXT.5 Extended: Password-based Authentication Mechanism

Hierarchical to: No other components.

Dependencies: No dependencies.

FIA_UAU_EXT.5.1 The TSF shall provide a local password-based authentication mechanism, *and authentication against Active Directory server* to perform user authentication.

FIA_UAU_EXT.5.2 The TSF shall ensure that users with expired passwords are required to create a new password after correctly entering the expired password.

6.2.4.4 FIA_UAU.6 Re-authenticating

Hierarchical to: No other components.

Dependencies: No dependencies.

FIA_UAU.6.1 The TSF shall re-authenticate the user under the conditions: *when the user changes their password, following TSF-initiated locking (FTA_SSL).*

Assurance Activity:

The evaluator shall perform the following test:

> *Test 1: The evaluator shall attempt to change their password as directed by the operational guidance. While making this attempt, the evaluator shall verify that re-authentication is required.*

6.2.4.5 FIA_UAU.7 Protected Authentication Feedback

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 Timing of authentication

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

Application Note: "Obscured feedback" implies the TSF does not produce a visible display of any authentication data entered by a user (such as the echoing of a password), although an obscured indication of progress may be provided (such as an asterisk for each character). It also implies that the TSF does not return any information during the authentication process to the user that may provide any indication of the authentication data.

6.2.5 Security Management (FMT)

6.2.5.1 FMT_MTD.1 Management of TSF Data (for general TSF data)

Hierarchical to: No other components.

Dependencies: FMT_SMR.1 Security roles

FMT_SMF.1 Specification of Management Functions

FMT_MTD.1.1 The TSF shall restrict the ability to <u>*manage*</u> the *TSF data* ~~to the *Security Administrators*~~ ***specified in the following table to authorised roles with the privilege specified in the following table.***

TSF Data	Operation	Role
All TSF data	Manage	Superuser
User password	Change	All roles: a user can change their own password
Users, groups, roles and permissions	Manage	Superuser
Superuser defined	Superuser defined	Limited-Access
Limited-access permissions inherited at upgrade from previous NIOS version	Limited-access operations inherited at upgrade from previous NIOS version	Default

Application Note: The word "manage" includes but is not limited to create, initialize, view, change default, modify, delete, clear, and append. This requirement is intended to be the "default" requirement for management of TSF data; other iterations of FMT_MTD should place different restrictions or operations available on the specifically-identified TSF data. TSF data includes cryptographic information as well; managing these data would include the association of a cryptographic protocol with an interface, for instance.

*Application Note: **Superusers** have access to all TSF data. **Limited-Access users** have Superuser defined access to Superuser defined sets of TSF data (there can be multipl Limited-Access groups). **Default users** are Limited-Access users imported at upgrade with previously defined permissions - included for completeness however, the evaluated configuration requires a fresh install and therefore the Default role will not be assigned).*

6.2.5.2 FMT_SMF.1 Specification of Management Functions

Hierarchical to: No other components.

Dependencies: No dependencies.

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

- <u>*Ability to configure the cryptographic functionality.*</u>

- *Ability to update the TOE, and to verify the updates using the digital signature capability (FCS_COP.1(2)) and [selection: no other functions]*
- *Ability to configure the security functions of the TOE*

Application Note: At a minimum the TOE must provide the functionality for the administrator to verify that the update received came from a trusted source; this is done using digital signatures. If other mechanisms are used, those should be specified in the assignment; otherwise "no other functions" should be selected. If the other mechanisms used are cryptographic, then the ST author should ensure they are specified using FCS components, and that those components are referenced in the assignment.

6.2.5.3 FMT_SMR.1 Security Roles

Hierarchical to: No other components.

Dependencies: No dependencies.

FMT_SMR.1.1 The TSF shall maintain the roles:

- *~~Security Administrator~~**Superuser***,
- *Limited-Access*
- *Default*

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

*Application Note: The TOE implements a groups & roles access control system. The above roles equate to groups on the TOE. The **Superuser** group and role are synonomous. There may be multiple **Limited-Access** groups assigned to multiple Superuser defined sets of permissions which are labelled roles. The **Default** group and role are synonomous. This role is included for completeness however, the evaluated configuration requires a fresh install and therefore the Default role will not be assigned).*

6.2.6 Protection of the TSF (FPT)

6.2.6.1 FPT_ITT.1(1) Basic Internal TSF Data Transfer Protection (Disclosure)

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_ITT.1.1(1) **Refinement:** The TSF shall protect TSF data from disclosure when it is transmitted between separate parts of the TOE **through the use of the TSF-provided cryptographic services: *FCS_TLS_EXT and FCS_HTTPS_EXT***.

6.2.6.2 FPT_ITT.1(2) Basic Internal TSF Data Transfer Protection (Modification)

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_ITT.1.1(2) **Refinement:** The TSF shall **detect modification of TSF data** when it is transmitted between separate parts of the TOE **through the use of the TSF-provided cryptographic services: *FCS_TLS_EXT and FCS_HTTPS_EXT***.

6.2.6.3 FPT_PTD.1(1) Management of TSF Data (for reading of authentication data)[2]

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_PTD.1.1(1) The TSF shall **prevent** *reading of* the *plaintext passwords.*

Application Note: The intent of the requirement is that no user or administrator be able to read the authentication data used to directly authenticate a user to the TSF (such as an unencrypted password) through "normal" interfaces if the reading of such data could lead to someone impersonating that user. An all-powerful administrator of course could directly read memory to capture a password but is trusted not to do so. Likewise, if a system relies on a public key for a user as part of the authentication process, that key could be considered "authentication data" but being able to read that key would not lead to a compromise of that user, and so would not fall under the purview of this requirement.

Assurance Activity:

The evaluator shall examine the TSS to determine that it details how any plaintext 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. If passwords are not stored in plaintext, the TSS shall describe how the passwords are protected.

6.2.6.4 FPT_PTD.1(2) Management of TSF Data (for reading of all symmetric keys)

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_PTD.1.1(2) The TSF shall **prevent** *reading of all pre-shared keys, symmetric key, and private keys.*

Application Note: The intent of the requirement is that no user or administrator be able to read or view the identified keys (stored or ephemeral) through "normal" interfaces. While the security administrator of course could directly read memory to view these keys, they are trusted not to do so.

Assurance Activity:

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.

6.2.6.5 FPT_RPL.1 Replay detection

Hierarchical to: No other components.

Dependencies: No dependencies.

[2] This is an extended requirement, but has not been marked as one in the PP.

FPT_RPL.1.1 The TSF shall detect replay for the following entities: *network packets terminated at the TOE*.

