Riverbed Technology
Cascade Shark v9.6 and Cascade Pilot v9.6

Security Target

Evaluation Assurance Level: EAL3+
Document Version: 0.24

Prepared for:

Riverbed Technology
199 Fremont St.
San Francisco, CA 94105
United States of America

Phone: +1 (415) 247-8800
Email: support@riverbed.com
http://www.riverbed.com

Prepared by:

Corsec Security, Inc.
13135 Lee Jackson Memorial Highway, Suite 220
Fairfax, VA 22033
United States of America

Phone:+1 (703) 267-6050
Email: info@corsec.com
http://www.corsec.com

Table of Contents

Table of Figures

List of Tables

1 Introduction

This section identifies the Security Target (ST), Target of Evaluation (TOE), and the ST organization. The Target of Evaluation is the Riverbed Cascade Shark v9.6 and Cascade Pilot v9.6 (Shark and Pilot), and will hereafter be referred to as the TOE or Shark and Pilot throughout this document. The TOE is a software-only network analysis solution providing high-performance, multi-gigabit-per-second network traffic analysis, recording, monitoring, and reporting. The TOE is produced by Riverbed Technology.

1.1 Purpose

This ST is divided into nine sections, as follows:

- Introduction (Section 1) – Provides a brief summary of the ST contents and describes the organization of other sections within this document. It also provides an overview of the TOE security functions and describes the physical and logical scope for the TOE, as well as the ST and TOE references.
- Conformance Claims (Section 2) – Provides the identification of any Common Criteria (CC), ST Protection Profile, and Evaluation Assurance Level (EAL) package claims. It also identifies whether the ST contains extended security requirements.
- Security Problem (Section 3) – Describes the threats, organizational security policies, and assumptions that pertain to the TOE and its environment.
- Security Objectives (Section 4) – Identifies the security objectives that are satisfied by the TOE and its environment.
- Extended Components (Section 5) – Identifies new components (extended Security Functional Requirements (SFRs) and extended Security Assurance Requirements (SARs)) that are not included in CC Part 2 or CC Part 3.
- Security Requirements (Section 6) – Presents the SFRs and SARs met by the TOE.
- TOE Specification (Section 7) – Describes the security functions provided by the TOE that satisfy the security functional requirements and objectives.
- Rationale (Section 8) - Presents the rationale for the security objectives, requirements, and SFR dependencies as to their consistency, completeness, and suitability.
- Acronyms (Section 9) – Defines the acronyms and terminology used within this ST.

1.2 Security Target and TOE References

Table 1 ST and TOE References

ST Title	Riverbed Technology Cascade Shark v9.6 and Cascade Pilot v9.6 Security Target
ST Version	Version 0.24
ST Author	Corsec Security, Inc.
ST Publication Date	1/14/2013
TOE Reference	Riverbed Cascade Shark v9.6 and Cascade Pilot v9.6, Version 9.6.1004.7291
Keywords	Network traffic analysis, recording, monitoring, and reporting

1.3 TOE Overview

The TOE provides network monitoring, network traffic recording and analysis, and performance management capabilities that collect traffic data on a network, calculate performance metrics and alert the administrators to problems or other conditions. It is a software-only TOE. The TOE includes the Cascade

Shark software and its underlying operating system and the Cascade Pilot software along with the FIPS[1] 140-2 validated cryptographic modules provided by the underlying Windows operating system on the Pilot system. The hardware on which the Cascade Shark is delivered is part of the operating environment. Except for the FIPS validated cryptographic modules, the operating system and hardware on which the Cascade Pilot executes is also in the operating environment.

Shark can be deployed in two different configurations. Shark may come installed on a Shark Appliance that is produced by Riverbed. The TOE may also be deployed on a virtual appliance called Virtual Cascade Shark. In the case of the virtual appliance, the hardware is simulated through a virtual machine. Riverbed supports VMware ESXi 4.1 to provide the virtualization environment. Functionality of both deployments is the same. In the case of the virtual appliance, VMware VMtools are installed to facilitate the needs of the virtual appliance and the virtual machine in which it is installed on. Hereafter, the term Shark applies to both the Cascade Shark Appliance and Virtual Cascade Shark unless noted otherwise.

Shark is a turnkey hardware and software solution capable of sustained, multi-gigabit per second recording of network traffic. Its main purpose is to facilitate the capture and metric calculations of network traffic at high speeds.

Pilot is a network analysis software product that seamlessly and securely integrates with one or more remote Shark appliances for a fully distributed, easy to manage packet capture solution. It is an analysis tool whose main purpose is to provide a centralized analysis and reporting GUI[2] to administrators of multiple Shark appliances.

The TOE is generally deployed in a one-to-many or many-to-many configuration, where one or more Pilot installations are used to manage one or more Shark appliances as depicted in Figure 1. Typically a Shark appliance is required for each LAN[3] segment of a network.

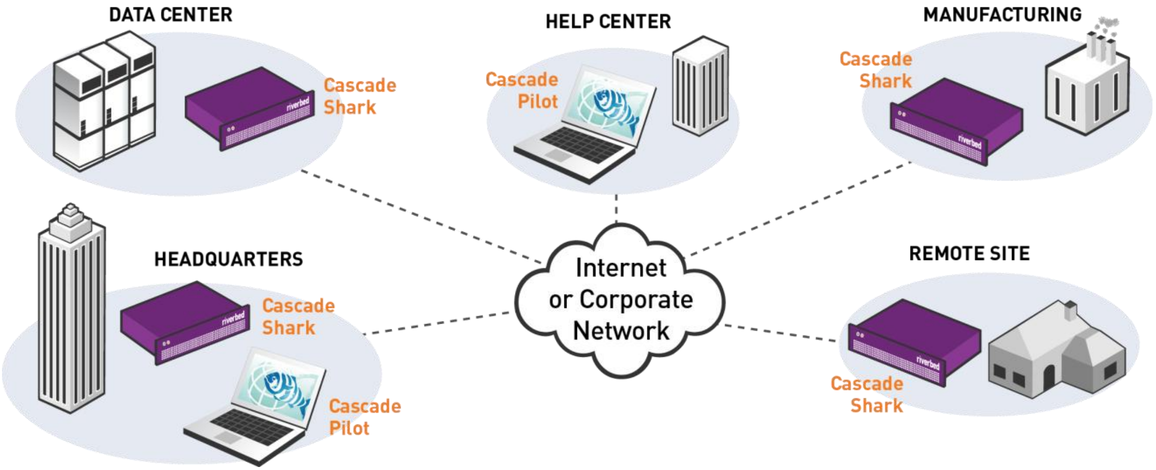

Figure 1 Typical Shark and Pilot Deployment

[1] Federal Information Processing Standard

[2] Graphical User Interface

[3] Local Area Network

1.3.1 TOE Environment

The hardware on which the Shark software operates includes an Intel processor and runs a Linux-based operating system. Shark is shipped as an appliance on three different hardware platforms. The Shark hardware is in the TOE environment.

Three models of the appliance are available, each with varying performance and storage. All models implement storage in a RAID[4]-0 striped set configuration for maximum throughput to disk. The partition of the array dedicated for the storage of captured packets is unformatted and utilized in RAW mode, allowing the TOE to leverage a proprietary storage system to maximize bulk read/write speeds and optimize seek times.

The hardware and operating system on which Pilot executes is in the TOE environment. The Pilot software executes on the Windows XP, Windows Vista, and Windows 7 operating systems. In the evaluated configuration, the Pilot software must be installed on one of the Windows operating systems that include a FIPS-validated cryptographic module or algorithms as specified in Table 2. The cryptographic module must be configured to be in FIPS approved mode. It is recommended that the Pilot hardware meet the following specifications:

- A dual-core 2.0 GHz CPU or better
- 2 GB RAM
- 300 MB free disk space plus additional space for trace files and reports
- Support for graphics cards with a minimum resolution of 1024x768

Table 2 Windows FIPS Certificates for Pilot

Note: See Table 15 for a listing of additional CAVP certificates.

Windows OS[5]	FIPS 140-2 Certificates CMVP	FIPS 140-2 Certificates CAVP

Microsoft Windows 7 Ultimate Edition (x86 Version);
Microsoft Windows 7 Ultimate Edition (x64 Version);
Microsoft Windows 7 Ultimate Edition SP1 (x86 Version);
	1328	AES (Certs. #1168 and #1178); AES GCM (Cert. #1168, vendor affirmed); AES GMAC (Cert. #1168, vendor-affirmed); DRBG (Certs. #23 and #24); ECDSA (Cert.#141); HMAC (Cert. 677); KAS (SP 800-56A, vendor affirmed, key agreement; key establishment methodology provides 80 to 256 bits of encryption strength; RNG (Cert. #649); RSA (Certs. #559 and #560); SHS (Cert. #1081); Triple-DES (Cert. #846)

Microsoft Windows 7 Ultimate Edition SP1 (x64 Version)(single-user mode)	1329	AES (Certs. #1168 and #1178); AES GCM (Cert. #1168, vendor affirmed); AES GMAC (Cert. #1168, vendor-affirmed); DRBG (Certs. #23 and #24); DSA (Cert.#386); ECDSA (Cert.#141); HMAC (Cert. 677); KAS (SP 800-56A, vendor affirmed, key agreement; key establishment methodology provides 80 to 256 bits of encryption strength; RNG (Cert. #649); RSA (Certs. #559 and #560); SHS (Cert.

[4] Redundant Array of Independent Disks

[5] Operating System

Windows OS[5]	FIPS 140-2 Certificates CMVP	FIPS 140-2 Certificates CAVP
		#1081); Triple-DES (Cert. #846)
Microsoft Windows XP, Professional SP 3 (in single user mode)	989	AES (Cert. #781); HMAC (Cert. #428); RNG (Cert. #447); RSA (Cert. #371) SHS (Cert. #783); Triple-DES (Cert. #675); Triple-DES MAC (Triple-DES cert. #675. vendor affirmed)
	990	DSA (Cert #292); RNG (Cert. #448); SHS (Cert. #784); Triple-DES (Cert.#676); Triple-DES MAC (Triple-DES Cert. #676, vendor affirmed)
	997	HMAC (Cert. #429); RNG (Cert. #449); SHS (Cert. #785): Triple-IDES (Cert. #677): Triple-DES MAC (Triple-DES Cert. #677, vendor affirmed)
Microsoft Windows Vista Ultimate Edition (x86 Version); Microsoft Windows Vista Ultimate Edition (x64 Version) (single-user mode)	893	AES (Cert. #553); HMAC (Cert. #297); RNG (Cert. #321); RSA (Cert. #255 and #258; SHS (Cert. #618; Triple-DES (Cert. #549)
	894	DSA (Cert. #226); RNG (Cert. #321); SHS (Cert. #618); Triple-DES (Cert. #549); Triple-DES MAC (Cert. #549, vendor affirmed)
Microsoft Windows Vista Ultimate Edition SP1 (x86 Version); Microsoft Windows Vista Ultimate Edition SP1 (x64 version) (single-user mode)	1000	AES (Cert. #739 and #756); ECDSA (Cert. #82); HMAC (Cert. #412); RNG (Cert. #435 and SP 800-90AES-CTR, vendor-affirmed); RSA (Cert. #353 and #357; SHS (Cert. #753); Triple-DES (Cert. #656)
	1002	AES (Cert. #739); HMAC (Cert. #407); RNG (SP 800-90, vendor affirmed); RSA (Cert. #353 and #354; SHS (Cert. #753); Triple-DES (Cert. #656)
	1003	DSA (Cert. #281); RNG (Cert. #435); SHS (Cert. #753); Triple-DES (Cert.#656); Triple-DES MAC (Cert. #656, vendor affirmed)

The TOE relies on the underlying hardware of both the Shark appliance and Pilot system to include a hardware clock to be retrieved by the operating system for use by the TOE in time stamping the audit records.

1.4 TOE Description

The TOE is the Cascade Shark v9.6 and Cascade Pilot v9.6 software. It is a software only TOE that provides a performance monitoring and management solution.

1.4.1 Architecture

The architecture of the TOE leverages various frameworks and programming languages. Shark and Pilot combine flow analysis, packet analysis, and retrospective packet capture to provide an integrated, top-down, application-aware performance monitoring and management solution.

1.4.1.1 Cascade Shark

The Shark's main purpose is to facilitate the capture and metric calculations of network traffic at high speeds. The Cascade Shark software is divided into two components: the Shark Probe and Shark Packet Recorder. Each component is coded in C and C++.

Utilizing a multithreaded design, the Shark Packet Recorder captures network traffic while the Shark Probe indexes and calculates flow metrics of the traffic with no performance degradation. The Shark Probe communicates with the Shark Packet Recorder using a proprietary RPC[6] protocol, which is only available to the TOE.

The Shark Packet Recorder moves network packets from the network interface into memory, time-stamps them and then saves the packets as objects called job traces onto disk. Capturing functionality is regulated by "capture jobs" which can be started manually, triggered, or scheduled, and are designed to isolate traffic by port, protocol, IP[7] address, etc. Shark Packet Recorder makes use of time metadata information to index network packet data for faster time-driven retrieval of the packets.

The Shark Probe performs network analysis, monitoring, and reporting. The Shark Probe is capable of performing analysis on both live and previously saved network traffic. The analysis functionality is regulated by "views", which are the core analysis and visualization paradigm. The system provides over 200 views that can be applied to live traffic or leveraged against historical packet captures and indices. The views compute specific metrics, such as bandwidth over time, IP conversations or protocol distributions.

The Shark Probe web server is implemented by Poco library.

