Tripwire, Inc. Tripwire Manager, Version 4.6.1, with Tripwire for Servers, Version 4.6.1 Security Target

Version 1.0

May 1, 2009

Prepared for:

Tripwire, Inc.

101 SW Main Street, Suite 1500
Portland, OR 97204

Prepared By:

Science Applications International Corporation

Common Criteria Testing Laboratory

7125 Columbia Gateway Drive, Suite 300
Columbia, MD 21046

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is Tripwire, Inc. Tripwire Manager/Tripwire for Servers provided by Tripwire, Inc. The TOE is a change audit assessment product that can assure the integrity of critical data on system(s) by monitoring file system object attributes for unauthorized or unexpected modification.

The Security Target contains the following additional sections:

- Section 2 – Target of Evaluation (TOE) Description
  This section gives an overview of the TOE, describes the TOE in terms of its physical and logical boundaries, and states the scope of the TOE.
- Section 3 – TOE Security Environment
  This section details the expectations of the environment, the threats that are countered by the TOE and IT environment, and the organizational policy that TOE must fulfill.
- Section 4 – TOE Security Objectives
  This section details the security objectives of the TOE and IT environment.
- Section 5 – IT Security Requirements
  The section presents the security functional requirements (SFR) for the TOE and IT Environment that supports the TOE, and details the assurance requirements.
- Section 6 – TOE Summary Specification
  The section describes the security functions represented in the TOE that satisfy the security requirements.
- Section 7 – Protection Profile Claims
  This section presents any protection profile claims.
- Section 8 – Rationale
  This section closes the ST with the justifications of the security objectives, requirements and TOE summary specifications as to their consistency, completeness, and suitability.

1.1 Security Target, TOE and CC Identification

ST Title – Tripwire, Inc. Tripwire Manager, Version 4.6.1, with Tripwire for Servers, Version 4.6.1 Security Target

ST Version – Version 1.0

ST Date –May 1, 2009

TOE Identification – Tripwire Manager, Version 4.6.1, with Tripwire for Servers, Version 4.6.1

CC Identification – Common Criteria for Information Technology Security Evaluation, Version 2.3, August 2005.

1.2 Conformance Claims

This TOE is conformant to the following CC specifications:

- Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Requirements, Version 2.3, August 2005.
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security Assurance Requirements, Version 2.3, August 2005.
  - Part 3 Conformant
  - Assurance Level: EAL 3 augmented with ALC_FLR.2

1.3 Conventions and Acronyms

This section specifies the formatting conventions used in the Security Target and provides a glossary of acronyms.

1.3.1 Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
    - Iteration: allows a component to be used more than once with varying operations. In the ST, iteration is indicated by a letter placed at the end of the component. For example FDP_ACC.1a and FDP_ACC.1b indicate that the ST includes two iterations of the FDP_ACC.1 requirement, a and b.
    - Assignment: allows the specification of an identified parameter. Assignments are indicated using bold and are surrounded by brackets (e.g., [**assignment**]).
    - Selection: allows the specification of one or more elements from a list. Selections are indicated using bold italics and are surrounded by brackets (e.g., [***selection***]).
    - Refinement: allows the addition of details. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …").
- Explicitly stated Security Functional Requirements (i.e., those not found in Part 2 of the CC) are identified with "**(EX)**".
- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.

1.3.2 Acronyms

1.3.2.1 Common Criteria Specific

CC	Common Criteria
EAL	Evaluation Assurance Level
IT	Information Technology
PP	Protection Profile
SF	Security Function
SFP	Security Function Policy
SOF	Strength of Function
ST	Security Target
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Functions
TSFI	TSF Interface
TSP	TOE Security Policy

1.3.2.2 TOE and IT Technology Specific

3DES	The NIST Data Encryption Standard block cipher used three times, with either 2 or 3 keys
ACL	Access Control List
ANSI	American National Standards Institute
CLI	Command Line Interface
DAC	Discretionary Access Control
DACL	Discretionary Access Control List
ElGamal	An asymmetric encryption algorithm for public key cryptography
FIPS	Federal Information Processing Standard (NIST)

GUI	Graphical User Interface
HTTPS	HyperText Transport Protocol Secure
IETF	Internet Engineering Task Force
ISO/IEC	International Organization for Standardization / International Engineering Consortium
JDBC	Java Data Base Connectivity
JVM	Java Virtual Machine
MD5	Message Digest 5, a hashing algorithm
MS-DOS	Microsoft Disk Operating System, an early personal computer operating system
NIST	National Institute of Standards and Technology
NTFS	[Microsoft Windows] NT File System (a file system with ACLs)
RAM	Random Access Memory. Non-volatile RAM keeps its data without power.
RMI	Remote Method Invocation
ROM	Read Only Memory. It keeps its data without power.
SACL	System Access Control List
SDC	Secure Download Cabinet (an application packaging and distribution method, like zip)
SHA-1	Secure Hash Algorithm 1 (NIST)
SID	Security Identifier (MS Windows internal user identier)
SMTP	Simple Mail Transport Protocol
SNMP	Simple Network Management Protocol
SQL	Structured Query Language for data base access
SSL	Secure Sockets Layer
TLS	Transport Layer Security
UI	User Interface
VPN	Virtual Private Network

2. TOE Description

The Target of Evaluation (TOE) is Tripwire Manager and Tripwire for Servers 4.6.1.