FPT_RPL.1.2 The TSF shall perform: *reject the data* when replay is detected.

Application Note: The intent of the first element is that communications of a trusted nature (administrator to TOE, IT entity to TOE, TOE to TOE) are covered by the element and not subject to replay attacks.

6.2.6.6 FPT_STM.1 Reliable time stamps

Hierarchical to: No other components.

Dependencies: No dependencies.

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

6.2.6.7 Extended: Trusted Update (FPT_TUD_(EXT).1)

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_TUD_(EXT).1.1 The TSF shall provide security administrators the ability to query the current version of the TOE firmware/software.

FPT_TUD_(EXT).1.2 The TSF shall provide security administrators the ability to initiate updates to TOE firmware/software.

FPT_TUD_(EXT).1.3 The TSF shall provide a means to verify firmware/software updates to the TOE using digital signature mechanism prior to installing those updates.

Application Note: The digital signature mechanism referenced in the third element is the one specified in FCS_COP.1(2).

Assurance Activity:

Updates to the TOE either have a hash associated with them, or are signed by an authorized source. If digital signatures are used, the definition of an authorized source is contained in the TSS, along with a description of how the certificates used by the update verification mechanism are contained on the device. The evaluator ensures this information is contained in the TSS. The evaluator also ensures that the TSS (or the operational guidance) describes how the candidate updates are obtained; the processing associated with verifying the digital signature or calculating the hash of the updates; and the actions that take place for successful (hash or signature was verified) and unsuccessful (hash or signature could not be verified) cases. The evaluator shall perform the following tests:

> *Test 1: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains a legitimate update using procedures described in the operational guidance and verifies that it is successfully installed on the TOE. Then, the evaluator performs a subset of other assurance activity tests to demonstrate that the update functions as expected. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update.*

> *Test 2: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains or produces an illegitimate update, and attempts to install it on the TOE. The evaluator verifies that the TOE rejects the update.*

6.2.6.8 FPT_TST_EXT.1: TSF Testing

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_TST_EXT.1.1 The TSF shall run a suite of self tests during initial start-up (on power on) to demonstrate the correct operation of the TSF.

Assurance Activity:

The evaluator shall examine the TSS to ensure that it details the self tests that are run by the TSF on start-up; 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.

6.2.7 Resource Utilization (FRU)

6.2.7.1 FRU_RSA.1 Maximum quotas

Hierarchical to: No other components.

Dependencies: No dependencies.

FRU_RSA.1.1 The TSF shall enforce maximum quotas of the following resources: log files, as defined for FAU_STG.3.1, space occupied by files uploaded for file distribution, size of uploaded files for any purpose, and number of administrator GUI and API sessions that subjects can use simultaneously.

6.2.8 TOE Access (FTA)

6.2.8.1 FTA_SSL_EXT.1 TSF-initiated session locking

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 Timing of authentication

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions,

- <u>terminate the session</u>

after a Security Administrator-specified time period of inactivity.

Assurance Activity:

The evaluator shall perform the following test:

> *Test 1: The evaluator follows the operational guidance 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.

6.2.8.2 FTA_SSL.3 TSF-initiated termination

Hierarchical to: No other components.

Dependencies: No dependencies.

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

Assurance Activity:

The evaluator shall perform the following test:

> *Test 1: The evaluator follows the operational guidance 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.*

6.2.8.3 FTA_TAB.1 Default TOE access banners

Hierarchical to: No other components.

Dependencies: No dependencies.

FTA_TAB.1.1 **Refinement:** Before establishing **a user/administrator** session the TSF shall display **a Security Administrator-specified** advisory **notice and consent** warning message regarding unauthorized use of the TOE.

Application Note: This requirement is intended to apply to interactive sessions between a human user and a TOE. IT entities establishing connections or programmatic connections (e.g., remote procedure calls over a network) are not required to be covered by this requirement.

Assurance Activity:

The evaluator shall check the TSS to ensure that it details each method of access (local and remote) available to the administrator (e.g., serial port, SSH, HTTPS). The evaluator shall also perform the following test:

> *Test 1: The evaluator follows the operational guidance 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.*

6.2.9 Trusted Path/Channels (FTP)

6.2.9.1 FTP_ITC.1(1) Inter-TSF trusted channel (Prevention of Disclosure)

Hierarchical to: No other components.

Dependencies: No dependencies.

FTP_ITC.1.1(1) **Refinement:** The TSF shall **use *FCS_TLS_EXT and FCS_HTTPS_EXT* to** provide a **trusted** communication channel between itself and **authorized IT entities** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure.

FTP_ITC.1.2(1) **Refinement:** The TSF shall permit *the TSF*, ***or the authorized IT entities*** to initiate communication via the trusted channel.

FTP_ITC.1.3(1) The TSF shall initiate communication via the trusted channel for *all authentication functions*, *TLS communications between the TOE and other TOE instances*.

6.2.9.2 FTP_ITC.1(2) Inter-TSF trusted channel (Detection of Modification)

Hierarchical to: No other components.

Dependencies: No dependencies.

FTP_ITC.1.1(2) **Refinement:** The TSF shall **use *FCS_TLS_EXT and FCS_HTTPS_EXT* in providing** a **trusted** communication channel between itself and **authorized IT entities** that is logically distinct from other communication channels and provides assured identification of its end points and **detection of modification of data**.

FTP_ITC.1.2(2) **Refinement:** The TSF shall permit *the TSF*, ***or the authorized IT entities*** to initiate communication via the trusted channel.

FTP_ITC.1.3(2) The TSF shall initiate communication via the trusted channel for *all authentication functions*, *TLS communications between the TOE and other TOE instances*.

6.2.9.3 FTP_ITC.1(3) Inter-TSF trusted channel (DNS)

Hierarchical to: No other components.

Dependencies: No dependencies.

FTP_ITC.1.1(3) The TSF shall provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification **and masquerading data origin as described below:**~~or disclosure~~.

Operation	Protocol	Standards	Algorithms
Validate received DNS server responses and sign DNS responses sent by the TOE	DNSSEC	RFC 4033 RFC 4034 RFC 4035	RSA/SHA-1 RSASHA1-NSEC3-SHA1 RSA/SHA-256 RSA/SHA-512
Secret key transaction authentication for DNS	TSIG	RFC 2845	HMAC-SHA256

Secret key transaction authentication for DNS (Microsoft environment)	GSS-TSIG	RFC 3645	AES128_CTS_HMAC_SHA1_96 AES256_CTS_HMAC_SHA1_96

Application Note: FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FCS_CKM.1, FCS_CKM_EXT.4, FCS_RBG_(EXT).1, shall be applied to the cryptographic algorithms used for the above.