The Cascade Shark appliances are based on general-purpose Intel-based server hardware running a Linux-based operating system. Command line access to the Cascade Shark is available via the console and remote shell access. In the CC evaluated configuration, remote shell access to Shark is disabled.

The appliance on which the Shark executes implements storage in a RAID-0 striped set configuration for maximum throughput to disk. The partition of the array dedicated for the storage of captured packets is unformatted and utilized in RAW mode, allowing the TOE to leverage a proprietary storage system to maximize bulk read/write speeds and optimize seek times. In the case of Virtual Cascade Shark, the Packet Storage volume utilizes a virtual disk. The NIC[8] on the hardware platforms is a proprietary PCI[9] Express TurboCap card with a 10 Gbps[10] fiber optic Ethernet interface to the monitored network or a 1 Gbps copper or fiber optic Ethernet interface to the monitored network. These cards are Intel-based NICs that have been programmed with a custom firmware and are controlled by a specialized device driver designed to optimize

[6] Remote Procedure Call

[7] Internet Protocol

[8] Network Interface Controller

[9] Peripheral Component Interconnect

[10] Gigabit per second

high-speed packet capture and implement de-duplication. The Cascade Shark hardware is in the TOE environment.

1.4.1.2 Cascade Pilot

Pilot is the visualization software half of the TOE, whose main purpose is to provide a reporting GUI for Cascade Shark. Cascade Pilot provides the ability to display, drill down into, rewind, configure alerts on, and report network traffic captured and/or analyzed by one or more Shark appliances. Pilot users can apply network traffic analysis metrics (views) aimed at the visualization and analysis of capture statistics, create trigger-alert mechanisms on one or more Shark appliances, and create reports. Alerts are triggered based on network traffic analysis, not security-related events. Views can be applied to remote traffic sources, including capture jobs configured on Shark appliances and historical trace clips[11].

Users of Pilot can also use the software to configure “watches” and “event” triggers on any criteria available in the set of 200+ canned views. The views compute specific metrics, such as bandwidth over time, IP conversations or protocol distributions. The results of the views are displayed in the form of charts (strip charts, bar charts, grids, etc).

Cascade Pilot is software written to run in the Windows installation of the user’s workstation. Pilot functionality can be divided into two components: the GUI written in .NET and the Pilot Server written in C and C++. The Pilot Server provides functionality that is similar to the Shark Probe functionality of the Cascade Shark appliance, but this is not included in the evaluated configuration.

In the evaluated configuration, Cascade Pilot will be configured to connect to one or more Shark appliances. The Shark appliances perform live traffic analysis on entire remote networks or historical analysis on saved capture jobs, or subsets of those capture jobs exported to Pilot as trace clips.

The Cascade Pilot GUI interacts with the Shark appliance over an HTTPS connection. Users of Cascade Pilot are able to authenticate a Shark appliance by accepting the self-signed certificate of the appliance’s web server. In addition, users of Cascade Pilot authenticate themselves to Shark by providing a valid username and password.

[11] A trace clip represents user-defined time intervals within a job trace. A job trace represents the network traffic saved in the packet data storage.

1.4.2 Physical Boundaries

This section identifies the components of the product that are in the TOE. Section 1.3.1 identifies the hardware and software components that the TOE relies upon and that are part of the IT[12] environment.

There are no hardware components that are part of the TOE. The TOE runs on the hardware appliances listed in Section 1.3.1.

The following software components constitute the entire TOE:

- Cascade Shark v9.6 and Cascade Pilot v9.6 software
- FIPS 140-2 validated cryptographic module included with the Windows OS on the Cascade Pilot system

The following diagram shows the physical TOE boundary.

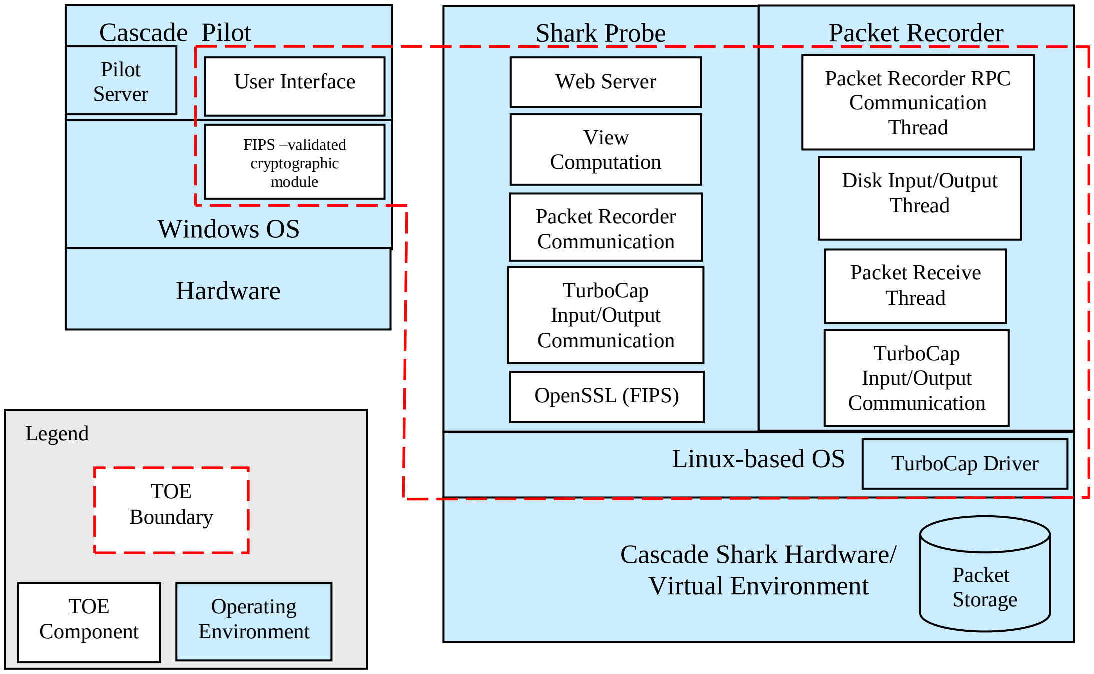

Figure 2 TOE Boundary

1.4.2.1 Guidance Documentation

The TOE includes the following guidance:

Cascade® Shark® Appliance User's Guide Version 9.6

[12] IT – Information Technology

Cascade® Shark Appliance Quick Start Guide Version 9.6
Cascade Shark Release Notes
Virtual Cascade® Shark® Appliance Quick Start Guide Version 9.6
Cascade Virtual Shark Release Notes
Cascade® Pilot Reference Manual Version 9.6
Cascade Pilot Release Notes
Cascade® Pilot Personal Edition Reference Manual Version 9.6
Cascade Pilot Personal Edition Release Notes

1.4.3 Logical Boundaries

The logical boundaries of the TOE include the security functions of the TOE interfaces. The TOE logically supports the following security functions:

- Security Audit
- Cryptographic Support
- User Data Protection
- Identification and Authentication
- Security Management
- Protection of the TOE Security Functions (TSF)
- Resource Utilization
- Network Performance Management

1.4.3.1 Security Audit

The TOE collects audit data on security-relevant user actions and provides an interface for reviewing the audit logs. Audit information generated by the system includes date and time of the event, user identifier (ID) that caused the event to be generated, computer where the event occurred, and other event-specific data. Shark provides a reliable time stamp, relying on the hardware appliance to include a hardware clock.

The Shark generates all audit records. All auditable actions/events performed by Pilot result in requests for services from Shark and Shark audits the action.

The Shark utilizes a Linux syslog system for all audit records. The syslog will contain all audit records produced by Shark and Pilot as described above. Audit records are saved in plain text files and are rotated when needed. The syslog is accessible to the Security Administrator (any administrator with isAdmin privileges) through a web interface.

1.4.3.2 Cryptographic Support

The cryptographic functions provided by the TOE include key generation, key zeroization, encryption/decryption, cryptographic signatures, cryptographic hashing, and keyed-hash message authentication. All cryptographic algorithm implementations have been validated by the NIST[13]-run Cryptographic Algorithm Validation Program (CAVP).

Shark includes a cryptographic module (OpenSSL Object Module 2.0) to implement the cryptographic functions on the Shark. All cryptographic functions in the OpenSSL Object Module are implemented in the TOE with the same executable module that was FIPS 140-2 validated, and per CMVP Implementation guidance G.5 maintains the FIPS 140-2 validation status in the TOE. For more information, see. http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm#1747 FIPS Certificate No. 1747. The OpenSSL Object Module used by the TOE was tested using the CascadeOS 6.1 (64-bit) on Intel Pentium T4200 (x86) processors.

[13] National Institute of Standards and Technology

Pilot uses a FIPS 140-2 validated cryptographic module that is part of the underlying Windows operating system on which the Pilot software operates to implement the cryptographic functions. Pilot executes on a system running either Windows XP, Windows Vista, or Windows 7 operating systems.

The cryptographic functions described above are used to protect all network communications between the Shark and Pilot components. All communications between the Shark and Pilot components are protected with TLS.

1.4.3.3 User Data Protection

Within the TOE, packets, traces, computed network metrics, capture jobs, and any saved reports are defined as user data. User data only flows to the Cascade Shark appliance or in some situations to the Cascade Pilot. User data does not flow through the Shark or Pilot.

The TOE provides the ability to control access to objects based on the user or group of the subject requesting access.

1.4.3.4 Identification and Authentication

The TOE requires users to provide unique identification and authentication (i.e.,ID and password) data before any administrative access to the TOE is granted.

The product supports both local and remote authentication via RADIUS[14] and TACACS+[15]. The local identification and authentication mechanism is based on usernames and passwords. In the evaluated configuration only local authentication is allowed.

Security Administrators can configure a password policy for the Shark, specifying the password complexity and composition requirements and the allowed number of failed authentication attempts before lockout occurs. In addition, passwords will expire after an administrator-configured expiration.

Security Administrators access the Shark appliance either via an SSH connection or directly at the console. On the Shark, Security Administrators are identified and authenticated by the underlying Linux-based operating system. Security Administrators and users that access the Shark appliance via a web-based browser are authenticated by the Shark software. The ability to access the Shark appliance via an SSH connection will be disabled in the evaluated configuration.

1.4.3.5 Security Management

The TOE is delivered with a Security Administrator role. The TOE must have at least one user with the Security Administrator role defined, so the last user with the Security Administrator role cannot be deleted. Granular role permissions can be assigned to groups and the users can be assigned to one or more groups.

The TOE implements a robust management system providing multiple administrative functions and restricting their use based on groups and privileges assigned to the groups. The TOE provides two graphical management interfaces: the Cascade Shark Web Interface and the Cascade Pilot user interface. The Cascade Shark Web Interface can be accessed from either Cascade Pilot or remotely from a web browser. Cascade Pilot displays data collected by remote Cascade Shark and local network traffic, but the collection of local network traffic on the Cascade Pilot is not included in the evaluated configuration.

The Cascade Shark Web Interface provides the capability to

- display the Shark appliance status, capture job status
- perform user/group management
- configure Shark capture board(s)
- add new protocol definitions/groups

[14] RADIUS – Remote Authentication Dial In User Service

[15] TACACS+ – Terminal Access Controller Access-Control System Plus

- retrieve/view Shark audit records.
- manage the Capture Jobs in the Jobs Repository[16]

The Cascade Pilot provides the ability to:

- display and analyze network traffic on data collected by a remote Shark appliance
- connect and manage one or more remote Shark appliances
- manage the list of keys used to decode encrypted traffic

The TOE also provides a CLI[17]. In the evaluated configuration, the CLI is accessible from the Shark local console. When accessing the TOE from the local console, a proprietary command line interface is provided. In the evaluated configuration, access to the shell provided by the underlying operating system is not allowed.

1.4.3.6 Protection of the TSF

The TOE provides protection mechanisms for its security functions. One of the protection mechanisms is that users must authenticate before any administrative operations can be performed on the system, whether those functions are related to the management of user accounts or the configuration of traffic analysis. Another protection mechanism is that all functions of the TOE are confined to the TOE itself. The TOE is completely self-contained and therefore maintains its own execution domain.

The TOE implements HTTPS for protection of the management user interfaces. HTTPS (SSL 3.1 / TLS 1.0) connections are used to protect all communication between Shark and Pilot. HTTPS protects data transfer and leverages cryptographic capabilities to prevent replay attacks. The management communication channels between the Shark and remote entity are distinct from other communication channels and provide assured identification of both endpoints. In addition, the communications are protected from modification and disclosure.

1.4.3.7 Resource Utilization

The TOE provides the ability to limit the amount of disk space used in storage for each capture job. A quota amount is assigned for each capture job.

1.4.3.8 Network Performance Management

The TOE collects traffic data on a network and performs analysis on the collected data. The collected data is analyzed against configured policies which allow the Security Administrator or user to perform flow analysis, packet analysis, and calculate performance metrics. The Shark Probe component is capable of performing analysis on both live and previously saved network traffic. The analysis functionality is regulated by “views”, which are the core analysis and visualization paradigm.

1.4.3.9 Physical/Logical Features and Functionality Not Included in the Evaluated Configuration of the TOE

The following features/functionality are not included in the evaluated configuration:

- User of external authentication servers (e.g., TACACS+ server, RADIUS server)
- Remote shell access (SSH)
- The capability for Pilot to apply live and historical views to local NICs and local trace files

[16] The Jobs Repository includes the name of each Job Trace in the appliance.

[17] Command Line Interface

2 Conformance Claims

This section provides the identification for any CC, Protection Profile (PP), and EAL package conformance claims. Rationale is provided for any extensions or augmentations to the conformance claims. Rationale for CC and PP conformance claims can be found in Section 8.1.