The TOE consists of a server application component (Tripwire for Servers), a client administrative console application component (Tripwire Manager), and a set of support programs (Tripwire utilities).

The remainder of this section summarizes the TOE architecture.

2.1 TOE Overview

The TOE is a change audit assessment product that can assure the integrity of critical data on system(s) by monitoring file system object attributes for unauthorized or unexpected modification. The TOE accomplishes this by detecting the corrupted or altered files and reporting the occurrence to the system administrators, so corrective actions can be taken. The TOE can monitor the attributes of UNIX files, Windows files, and Windows Registry keys and values for unauthorized or unexpected modification.

The TOE is designed to monitor servers in general. It can monitor servers that run on either Windows or several types of UNIX operating systems. The TOE does not interact with the server as a server but as a program running on an operating system. The TOE administrator configures the server objects to be monitored but the TOE does not provide general user services.

2.2 TOE Architecture

The Tripwire TOE includes three components: Tripwire for Servers, Tripwire Manager, and Tripwire Utilities. The security relevant portions of the TOE are the Tripwire for Servers and the Tripwire Manager components. The Tripwire Utilities component is part of the TOE but does not enforce any security functions. Both of the security relevant components include a crypto module subcomponent[1].

Authorized administrators configure the TOE by creating integrity check rules that specify objects and corresponding object attributes to monitor. These are stored in a policy file configuration file, thus creating a baseline. After making changes to the baseline the administrator can use the TOE to perform integrity checks at regular intervals using IT Environment support such as *crontab* for example on UNIX operating systems or scheduled checks using TOE interfaces.

The TOE provides its own audit mechanism that can generate audit records containing integrity check results and TOE management actions. The TOE does not maintain its own audit trail however – audit records are sent by the TOE to the underlying operating system audit mechanism to add to its audit trail. Auditing in the TOE is not enabled by default and must be enabled by an authorized administrator in the evaluated configuration.

The TOE provides its own crypto module that can generate cryptographic keys and can digitally sign/verify files when stored in its environment. Files that are signed during storage are attribute baselines for objects, configuration files and policy files. The crypto module performs SSL[2] operations to protect communication between TOE components.

The TOE provides two sets of interfaces to control how it operates: a Graphical User Interface (GUI) in the Tripwire Manager component and a Command-Line Interface (CLI) in each Tripwire for Servers component. Both interfaces provide the same administrative functions and have the same restrictions. Administrators can either connect to each Tripwire for Server component and administer it with its CLI or connect to the Tripwire Manager and use its GUI to administer multiple Tripwire for Server components.

The TOE can be described in terms of the following components:

[1] The vendor asserts that correctness of the cryptographic functions provided by the TOE. The cryptographic functions have not been FIPS validated.
[2] The TOE utilizes the TLSv1 protocol; any reference to SSL in the Security Target is a generalization and actually refers to the TLSv1 protocol.

- Tripwire for Servers component – Monitors the object attributes of file system objects for unauthorized or unexpected modification. The set of objects monitored is configurable, but are generally those objects that are critical to the secure operation of the particular server that the TOE is protecting. These may be objects generated by the host operating system or objects generated by the server application. There is a UNIX version of this component as well as a Windows version of this component. The UNIX version is used to monitor UNIX files. The Windows version is used to monitor Windows files and registry keys and values.
    - *twadmin* subcomponent – Used to create and sign the configuration and policy files used by Tripwire for Servers
    - *twprint* subcomponent – Used for reporting when management is performed locally
    - *tripwire* subcomponent – Used for creating a baseline of the target server and for performing integrity checking
    - *twagent* subcomponent – Provides a network interface to the above three Tripwire for Servers subcomponents that is accessible using the Tripwire Manager component.
- Tripwire Manager component – Provides graphical user interface (GUI) administrative console that can be used to manage the Tripwire for Servers component.
- Tripwire Utilities – Provides useful functions for troubleshooting problems with Tripwire for Servers configuration.

The IT Environment of the TOE is composed of the following components

- Operating system – Provides runtime environment for Tripwire for Servers component and JVM. Also supports TOE audit security function by storing audit records.
- Java Virtual Machine (JVM) – Provides runtime environment for Tripwire Manager component.
- Email server – Supports the TOE's ability to send alerts generated by the TOE to administrators using SMTP email.
- SNMP server – Supports the TOE's ability to send alerts generated by the TOE to administrators using SNMPv1.