Application Note: The above protocols are only invoked where the client and server are both configured to support them, such as between instances of the TOE.

FTP_ITC.1.2(3) The TSF shall permit the TSF or another trusted IT product to initiate communication via the trusted channel.

FTP_ITC.1.3(3) The TSF shall initiate communication via the trusted channel *for securing information provided by DNS where both the client and server are configured to support the identified protocols*.

Application Note: *FTP_ITC.1(3) is not defined in the PP, but is added to make a special requirement of secure DNS support. Thus, application note for FTP_ITC.1 in the PP is not applicable here.*

6.2.9.4 FTP_TRP.1(1) Trusted path

Hierarchical to: No other components.

Dependencies: No dependencies.

FTP_TRP.1.1(1) **Refinement:** The TSF shall provide a communication path between itself and *remote administrators* **using *FCS_TLS_EXT and FCS_HTTPS_EXT*** that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from disclosure.

FTP_TRP.1.2(1) The TSF shall permit *remote administrators* to initiate communication via the trusted path.

FTP_TRP.1.3(1) **Refinement:** The TSF shall require the use of the trusted path for *all remote administrative actions*.

6.2.9.5 FTP_TRP.1(2) Trusted path

Hierarchical to: No other components.

Dependencies: No dependencies.

FTP_TRP.1.1(2) **Refinement:** The TSF shall provide a communication path between itself and *remote administrators* **using *FCS_TLS_EXT and FCS_HTTPS_EXT*** that is logically distinct from other communication paths and provides assured identification of its end points and **detection of modification of**

the communicated data.

FTP_TRP.1.2(2) The TSF shall permit *remote administrators* to initiate communication via the trusted path.

FTP_TRP.1.3(2) **Refinement:** The TSF shall require the use of the trusted path for *all remote administrative actions*.

Application Note: The refinement is necessary because it is not required (and in most cases impractical) for the TSF to prevent the data from being modified; it is sufficient to detect this occurrence.

6.3 Security Assurance Requirements for the TOE

This section contains the complete set of SARs from the CC Part 3. The TOE security assurance requirements, summarized in Table 10, identify the management and evaluative activities required at EAL2+.

Table 10: TOE Security Assurance Requirements

Assurance Class	Assurance Components	Assurance Components Description
Development	ADV_ARC.1	Security architecture description
	ADV_FSP.2	Security-enforcing functional specification
	ADV_TDS.1	Basic design
Guidance Documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative User guidance
Life Cycle Support	ALC_CMC.2	Use of a CM system
	ALC_CMS.2	Parts of the TOE CM coverage
	ALC_DEL.1	Delivery procedures
	ALC_FLR.2	Flaw reporting procedures
	ALC_DVS.1	Developer security procedures
Tests	ATE_COV.1	Evidence of coverage
	ATE_FUN.1	Functional testing
	ATE_IND.2	Independent testing - sample
Vulnerability Assessment	AVA_VAN.2	Vulnerability analysis

Note that as SFRs from the *Security Requirements for Network Devices* PP are included in the ST, the additional assurance activities, which are identified by “Assurance Activity” in Sections 4.2 and Annex C of *Security Requirements for Network Devices* PP shall be applicable to the evaluation of the TOE, in addition to the assurance activities required for EAL2+ (ALC_FLR.2 and ALC_DVS.1).

6.4 Security Requirements for the IT Environment

There are no security functional requirements for the IT Environment.

6.5 Rationale for Security Functional Requirements

The tables below demonstrate the completeness and sufficiency of SFRs that fulfill the objectives of the TOE.

Table 11: Completeness of Security Functional Requirements

SFRs	O.PROTECTED_COMMUNICATIONS	O.VERIFIABLE_UPDATES	O.SYSTEM_MONITORING	O.DISPLAY_BANNER	O.TOE_ADMINISTRATION	O.RESIDUAL_INFORMATION_CLEARING	O.RESOURCE_AVAILABILITY	O.SESSION_LOCK	O.TSF_SELF_TEST
FAU_GEN.1			X						
FAU_GEN.2			X						
FAU_STG.1			X						
FAU_STG.3			X						
FCS_CKM.1	X								
FCS_CKM_EXT.4	X								
FCS_COP.1(1)	X								
FCS_COP.1(2)	X	X							
FCS_COP.1(3)	X	X							
FCS_COP.1(4)	X								
FCS_COMM_PROT_EXT.1	X								
FCS_TLS_EXT.1	X								
FCS_HTTPS_EXT.1	X								
FCS_RBG_EXT.1	X								
FDP_RIP.2						X			
FIA_PMG_EXT.1					X				
FIA_UIA_EXT.1					X				
FIA_UAU_EXT.5					X				
FIA_UAU.6					X				
FIA_UAU.7					X				

SFRs	O.PROTECTED_COMMUNICATIONS	O.VERIFIABLE_UPDATES	O.SYSTEM_MONITORING	O.DISPLAY_BANNER	O.TOE_ADMINISTRATION	O.RESIDUAL_INFORMATION_CLEARING	O.RESOURCE_AVAILABILITY	O.SESSION_LOCK	O.TSF_SELF_TEST
FMT_MTD.1					X				
FMT_SMF.1					X				
FMT_SMR.1					X				
FPT_ITT.1(1)	X								
FPT_ITT.1(2)	X								
FPT_PTD.1(1)	X				X				
FPT_PTD.1(2)	X								
FPT_RPL.1	X								
FPT_STM.1			X						
FPT_TST_EXT.1									X
FPT_TUD_EXT.1		X							
FRU_RSA.1							X		
FTA_SSL_EXT.1								X	
FTA_SSL.3								X	
FTA_TAB.1				X					
FTP_ITC.1(1)	X								
FTP_ITC.1(2)	X								
FTP_ITC.1(3)	X								
FTP_TRP.1(1)	X								
FTP_TRP.1(2)	X								

Table 12: Sufficiency of Security Functional Requirements

Objectives	Description	SFRs	Purpose
O.PROTECTED_COMMUNICATIONS	The TOE provides trusted communications channels	FCS_CKM.1	Provides the capability for generating asymmetric keys to be used in secure communications mechanisms.
		FCS_CKM_EXT.4	Ensures that keys are zeroized when no longer in use.
		FCS_COP.1(1)	Provides AES encryption and decryption.