Table 3 CC and PP Conformance

Common Criteria (CC) Identification and Conformance	Common Criteria for Information Technology Security Evaluation, Version 3.1, [Revision 3], [July 2009]; CC Part 2 extended; CC Part 3 conformant; PP claim (none); Parts 2 and 3 Interpretations from the CEM as of 8/30/2011 were reviewed, and no interpretations apply to the claims made in this ST.
PP Identification	None[18]
Evaluation Assurance Level	EAL3 Augmented with Flaw Remediation (ALC_FLR.2)

[18] This ST is not claiming conformance with any Protection Profiles. However, this ST incorporates as many of the threats, assumptions, organizational security policies, security objectives, and security functional requirements defined in the Security Requirements for Network Devices, 10 December 2010, Version 1.0 (NDPP) with which the TOE defined in this ST complies.

3 Security Problem

The TOE security environment describes the security aspects of the intended environment in which the TOE is to be used and the manner in which it is expected to be employed. The statement of TOE security environment defines the following:

- Known and presumed threats countered by either the TOE or by the security environment
- Organizational security policies with which the TOE must comply
- Assumptions made on the operational environment and the method of use intended for the product

The TOE is intended to be used in environments where the TOE components can be physically protected from tampering and where necessary information will be available via other network components (e.g. routers).

3.1 Threats to Security

This section identifies the threats to the IT[19] assets against which protection is required by the TOE or by the security environment. The threat agents are divided into two categories:

- Attackers who are not TOE users: They have public knowledge of how the TOE operates and are assumed to possess a low skill level, limited resources to alter TOE configuration settings or parameters and no physical access to the TOE.
- TOE users: They have extensive knowledge of how the TOE operates and are assumed to possess a high skill level, moderate resources to alter TOE configuration settings or parameters and physical access to the TOE. (TOE users are, however, assumed not to be willfully hostile to the TOE.)

Both are assumed to have a low level of motivation. The IT assets requiring protection are the TSF[20] and user data saved on or transitioning through the TOE and the hosts on the protected network. Removal, diminution and mitigation of the threats are through the objectives identified in Section 4 Security Objectives. The following threats are applicable:

Table 4 Threats

Name	Description
T.ADMIN_ERROR	A Security Administrator may incorrectly install or configure the TOE resulting in ineffective security mechanisms.
T.FAIL_NETANAL	The TOE may fail to identify the network traffic flow conditions as requested by the Security Administrator.
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.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

[19] IT – Information Technology

[20] TSF – TOE Security Functionality

Name	Description
	identification and authentication data.
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.

3.2 Organizational Security Policies

An Organizational Security Policy (OSP) is a set of security rules, procedures, or guidelines imposed by an organization on the operational environment of the TOE. There are no OSPs defined for this ST.

3.3 Assumptions

This section describes the security aspects of the intended environment for the evaluated TOE. The operational environment must be managed in accordance with assurance requirement documentation for delivery, operation, and user guidance. The following specific conditions are required to ensure the security of the TOE and are assumed to exist in an environment where this TOE is employed.

Table 5 Assumptions

Name	Description
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 Security Administrators are trusted to follow and apply all Security Administrator guidance in a trusted manner.

4 Security Objectives

This section defines the security objectives of the TOE and its supporting environment. Security objectives, categorized as either IT security objectives or non-IT security objectives, reflect the stated intent to counter identified threats and/or comply with any organizational security policies identified. All of the identified threats and organizational policies are addressed under one of the categories below.

4.1 Security Objectives for the TOE

The specific security objectives for the TOE are as follows:

Table 6 Security Objectives for the TOE

Name	Description
O.ANALYZE	The TOE will apply analytical processes and information to derive conclusions about the network (past, present, or future).
O.PROTECTED_COMMUNICATIONS	The TOE will provide protected communication channels for Security Administrators and users, other parts of a distributed TOE, and authorized IT entities.
O.RESOURCE_AVAILABILITY	The TOE shall provide mechanisms that mitigate user attempts to exhaust TOE resources (e.g., persistent storage).
O.SCAN	The TOE will collect network traffic information from the network interface card.
O.SYSTEM_MONITORING	The TOE will provide the capability to generate audit data.
O.TOE_ADMINISTRATION	The TOE will provide mechanisms to ensure that only Security Administrators are able to log in and configure the TOE, and provide protections for logged-in Security Administrators and users.
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

4.2.1 IT Security Objectives

The specific security objectives for the operational environment of the TOE are as follows:

Table 7 IT Security Objectives

Name	Description
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.

4.2.2 Non-IT Security Objectives

The following non-IT environment security objectives are to be satisfied without imposing technical requirements on the TOE. That is, they will not require the implementation of functions in the TOE hardware and/or software. Thus, they will be satisfied largely through application of procedural or administrative measures.

Table 8 Non-IT Security Objectives

Name	Description
OE.TRUSTED_ADMIN	TOE Security Administrators are trusted to follow and apply all Security Administrator guidance in a trusted manner.
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.

5 Extended Components

This section defines the extended SFRs and extended SARs met by the TOE. These requirements are presented following the conventions identified in Section 6.

5.1 Extended TOE Security Functional Components

This section specifies the extended SFRs for the TOE. The extended SFRs are organized by class. Table 9 identifies all extended SFRs implemented by the TOE

Table 9 Extended TOE Security Functional Requirements

Name	Description
FCS_CKM_EXT.4	Cryptographic Key Zeroization
FCS_TLS_EXT.1	Extended:: TLS
FCS_HTTPS_EXT.1	Extended: HTTPS
FIA_UAU_EXT.5	Extended: Password-based Authentication Mechanism
FPT_PTD_EXT.1	Extended: Management of TSF Data
FPT_TST_EXT.1	Extended: TSF testing
NPM_SDC_EXT.1	System data collection
NPM_ANL_EXT.1	Analysis

5.1.1 Class FCS: Cryptographic Support

Families in this class address the requirements for functions to implement cryptographic functionality as defined in CC Part 2.

5.1.1.1 Family FCS_CKM_EXT[21]: Extended: Cryptographic Key Management

Family Behaviour

A cryptographic key must be managed throughout its life cycle. This family is intended to support that lifecycle and consequently defines requirements for the following activities: cryptographic key generation, cryptographic key distribution, cryptographic key access and cryptographic key destruction. This family should be included whenever there are functional requirements for the management of cryptographic keys.

Components in this family address the requirements for managing cryptographic keys as defined in CC Part 2. This section defines the extended components for the FCS_CKM_EXT family and is considered to be part of the CC Part 2 FCS_CKM family.

Component Leveling

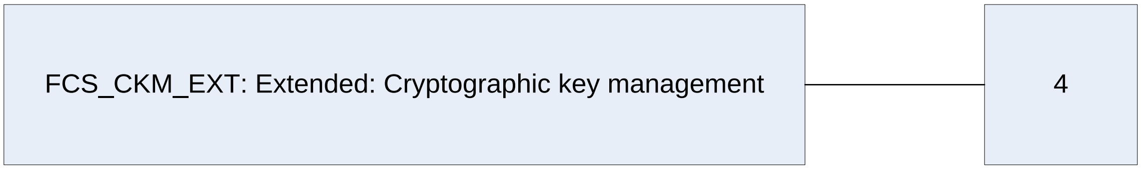

Figure 3 Extended: Cryptographic key management family decomposition

The extended FCS_CKM_EXT.4 component is considered to be part of the CC Part 2 FCS_CKM family.

FCS_CKM_EXT.4 Cryptographic key zeroization, requires cryptographic keys and cryptographic critical security parameters to be zeroized. It was modeled after FCS_CKM.4.

Management: FCS_CKM_EXT.4

a) There are no management activities foreseen.

Audit: FCS_CKM_EXT.4

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

a) Minimal: Failure on invoking the cryptographic key zeroization functionality.

FCS_CKM_EXT.4 Cryptographic Key Zeroization

Hierarchical to: No other components

FCS_CKM_EXT.4.1

The TSF shall zeroize all plaintext secret and private cryptographic keys and CSP[22]s when no longer required.

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]

[21] FCS_CKM_EXT is considered to be included in the existing CC Part 2 FCS_CKM family. The "EXT" defines this is an extended component only and is included in the component leveling diagram and family name for consistency purposes.

[22] Critical Security Parameters

5.1.1.2 Family FCS_TLS_EXT: Extended: TLS

Family Behaviour

Components in this family address the requirements for protecting communications using TLS. This is a new family defined for the FCS Class and is modeled after the CC Part 2 FCS_CKM family.

Component Leveling

Figure 4 Extended: TLS family decomposition

FCS_TLS_EXT.1 Extended: TLS, requires that TLS be implemented. It was modeled after the CC Part 2 FCS_CKM.1 component.

Management: FCS_TLS_EXT.1

There are no management activities foreseen.

Audit: FCS_TLS_EXT.1

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

a) Minimal: Failure to establish a TLS Session
b) Basic: Establishment/Termination of a TLS session.

FCS_TLS_EXT.1 Extended: TLS

Hierarchical to: No other components

FCS_TLS_EXT.1.1

The TSF shall implement one or more of the following protocols [selection: TLS 1.0 (RFC 2346), 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
Optional Ciphersuites:
[selection:
None
TLS_RSA_WITH_3DES_EDE_CBC_SHA
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
].

Dependencies: FCS_COP.1 Cryptographic operation

5.1.1.3 Family FCS_HTTPS_EXT: Extended: HTTPS

Family Behaviour

Components in this family address the requirements for protecting communications using HTTPS. This is a new family defined for the FCS Class.

Component Leveling

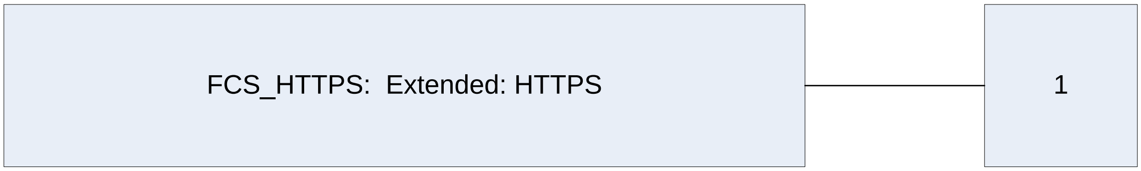

Figure 5 Extended: HTTPS family decomposition

FCS_HTTPS_EXT.1 Extended: HTTPS, requires that HTTPS be implemented.

Management: FCS_HTTPS_EXT.1

a) There are no management activities foreseen.

Audit: FCS_ HTTPS_EXT.1

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

a) There are no auditable events foreseen.

FCS_HTTPS_EXT.1 Extended: HTTPS

Hierarchical to: No other components

FCS_HTTPS_EXT.1.1

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

FCS_HTTPS_EXT.1.2

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

Dependencies: FCS_TLS_EXT.1 Extended: TLS

5.1.2 Class FIA: Identification and Authentication

Families in this class address the requirements for functions to establish and verify a claimed user identity as defined in CC Part 2.

5.1.2.1 Family FIA_UAU_EXT[23]: Extended: User authentication

Family Behaviour

This family defines the types of user authentication mechanisms supported by the TSF.

This section defines the extended components for the FIA_UAU_EXT family and is modeled after CC Part 2 FIA_UAU family.

Component Leveling

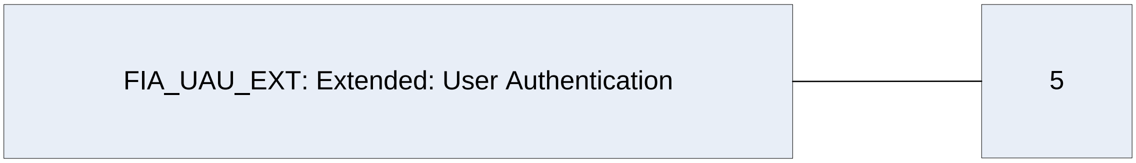

Figure 6 Extended: User Authentication family decomposition

The extended FIA_UAU_EXT.5 component is considered to be part of the FIA_UAU family as defined in CC Part 2.

FIA_UAU_EXT.5 Extended: Password-based Authentication Mechanism, requires a local password-based authentication mechanism and the capability for passwords to expire. In addition, other authentication mechanisms can be specified. It was modeled after the CC Part 2 FIA_UAU.5 component.

Management: FIA_UAU_EXT.5

The following actions could be considered for the management functions in FMT:

a) reset a user password by an administrator;
b) management of the authentication mechanisms;
c) management of the rules for authentication if multiple authentication mechanisms are provided.

Audit: FIA_UAU_EXT.5

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

a) Minimal: Unsuccessful use of the authentication mechanism;
b) Basic: All use of the authentication mechanisms.

FIA_UAU_EXT.5 Extended: Password-based Authentication Mechanism

Hierarchical to: No other components

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.

[23] FIA_UAU_EXT is considered to be part of the existing CC Part 2 FIA_UAU family. The "EXT" defines this is an extended component only and is included in the component leveling diagram and family name for consistency purposes.

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].

Dependencies: No dependencies.

5.1.3 Class FPT: Protection of the TSF

Families in this class address the requirements for functions providing integrity and management of mechanisms that constitute the TSF and of the TSF data as defined in CC Part 2.

5.1.3.1 Family FPT_PTD_EXT: Extended: Management of TSF Data

Family Behaviour

Components in this family address the requirements for managing and protecting TSF data, such as passwords and keys. This is a new family defined for the FPT Class and is modeled after the CC Part 2 FPT_PHP family.

Component Leveling

FPT_PTD_EXT: Extended: Management of TSF Data
1

Figure 7 Extended: Management of TSF Data family decomposition

FPT_PTD_EXT.1 Extended: Management of TSF Data, requires preventing selected TSF data from being read by any user or subject. It was modeled after the CC Part 2 FPT_PHP.1 component.