2.2.1 Physical Boundaries

The components that make up the TOE are:

- Tripwire for Servers component.
    - *twadmin* subcomponent
    - *twprint* subcomponent
    - *tripwire* subcomponent
    - *twagent* subcomponent
- Tripwire Manager component
- Tripwire Utilities

The TOE depends on the following in the IT Environment:

- Operating system – Tripwire For Servers - Solaris 2.6, 7, 8, 9, 10; AIX 5.2, 5.3; Windows NT, 2000, XP Professional, 2003; Red Hat Linux 9.0; Red Hat Enterprise Linux 3.0, 4.0; HP-UX 11, 11i; HP-UX Itanium 11iv2. Tripwire Manager - Solaris 7,8, 9; Windows 2000, XP Professional, 2003; Red Hat Linux 9.0; Red Hat Enterprise Linux 3.0, 4.0.
- JVM – Sun Java 2 JRE v1.4

- Email server – SMTP protocol compatible email server[3].
- SNMP server – SNMPv1 protocol compatible notification server.

The TOE in its intended environment is depicted in the figure below.

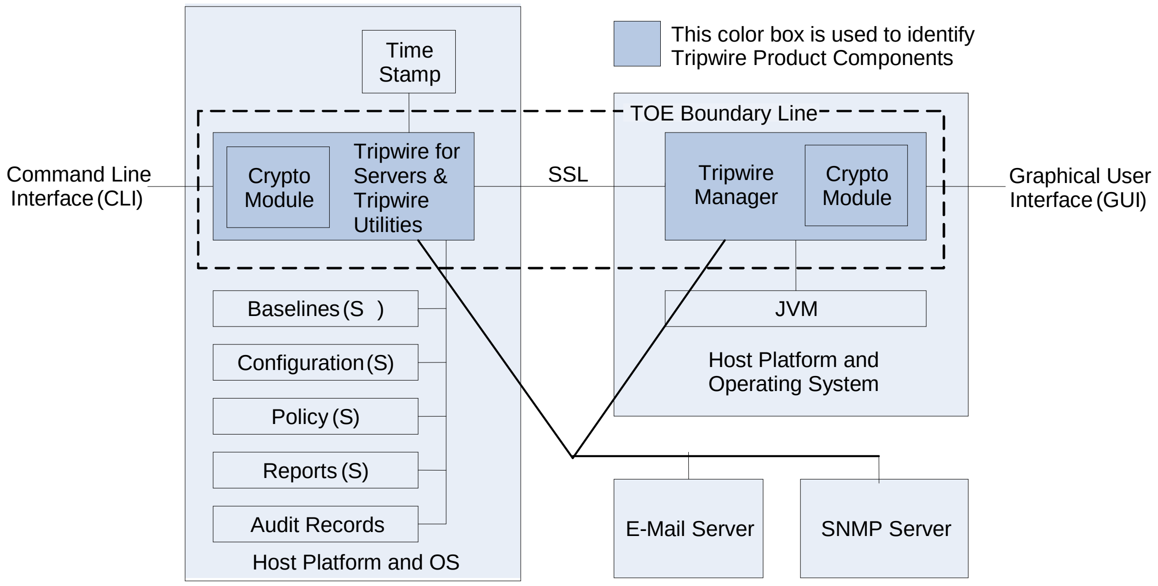
Time Stamp
This color box is used to identify Tripwire Product Components
TOE Boundary Line
Command Line Interface (CLI)
Crypto Module
Tripwire for Servers & Tripwire Utilities
SSL
Tripwire Manager
Crypto Module
Graphical User Interface (GUI)
Baselines (S)
Configuration (S)
Policy (S)
Reports (S)
Audit Records
Host Platform and OS
JVM
Host Platform and Operating System
E-Mail Server
SNMP Server

Figure 1: The TOE and its Environment

Tripwire for Servers and Tripwire Manager may be installed on the same machine or on different machines. (S) indicates files that are signed in storage.

2.2.2 Logical Boundaries

This section identifies the security functions that the TSF provides.

- Change Audit Assessment (EX)
- Audit
- Cryptographic Support
- Identification and Authentication
- Security Management
- TSF Protection

2.2.2.1 Change Audit Assessment (EX)

The TOE is a change audit assessment product that can assure the integrity of critical data on system(s) by monitoring critical data for unauthorized or unexpected modification. The TOE can use the SNMP or email servers provided by the IT Environment to send alert messages.

[3] Note that while the TOE supports the use of MAPI and sendmail for email notifications they were not part of the evaluation.

2.2.2.2 Security audit

The TOE can generate audit events for TOE management events. The audit trail is stored in and protected by the IT environment.

2.2.2.3 Cryptographic support

The TOE digitally signs stored attribute baselines for objects, as well as configuration files and reports written to files. The TOE also uses SSL to protect communication between its components.

2.2.2.4 Identification and authentication

The TOE requires the use of passphrases that conform to certain composition rules before allowing access to its services. The only logon into the TOE is an administrator role logon. Individual users do not login to the TOE.