Objectives	Description	SFRs	Purpose
		FCS_COP.1(2)	Provides digital signatures.
		FCS_COP.1(3)	Provides cryptographic hashing.
		FCS_COP.1(4)	Provides message authentication.
		FCS_COMM_PROT_EXT.1	Provides secure protocols for use in communication with the TOE.
		FCS_TLS_EXT.1	Provides a TLS connection for use in communications with the TOE.
		FPT_RPL.1	Provides detection of replayed data.
		FCS_HTTPS_EXT.1	Ensures that HTTPS connections with the TOE using TLS.
		FCS_RBG_EXT.1	Provides an appropriate random bit generator.
		FPT_ITT.1(1)	Provides protection against disclosure when TSF data is in transit.
		FPT_ITT.1(2)	Provides protection against modification when TSF data is in transit.
		FPT_PTD.1(1)	Provides protection against disclosure when passwords are in transit.
		FPT_PTD.1(2)	Provides protection against disclosure when pre-shared keys, and symmetric keys are in transit.
		FTP_ITC.1(1)	Provides trusted communication channels to protect data from disclosure.
		FTP_ITC.1(2)	Provides trusted communication channels to protect data from modification.
		FTP_ITC.1(3)	Provides trusted communication channels to protect data modification and data origin masquerading.
		FTP_TRP.1(1)	Provides trusted path to protect data from disclosure.
		FTP_TRP.1(2)	Provides trusted path to protect data from modification.
O.VERIFIABLE_UPDATES	Updates can be verified before install.	FCS_COP.1(2)	Provides hash of update to the TOE.
		FCS_COP.1(3)	Provides digital signature of the update to the TOE.

Objectives	Description	SFRs	Purpose
		FPT_TUD_EXT.1	Provides a means to verify TOE updates.
O.SYSTEM_MONITORING	The TOE will provide local audit capability.	FAU_GEN.1	Provides generation of audit logs.
		FAU_GEN.2	Provides identification of the actor associated with each auditable event.
		FAU_STG.1	Provides capability of storing audit logs locally and preventing unauthorized changes to the stored audit records.
		FAU_STG.3	Provides capability of taking action when the local storage of audit records gets full.
		FPT_STM.1	Provides reliable time stamps for the audit logs.
O.DISPLAY_BANNER	The TOE will display a warning banner.	FTA_TAB.1	Provides the capability of displaying a warning banner before establishing a communication session with TOE.
O.TOE_ADMINISTRATION	Only administrators can manage the TOE.	FIA_UIA_EXT.1	Provides identification and authentication mechanisms for authorized TOE access.
		FIA_UAU_EXT.5	Provides password-based authentication mechanism.
		FIA_UAU.6	Provides the capability of re-authenticating users when changing passwords, or after session being locked.
		FIA_UAU.7	Provides only obscured feedback when password-based authentication is in progress.
		FMT_MTD.1	Provides the capability of managing TSF data by authorized users.
		FMT_SMF.1	Provides TOE management functions.
		FMT_SMR.1	Provides roles of TOE users.
		FPT_PTD.1(1)	Provides protection of passwords.
		FIA_PMG_EXT.1	Provides password management.

Objectives	Description	SFRs	Purpose
O.RESIDUAL_INFORMATION_CLEARING	Residual data will be removed before reallocation.	FDP_RIP.2	Provides the capability of removal of residual data.
O.RESOURCE_AVAILABILITY	Users cannot exhaust TOE resources.	FRU_RSA.1	Provides quotas for resources.
O.SESSION_LOCK	Unattended sessions will be locked.	FTA_SSL_EXT.1	Provides session locking for local interactive session with TOE.
		FTA_SSL.3	Provides session locking for remote interactive session with the TOE.
O.TSF_SELF_TEST	The TOE will be capable of testing it's security functions.	FPT_TST_EXT.1	Provides selft tests on TOE security functions.

6.6 Rationale for Security Assurance Requirements

The sponsor of the TOE decided that the TOE shall be evaluated at EAL2 augmented with ALC_FLR.2 and ALC_DVS.1, which provides a low to moderate level of independently assured security in the absence of ready availability of the complete development record. Meanwhile, the choice of evaluation at EAL2 augmented with ALC_FLR.2 and ALC_DVS.1 is also driven by the demand of end user of the TOE. Security Assurance Requirements are further chosen to claim conformance at the Evaluation Asurance level in accordance to current applicable NIAP policies concerning allowable EALs for evaluation in the United States.

6.7 Rationale for Dependencies

6.7.1 Security Functional Requirement Dependencies

The table below is a cross-reference of the functional components, their related dependencies, and whether the dependency was satisfied.

Table 13: SFR Dependencies Satisfied

Functional Component ID	Dependency (ies)	Satisfied
FAU_GEN.1	FPT_STM.1	Yes
FAU_GEN.2	FAU_GEN.1	Yes
	FAU_UID.1	Yes
FAU_STG.1	FAU_GEN.1	Yes
FAU_STG.3	FAU_STG.1	Yes
FCS_CKM.1	FCS_COP.1 FCS_CKM.4	Yes, FCS_CKM_EXT.4 (FCS_CKM_EXT.4 is modeled after FCS_CKM.4. However, it is explicitly stated because the TOE is required to support special key destruction method, i.e. key zeroization).
FCS_CKM_EXT.4	FCS_CKM.1	Yes
FCS_COMM_PROT_EXT.1	None	

Functional Component ID	Dependency (ies)	Satisfied
FCS_COP.1(1)	FCS_CKM.1 FCS_CKM.4	Yes, FCS_CKM_EXT.4 (FCS_CKM_EXT.4 is modeled after FCS_CKM.4. However, it is explicitly stated because the TOE is required to support special key destruction method, i.e. key zeroization).
FCS_COP.1(2)	FCS_CKM.1 FCS_CKM.4	Yes, FCS_CKM_EXT.4 (FCS_CKM_EXT.4 is modeled after FCS_CKM.4. However, it is explicitly stated because the TOE is required to support special key destruction method, i.e. key zeroization).
FCS_COP.1(3)	FCS_CKM.1 FCS_CKM.4	Yes, FCS_CKM_EXT.4 (FCS_CKM_EXT.4 is modeled after FCS_CKM.4. However, it is explicitly stated because the TOE is required to support special key destruction method, i.e. key zeroization).
FCS_COP.1(4)	FCS_CKM.1 FCS_CKM.4	Yes, FCS_CKM_EXT.4 (FCS_CKM_EXT.4 is modeled after FCS_CKM.4. However, it is explicitly stated because the TOE is required to support special key destruction method, i.e. key zeroization).
FCS_TLS_EXT.1	None	
FCS_HTTPS_EXT.1	None	
FCS_RBG_EXT.1	None	
FDP_RIP.2	None	
FIA_PMG_EXT.1	None	
FIA_UIA_EXT.1	None	
FIA_UAU_EXT.5	None	
FIA_UAU.6	None	
FIA_UAU.7	FIA_UAU.1	Yes, FIA_UIA_EXT.1 (which is equivalent to a combination of FIA_UID.1 and FIA_UAU.1, except that it allows list of TOE-provided services, not list of TSF-mediated actions, before identification and authentication).
FMT_MTD.1	FMT_SMF.1	Yes
	FMT_SMR.1	Yes
FMT_SMF.1	None	
FMT_SMR.1	FIA_UID.1	Yes, FIA_UIA_EXT.1 (which is equivalent to a combination of FIA_UID.1 and FIA_UAU.1, except that it allows list of TOE-provided services, not list of TSF-mediated actions, before identification and authentication).
FPT_ITT.1(1)	None	
FPT_ITT.1(2)	None	
FPT_PTD.1(1)	None	