Management: FPT_PTD_EXT.1

a) There are no management activities foreseen.

Audit: FPT_PTD_EXT.1

a) There are no auditable activities foreseen.

FPT_PTD_EXT.1 Extended: Management of TSF Data
Hierarchical to: No other components
FPT_PTD_EXT.1.1
The TSF shall prevent reading of [assignment: *TSF data*].
Dependencies: No dependencies.

5.1.3.2 Family FPT_TST_EXT[24]: Extended: TSF self test

Family Behaviour

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

The extended FPT_TST_EXT.1 component is considered to be part of the FPT_TST_EXT family, and is considered part of the CC Part 2 FPT_TST family.

Component Leveling

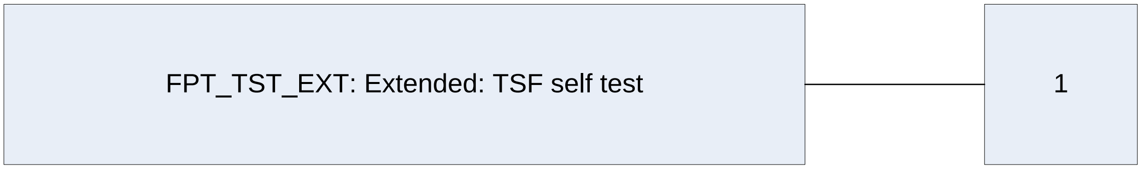

Figure 8 TSF testing family decomposition

FPT_TST _EXT.1 Extended: TSF testing, requires a suite of self tests to be run during initial start-up in order to demonstrate correct operation of the TSF. It was modeled after FPT_TST.1.

Management: FPT_TST _EXT.1

a) management of the conditions under which TSF self testing occurs, such as during initial start-up, regular interval, or under specified conditions.

Audit: FPT_TST _EXT.1

a) Minimal: Indication that TSF self-test was completed.

FPT_TST_EXT.1 TSF testing
Hierarchical to: No other components
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.

Dependencies: No dependencies.

[24] FPT_TST_EXT is considered to be included in the existing CC Part 2 FPT_TST family. The "EXT" defines this is an extended component only and is included in the component leveling diagram and family name for consistency purposes.

5.1.4 Class NPM: Network Performance Management

Families in this class address the requirements for functions to implement network performance functionality as defined in CC Part 2.

Network Performance Management functions involve the collection of network packet data and the analysis of this collected data. The NPM: Network Performance Management function class was modeled after the CC FAU: Security audit class. The extended family and related components for NPM_SDC_EXT: System data collection was modeled after the CC family and related components for FAU_GEN: Security audit data generation. The extended family NPM_ANL_EXT: Analysis was modeled after the family FAU_SAA: Potential violation analysis.

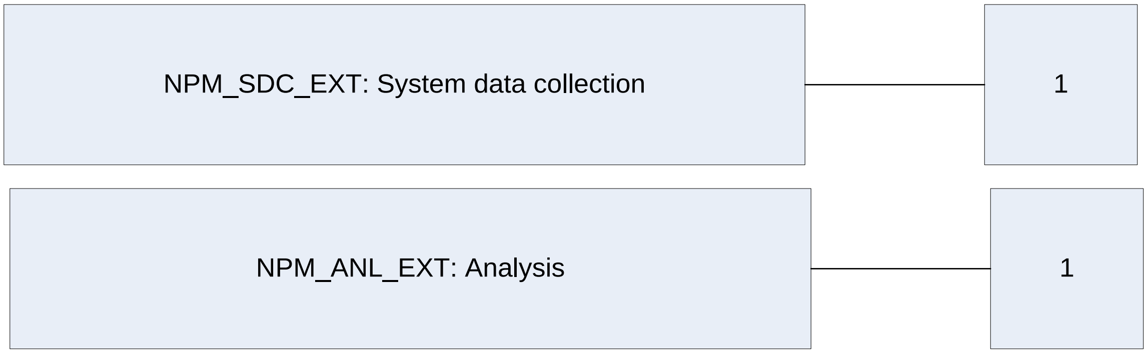

Figure 9 NPM: Network Performance Management Function Class Decomposition

5.1.4.1 Family NPM_SDC_EXT: System data collection

Family Behaviour

This family defines the requirements for recording the occurrence of network performance management events that take place under TSF control. This family identifies the level of system data collection, enumerates the types of events that shall be collected by the TSF, and identifies the minimum set of related information that should be provided within various network performance management event record types.

Component Leveling

Figure 10 System data collection family decomposition

NPM_SDC_EXT.1 System data collection, defines the level of NPM events, and specifies the list of data that shall be recorded in each record. It was modeled after the CC Part 2 FAU_GEN.1 component.

Management: NPM_SDC_EXT.1

a) There are no management activities foreseen.

Audit: NPM_SDC_EXT.1

a) There are no auditable events foreseen.

NPM_SDC_EXT.1 System data collection
Hierarchical to: No other components
NPM_SDC_EXT.1.1

The TSF shall be able to collect the following information from the targeted IT System resource(s):
[assignment: *data accesses, service requests, network traffic, security configuration changes, attempts to breach IPS policy; and no other events*.]

NPM_SDC_EXT.1.2

At a minimum, the TSF shall collect and record the following information:

a) Date and time of the event, type of event, and subject identity.

Dependencies: FPT_STM.1 Reliable time stamps

5.1.4.2 Family NPM_ANL_EXT: Analysis

Family Behaviour

This family defines the analysis the TOE performs on the collected network packet data. This family enumerates the types of analytical functions that shall be executed on the data collected.

Component Leveling

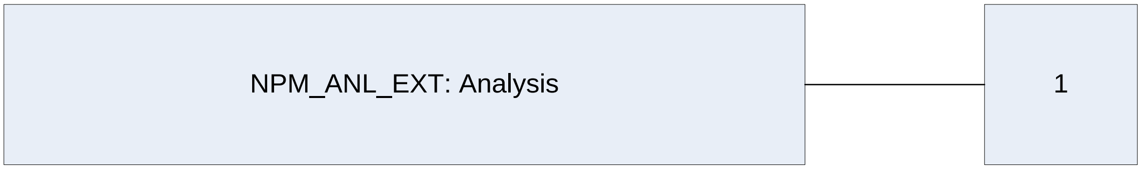

Figure 11 Analysis family decomposition

NPM_ANL_EXT.1 analysis, specifies the list of analyses the TOE will perform on the collected application data. It was modeled after the CC Part 2 FAU_SAA.4 component.

Management: NPM_ANL_EXT.1

a) Maintenance of the analysis functions by (adding, modifying, deletion) of policies from the set of policies.

Audit: NPM_ANL_EXT.1

a) Minimal: Enabling and disabling of any of the analysis mechanisms.

NPM_ANL_EXT.1 Analysis
Hierarchical to: No other components
NPM_ANL_EXT.1.1

The TSF shall perform the following analysis function(s) on all packet data collected:

a) [*assignment: analytical functions*.]

Dependencies: NPM_SDC_EXT.1

5.2 Extended TOE Security Assurance Components

There are no extended TOE security assurance components defined for this evaluation.

6 Security Requirements

This section defines the SFRs and SARs met by the TOE.

6.1 Conventions

There are several font variations used within this ST. Selected presentation choices are discussed here to aid the Security Target reader.

The CC allows for assignment, refinement, selection and iteration operations to be performed on security functional requirements. All of these operations are used within this ST. These operations are performed as described in Part 2 of the CC, and are shown as follows:

- Completed assignment statements are identified using [*italicized text within brackets*].
- Completed selection statements are identified using [*underlined italicized text within brackets*].
- Refinements are identified using **bold text**. Any text removed is stricken (Example: ~~TSF Data~~) and should be considered as a refinement.
- Extended Functional and Assurance Requirements are identified using "EXT_" at the beginning of the short name.
- Iterations are identified by appending a number in parentheses following the component title. For example, FAU_GEN.1(1) Audit Data Generation would be the first iteration and FAU_GEN.1(2) Audit Data Generation would be the second iteration.

6.2 Security Functional Requirements

This section specifies the SFRs for the TOE. This section organizes the SFRs by CC class. Table 10 identifies all SFRs implemented by the TOE and indicates the ST operations performed on each requirement.

Table 10 TOE Security Functional Requirements

Name	Description	S	A	R	I
FAU_GEN.1	Audit data generation	✓	✓		
FAU_GEN.2	User identity association				
FAU_SAR.1	Audit review		✓		
FCS_CKM.1	Cryptographic key generation		✓	✓	
FCS_CKM_EXT.4	Extended: Cryptographic Key Zeroization				
FCS_COP.1(1)	Cryptographic operation (for data encryption/decryption)		✓		✓
FCS_COP.1(2)	Cryptographic operation (for cryptographic signature)		✓	✓	✓
FCS_COP.1(3)	Cryptographic operation (for cryptographic hashing)		✓	✓	✓
FCS_COP.1(4)	Cryptographic operation (for keyed-hash message authentication)		✓	✓	✓
FCS_HTTPS_EXT.1	Extended: HTTPS				
FCS_TLS_EXT.1	Extended: TLS	✓			
FDP_ACC.1	Subset Access Control		✓		
FDP_ACF.1	Security Attribute Based Access Control		✓		
FIA_AFL.1	Authentication failure handling	✓	✓		

Name	Description	S	A	R	I
FIA_ATD.1	User attribute definition		✓		
FIA_SOS.1	Verification of secrets		✓	✓	
FIA_UAU.2	User authentication before any action				
FIA_UAU_EXT.5	Extended: Password-based Authentication Mechanism	✓			
FIA_UAU.6	Re-authenticating		✓		
FIA_UAU.7	Protected Authentication Feedback		✓		
FIA_UID.2	User identification before any action				
FMT_MSA.1	Management of security attributes	✓	✓		
FMT_MSA.3	Static attribute initialization	✓	✓		
FMT_MTD.1	Management of TSF data	✓	✓		
FMT_SMF.1	Specification of management functions		✓		
FMT_SMR.1	Security roles		✓		
FPT_ITT.1(1)	Basic Internal TSF Data Transfer Protection (Disclosure)	✓		✓	✓
FPT_ITT.1(2)	Basic Internal TSF Data Transfer Protection (Modification)	✓		✓	✓
FPT_PTD_EXT.1(1)	Extended: Management of TSF Data (for reading of authentication data)		✓		✓
FPT_PTD_EXT.1(2)	Extended: Management of TSF Data (for reading of all symmetric keys)		✓		✓
FPT_RPL.1	Replay Detection		✓		
FPT_STM.1	Reliable Time Stamps			✓	
FPT_TST_EXT.1	Extended: TSF testing				
FRU_RSA.1	Maximum Quotas	✓	✓		
FTP_ITC.1(1)	Inter-TSF Trusted Channel (Prevention of Disclosure)	✓	✓	✓	
FTP_ITC.1(2)	Inter-TSF Trusted Channel (Detection of Modification)	✓	✓	✓	
FTP_TRP.1(1)	Trusted Path (Prevention of Disclosure)	✓	✓	✓	✓
FTP_TRP.1(2)	Trusted Path (Detection of Modification)	✓	✓	✓	✓
NPM_SDC_EXT.1	Extended: System data collection		✓		
NPM_ANL_EXT.1	Extended: Analysis		✓		

Note: S=Selection; A=Assignment; R=Refinement; I=Iteration

6.2.1 Class FAU: Security Audit

FAU_GEN.1 Audit Data Generation
Hierarchical to: No other components.

FAU_GEN.1.1

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

a) Start-up and shutdown of the audit functions;
b) All auditable events, for the [not specified] level of audit; and
c) [*All administrative actions;*
d) *Specifically defined auditable events listed in Table 11*].

FAU_GEN.1.2

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

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and
b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [*no additional information*].

Dependencies: FPT_STM.1 Reliable time stamps

Table 11 TOE Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	
FAU_GEN.2	None.	
FAU_SAR.1	Reading of information from the audit records.	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.
FCS_COP.1(3)	Failure on invoking functionality.	No additional information.
FCS_COP.1(4)	Failure on invoking functionality.	No additional information.
FDP_ACC.1	None.	
FDP_ACF.1	All requests to perform an operation on an object covered by the SFP.	No additional information.
FIA_SOS.1	None.	
FIA_AFL.1	The reaching of the threshold for the unsuccessful authentication attempts and the actions take and the subsequent, if appropriate restoration to the normal state.	No additional information.
FIA_UAU.2	All use of the Shark user authentication mechanism.	Provided user identity, origin of the attempt (e.g., IP address).
FIA_UAU_EXT.5	All use of the Shark authentication mechanism.	Origin of the attempt (e.g., IP address).

Requirement	Auditable Events	Additional Audit Record Contents
FIA_UAU.6	Attempt to re-authenticate via the Shark.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None.	
FIA_UID.2	All use of the Shark user identification mechanism.	Provided user identity, origin of the attempt (e.g., IP address).
FMT_MTD.1	None.	
FMT_SMF.1	None.	
FMT_SMR.1	None.	
FPT_ITT.1(1)	None.	
FPT_ITT.1(2)	None.	
FPT_PTD_EXT.1(1)	None.	
FPT_PTD_EXT.1(2)	None.	
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".
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_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.

FAU_GEN.2 User Identity Association

Hierarchical to: No other components.

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.

Dependencies: FAU_GEN.1 Audit data generation
FIA_UID.1 Timing of identification

FAU_SAR.1 Audit review

Hierarchical to: No other components.