2.2.2.5 Security management

The TOE provides administrator console interfaces that can be used by authorized administrators to perform all management functions including: starting/stopping of the audit function, specification of integrity check rules and reporting actions, promoting object attribute snapshots to baselines, and reviewing integrity check reports written to files.

2.2.2.6 Protection of the TSF

The TOE uses mutually authenticated SSL communications between Tripwire Manager and Tripwire for Servers components to protect communication between these components.

The Tripwire for Servers component executes as a trusted process within the operating system that is the IT environment. That is, in Unix based operating systems, the TOE executes as ROOT, while on Windows platforms the TOE executes as a SYSTEM process.

The TOE uses features provided by the IT environment to protect itself from external tampering. The TOE utilizes the process mechanism in the IT environment as a protected domain of execution. Also, the TOE uses the abstraction of files and a file protection mechanism (e.g., access control lists) in the IT environment to protect TOE executables, TOE configuration data, and TOE output data. The IT Environment also provides the timestamp used in the audit records.

2.2.2.7 TOE Documentation

Tripwire offers a series of documents that describe the installation process for Tripwire for Servers/Tripwire Manager as well as guidance for subsequent use and administration of the applicable security features. Refer to Section 6 for information about these and other documentation associated with Tripwire for Servers/Tripwire Manager.

3. Security Environment

This section summarizes the threats addressed by the TOE and its environment and assumptions made about the intended environment of the TOE. While the identified threats are mitigated by the security functions implemented in the TOE, with help from its environment, the overall assurance level (EAL3 augmented with ALC_FLR.2) also serves as an indicator of whether the TOE would be suitable for a given environment.

3.1 Threats

T.AUTHENT	An authorized user may incorrectly change TOE data or functions they are authorized to modify.
T.COLLECT	An attacker may be able to change attribute information for targeted objects and have that change go undetected.
T.MANAGE	An administrator may incorrectly install or configure the TOE resulting in ineffective security mechanisms.
T.PROTECT	An attacker may be able to gain unauthorized access to data collected from targeted objects.

3.2 Organizational Security Policies

There are no organizational security policies.

3.3 Secure Usage Assumptions

3.3.1 Intended Usage Assumptions

A.ACCESS	The TOE has access to all the IT System data it needs to perform its functions.
A.ASCOPE	The TOE will be configured to monitor products that it is compatible with and in quantities it can handle.
A.DYNMIC	The TOE will be managed in a manner that allows it to appropriately address changes in the IT System the TOE monitors.

3.3.2 Physical Assumptions

A.LOCATE	The processing resources of the TOE will be located within controlled access facilities, which will prevent unauthorized physical access.
A.PROTCT	The TOE software critical to security policy enforcement will be protected from unauthorized physical modification.

3.3.3 Personnel Assumptions

A.MANAGE	There will be one or more competent individuals assigned to manage the TOE and its supporting platforms and the security of the information they contain.
A.NOEVIL	The authorized administrators are not willfully negligent, or hostile, and will follow and abide by the instructions provided by the TOE and its supporting platforms documentation.
A.NOTRST	The TOE and its supporting platforms can only be accessed by authorized users.

4. Security Objectives

This section summarizes the security objectives for the TOE and its environment.

4.1 Security Objectives for the TOE

O.AUDITING	The TOE shall provide the capability to create records containing integrity check results and security-relevant events associated with roles.
O.AUTHENT	The TOE shall verify the claimed authority of users to assume the administrator role.
O.COLLECT	The TOE shall collect attribute information for targeted objects and maintain a baseline of attributes for each.
O.COMPARE	The TOE shall perform integrity checks on targeted objects by comparing collected attributes of each object against its stored baseline and generating a report containing integrity check results.
O.MANAGE	The TOE shall provide functions such that it can be managed by authorized administrators.
O.PROTECT	The TOE shall protect collected attribute information for targeted objects from external interference or tampering.

4.2 Security Objectives for the IT Environment

OE.AUDRPT	The IT environment will provide functions to select and display audit records.
OE.PROTECT	The IT environment will provide domain separation for the TOE that protects it from external interference and tampering by untrusted users.
OE.STORAGE	The IT environment will provide storage for audit records.
OE.TIME	The IT environment will provide a time source that provides reliable time stamps.

4.3 Security Objectives for the Non-IT Environment

OE.CREDEN	Those responsible for the TOE must ensure that all key file passphrases are protected by the users in a manner that is consistent with IT security.
OE.INSTAL	Those responsible for the TOE must ensure that the TOE is delivered, installed, managed, and operated in a manner which is consistent with IT security.
OE.INTROP	The TOE is configured to monitor on compatible products and has all necessary access to the data it is monitoring.
OE.PERSON	Personnel working as authorized administrators shall be carefully selected and trained for proper operation of the System.
OE.PHYCAL	Those responsible for the TOE must ensure that those parts of the TOE critical to security policy are protected from any physical attack.