Functional Component ID	Dependency (ies)	Satisfied
FPT_PTD.1(2)	None	
FPT_RPL.1	None	
FPT_STM.1	None	
FPT_TST_EXT.1	None	
FPT_TUD_EXT.1	None	
FRU_RSA.1	None	
FTA_SSL_EXT.1	FIA_UAU.1	Yes, FIA_UIA_EXT.1 (which is equivalent to a combination of FIA_UID.1 and FIA_UAU.1, except that it allows list of TOE-provided services, not list of TSF-mediated actions, before identification and authentication).
FTA_SSL.3	None	
FTA_TAB.1	None	
FTP_ITC.1(1)	None	
FTP_ITC.1(2)	None	
FTP_ITC.1(3)	None	
FTP_TRP.1(1)	None	
FTP_TRP.1(2)	None	

6.7.2 Security Assurance Requirement Dependencies

SAR dependencies identified in the CC have been met by this ST as shown in the table below.

Table 14: EAL2+ SAR Dependencies Satisfied

Assurance Component ID	Dependencies	Satisfied
ADV_ARC.1	ADV_FSP.1 ADV_TDS.1	Yes
ADV_FSP.2	ADV_TDS.1	Yes
ADV_TDS.1	ADV_FSP.2	Yes
AGD_OPE.1	ADV_FSP.1	Yes, by ADV_FSP.2 which is hierarchical to ADV_FSP.1
AGD_PRE.1	None	
ALC_CMC.2	ALC_CMS.1	Yes, by ALC_CMS.2 which is hierarchical to ALC_CMS.1
ALC_CMS.2	None	
ALC_DEL.1	None	

Assurance Component ID	Dependencies	Satisfied
ALC_FLR.2	None	
ALC_DVS.1	None	
ATE_IND.2	ADV_FSP.2 AGD_OPE.1 AGD_PRE.1 ATE_FUN.1 ATE_COV.1	Yes
ATE_FUN.1	ATE_COV.1	Yes
ATE_COV.1	ATE_FUN.1 ADV_FSP.2	Yes
AVA_VAN.2	ADV_ARC.1 ADV_FSP.2 ADV_TDS.1 AGD_OPE.1 AGD_PRE.1	Yes

7 TOE Summary Specification

This section presents an overview of the security functions implemented by the TOE.

7.1 TOE Security Functions

The following security functions are implemented by the TOE:

- TSF_TOE_COMM
- TSF_TRUSTED_UPDATES
- TSF_AUDIT
- TSF_TOE_ACCESS
- TSF_RESOURCE_EXHAUSTION
- TSF_USER_DATA_DISCLOSURE
- TSF_SELF_TEST

7.1.1 TSF_TOE_COMM

Related SFRs:	FCS_COMM_PROT_EXT.1, FCS_TLS_EXT.1, FCS_HTTPS_EXT.1, FCS_CKM.1, FCS_CKM_EXT.4, FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FCS_RBG_EXT.1, FPT_ITT.1(1), FPT_ITT.1(2), FPT_PTD.1(1), FPT_PTD.1(2), FPT_RPL.1, FTP_ITC.1(1), FTP_ITC.1(2), FTP_ITC.1(3), FPT_TRP.1(1), FPT_TRP.1(2)

The TOE communicates with other network devices, as well as administrators, over the network. The critical communication paths and their related protection mechanisms are as follows:

- **Grid communication.** The TOE may be configured to communicate with other TOE instances in a grid. This communication is protected via an SSL/TLS VPN.
- **Remote Administration.** Remote administrators configure the TOE via a web based GUI that is protected using TLS/HTTPS.
- **Application Programming Interface.** The TOE provides a Perl API to assist integration of the Infoblox device into network environments. The API is protected using TLS/HTTPS. The Perl API provides interfaces to DHCP, DNS, Grid and IPAM services.
- **DNS.** The TOE implements DNSSEC, TSIG (Transaction SIGnature) and GSS-TSIG (Generic Security Service Algorithm for Secret Key Transaction) to secure DNS traffic between itself and other trusted IT products.

The following sections provide further detail on each of the secure communication protocols identified above and the associated supporting cryptography

7.1.1.1 SSL/TLS VPN

The TOE uses the OpenVPN/OpenSSL implementation of TLS to establish a VPN for Grid communication between instances of the TOE (when so configured). This implementation has the following characteristics:

- Key exchange (128 bit) is as per TLS RFC 2246 with OpenVPN specified as the transport protocol.
- All packets sent over the VPN after the key exchange are encrypted with AES-128-CBC.
- Authentication and integrity are provided using HMAC as per IPSec Authentication Header (AH) described in RFC 2402. **Note:** The TOE does not implement the full IPSec protocol.

- Replay protection is provided as per IPSec Encapsulating Security Payload (ESP) described in RFC 2406. **Note:** The TOE does not implement the full IPSec protocol.

7.1.1.2 TLS/HTTPS

The TOE implements TLS 1.0 (RFC 2246) without extensions, supporting the following ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA,
- TLS_RSA_WITH_AES_256_CBC_SHA,
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA, and
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA.

The TOE implements the HTTPS protocol that complies with RFC 2818 using TLS 1.0. This implementation uses Apache 2.2 (mod_ssl) and OpenSSL 1.0.0e (with fips extensions) on port 443. TLS is configured to only support TLS 1.0 and allow only the cipher suites listed above. The TOE (server) uses a 2048 bit RSA key. The server certificate may be self-signed or signed by an external CA using a CSR generated by the TOE. Client TLS authentication is not used. Session resumption is supported.