FAU_SAR.1.1

The TSF shall provide [*Security Administrators*] with the capability to read [*all audit information*] from the audit records.

FAU_SAR.1.2

The TSF shall provide the audit records in a manner suitable for the user to interpret the information.

Dependencies: FAU_GEN.1 Audit data generation

6.2.2 Class FCS: Cryptographic Support

FCS_CKM.1 Cryptographic Key Generation (for asymmetric keys)

Hierarchical to: No other components.

FCS_CKM.1.1

The TSF shall generate **asymmetric** cryptographic keys in accordance with a ~~specified cryptographic key generation algorithm~~ **domain parameter generator and a random number generator** ~~and specified cryptographic key sizes~~ that meet the following:.[

a) Key generated shall provide a minimum of 112 bits of encryption strength.

b) Generated 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"*
- *Used for cryptographic public/private key pair generation.*]

Dependencies: [FCS_CKM.2 Cryptographic key distribution, or
FCS_COP.1 Cryptographic operation]
FCS_CKM.4 Cryptographic key destruction

FCS_CKM_EXT.4 Cryptographic Key Zeroization

Hierarchical to: No other components.

FCS_CKM_EXT.4.1

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

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_COP.1(1) Cryptographic Operation (for data encryption/decryption)

Hierarchical to: No other components.

FCS_COP.1.1(1)

The TSF shall perform [*encryption and decryption*] in accordance with a specified cryptographic algorithm [*AES operating in ECB, CBC, CFB8, CFB128, OFB modes, and 3DES operating in EDE, CBC modes*] and cryptographic key sizes [*128-bits, 256-bits, and 192 bits (AES), 168-bits (3DES)*] that meet the following: [

- *FIPS PUB 197, "Advanced Encryption Standard (AES)"*
- *FIPS PUB 46-3, "Data Encryption Standard (DES)"*
- *NIST SP800-67 Revision 1*
- *NIST SP800-38A*]

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.4 Cryptographic key destruction

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

Hierarchical to: No other components.

FCS_COP.1.1(2)

The TSF shall perform [*cryptographic signature services*] in accordance with a specified cryptographic algorithm [*Rivest Shamir Adleman (RSA) with a key size (modulus) of 2048 bits*] ~~and cryptographic key sizes~~ that meet the following: [*FIPS PUB 186-3, "Digital Signature Standard" , FIPS PUB 186-2, "Digital Signature Standard"*].

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.4 Cryptographic key destruction

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

Hierarchical to: No other components.

FCS_COP.1.1(3)

The TSF shall perform [*cryptographic hashing services*] in accordance with a specified cryptographic algorithm [*SHS*] and ~~cryptographic key~~ **message digest** sizes [*160, 256, 384, 512 bits*] that meet the following: [*FIPS PUB 180-3, "Secure Hash Standard"*].

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.4 Cryptographic key destruction

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

Hierarchical to: No other components.

FCS_COP.1.1(4)

The TSF shall perform [*keyed-hash message authentication*] in accordance with a specified cryptographic algorithm [*HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512*], ~~and~~ cryptographic key sizes [*160, 256, 384, 512 bits*]**, and message digest sizes 160, 256, 384, 512 bits** that meet the following: [*FIPS PUB 198-1, "The Keyed-Hash Message Authentication Code", and FIPS-PUB 180-3, "Secure Hash Standard"*].

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.4 Cryptographic key destruction

FCS_HTTPS_EXT.1 Extended: HTTPS

Hierarchical to: No other components.

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.

Dependencies: FCS_TLS_EXT.1 Extended: TLS

FCS_TLS_EXT.1 Extended: TLS

Hierarchical to: No other components.

FCS_TLS_EXT.1.1

The TSF shall implement one or more of the following protocols [TLS 1.0 (RFC 2346)] supporting the following ciphersuites:
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
[
TLS_RSA_WITH_3DES_EDE_CBC_SHA
].

Dependencies: FCS_COP.1(2) Cryptographic Operation (for cryptographic signature)

6.2.3 Class FDP: User Data Protection

FDP_ACC.1 Subset access control
Hierarchical to: No other components.

FDP_ACC.1.1
The TSF shall enforce the [*Cascade Resource SFP*] on [
subjects: users of the management interface,
objects: files, views; and,
operations: read, modify, delete, take ownership, share a resource].

Dependencies: FDP_ACF.1 Security attribute based access control

FDP_ACF.1 Security attribute based access control
Hierarchical to: No other components.

FDP_ACF.1.1
The TSF shall enforce the [*Cascade Resource SFP*] to objects based on the following:
[a. *subject security attributes:*
- *user name*
- *group name*
- *privileges*

b. *object security attributes:*
- *owner*
- *share with list for views]*.

FDP_ACF.1.2
The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [
- *the object owner can read, modify, or delete the file or view*
- *the object owner can share a file or view with the groups of which the user is a member*
- *the object owner can share a view with any group if the owner has the CanShareViews privilege*
- *for files, members of the group which owns the file can read, modify, or delete the file*
- *for files, members of the group which owns the file can take ownership of the file by unsharing the file with the group.*
- *for views, the user or group with which the view was shared can read the view.*]

FDP_ACF.1.3
The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: [*access is granted if the user is a member of a group that has the IsAdmin privilege*].

FDP_ACF.1.4
The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [*no rules*].

Dependencies: FDP_ACC.1 Subset access control
FMT_MSA.3 Static attribute initialisation

6.2.4 Class FIA: Identification and Authentication

FIA_AFL.1 Authentication failure handling
Hierarchical to: No other components.

FIA_AFL.1.1

The TSF shall detect when [an administrator configurable positive integer within *the range of 0 to 10*[25]] unsuccessful authentication attempts occur related to [*user's attempts to use a management interface*].

FIA_AFL.1.2

When the defined number of unsuccessful authentication attempts has been [met], the TSF shall [*disable the user until enabled by a Security Administrator*].

Dependencies: FIA_UAU.1 Timing of authentication

FIA_ATD.1 User attribute definition
Hierarchical to: No other components.

FIA_ATD.1.1

The TSF shall maintain the following list of security attributes belonging to individual users: [*user identity, group identity, authentication data, and privilege*].

Dependencies: No dependencies.

FIA_SOS.1 Verification of secrets[26]
Hierarchical to: No other components.

FIA_SOS.1.1

The TSF shall provide a mechanism to verify that secrets meet [*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.*

[25] A value of zero (0) means that the user account will never be disabled.

[26] This SFR requires the same functionality of that defined in NDPP FIA_PMG_EXT.1.

5. *New passwords must ~~contain a minimum of 4 character changes from the previous password~~* **be different than the last N passwords where N is 0 - 12.**

].

Dependencies: No dependencies.

FIA_UAU.2 User authentication before any action[27]

Hierarchical to: FIA_UAU.1 Timing of authentication.

FIA_UAU.2.1

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

Dependencies: FIA_UID.1 Timing of identification.

FIA_UAU_EXT.5 Extended: Password-based Authentication Mechanism

Hierarchical to: No other components.

FIA_UAU_EXT.5.1

The TSF shall provide a local password-based authentication mechanism, [none] 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].

Dependencies: No dependencies.

FIA_UAU.6 Re-authenticating

Hierarchical to: No other components.

FIA_UAU.6.1

The TSF shall re-authenticate the user under the conditions *[when the user changes their password and at the next login after password expiration*].

Dependencies: No dependencies.

FIA_UAU.7 Protected Authentication Feedback

Hierarchical to: No other components.

FIA_UAU.7.1[28]

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

Dependencies: FIA_UAU.1 Timing of authentication.

FIA_UID.2 User identification before any action

Hierarchical to: FIA_UID.1 Timing of identification.

FIA_UID.2.1

[27] The statement of the SFR in the NDPP only applies to local console authentication; this SFR applies to local console and web interfaces.

[28] In the NDPP, this requirement was limited to the local console. This requirement is more restrictive because it applies to all administrative interfaces.

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

Dependencies: No dependencies

6.2.5 Class FMT: Security Management

FMT_MSA.1 Management of security attributes
Hierarchical to: No other components.

FMT_MSA.1.1

The TSF shall enforce the [*Cascade Resource SFP*] to restrict the ability to [*query, modify*] the security attributes [*user group, user privileges, object owner, share with list*] to [*Security Administrators for all security attributes; object owners can change the object owner and share with list*].

Dependencies: [FDP_ACC.1 Subset access control or
FDP_IFC.1 Subset information flow control]
FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_MSA.3 Static attribute initialisation
Hierarchical to: No other components.

FMT_MSA.3.1

The TSF shall enforce the [*Cascade Resource SFP*] to provide [restrictive] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2

The TSF shall allow the [*Security Administrator*] to specify alternative initial values to override the default values when an object or information is created.

Dependencies: FMT_MSA.1 Management of security attributes
FMT_SMR.1 Security roles

FMT_MTD.1 Management of TSF data
Hierarchical to: No other components.

FMT_MTD.1.1

The TSF shall restrict the ability to [manage] the [*TSF data*] to [*Security Administrators and users as detailed in Table 12 below*].

Table 12 Management of TSF Data

Operation	TSF Data	Users with the following privileges
Create (apply)	Views applied to trace files, capture jobs, trace clips	CanApplyViewsOnFile
Create (apply)	Views to any interface	CanApplyViewsOnInterfaces
Create, delete, modify, *start, stop*	Jobs	CanCreateJobs
query, modify, delete	Mounted folders (file system folders on the	CanAccessProbeFiles

Operation	TSF Data	Users with the following privileges
	Shark and jobs repository virtual folder)	
Create	Trace file	CanCreateFiles
Send from Pilot	Trace file	CanImportFiles
Create, modify, delete	Watch on a view	CanScheduleWatches
Send to Pilot	Trace clip, capture job, and file	CanExportFiles
Share	Views	CanShareViews

Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles

FMT_SMF.1 Specification of Management Functions
Hierarchical to: No other components.

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

- *Manage TSF Data as specified in FMT_MTD.1.*
- *Manage and initialize security attributes*.
- *Ability to configure the cryptographic functionality.*]

Dependencies: No Dependencies

FMT_SMR.1 Security roles
Hierarchical to: No other components.

FMT_SMR.1.1
The TSF shall maintain the roles [*Security Administrator, users without administrator privileges*].

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

Dependencies: FIA_UID.1 Timing of identification

6.2.6 Class FPT: Protection of the TSF

FPT_ITT.1(1) Basic Internal TSF Data Transfer Protection (Disclosure)
Hierarchical to: No other components.

FPT_ITT.1.1(1)
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: TLS, HTTPS**.

Dependencies: No dependencies

FPT_ITT.1(2) Basic Internal TSF Data Transfer Protection (Modification)
Hierarchical to: No other components.

FPT_ITT.1.1(2)

The TSF shall ~~protect detect TSF data from~~ **detect** [modification] **of TSF data** when it is transmitted between separate parts of the TOE **through the use of the TSF-provided cryptographic services: TLS, HTTPS**.

Dependencies: No dependencies

FPT_PTD_EXT.1(1) Extended: Management of TSF Data (for reading authentication data)

Hierarchical to: No other components.

FPT_PTD_EXT.1.1(1)

The TSF shall prevent reading of [*plaintext passwords*].

Dependencies: No dependencies

FPT_PTD_EXT.1(2) Extended: Management of TSF Data (for reading of all symmetric keys)

Hierarchical to: No other components.

FPT_PTD_EXT.1.1(2)

The TSF shall prevent reading of [*all pre-shared keys, symmetric key, and private keys*].

Dependencies: No dependencies

FPT_RPL.1 Replay Detection

Hierarchical to: No other components.

FPT_RPL.1.1

The TSF shall detect replay for the following entities: [*network TLS packets terminated at the TOE*].

FPT_RPL.1.2

The TSF shall perform: [*reject the data*] when replay is detected.

Dependencies: No dependencies

FPT_STM.1 Reliable time stamps

Hierarchical to: No other components.

FPT_STM.1.1

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

Dependencies: No dependencies

FPT_TST_EXT.1 Extended: TSF Testing

Hierarchical to: No other components.

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.

Dependencies: No dependencies

6.2.7 Class FRU: Resource Utilization

FRU_RSA.1 Maximum Quota

Hierarchical to: No other components.

FRU_RSA.1.1

The TSF shall enforce maximum quotas of the following resources: [*storage size for packet captures by capture jobs*] that [subjects] can use [over a specified period of time].

Dependencies: No dependencies

6.2.8 Class FTP: Trusted Path/Channels

FTP_ITC.1(1) Inter-TSF Trusted Channel (Prevention of Disclosure)

Hierarchical to: No other components.

FTP_ITC.1.1(1)

The TSF shall **use TLS or HTTPS to** provide a communication channel between itself and ~~another trusted~~ **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 ~~modification or~~ disclosure.

FTP_ITC.1.2(1)

The TSF shall permit [*the TSF*, ***or the*** *~~another trusted~~* ***authorized*** *IT* ***entities*** *~~product~~*] 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, communications between TOE components and external authorized IT entities*].

Dependencies: No dependencies

FTP_ITC.1(2) Inter-TSF Trusted Channel (Detection of Modification)

Hierarchical to: No other components.

FTP_ITC.1.1(2)

The TSF shall **use TLS or HTTPS in providing** ~~provide~~ a **trusted** communication channel between itself and ~~another trusted~~ **authorized** IT ~~product~~ **entities** that is logically distinct from other communication channels and provides assured identification of its end points and ~~protection of the channel data from~~ **detection of the** modification **of data** ~~or disclosure~~.

FTP_ITC.1.2(2)

The TSF shall permit [*the TSF*, ***or the*** *~~another trusted~~* ***authorized*** *IT* ***entities*** *~~product~~*] 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, communications between TOE components and external authorized IT entities*].