5. IT Security Requirements

This section defines the security functional and security assurance requirements for the TOE and associated IT environment components.

In addition to these requirements, the TOE also satisfies a minimum strength of function ‘SOF-medium’. The only applicable (i.e., probabilistic or permutational) security functions are FIA_SOS.1, and FIA_RAU.2(EX), which are levied on the TOE.

5.1 TOE Security Functional Requirements

The following table describes the SFRs that are candidates to be satisfied by Tripwire Manager/Tripwire for Servers.

Requirement Class	Requirement Component
CHG(EX): Change audit assessment	CHG_COL.1(EX): Change Audit Collection
	CHG_ASM.1(EX): Change Audit Assessment
	CHG_REP.1(EX): Change Audit Reporting
FAU: Security audit	FAU_GEN.1(EX): Audit data generation
FCS: Cryptographic support	FCS_CKM.1: Cryptographic key generation
	FCS_CKM.4: Cryptographic key destruction
	FCS_COP.1: Cryptographic operation
FIA: Identification and authentication	FIA_RAU.2(EX): Role authentication before any action
	FIA_SOS.1: Verification of secrets
	FMT_SMF.1: Specification of Management Functions
FPT: Protection of the TSF	FPT_ITT.1: Basic internal TSF data transfer protection
	FPT_RVM.1a: Non-bypassability of the TSP

Table 1 TOE Security Functional Components

5.1.1 Change Audit Assessment (CHG(EX))

5.1.1.1 Change Audit Collection (CHG_COL.1(EX))

Dependencies: FAU_GEN.1 Audit data generation

CHG_COL.1.1(EX) The TSF shall be able to collect the following information from the targeted IT system resource(s):

a.) Configuration information for the establishment of baselines in the TOE and for comparison with previously established baselines, and

b.) Reports of changes to configuration information.

CHG_COL.1.2(EX) The TSF shall collect and record the following information from targeted IT system resource(s):

a.) Date and time of the collection,

b.) Type of information collected,

c.) Identity of the IT system resource from which the information was collected,

d.) Administratively configurable, rule-defined information that is specific to each targeted IT system resource type.

5.1.1.1 Change Audit Assessment (CHG_ASM.1(EX))

CHG_ASM.1.1(EX) The TSF shall compare collected attributes of objects against stored baselines using integrity check rules (rules that specify how comparisons are performed).

5.1.1.2 Change Audit Reporting (CHG_REP.1(EX))

CHG_REP.1.1(EX) The TSF shall take one or more of the following actions if a new element version is not equal to the stored baseline of an object:

a.) Display integrity check results to the console
b.) Write integrity check results to a file
c.) Send integrity check results to administrators using email
d.) Send integrity check results to administrators using SNMP
e.) Generate operating system audit events containing integrity check results
f.) Execute a command
g.) Update or create a baseline

5.1.2 Security audit (FAU)

5.1.2.1 Audit data generation (FAU_GEN.1(EX))

FAU_GEN.1.1(EX) The TSF shall be able to generate an audit record of the auditable events listed in Table 2 Auditable Events.

FAU_GEN.1.2(EX) 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, **[the additional information specified in the Details column of the following Table**:

Component	Event	Details
CHG_COL.1	Integrity Check Result	Any integrity check violations
FCS_CKM.1	Cryptographic key generation	Success and failure of the activity.
FCS_CKM.4	Cryptographic key destruction	Success and failure of the activity.
FCS_COP.1	Cryptographic operation	Success and failure, and the type of cryptographic operation.
FIA_RAU.2(EX)	Role authentication before any action.	None. (None of the FIA_UAU.2 events apply.)
FIA_SOS.1	Verification of secrets	Rejection by the TSF of any rejected secret.
FMT_SMF.1	Specification of management functions.	Use of management functions.
FPT_ITT.1	Basic internal TSF data transfer protection.	None

Table 2 Auditable Events.

5.1.3 Cryptographic support (FCS)

5.1.3.1 Cryptographic key generation (FCS_CKM.1)

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **[listed below]** and specified cryptographic key sizes **[specified for each algorithm]** that meet the following: **[standards noted for each algorithm]**

a.) **El Gamal 1024 bit key pairs in accordance with ISO/IEC 18033-2**
b.) **RNG (random key generation) for 3DES 168 bits in accordance with ANSI X9.52**

5.1.3.2 Cryptographic key destruction (FCS_CKM.4)

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [**overwriting files**] that meets the following: [**none**].