7.1.1.3 Secure DNS Protocols

The TOE implements the following protocols to verify and authenticate DNS updates (standards and algorithm detail are provided at FTP_ITC.1.1(3)section 6.2.9.3):

- **DNSSEC.** DNSSEC is used to verify digitally signed DNS server responses from external servers and sign DNS responses sent by the TOE.

- **TSIG.** TSIG is used to authenticate dynamic DNS updates between pre-configured servers in ISC environments.

- **GSS-TSIG.** GSS-TSIG is an extension to TSIG for authenticating DNS updates in Microsoft/Kerberos environments.

7.1.1.4 Cryptographic Support

The TOE generates asymmetric cryptographic keys in accordance with a domain parameter generator, a random number generator and a prime number generator that meet ANSI X9.80. Generated key strength is equivalent to, or greater than, a symmetric key strength of 112 bits using conservative estimates.

Meanwhile, for domain parameters used in finite field-based key establishment schemes, the TOE meets NIST Special Publication 800-56A. For domain parameters used in RSA-based key establishment schemes, the TOE meets NIST Special Publication 800-56B (RSA CAVP certificate #1085).

The TOE performs encryption and decryption in accordance with cryptographic algorithm AES operating in CBC mode and cryptographic key sizes 128-bits, 256-bits, and 192-bits. FIPS PUB 197 and NIST SP 800-38B are met for AES implementation (AES CAVP certifate #2115).

The TOE performs cryptographic signature services in accordance with RSA Digital Signature Algorithm (rDSA) with a key size (modulus) of 2048 bits or greater. It meets the requirements of RSA Digital Signature in FIPS PUB 186-3 (RSA CAVP certificate #1085).

The TOE performs cryptographic hashing services in accordance with SHA-1, SHA-256, SHA-384, and SHA-512 and message digest sizes 160, 256, 384, and 512 bits. FIPS Pub 180-2 is met for SHA implementation (SHA CAVP certificate #1839).

The TOE performs keyed-hash message authentication in accordance with HMAC-SHA-1, SHA-256, SHA-384, SHA-512, key size 128, 256 and 512 bits, and message digest sizes 160, 256, 384, 512 bits. FIPS Pub 198 and FIPS Pub 180-2 are met for HMAC implementation (HMAC CAVP certificate #1287).

The TOE performs all random bit generation (RBG) services in accordance with FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using AES seeded by an entropy source that accumulates entropy from at least one independent TSF-hardware-based noise sources. The deterministic RBG is seeded with a minimum 256 bits of entropy at least equal to the greatest bit length of the keys and authorization factors that it will generate (RNG CAVP certificate #1287).

Specifically, random number generation is initialized and re-seeded using the timing of the following events:

- Disk I/O
- Key and mouse input (does not normally occur on NIOS)
- Interrupt handling (IRQ events)

The seeding is put into an entropy pool (4096 bits) and are mixed there using a CRC-like function.

The random bytes is generated from the entropy pool: A SHA1 hash is taken of the contents of the entropy pool and the resulting hash is used as the random bytes. The result is also re-mixed into the entropy pool (using same method as above). Particularly, the TOE implements two devices to generate random bytes, and they are slightly different but still meet the aforementioned requirements of random bytes generation:

- /dev/random keeps a count of the number of bytes of effective seeding and the number of random bytes generated and will block until more seeding occurs to make certain that the entropy is not overused. The effective seeding is estimated by looking at the timing between seedings for a device /irq/etc.
- /dev/urandom does NOT block in this way and will continue to deliver random bytes.

On system boot the TOE's random handling will be initialized with saved random state (from the previous time the system was up, if any).

The TOE zeroizes all plaintext secret and private cryptographic keys and Cryptographic Critical Security Parameters (CSPs) when no longer required. While the administrator could directly read RAM or persistent memory to view CSPs, they are trusted not to do so.

Table 15 below identifies all secret and private keys and CSPs used to generate key, the related zeroization procedures and whether any interface is available to view the plaintect key.

Table 15: Keys, zeroization and interfaces

Key/CSP	Location	Zeroization procedure	Interface
HTTPS server private key: 2048 bit RSA	Encrypted database file. Encrypted in database backups.	Overwritten when no longer in use; three times with a random pattern, and once with zeroes.	Not readable via any normal interface.
TLS/VPN private key: 2048 bit RSA	File	Overwritten when no longer in use; three times with a random pattern, and once with zeroes.	Not readable via any normal interface.
Reporting CA private key: RSA 2048 bits	Encrypted database file. Encrypted in	Overwritten when no longer in use; three	Not readable via any normal interface.

Key/CSP	Location	Zeroization procedure	Interface
	database backups.	times with a random pattern, and once with zeroes.	
Reporting TLS node private keys of forwarders and indexer: RSA 2048 bits	Stored in database and in files. Encrypted in database backups.	Overwritten when no longer in use; three times with a random pattern, and once with zeroes.	Not readable by any normal interface.
DNSSEC private keys	Stored in database and in files. Encrypted in database backups.	Overwritten when no longer in use; three times with a random pattern, and once with zeroes.	Not readable by any normal interface.
Static symmetric key for decrypting software updates (AES-256-CBC)	Compiled into a library binary.	None. Key remains static.	Not readable by any normal interface.
Static symmetric key for encrypting/decrypting database backups (AES-128-CBC)	Compiled into a library binary.	None. Key remains static.	Not readable by any normal interface.
DNS GSS-TSIG shared keys	Stored in the database and in files. Encrypted in database backups.	None. Key remains static.	Not readable by any normal interface.
Grid shared secret	Stored in the database and in files. Encrypted in database backups.	Overwritten when no longer in use; three times with a random pattern, and once with zeroes.	Not readable by any normal interface.
Administration session cookie HMAC key: HMAC-SHA1	File	Overwritten when no longer in use; three times with a random pattern, and once with zeroes.	Not readable by any normal interface.
TLS session keys	Stored in memory or on disk	Overwritten with zeroes when no longer in use	Not readable by any normal interface.
RBG state seed	Process memory	The generator state is overwritten with zeroes when the generator process exits, at system shutdown.	Not readable by any normal interface.

7.1.2 TSF_TRUSTED_UPDATES

Related SFRs: FCS_COP.1(2), FCS_COP.1(3), FPT_TUD_EXT.1

The TOE provides security administrators the ability to query the current version of the TOE firmware/software. The security administrators are provided interfaces to initiate updates to TOE firmware/software, and the TOE uses digital signature mechanism to verify firmware/software updates prior to installing those updates.

Specifically, Infoblox generates RSA digital signature for TOE updates to ensure that the update can be trusted. The TOE verifies the digital signature associated with the TOE update. The certificate used for validation is stored in a protected file on the appliance.