Dependencies: No dependencies

FTP_TRP.1(1) Trusted Path (Prevention of Disclosure)

Hierarchical to: No other components.

FTP_TRP.1.1(1)

The TSF shall provide a communication path between itself and [remote] ~~users~~ **administrators using TLS or HTTPS** 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 ~~*users*~~ ***administrators***] to initiate communication via the trusted path.

FTP_TRP.1.3(1)

The TSF shall require the use of the trusted path for [*all remote administrative actions*].

Dependencies: No dependencies

FTP_TRP.1(2) Trusted Path (Detection of Modification)

Hierarchical to: No other components.

FTP_TRP.1.1(2)

The TSF shall provide a communication path between itself and [*remote*] ~~users~~ **administrators using TLS or HTTPS** that is logically distinct from other communication paths and provides assured identification of its end points and ~~protection of the communicated data from~~ **detection of** [*modification*] **of the communicated data**.

FTP_TRP.1.2(2)

The TSF shall permit [remote ~~*users*~~ ***administrators***] to initiate communication via the trusted path.

FTP_TRP.1.3(2)

The TSF shall require the use of the trusted path for [*all remote administrative actions*].

Dependencies: No dependencies

6.2.9 Class NPM: Network Performance Management

NPM_SDC_EXT.1 System data collection

Hierarchical to: No other components.

NPM_SDC_EXT.1.1

The System shall be able to collect the following information from the targeted IT System resource(s): [*network packets*].

NPM_SDC_EXT.1.2

At a minimum, the System shall collect and record the following information:

a) Date and time of the event, type of event, and subject identity.

Dependencies: FPT_STM.1 Reliable time stamps

NPM_ANL_EXT.1 Analysis

Hierarchical to: No other components.

NPM_ANL_EXT.1.1

The TSF shall perform the following analysis function(s) on all system data collected: [*Statistical analysis metrics*].

Dependencies: NPM_SDC_EXT.1 System data collection

6.3 Security Assurance Requirements

This section defines the assurance requirements for the TOE. Assurance requirements are taken from the CC Part 3 and are EAL3 augmented with ALC_FLR.2.

Table 13 Assurance Requirements

Assurance Requirements	
Class ASE: Security Target evaluation	ASE_CCL.1 Conformance claims
	ASE_ECD.1 Extended components definition
	ASE_INT.1 ST introduction
	ASE_OBJ.2 Security objectives
	ASE_REQ.2 Derived security requirements
	ASE_SPD.1 Security problem definition
	ASE_TSS.1 TOE summary specification
Class ALC : Life Cycle Support	ALC_CMC.3 Authorization controls
	ALC_CMS.3 Implementation representation CM[29] coverage
	ALC_DEL.1 Delivery Procedures
	ALC_DVS.1 Identification of security measures
	ALC_LCD.1 Developer defined life-cycle model
	ALC_FLR.2 Flaw Reporting Procedures
Class ADV: Development	ADV_ARC.1 Security Architecture Description
	ADV_FSP.3 Functional specification with complete summary
	ADV_TDS.2 Architectural design
Class AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	AGD_PRE.1 Preparative procedures
Class ATE: Tests	ATE_COV.2 Analysis of coverage
	ATE_DPT.1 Testing: basic design
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing – sample
Class AVA: Vulnerability assessment	AVA_VAN.2 Vulnerability analysis

[29] CM – Configuration Management

7 TOE Specification

This section presents information to detail how the TOE meets the functional requirements described in previous sections of this ST.

7.1 TOE Security Functions

Each of the security requirements and the associated descriptions correspond to the security functions. Hence, each function is described by how it specifically satisfies each of its related requirements. This serves to both describe the security functions and rationalize that the security functions satisfy the necessary requirements.

Table 14 Mapping of TOE Security Functions to Security Functional Requirements

TOE Security Function	SFR ID	Description
Security Audit	FAU_GEN.1	Audit data generation
	FAU_GEN.2	User identity association
	FAU_SAR.1	Audit review
Cryptographic Support	FCS_CKM.1	Cryptographic key generation
	FCS_CKM_EXT.4	Extended: Cryptographic Key Zeroization
	FCS_COP.1(1)	Cryptographic operation (for data encryption/decryption)
	FCS_COP.1(2)	Cryptographic operation (for cryptographic signature)
	FCS_COP.1(3)	Cryptographic operation (for cryptographic hashing)
	FCS_COP.1(4)	Cryptographic operation (for keyed-hash message authentication)
	FCS_HTTPS_EXT.1	Extended: HTTPS
	FCS_TLS_EXT.1	Extended: TLS
Identification and Authentication	FIA_AFL.1	Authentication failure handling
	FIA_ATD.1	User attribute definition
	FIA_SOS.1	Verification of secrets
	FIA_UAU.2	User authentication before any action
	FIA_UAU_EXT.5	Extended: Password-based Authentication Mechanism
	FIA_UAU.6	Re-authenticating
	FIA_UAU.7	Protected Authentication Feedback
	FIA_UID.2	User identification before any action

TOE Security Function	SFR ID	Description
Network Performance Management	NPM_SDC_EXT.1	Extended: System data collection
	NPM_ANL_EXT.1	Extended: Analysis
Security Management	FMT_MSA.1	Management of security attributes
	FMT_MSA.3	Static attribute initialization
	FMT_MTD.1	Management of TSF data
	FMT_SMF.1	Specification of management functions
	FMT_SMR.1	Security roles
Protection of the TSF	FPT_ITT.1(1)	Basic Internal TSF Data Transfer Protection (Disclosure)
	FPT_ITT.1(2)	Basic Internal TSF Data Transfer Protection (Modification)
	FPT_PTD_EXT.1(1)	Extended: Management of TSF Data (for reading of authentication data)
	FPT_PTD_EXT.1(2)	Extended: Management of TSF Data (for reading of all symmetric keys)
	FPT_RPL.1	Replay Detection
	FPT_STM.1	Reliable Time Stamps
	FPT_TST_EXT.1	Extended: TSF testing
Resource Utilization	FRU_RSA.1	Maximum Quotas
Trusted path/channels	FTP_ITC.1(1)	Inter-TSF Trusted Channel (Prevention of Disclosure)
	FTP_ITC.1(2)	Inter-TSF Trusted Channel (Detection of Modification)
	FTP_TRP.1(1)	Trusted Path (Prevention of Disclosure)
	FTP_TRP.1(2)	Trusted Path (Detection of Modification)
User Data Protection	FDP_ACC.1	Subset Access Control
	FDP_ACF.1	Security Attribute Based Access Control

7.1.1 Security Audit

The TOE generates audit records for security-relevant user actions and provides an interface for reviewing the audit logs. As Security Administrators manage and configure the TOE, their activities are tracked and recorded as audit records. Audit records generated by the system include date and time of the event, user

ID[30] that caused the event to be generated (in the case of events by the system, SYSTEM is used for the ID), where the event occurred, and other event-specific data. Table 11 provides a detailed listing of the auditable events for each security functional requirement of the TOE. Activities generated by users are also tracked and recorded as audit records.

The TSF uses the user ID in the audit record to identify the Security Administrator and user that performed the action. The TOE maintains the time from a reliable source. The TOE software uses the time from the CascadeOS. The CascadeOS of the TOE relies on the external NTP server in the environment in order to maintain the correct time.
The Shark uses a Linux syslog daemon to record audit records. The syslog is used to diagnose Cascade Shark issues and includes audit log messages. It is a rotated plaintext file (one or more) stored in the Shark file system.

The TOE provides auditing of Security Administrator and user actions that occur within the management interfaces. The Cascade Shark Web Interface provides an authorized Security Administrator access to view the audit logs created as a result of Security Administrator actions and reporting features. Only Security Administrators can review the security audit logs.

TOE Security Functional Requirements Satisfied: FAU_GEN.1, FAU_GEN.2, FAU_SAR.1

7.1.2 Cryptographic Support

The Cryptographic Support TSF function provides cryptographic functions to implement SSL v3.1 (TLS[31] v1.0) that secures the communication channel between the Shark and Pilot Components, and between Shark and web browsers. SSL is a network protocol that allows data to be exchanged using a secure channel between two networked devices and provides confidentiality and integrity of data sent over an unsecure network.

The TOE implements HTTPS[32] for protection of the management user interfaces. HTTPS using SSLv3.1 (TLS v1.0) connections are used to protect all communication between Shark and Pilot.

The TLS implementation used by Shark is provided by the OpenSSL Object Module v2.0 which is FIPS[33] 140-2 validated. For more information, see. http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm#1747 FIPS Certificate No. 1747. The OpenSSL Object Module used by the TOE was tested using the CascadeOS 6.1 (64-bit) on Intel Pentium T4200 (x86) processors.

Shark uses the following cryptographic algorithms that have been FIPS validated:

Table 15 Algorithm and Validation Certificates

Algorithm	Validation Certificate	Usage
AES	1884	Encrypt/decrypt
3DES	1223	Encrypt/decrypt
DSA	589	Sign and verify
PRNG (ANSI X9.31 Appendix A.2.4 using AES)	985	Random number generation
RSA (X9.31, PKCS #1.5, PSS)	960	Sign and verify

[30] ID – Identifier
[31] TLS – Transport Layer Security
[32] HTTPS – Hyper Text Transfer Protocol Secure
[33] FIPS – Federal Information Processing Standards

Algorithm	Validation Certificate	Usage
SHA-1 SHA-256 SHA-384 SHA-512	1655	Hashing
SHS	1655	Secure hashing
HMAC-SHA-1 HMAC-SHA-256 HMAC-SHA-384 HMAC-SHA-512	1884	Message integrity

In addition, the cryptographic functionality within Shark is used to secure the sessions between Shark and trusted IT[34] components, such as Profiler which is another Riverbed product in the Cascade product line. The secure sessions between Shark and Profiler uses a proprietary protocol over TLS v1.0 as well. The Cascade Profiler product is seeking a Common Criteria evaluation separate from this evaluation effort.

Shark creates a self-signed certificate and cryptographic key pair when it gets installed on the hardware. The certificate and private key are stored and remain on Shark.

Pilot uses a FIPS 140-2 validated cryptographic module that is part of the underlying Windows operating system on which the Pilot software operates to implement its cryptographic functions, TLS. Pilot executes on a system running either Windows XP, Windows Vista, or Windows 7 operating systems which provide FIPS validated cryptographic modules.

The cryptographic modules implement mechanisms for protecting the pre-shared keys, symmetric key, and private keys from unauthorized disclosure.

TOE Security Functional Requirements Satisfied: FCS_CKM.1, FCS_CKM_EXT.4, FCS_COP.1(1) – (4), FCS_HTTPS_EXT.1, FCS_TLS_EXT.1, FPT_PTD_EXT.1(1), (2)

7.1.3 User Data Protection

Each user owns the resources (i.e., files and folders) that the user has created and the views that the user has applied. Except for Security Administrators, a user has access to only the resources owned by the user, unless the resource has been shared:

- A user cannot see a file or a view created by another user unless that file is located in one of the group folders the user belongs to.
- A user cannot close a view or delete a file that has been created by another user unless that file is located in one of the common group folders the user belongs to.
- A user cannot see a view created by another user unless the owner of the view shared the view with a group to which the user requesting access belongs.

File ownership is determined by the location of the file. Files can be owned by users or groups. Ownership of a file is changed by moving the file to a user's folder or a group's folder.

Resources (views and trace files) can be *shared*. Members of a group share a common folder that has the same name of the group. This folder can be used for trace file sharing, and all the users in the group have read and write access to the folder. When a file is moved into a common folder, all the other members of

[34] IT – Information Technology

the group will immediately see and be able to manipulate it (this feature is called sharing a file). Pilot provides a graphical interface that allows the owner to share the views with groups.

TOE Security Functional Requirements Satisfied: FDP_ACC.1, FDP_ACF.1.

7.1.4 Identification and Authentication

The TOE requires users to provide unique identification and authentication data (i.e., ID, password) before any administrative access to the TOE is granted. While the Security Administrator or user password is being entered, it is provided on screen as dots to avoid over the shoulder password compromise.

All Security Administrators and users that access the Shark management interfaces are authenticated by the Shark software. The Shark supports both local and remote authentication via RADIUS[35] and TACACS+[36]. In the evaluated configuration, only local authentication is allowed. The local identification and authentication mechanism is based on usernames and passwords. The management of credentials is controlled by the user configuration system located on the Shark appliance. Each request from a Cascade Pilot requiring authentication uses HTTP cookies for authentication.

For remote access (both web-based administrative sessions and Cascade Pilot sessions), Shark Probe passes the authentication credentials to the Credential Manager on the Shark. The Credential Manager determines if the user has a privilege that permits the execution of the requested operation. If the Credential Manager rejects the operation, the Shark appliance returns an error to the interface making the request. Otherwise the Shark appliance executes the operation.

The password policy for the product specifies the password complexity, password composition requirements, and the allowed number of failed authentication attempts before lockout occurs as defined by FIA_SOS.1. In addition, passwords will expire after a Security Administrator-configured expiration. When a user password expires, the user must create a new password after correctly entering the expired password.

The TOE enforces an authentication failure limit for the Security Administrators and users and will prevent further login attempts for the offending user by disabling the Security Administrator’s or user’s account. The authentication failure limit is configurable by Security Administrator and the value must be in the range of zero to ten, where a value of zero means that the user account will never be disabled. Only a Security Administrator can re-enable a disabled account.