5.1.3.3 Cryptographic operation (FCS_COP.1)

FCS_COP.1.1 The TSF shall perform **[signature generation, signature verification, hash generation, encryption, decryption]** in accordance with a specified cryptographic algorithm **[listed below]** and cryptographic key sizes **[specified for each algorithm]** that meet the following: **[standards noted for each algorithm].**

a.) **Signature generation/verification El Gamal 1024 (ISO/IEC 18033-2)**
b.) **Hash generation SHA-1 (FIPS PUB 180-1, ANSI X9.30 Part 2)**
c.) **Hash generation MD5 (RFC 1321)**
d.) **Encryption/Decryption 3DES 168 (ANSI X9.52)**
e.) **Encryption/Decryption RC4 2048 (RSA Data Security Inc, 1992)**

5.1.4 Identification and authentication (FIA)

5.1.4.1 Role authentication before any action (FIA_RAU.2(EX))

FIA_RAU.2.1(EX) The TSF shall require each user to be successfully authenticated to the administrator role before allowing any other TSF-mediated actions on behalf of that user. (EX)

5.1.4.2 Verification of secrets (FIA_SOS.1)

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that secrets meet **[the following: a) for each attempt to use the authentication mechanisms, the probability that a random attempt will succeed is less than one in 1,000,000,000; and b) any feedback given during each attempt to use the authentication mechanism will reduce the probability of the above metric by only one.]**.

5.1.5 Security management (FMT)

5.1.5.1 Specification of Management Functions (FMT_SMF.1)

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

a.) **Manage change audit security function, including specifying which machines and attributes will be checked**
b.) **Manage audit security function**
c.) **Query and clear integrity check reports**
d.) **Change the passphrase]**.

5.1.6 Protection of the TSF (FPT)

5.1.6.1 Basic internal TSF data transfer protection (FPT_ITT.1)

FPT_ITT.1.1 The TSF shall protect TSF data from **[*disclosure and modification*]** when it is transmitted between separate parts of the TOE.

5.1.6.2 Non-bypassability of the TSP (FPT_RVM.1a)

FPT_RVM.1a The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

5.2 IT Environment Security Functional Requirements

The following table describes the SFRs that are candidates to be satisfied by the IT environment.

Requirement Class	Requirement Component
FAU: Security audit	FAU_SAR.1: Audit review
	FAU_SAR.3: Selectable audit review
	FAU_STG.1: Protected audit trail storage
FMT: Security management	FMT_SMF.1: Specification of management functions
FPT: Protection of the TSF	FPT_RVM.1b: Non-bypassability of the TSP
	FPT_SEP.1: TSF domain separation
	FPT_STM.1: Reliable time stamps

Table 3: IT Environment Security Functional Components

5.2.1 Security audit (FAU)

5.2.1.1 Audit review (FAU_SAR.1)

FAU_SAR.1.1 The ~~TSF~~ IT Environment shall provide **[administrators]** with the capability to read **[all audit information]** from the audit records.

FAU_SAR.1.2 The ~~TSF~~ IT Environment shall provide the audit records in a manner suitable for the user to interpret the information.

5.2.1.2 Selectable audit review (FAU_SAR.3)

FAU_SAR.3.1 The ~~TSF~~ IT Environment shall provide the ability to perform [***searches, sorting***] of audit data based on **[the following criteria:**

a.) **date and time of the event**
b.) **type of event**
c.) **subject identity, and**
d.) **the outcome (success or failure) of the event]**

5.2.1.3 Protected audit trail storage (FAU_STG.1)

FAU_STG.1.1 The ~~TSF~~ IT Environment shall protect the stored audit records from unauthorized deletion.

FAU_STG.1.2 The ~~TSF~~ IT Environment shall be able to [***prevent***] unauthorized modifications to the audit records in the audit trail.

5.2.2 Security management (FMT)

5.2.2.1 Specification of Management Functions (FMT_SMF.1)

FMT_SMF.1.1 The ~~TSF~~ IT Environment shall be capable of performing the following security management functions: **[select and display TOE security audit records]**.

5.2.3 Protection of the TSF (FPT)

5.2.3.1 Non-bypassability of the TSP (FPT_RVM.1b)

FPT_RVM.1b The ~~TSF~~ IT Environment shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

5.2.3.2 TSF domain separation (FPT_SEP.1)

FPT_SEP.1.1 The ~~TSF~~ IT Environment shall maintain a security domain for ~~its own~~ TOE execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.1.2 The ~~TSF~~ IT Environment shall enforce separation between the security domains of subjects in the TSC.

5.2.3.3 Reliable time stamps (FPT_STM.1)

FPT_STM.1.1 The ~~TSF~~ IT Environment shall be able to provide reliable time stamps for ~~its own~~ use by the TOE.

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are the EAL 3 augmented with ALC_FLR.2 components as specified in Part 3 of the Common Criteria. No operations are applied to the assurance components.