The RSA digital signature used by the TOE meets FIPS PUB 186-3 with a key size (modulus) of 2048 bits or greater. When generating RSA digital signature, SHA is used to perform cryptographic hashing services in accordance with the FIPS Pub 180-2.

The signature is used by the product and will not allow an update to be uploaded that is not properly signed. The key (not certificate) used by validation is kept in a file on the root disk. There is no means for an administrator to access that file. The software updates are pointed to from the Infoblox Support web site and are hosted on the Infoblox FTP server.

7.1.3 TSF_AUDIT

Related SFRs:	FAU_GEN.1, FAU_GEN.2, FAU_STG.1, FAU_STG.3, FPT_STM.1

The TOE generates an audit record of the following auditable events: Start-up and shutdown of the audit functions; All auditable events for the basic level of audit; All administrative actions; and specifically defined auditable events listed in Table 9.

The TOE records within each audit record the following information: Date and time of the event, type of event, subject identity (if applicable),and the outcome (success or failure) of the event; and for each audit event type, based on the auditable event definitions of the functional components, information specified in column three of the Table 9.

For audit events resulting from actions of identified users, the TOE associates each auditable event with the identity of the user that caused the event.

The audit log is stored locally by default, and the TOE protects the stored audit records in the audit trail from unauthorized deletion and prevent unauthorized modifications to the stored audit records in the audit trail.

Specifically, the audit logs are kept in two separate log files:

- Audit Log
- sys log (this refers to an internal system log file, not a Syslog server, however the TOE may be configured to send Audit Log and sys log events to a Syslog server)

The Audit Log is accessible only to the superuser. The sys log is accessible to the superuser plus users with privileges. The TOE deletes the oldest audit file if the audit trail exceeds maximum of 10 audit log files.

The local time source supports the reliable time stamp for audit function.

7.1.4 TSF_TOE_ACCESS

Related SFRs:	FIA_PMG_EXT.1, FIA_UIA_EXT.1, FIA_UAU_EXT.5, FIA_UAU.6, FIA_UAU.7, FMT_MTD.1, FMT_SMF.1, FMT_SMR.1, FTA_TAB.1, FTA_SSL_EXT.1, FTA_SSL.3, FPT_PTD.1(1)

The TOE provides the administrator access to the TOE via console port (locally) and HTTPS (remotely). The TOE provides a password-based logon mechanism for authorized access to the TOE locally or remotely.

The TOE allows no services on behalf of the user to be performed before the user is identified and authenticated, and requires each user to be successfully identified and authenticated before allowing any other actions on behalf of that user.

The TOE enforces the following password policy for administrative passwords:

- Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and special characters (that include: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”);
- Minimum password length shall settable by the Security Administrator, and support passwords of 8 characters or greater;
- Passwords composition rules specifying the types and number of required characters that comprise the password shall be settable by the Security Administrator.
- Passwords shall have a maximum lifetime, configurable by the Security Administrator.
- New passwords must contain a minimum of 4 character changes from the previous password.

The TOE not only provides a local password-based authentication mechanism as described above, but also supports authentication against Active Directory server to perform user authentication.

The TOE implements that users with expired passwords are required to create a new password after correctly entering the expired password. The TOE re-authenticates the user when the user changes their password, or following session locking.

The TSF provides only obscured feedback to the user while the authentication is in progress at the local console.

The TOE restricts the ability to manage the TSF data to the System Administrators.

The TSF is capable of performing the following management functions:

- Ability to configure the cryptographic functionality.
- Ability to update the TOE, and to verify the updates using the digital signature capability.
- Ability to configure the security functions of the TOE.

The TOE maintains the following roles: Superuser, Limited-Access and Default and is able to associate users with roles.

Only superusers can create admin groups and define their administrative permissions. There are two ways to define the permissions of an admin group. You can create an admin group and assign permissions directly to the group, or you can create roles that contain permissions and assign the roles to an admin group.

• **Superuser** – Superuser admin groups provide their members with unlimited access and control of all the operations that a NIOS appliance performs. There is a default superuser admin group, called admin-group, with one superuser administrator, admin. Superusers can access the appliance through its console, GUI, and API. In addition, only superusers can create admin groups.

• **Limited-Access** – All limited-access admin groups require either read-only or read/write permission to access certain resources, such as grid members, and DNS and DHCP resources, to perform certain tasks. Therefore, when you create an admin group, you must specify which resources the group is authorized to access and their level of access.

• **Default** – When upgrading from previous NIOS releases, the appliance converts the ALL USERS group to the Default Group when the ALL USERS Group contains admin accounts. The evaluated configuration requires a fresh install and therefore the Default role will not be assigned.

The TOE prevents unauthorized user access to passwords and critical security parameters (seeds, seeding keys, pre-shared keys, symmetric key, and private keys) and prevents reading of the plaintext passwords, and all pre-shared keys, symmetric key, and private keys.

For local interactive sessions, the TOE terminates the session after a Security Administrator-specified time period of inactivity. The TOE also terminates a remote interactive session after a Security Administrator-configurable time interval of session inactivity.

Before establishing a user/administrator session, the TOE displays a Security Administrator-specified advisory notice and consent warning message regarding unauthorized use of the TOE.

Password representations are stored in SHA-1 hash format.

7.1.5 TSF_RESOURCE_EXHAUSTION

Related SFRs: FRU_RSA.1

The TOE implements quotas which are placed on the amount of exhaustible resources. Specifically, The TOE enforces maximum quotas of the following resources: log file size and number, upload file size and number, number of files stored on the TOE, and space occupied by downloadable files (for file distribution), that individual user, subjects can use simultaneously, or over a specified period of time. The file distribution feature allows files to be uploaded to the appliance (using secure GUI or API) and then served to clients using TFTP, FTP, or HTTP. The files to be downloaded are supplied by the customer and are typically VOIP phone configurations and firmware.

For log files, there is a configurable maximum size of each log file and then roll them over, keeping no more than ten of each log. For uploaded files (for file distribution), there is a limit that can be set by the administrator in the grid file distribution properties editor.

7.1.6 TSF_USER_DATA_DISCLOSURE

Related SFRs: FDP_RIP.2

The TOE ensures that any previous information content of a resource is made unavailable upon the allocation of the resource to all objects by clearing the residual information before network packets are sent from the TOE.

Specifically, all network traffic goes through socket buffers allocated and managed by the kernel. The routines for allocation are centralized and all appropriate zeroing and initialization is handled by routines in the sk_buff routines.

The Kernel guarantees the clearing of the memory from one process invocation to the next and thereby makes certain that there can be no leaking of information between connections managed by different processes.