Users and groups are configured using the Cascade Shark Web Interface. Usernames (including Security Administrator usernames) are stored in a clear-text file local to Cascade Shark with stored passwords obscured via an SHA-512[37] hash to prevent disclosure of user passwords. To change a user password, the user must supply their old password prior to changing their password. The Cascade Pilot has the ability to cache the passwords, but this capability is not allowed in the evaluated configuration.

A user can be a member of one or more groups. A group is a set of users. Each group is assigned zero or more privileges that are available to the members of the group. Privileges can only be assigned to groups, not to users. A privilege is granted to the user if the privilege is enabled for any group of which the user is a member.

The Cascade Shark Web Interface is used to configure the privileges for groups. A privilege is a capability that can be granted to or revoked from a group on Cascade Shark.

Security Administrators can access the Shark appliance directly at the console. On the Shark console, Security Administrators and users are identified and authenticated by the underlying operating system PAM authentication system.

[35] RADIUS – Remote Authentication Dial In User Service
[36] TACACS+ – Terminal Access Controller Access-Control System Plus
[37] SHA – Secure Hash Algorithm

The privileges that the Shark appliance currently implements are defined in the table below:

Table 16 Shark Privileges

Privilege	Description
IsAdmin	This is the Security Administrator privilege. If set to true, gives members of a group full access to Shark. Administrators see all the resources in the system, including views, files and folders that have been created by other users. Administrators have full control on all these resources.
CanApplyViewsOnFiles	if set to true, allows members of a group to apply views to trace files residing on the Shark appliance, capture jobs, and trace clips. In order to apply a view to a capture job or trace clip, CanAccessProbeFiles is also required.
CanApplyViewsOnInterfaces	if set to true, allows members of a group to apply views to the capture ports and job interfaces on the Shark appliance.
CanCreateFiles	if set to true, the members of a group can create files on the Shark appliance, by selecting the "send to file" buttons in the Cascade Pilot.
CanImportFiles	if set to true, the members of a group can import files into the Shark appliance, through drag and drop or by clicking on the "Import Files Into Shark Appliance" button in the Remote ribbon.
CanCreateJobs	if set to true, the members of a group have full access (create, delete, change parameters, start, stop) to manage capture jobs from the Web Interface.
CanExportFiles	if set to true, allows members of a group to export files from the Shark appliance, and move them to the Cascade Pilot or to another Shark (assuming the user has sufficient privilege on the target Shark to create a trace file). When this privilege is not granted, the user is not able to export a trace file to Wireshark, because that involves exporting packets out of Cascade Shark to Cascade Pilot.
CanShareViews	if set to true, the members of a group can share the views that they created on the Shark appliance with any groups on the same Shark appliance. Without this privilege, users can only share views with the groups of which they are a member.
CanAccessProbeFiles	if set to true, the members of a group will be able to "see" the trace files located on the Shark appliance from Pilot.
CanScheduleWatches	if set to true, the members of a group can create a watch on a view. Since watches are attached to views, to create a watch, the user also needs the CanApplyView capability.

TOE Security Functional Requirements Satisfied: FIA_AFL.1, FIA_ATD.1, FIA_SOS.1 FIA_UAU_EXT.5, FIA_UAU.2, FIA_UAU.6, FIA_UAU.7, FIA_UID.2, FPT_PTD_EXT.1(1)

7.1.5 Security Management

The Linux-based operating system on the Shark appliance is delivered with the root account that is not available in the evaluated configuration. The Shark software is delivered with a Security Administrator role. The Shark must have at least one user with the Security Administrator role defined, so the last user with the Security Administrator role cannot be deleted. Granular Shark privileges can be assigned to groups and the users can be assigned to one or many groups. The TOE implements two types of roles: Security Administrators and users without administrator privileges. The Shark graphical management interfaces implement roles by checking the privileges assigned to the groups of which the Security Administrators and users are a member. Groups assigned the IsAdmin privilege are considered to be the Security Administrator.

Command line access to the Cascade Shark is available via the CLI. When accessing the TOE from the local console, a proprietary command line interface is provided. In the evaluated configuration, access to the shell provided by the underlying operating system is not allowed. The command line interface available via the Shark console allows Security Administrators to perform the initial appliance (operating system) configuration, such as network configuration.

The TOE implements a robust management system providing multiple administrative functions and restricting their use based on roles. The TOE provides two graphical management interfaces: the Cascade Shark Web Interface and the Cascade Pilot UI. The Web Interface can be accessed from either remotely from Cascade Pilot or remotely from a web browser. The Pilot UI displays data collected by remote Shark appliances. Cascade Pilot sends administrative operations to the Shark for processing.

The Cascade Shark Web Interface provides the capability to

- display the Shark appliance status, capture job status
- perform user and group management
- configure Shark capture board(s)
- modify protocol definitions/groups
- retrieve the appliance logs and audit trails.
- manage the Capture Jobs in the Jobs Repository[38]

The Pilot UI provides the ability to:

- display and analyze network traffic on data collected by a remote Shark appliance,
- connect and manage one or more remote Shark appliances

Security Administrators and users with the appropriate privilege as defined in Table 16 can view and modify a user’s group, user privileges, object owner, and share with list. There is no method provided for changing the initial default security attributes when a view or file is created. By default, files and views created can only be accessed by their owner or a user with the IsAdmin privilege.

TOE Security Functional Requirements Satisfied: FMT_MSA.1, FMT_MSA.3, FMT_MTD.1, FMT_SMF.1, FMT_SMR.1

7.1.6 Protection of the TSF

The TOE provides protection mechanisms for its security functions. One of the protection mechanisms is that Security Administrators and users must authenticate before any administrative operations can be performed on the system, whether those functions are related to the management of user accounts or the configuration of traffic analysis.

[38] The Jobs Repository includes the name of each Job Trace in the appliance.

SSL is used to protect the transmissions between separate parts of the TOE and to prevent the data from disclosure and modification. The TOE implements HTTPS and TLS for protection of the management user interfaces. HTTPS (SSLv3.1 (TLS v1.0)) connections are used to protect all communication between Shark and Pilot. HTTPS protects data transfer and leverages cryptographic capabilities to prevent replay attacks. The TOE uses FIPS-approved and validated cryptographic algorithms to implement TLS.

In addition, the cryptographic functionality within the TOE component is used to secure the sessions between the TOE and trusted IT components, such as Profiler which is another Riverbed product in the Cascade product line.

The cryptographic operations used to protect the transmissions are provided by FIPS validated algorithms. The TOE provides self-tests for the cryptographic modules.

The TOE maintains the time from a reliable source. The TOE software uses the time from the CascadeOS. The CascadeOS of the TOE relies on the external NTP server in the environment in order to maintain the correct time.

TOE Security Functional Requirements Satisfied: FPT_ITT.1(1), FPT_ITT.1(2), FPT_RPL.1, FPT_STM.1, FPT_TST_EXT.1

7.1.7 Resource Utilization

The management interface allows Security Administrators and users to create capture jobs that store packet captures. Each capture job can have an assigned maximum size for the amount of disk space used to store each capture job in the Packet Storage. Oldest information in a capture job is overwritten when used disk space reaches the configured storage available.

TOE Security Functional Requirements Satisfied: FRU_RSA.1

7.1.8 Trusted Path/Channels

The TOE provides trusted channels for all data from disclosure or modification while in transit between TOE components and between TOE components and some authorized IT entities.

SSL is used to provide trusted channels between the TOE and authorized IT entities and to prevent the data from disclosure and modification. The TOE implements HTTPS and TLS for protection of the management user interfaces. The TOE uses FIPS approved and validated cryptographic algorithms to implement TLS.

In addition, the cryptographic functionality within the TOE component is used to secure the sessions between the TOE and trusted IT components, such as Profiler, which is another Riverbed product in the Cascade product line.

The cryptographic operations used to implement the trusted channels are provided by FIPS validated algorithms.

TOE Security Functional Requirements Satisfied: FTP_ITC.1(1), (2), FTP_TRP.1(1), (2)

7.1.9 Network Performance Management

Cascade Shark collects network data and analyzes the collected network packets. Cascade Shark can provide both real-time and historical traffic analysis monitoring. Network traffic analysis metrics (views) can be applied to live (network interface cards) and off-line traffic sources. The network traffic analysis metrics can be computed based on a variety of attributes. Examples of possible views are:

- Bandwidth over time
- IP conversations
- Protocol distributions for either a live or off-line source of network traffic
  - Protocol distribution by bits
  - Protocol distribution by bytes
  - Protocol distribution by packets
- Network usage by traffic type

The Shark Packet Recorder is used to collect high-speed and/or long duration network traffic. The Shark Packet Recorder uses an optimized packet data store, which saves network traffic as capture jobs and uses time filters to efficiently index the network packet data.

Cascade Shark collects the following data: date and time of the network packet collection, type of data (network protocol by default), subject identity (IP address) if applicable.

TOE Security Functional Requirements Satisfied: NPM_SDC_EXT.1, NPM_ANL_EXT.1.

8 Rationale

8.1 Conformance Claims Rationale

This Security Target conforms to Part 2 and Part 3 of the Common Criteria Standard for Information Technology Security Evaluations, version 3.1 revision 3.

8.2 Security Objectives Rationale

This section provides a rationale for the existence of each threat and assumption that compose the Security Target. Sections 8.2.1, and 8.2.2 demonstrate that the mappings between the threats, polices, and assumptions to the security objectives is complete. The following discussion provides detailed evidence of coverage for each threat, policy, and assumption.

8.2.1 Security Objectives Rationale Relating to Threats

Table 17 displays the mapping of threats to objectives.

Table 17 Threats:Objectives Mapping

Threats	Objectives	Rationale
T.ADMIN_ERROR An administrator may incorrectly install or configure the TOE resulting in ineffective security mechanisms.	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.TOE_ADMINISTRATION counters this threat by ensuring that only authorized Administrators are able to log in and configure the TOE, and the TOE provides protections for logged-in Administrators.
T.FAIL_NETANAL The TOE may fail to identify the network traffic flow conditions as requested by the administrator.	O.ANALYZE The TOE will apply analytical processes and information to derive conclusions about the network (past, present, or future).	O.ANALYZE counters this threat by applying analytical processes and information on collected network traffic to derive conclusions (past, present, or future).
	O.SCAN The TOE will collect network traffic information from the network interface card.	O.SCAN counters this threat by collecting information from the network interface card for use in applying analytical processes on the information.
T.RESOURCE_EXHAUSTION A process or user may deny access to TOE services by exhausting critical resources on the TOE.	O.RESOURCE_AVAILABILITY The TOE shall provide mechanisms that mitigate user attempts to exhaust TOE resources (e.g., persistent storage).	O.RESOURCE_AVAILABILITY counters this threat by providing mechanisms that mitigate user attempts to exhaust TOE resources (e.g., persistent storage).
T.TSF_FAILURE Security mechanisms of the TOE may fail, leading to a compromise of the TSF.	O.TSF_SELF_TEST The TOE will provide the capability to test some subset of its security functionality to ensure it is operating properly.	O.TSF_SELF_TEST counters this threat by ensuring that the TOE provides self-tests on a subset of its security functionality to ensure it is operating properly.

Threats	Objectives	Rationale
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.	O.PROTECTED_COMMUNICATIONS The TOE will provide protected communication channels for administrators, other parts of a distributed TOE, and authorized IT entities.	O.PROTECTED_COMMUNICATIONS counters this threat by providing protected communication channels for administrators, other parts of a distributed TOE, and authorized IT entities.
	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.TOE_ADMINISTRATION counters this treat by ensuring that only administrators are able to log in and configure the TOE and providing protections for logged-in administrators.
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.	O.SYSTEM_MONITORING The TOE will provide the capability to generate audit data.	O.SYSTEM_MONITORING counters this threat by ensuring that unauthorized attempts to access the TOE are recorded.

Every Threat is mapped to one or more Objectives in the table above. This complete mapping demonstrates that the defined security objectives counter all defined threats.

8.2.2 Security Objectives Rationale Relating to Assumptions

Table 18 Assumptions:Objectives Mapping

Assumptions	Objectives	Rationale
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.	OE.NO_GENERAL_PURPOSE There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.	OE.NO_GENERAL_PURPOSE satisfies this assumption by ensuring 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.	OE.PHYSICAL Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.	OE.PHYSICAL satisifes the assumption that the TOE environment provides physical security commensurate with the value of the TOE and the data it contains.
A.TRUSTED_ADMIN TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.	OE.TRUSTED_ADMIN TOE Administrators are trusted to follow and apply all administrator guidance in a	OE.TRUSTED_ADMIN satisfies the assumption that the users who manage the TOE are trusted and follow all guidance.

Assumptions	Objectives	Rationale
	trusted manner.	

Every assumption is mapped to one or more Objectives in the table above. This complete mapping demonstrates that the defined security objectives uphold all defined assumptions.

8.3 Rationale for Extended Security Functional Requirements

The extended requirements are defined in section 5. These SFRs exhibit functionality that can be easily documented in the ADV assurance evidence and thus do not require any additional Assurance Documentation.

Although this ST is not compliant with the NDPP, some of the NDPP SFRs have been included. The following explicitly stated SFRs were taken directly from the NDPP: FCS_CKM_EXT.4, and FIA_UAU_EXT.5.

8.4 Rationale for Extended TOE Security Assurance Requirements

There are no Extended SARs defined for this ST.

8.5 Security Requirements Rationale

This section provides evidence supporting the internal consistency and completeness of the components (requirements) in the Security Target.