Requirement Class	Requirement Component
ACM: Configuration management	ACM_CAP.3: Authorization controls
	ACM_SCP.1: TOE CM coverage
ADO: Delivery and operation	ADO_DEL.1: Delivery procedures
	ADO_IGS.1: Installation, generation, and start-up procedures
ADV: Development	ADV_FSP.1: Informal functional specification
	ADV_HLD.2: Security enforcing high-level design
	ADV_RCR.1: Informal correspondence demonstration
AGD: Guidance documents	AGD_ADM.1: Administrator guidance
	AGD_USR.1: User guidance
ALC: Life cycle support	ALC_DVS.1: Identification of security measures
	ALC_FLR.2: Flaw reporting procedures
ATE: Tests	ATE_COV.2: Analysis of coverage
	ATE_DPT.1: Testing: high-level design
	ATE_FUN.1: Functional testing
	ATE_IND.2: Independent testing - sample
AVA: Vulnerability assessment	AVA_MSU.1: Examination of guidance
	AVA_SOF.1: Strength of TOE security function evaluation
	AVA_VLA.1: Developer vulnerability analysis

Table 4 EAL 3 augmented with ALC_FLR.2 Assurance Components

5.3.1 Configuration management (ACM)

5.3.1.1 Authorization controls (ACM_CAP.3)

ACM_CAP.3.1d The developer shall provide a reference for the TOE.
ACM_CAP.3.2d The developer shall use a CM system.
ACM_CAP.3.3d The developer shall provide CM documentation.
ACM_CAP.3.1c The reference for the TOE shall be unique to each version of the TOE.
ACM_CAP.3.2c The TOE shall be labeled with its reference.
ACM_CAP.3.3c The CM documentation shall include a configuration list and a CM plan.
ACM_CAP.3.4c The configuration list shall uniquely identify all configuration items that comprise the TOE.
ACM_CAP.3.5c The configuration list shall describe the configuration items that comprise the TOE.
ACM_CAP.3.6c The CM documentation shall describe the method used to uniquely identify the configuration items.
ACM_CAP.3.7c The CM system shall uniquely identify all configuration items.
ACM_CAP.3.8c The CM plan shall describe how the CM system is used.
ACM_CAP.3.9c The evidence shall demonstrate that the CM system is operating in accordance with the CM plan.

ACM_CAP.3.10c The CM documentation shall provide evidence that all configuration items have been and are being effectively maintained under the CM system.
ACM_CAP.3.11c The CM system shall provide measures such that only authorized changes are made to the configuration items.
ACM_CAP.3.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.1.2 TOE CM coverage (ACM_SCP.1)

ACM_SCP.1.1d The developer shall provide a list of configuration items for the TOE.
ACM_SCP.1.1c The list of configuration items shall include the following: implementation representation and the evaluation evidence required by the assurance components in the ST.
ACM_SCP.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.2 Delivery and operation (ADO)

5.3.2.1 Delivery procedures (ADO_DEL.1)

ADO_DEL.1.1d The developer shall document procedures for delivery of the TOE or parts of it to the user.
ADO_DEL.1.2d The developer shall use the delivery procedures.
ADO_DEL.1.1c The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to a user's site.
ADO_DEL.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.2.2 Installation, generation, and start-up procedures (ADO_IGS.1)

ADO_IGS.1.1d The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE.
ADO_IGS.1.1c The installation, generation and start-up documentation shall describe all the steps necessary for secure installation, generation and start-up of the TOE.
ADO_IGS.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ADO_IGS.1.2e The evaluator shall determine that the installation, generation, and start-up procedures result in a secure configuration.

5.3.3 Development (ADV)

5.3.3.1 Informal functional specification (ADV_FSP.1)

ADV_FSP.1.1d The developer shall provide a functional specification.
ADV_FSP.1.1c The functional specification shall describe the TSF and its external interfaces using an informal style.
ADV_FSP.1.2c The functional specification shall be internally consistent.
ADV_FSP.1.3c The functional specification shall describe the purpose and method of use of all external TSF interfaces, providing details of effects, exceptions and error messages, as appropriate.
ADV_FSP.1.4c The functional specification shall completely represent the TSF.
ADV_FSP.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ADV_FSP.1.2e The evaluator shall determine that the functional specification is an accurate and complete instantiation of the TOE security functional requirements.

5.3.3.2 Security enforcing high-level design (ADV_HLD.2)

ADV_HLD.2.1d The developer shall provide the high-level design of the TSF.
ADV_HLD.2.1c The presentation of the high-level design shall be informal.
ADV_HLD.2.2c The high-level design shall be internally consistent.

ADV_HLD.2.3c	The high-level design shall describe the structure of the TSF in terms of subsystems.
ADV_HLD.2.4c	The high-level design shall describe the security functionality provided by each subsystem of the TSF.
ADV_HLD.2.5c	The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.
ADV_HLD.2.6c	The high-level design shall identify all interfaces to the subsystems of the TSF.
ADV_HLD.2.7c	The high-level design shall identify which of the interfaces to the subsystems of the TSF are externally visible.
ADV_HLD.2.8c	The high-level design shall describe the purpose and method of use of all interfaces to the subsystems of the TSF, providing details of effects, exceptions and error messages, as appropriate.
ADV_HLD.2.9c	The high-level design shall describe the separation of the TOE into TSP-enforcing and other subsystems.
ADV_HLD.2.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ADV_HLD.2.2e	The evaluator shall determine that the high-level design is an accurate and complete instantiation of the TOE security functional requirements.