The user level code, i.e. programs listed below, also implements mechanisms to ensure that residual information shall not be leaked:

- named (DNS):

  TCP replies use buffers managed by a Bind specific package that handles the initialization (zeroing) of all buffers.

  UDP replies use buffers that are re-used without explicit clearing, but the code is specifically written to always write in new data for all parts of the buffer that is actually sent (i.e. the buffer is overwritten).

  Named, as part of its base functionality, uses internal caches to store information which is used in multiple different replies (to different clients), the information here is however not private to any client and the caches does not re-use any actual reply packages or query information.

- dhcpd (DHCP):

The DHCP server only sends UDP messages to clients and uses the same technique as described above (and the same underlying library).

- httpd (Apache, web server):

  Httpd has been written to use buffer and memory management from the Apache Portable Runtime Library (APR). The APR overwrites the buffer to prevent it from re-use.

- syslog-ng (syslog):

  The syslog daemon can be used to forward log messages. All buffers used by syslog-ng for message sent over the network are handled by the GString package from glib (GNOME C lib), this package handles strings with lengths and is used in such a way that all possible previous data is overwritten with new data before anything is sent.

- ntpd (NTP):

  Ntpd sends packets by filling in structures representing the protocol on the wire, i.e. the data buffer is overwritten.

  Note that, the reqirement of reliable time stamp is not implemented by NTP. Instead, it is supported by local time source.

7.1.7 TSF_SELF_TEST

Related SFRs: FPT_TST_EXT.1

The TOE implements self-test, during initial startup, to verify basic hardware components the TOE is relient upon, as illustrated in the table below, and also verify the integrity of the cryptographic module in the TOE in accordance to FIPS 140-2 power-up self-test requirements, i.e. Known Answer Tests for cryptographic algorithms supported by the TOE, Random Number Generation Known Answer Test, and cryptographic module integrity test. The TOE will log to the syslog when the test runs on power-up, either successful or failure. In the failure case, it is not permitted to use cryptography services, the unit will start in a broken state where the only permitted access is via the physical serial port. For units with an LCD panel on the front, they will display a failure message.

Table 16: BIOS POST Procedures

NMI Disable	NMI interrupt line to the CPU is disabled by setting bit 7 I?O port 70h (CMOS)
Power On Delay	Once the keyboard controller gets power, it sets the hard and soft reset bits. Check the keyboard controller or clock generator if a failure occurs
Initialize Chipsets	Check the BIOS, CLOCK and chipsets
Reset Determination	The BIOS reads the bits in the keyboard controller to see if a hard or soft reset is required (a soft reset will not test memory above 64K). Failure could be the BIOS or keyboard controller
ROM BIOS Checksum	The BIOS performs a checksum on itself and adds a preset factory value that should make it equal to 00. If a failure occurs, check the BIOS chips
Keyboard Test	A command is sent to the 8042 keyboard controller which performs a test and sets a buffer space for commands. After the buffer is defined the BIOS sends a command byte, writes data to the buffer, checks the high order bits of the internal keyboard controller and issues a No Operation (NOP) command
CMOS	Shutdown byte in CMOS RAM offset 0F is tested, the BIOS checksum calculated and diagnostic byte 0E updated before the CMOS RAM area is initialized and updated for date and time. Check the RTC and CMOS chip or battery if a failure occurs
DMA (8237) and PIC (8259) Disable	The DMA and Programmable Interrupt Controller are disabled before the POST proceeds and further. Check the 8237 or 8259 chips if a failure

	occurs
Video Disable	The video controller is disabled and port B initialized. Check the video adapter if a failure occurs
Chipset Initialized and Memory Detected	Memory addressed in 64K blocks. Failure would be in the chipset. If all memory is not seen, failure could be in a chip in the block after the last one seen
PIT Test	The timing functions of the 8254 Programmable Interrupt Timer are tested. The PIT and RTC chips normally cause errors here
Memory Refresh	PIT's ability to refresh memory is tested. If an XT, DMA controller #1 handles this. Failure is normally the PIT (8254) in AT's or the 8237, DMA #1, in XT's
Address Line	Test the address lines in the first 64K of RAM. If a failure occurs, an address line may be the problem
Base 64K	Test the address lines in the first 64K of RAM. If a failure occurs, an address line may be the problem
Chipset Initialization	The PIT, PIC and DMA controllers are initialized
Set Interrupt Table	Interrupt vector table used by PIC is installed in low memory, the first 2K
8042 Keyboard Controller Check	The BIOS reads the buffer area in the keyboard controller I/O port 60. Failure here is normally the keyboard controller
Video Tests	The type of video adapter is checked for, then a series of tests are performed on the adapter and monitor
BIOS Data Area	The vector table is checked for proper operation and video memory verified before protected mode tests are entered into. This is done so that any errors found are displayed on the monitor
Protected Mode Tests	Perform reads and writes to all memory locations below 1MB. Failure at this point indicate a bad RAM chip, the 8042 Keyboard Controller or a data line
DMA Chips	The DMA registers are tested using a data pattern
Final Initialization	Typically, the floppy and hard drives are tested and initialized and a check is made for serial and parallel devices. The information gathered is then compared against the contents of the CMOS and you will see the results of any failures on the monitor
BOOT	The BIOS hands over control to the Int 19 bootloader. This is where you would see error messages such as non-system disk

8 Glossary

For the purposes of this document, the following terms and definitions apply.

8.1 *Acronyms*

Acronym	Definition
A.	assumption (when used in hierarchical naming)
CC	Common Criteria
EAL	evaluation assurance level
IT	information technology
O.	security objective (of the TOE) (when used in hierarchical naming)
OE.	security objective (of the operational environment) (when used in hierarchical naming)
OSP	organizational security policy
P.	organizational security policy (when used in hierarchical naming)
PP	protection profile
SFP	security function policy
SFR	security functional requirement
ST	security target
T.	threat (when used in hierarchical naming)
TOE	Target of Evaluation
TSF	TOE security functionality
TSP	TOE security policy

9 References

1. Common Criteria for Information Technology Security Evaluation Version 3.1 Release 3 - Part 1: Introduction and General Model

2. Common Criteria for Information Technology Security Evaluation Version 3.1 Release 3 – Part 2: Security Functional Requirements

3. Common Criteria for Information Technology Security Evaluation Version 3.1 Release 3 – Security Assurance Requirements

4. Common Methodology for Information Technology Security Evaluation Version 3.1 Release 3 - Evaluation Methodology

5. Security Requirements for Network Devices, Version 1.0, dated December 10, 2010