8.5.1 Security Functional Requirements Rationale

Table 19 Objectives:SFRs Mapping

Objective	Requirements Addressing the Objective	Rationale
O.ANALYZE The TOE will apply analytical processes and information to derive conclusions about the network (past, present, or future).	NPM_ANL_EXT.1 Extended: Analysis	The requirement meets the objective by ensuring the TOE analyzes the collected data.
O.PROTECTED_COMMUNICATIONS The TOE will provide protected communication channels for administrators, other parts of a distributed TOE, and authorized IT	FCS_CKM.1 Cryptographic key generation	The requirement meets the objective by ensuring that the TOE can generate cryptographic keys for use during cryptographic operations.
	FCS_CKM_EXT.4	The requirement meets the

Objective	Requirements Addressing the Objective	Rationale
entities.	Extended: Cryptographic Key Zeroization	objective by ensuring that the TOE can zeroize cryptographic keys.
	FCS_COP.1(1) Cryptographic operation (for data encryption/decryption)	The requirement meets the objective by ensuring that the TOE can perform encryption and decryption in accordance with the defined algorithms and key sizes.
	FCS_COP.1(2) Cryptographic operation (for cryptographic signature)	The requirement meets the objective by ensuring that the TOE can perform cryptographic signature services in accordance with the defined algorithms and key sizes.
	FCS_COP.1(3) Cryptographic operation (for cryptographic hashing)	The requirement meets the objective by ensuring that the TOE can perform cryptographic hashing services in accordance with the defined algorithms and key sizes.
	FCS_COP.1(4) Cryptographic operation (for keyed-hash message authentication)	The requirement meets the objective by ensuring that the TOE can perform cryptographic hashing services in accordance with the defined algorithms and key sizes.
	FCS_HTTPS_EXT.1 Extended: HTTPS	The requirement meets the objective by ensuring that the TOE provides the HTTPS protocol in compliance with RFC 2818.
	FCS_TLS_EXT.1 Extended: TLS	The requirement meets the objective by ensuring that the TOE provides TLS in accordance with the defined ciphersuites.
	FPT_ITT.1(1) Basic Internal TSF Data Transfer Protection (Disclosure)	The requirement meets the objective by ensuring that the TOE protects TSF data from disclosure when transmitted between separate parts of the TOE.
	FPT_ITT.1(2) Basic Internal TSF Data Transfer Protection (Modification)	The requirement meets the objective by ensuring that the TOE detects modification of TSF data when transmitted between separate parts of the TOE.
	FPT_PTD_EXT.1(1)	The requirement meets the

Objective	Requirements Addressing the Objective	Rationale
	Extended: Management of TSF Data (for reading of authentication data)	objective by ensuring that the TOE prevents reading of plaintext passwords.
O.PROTECTED_COMMUNICATIONS The TOE will provide protected communication channels for administrators, other parts of a distributed TOE, and authorized IT entities.	FPT_PTD_EXT.1(2) Extended: Management of TSF Data (for reading of all symmetric keys)	The requirement meets the objective by ensuring that the TOE prevents reading of all specificed cryptographic keys.
	FPT_RPL.1 Replay Detection	The requirement meets the objective by ensuring that the TOE detects replay of network packets that have been encrypted via SSL/TLS.
	FTP_TRP.1(1) Trusted Path (Prevention of Disclosure)	The requirement meets the objective by ensuring that the TOE provides a trusted path between itself and authorized IT entities from disclosure.
	FTP_ITC.1(1) Inter-TSF Trusted Channel (Prevention of Disclosure)	The requirement meets the objective by ensuring that the TOE provides a trusted communication channel between itself and authorized IT entities from disclosure.
	FTP_ITC.1(2) Inter-TSF Trusted Channel (Detection of Modification)	The requirement meets the objective by ensuring that the TOE provides a a trusted communication channel between itself and authorized IT entities from modification.
	FTP_TRP.1(2) Trusted Path (Detection of Modification)	The requirement meets the objective by ensuring that the TOE provides a trusted path between itself and authorized IT entities from modification.
O.RESOURCE_AVAILABILITY The TOE shall provide mechanisms that mitigate user attempts to exhaust TOE resources (e.g., persistent storage).	FRU_RSA.1 Maximum Quotas	FRU_RSA.1 supports this objective by ensuring the TOE does not exhaust TOE resources.
O.SCAN The TOE will collect network traffic information from the network interface card.	NPM_SDC_EXT.1 Extended: System data collection	The requirement meets the objective by ensuring that the TOE collects system data from the network interface card.
O.SYSTEM_MONITORING The TOE will provide the capability to generate audit data.	FAU_GEN.1 Audit data generation	The requirement meets this objective by ensuring that the TOE maintains a record of defined security related events, including

Objective	Requirements Addressing the Objective	Rationale
		relevant details about the event.
	FAU_GEN.2 User identity association	The requirement meets the objective by ensuring that the TOE associates each auditable event with the identity of the user that caused the event.
	FAU_SAR.1 Audit review	The requirement meets the objective by ensuring that the TOE provides the ability to review logs.
	FPT_STM.1 Reliable Time Stamps	The requirement meets the objective by ensuring that the TOE provides reliable timestamps.
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.	FDP_ACC.1 Subset Access Control	The requirement meets the objective by defining an access control SFP based on subjects and objects (files and views).
	FDP_ACF.1 Security Attribute Based Access Control	The requirement meets the objective by enforcing an access control SFP based on the security attributes of the subjects and objects (files and views).
	FIA_AFL.1 Authentication failure handling	The requirement meets the objective by ensuring after an administrator-specified number of unsuccessful authentication attempts, the user account is disabled.
	FIA_ATD.1 User attribute definition	The requirement meets the objective by ensuring that the TOE maintains the user's security attributes.
	FIA_SOS.1 Verification of secrets	The requirement meets the objective by ensuring that the TOE enforces that passwords meet the required password quality metrics.
	FIA_UAU.2 User authentication before any action	The requirement meets the objective by ensuring that the TOE ensures that a user must be successfully authenticated before being allowed access to TOE management functions.
	FIA_UAU_EXT.5 Extended: Password-based Authentication Mechanism	The requirement meets the objective by ensuring that the TOE provides a local password

Objective	Requirements Addressing the Objective	Rationale
		based authentication.
	FIA_UAU.6 Re-authenticating	The requirement meets the objective by ensuring that the TOE re-authenticates the user under the specified conditions.
	FIA_UAU.7 Protected Authentication Feedback	The requirement meets the objective by ensuring that the TOE provides obscured feedback while the user is authenticating.
	FIA_UID.2 User identification before any action	The requirement meets the objective by ensuring that the TOE ensures that a user must be successfully identified before being allowed access to the TOE management functions.
	FMT_MSA.1 Management of security attributes	The requirement meets the objective by ensuring that the TOE restricts which users are allowed to query and modify security attributes.
	FMT_MSA.3 Static attribute initialization	The requirement meets the objective by providing restricting default values for the security attributes. The TOE does not provide a mechanism to allow the initial values for security attributes to be changed.
	FMT_MTD.1 Management of TSF data	The requirement meets the objective by ensuring that only authorized users are allowed access to TSF data.
	FMT_SMF.1 Specification of management functions	The requirement meets the objective by ensuring that the TOE includes administrative functions to facilitate the management of the TSF.
	FMT_SMR.1 Security roles	The requirement meets the objective by ensuring that the TOE associates users with roles to provide access to TSF management functions and data.
O.TSF_SELF_TEST The TOE will provide the capability to test some subset of its security functionality to ensure it is operating properly.	FPT_TST_EXT.1 Extended: TSF testing	The requirement meets the objective by ensuring that the TOE provides some self-tests on a subset of its security functionality to ensure it is operating properly.

All Security Functional Requirements (SFR) identified in this Security Target are fully addressed in this section and each SFR is mapped to the objective for which it is intended to satisfy.

8.5.2 Security Assurance Requirements Rationale

EAL3+ was chosen to provide a moderate level of assurance that is consistent with good commercial practices. As such minimal additional tasks are placed upon the vendor assuming the vendor follows reasonable software engineering practices and can provide support to the evaluation for design and testing efforts. The chosen assurance level is appropriate with the threats defined for the environment. While the TOE may operate in a hostile environment, it is expected to be in a non-hostile position and embedded in or protected by other products designed to address threats that correspond with the intended environment. At EAL3+ the TOE will have incurred a search for obvious flaws to support its introduction into the non-hostile environment.

The augmentation of ALC_FLR.2 was chosen to give greater assurance of the developer’s on-going flaw remediation processes.

8.5.3 Requirement Dependency Rationale

This ST does satisfy all the requirement dependencies of the Common Criteria. Table 20 lists each requirement to which the TOE claims conformance with a dependency and indicates whether the dependent requirement was included. As the table indicates, all dependencies have been met.

Table 20 Functional Requirements Dependencies

SFR ID	Dependencies	Dependency Met	Rationale
FAU_GEN.1	FPT_STM.1	✓	
FAU_GEN.2	FIA_UID.1	✓	Although FIA_UID.1 is not included, FIA_UID.2 is hierarchical to FIA_UID.1.
	FAU_GEN.1	✓	
FAU_SAR.1	FAU_GEN.1	✓	
FCS_CKM.1	FCS_COP.1(2)	✓	
	FCS_COP.1(3)	✓	
	FCS_COP.1(1)	✓	
	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
	FCS_COP.1(4)	✓	
FCS_CKM_EXT.4	FCS_CKM.1	✓	
FCS_COP.1(1)	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.

SFR ID	Dependencies	Dependency Met	Rationale
	FCS_CKM.1	✓	
FCS_COP.1(2)	FCS_CKM.1	✓	
	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
FCS_COP.1(3)	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
	FCS_CKM.1	✓	
FCS_COP.1(4)	FCS_CKM.1	✓	
	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
FCS_HTTPS_EXT.1	FCS_TLS_EXT.1	✓	
FCS_TLS_EXT.1	FCS_COP.1(2)	✓	
FDP_ACC.1	FDP_ACF.1	✓	
FDP_ACF.1	FDP_ACC.1	✓	
	FMT_MSA.3	✓	
FIA_AFL.1	FIA_UAU.1	✓	Although FIA_UAU.1 is not included, FIA_UAU.2 is hierarchical to FIA_UAU.1.
FIA_ATD.1	No dependencies	✓	
FIA_SOS.1	No dependencies	✓	
FIA_UAU.2	FIA_UID.1	✓	Although FIA_UID.1 is not included, FIA_UID.2 is hierarchical to FIA_UID.1.
FIA_UAU_EXT.5	No dependencies	✓	
FIA_UAU.6	No dependencies	✓	
FIA_UAU.7	FIA_UAU.1	✓	Although FIA_UAU.1 is not included, FIA_UAU.2 is hierarchical to FIA_UAU.1.
FIA_UID.2	No dependencies	✓	
FMT_MSA.1	FDP_ACC.1	✓	
	FMT_SMF.1	✓	

SFR ID	Dependencies	Dependency Met	Rationale
FMT_MSA.3	FMT_MSA.1	✓	
	FMT_SMR.1	✓	
FMT_MTD.1	FMT_SMR.1	✓	
	FMT_SMF.1	✓	
FMT_SMF.1	No dependencies	✓	
FMT_SMR.1	FIA_UID.1	✓	Although FIA_UID.1 is not included, FIA_UIA_EXT.1 provides coverage for user identification and authentication which supersedes FIA_UID.1.
FPT_ITT.1(1)	No dependencies	✓	
FPT_ITT.1(2)	No dependencies	✓	
FPT_PTD_EXT.1(1)	No dependencies	✓	
FPT_PTD_EXT.1(2)	No dependencies	✓	
FPT_RPL.1	No dependencies	✓	
FPT_STM.1	No dependencies	✓	
FPT_TST_EXT.1	No dependencies	✓	
FRU_RSA.1	No dependencies	✓	
FTP_TRP.1(1)	No dependencies	✓	
FTP_ITC.1(2)	No dependencies	✓	
FTP_ITC.1(1)	No dependencies	✓	
FTP_TRP.1(2)	No dependencies	✓	
NPM_SDC_EXT.1	FPT_STM.1	✓	
NPM_ANL_EXT.1	NPM_SDC_EXT.1	✓	

9 Acronyms

This section describes the acronyms used in this document.

Table 21 Acronyms

Acronym	Definition
ACL	Access Control List
AES	Advanced Encryption Standard
API	Application Programming Interface
CC	Common Criteria
CLI	Command Line Interface
CM	Configuration Management
CSP	Critical Security Parameter
EAL	Evaluation Assurance Level
FIPS	Federal Information Processing Standard
Gbit	Gigabit
Gbps	Gigabit per second
GUI	Graphical User Interface
HTTP	Hypertext Transport Protocol
HTTPS	Secure HTTP
ICMP	Internet Control Message Protocol
I/O	Input/Output
IP	Internet Protocol
IT	Information Technology
LAN	Local Area Network
N/A	Not Applicable
NTP	Network Time Protocol
NVRAM	Non-volatile Random Access Memory
OS	Operating System
OSP	Organizational Security Policy
PCI	Peripheral Component Interconnect
PP	Protection Profile
RADIUS	Remote Authentication Dial In User Service
RIP	Routing Information Protocol
RPC	Remote Procedure Call
SF	Switch Fabric

Acronym	Definition
SFP	Security Function Policy
SSH	Secure Shell
SSL	Secure Sockets Layer
ST	Security Target
TACACS+	Terminal Access Controller Access-Control System Plus
TCP	Transport Control Protocol
TOE	Target of Evaluation
TSF	TOE Security Functions
TLS	Transport Layer Security
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network
XML	eXtensible Markup Language

Prepared by:
Corsec Security, Inc.

13135 Lee Jackson Memorial Highway, Suite 220
Fairfax, VA 22033
United States of America

Phone: (703) 267-6050
Email: info@corsec.com
http://www.corsec.com