5.3.3.3 Informal correspondence demonstration (ADV_RCR.1)

ADV_RCR.1.1d	The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided.
ADV_RCR.1.1c	For each adjacent pair of provided TSF representations, the analysis shall demonstrate that all relevant security functionality of the more abstract TSF representation is correctly and completely refined in the less abstract TSF representation.
ADV_RCR.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.4 Guidance documents (AGD)

5.3.4.1 Administrator guidance (AGD_ADM.1)

AGD_ADM.1.1d	The developer shall provide administrator guidance addressed to system administrative personnel.
AGD_ADM.1.1c	The administrator guidance shall describe the administrative functions and interfaces available to the administrator of the TOE.
AGD_ADM.1.2c	The administrator guidance shall describe how to administer the TOE in a secure manner.
AGD_ADM.1.3c	The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.
AGD_ADM.1.4c	The administrator guidance shall describe all assumptions regarding user behavior that are relevant to secure operation of the TOE.
AGD_ADM.1.5c	The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate.
AGD_ADM.1.6c	The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.
AGD_ADM.1.7c	The administrator guidance shall be consistent with all other documentation supplied for evaluation.
AGD_ADM.1.8c	The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator.
AGD_ADM.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.4.2 User guidance (AGD_USR.1)

AGD_USR.1.1d	The developer shall provide user guidance.
AGD_USR.1.1c	The user guidance shall describe the functions and interfaces available to the non-administrative users of the TOE.

AGD_USR.1.2c	The user guidance shall describe the use of user-accessible security functions provided by the TOE.
AGD_USR.1.3c	The user guidance shall contain warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.
AGD_USR.1.4c	The user guidance shall clearly present all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behavior found in the statement of TOE security environment.
AGD_USR.1.5c	The user guidance shall be consistent with all other documentation supplied for evaluation.
AGD_USR.1.6c	The user guidance shall describe all security requirements for the IT environment that are relevant to the user.
AGD_USR.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.5 Life cycle support (ALC)

5.3.5.1 Identification of security measures (ALC_DVS.1)

ALC_DVS.1.1d	The developer shall produce development security documentation.
ALC_DVS.1.1c	The development security documentation shall describe all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.
ALC_DVS.1.2c	The development security documentation shall provide evidence that these security measures are followed during the development and maintenance of the TOE.
ALC_DVS.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ALC_DVS.1.2e	The evaluator shall confirm that the security measures are being applied.

5.3.5.2 Flaw reporting procedures (ALC_FLR.2)

ALC_FLR.2.1d	The developer shall provide flaw remediation procedures addressed to TOE developers.
ALC_FLR.2.2d	The developer shall establish a procedure for accepting and acting upon all reports of security flaws and requests for corrections to those flaws.
ALC_FLR.2.3d	The developer shall provide flaw remediation guidance addressed to TOE users.
ALC_FLR.2.1c	The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.
ALC_FLR.2.2c	The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.
ALC_FLR.2.3c	The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.
ALC_FLR.2.4c	The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to TOE users.
ALC_FLR.2.5c	The flaw remediation procedures documentation shall describe a means by which the developer receives from TOE users reports and enquiries of suspected security flaws in the TOE.
ALC_FLR.2.6c	The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.
ALC_FLR.2.7c	The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.
ALC_FLR.2.8c	The flaw remediation guidance shall describe a means by which TOE users report to the developer any suspected security flaws in the TOE.
ALC_FLR.2.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.6 Tests (ATE)

5.3.6.1 Analysis of coverage (ATE_COV.2)

ATE_COV.2.1d	The developer shall provide an analysis of the test coverage.
ATE_COV.2.1c	The analysis of the test coverage shall demonstrate the correspondence between the tests identified in the test documentation and the TSF as described in the functional specification.
ATE_COV.2.2c	The analysis of the test coverage shall demonstrate that the correspondence between the TSF as described in the functional specification and the tests identified in the test documentation is complete.
ATE_COV.2.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.6.2 Testing: high-level design (ATE_DPT.1)

ATE_DPT.1.1d	The developer shall provide the analysis of the depth of testing.
ATE_DPT.1.1c	The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TSF operates in accordance with its high-level design.
ATE_DPT.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.6.3 Functional testing (ATE_FUN.1)

ATE_FUN.1.1d	The developer shall test the TSF and document the results.
ATE_FUN.1.2d	The developer shall provide test documentation.
ATE_FUN.1.1c	The test documentation shall consist of test plans, test procedure descriptions, expected test results and actual test results.
ATE_FUN.1.2c	The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed.
ATE_FUN.1.3c	The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests.
ATE_FUN.1.4c	The expected test results shall show the anticipated outputs from a successful execution of the tests.
ATE_FUN.1.5c	The test results from the developer execution of the tests shall demonstrate that each tested security function behaved as specified.
ATE_FUN.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.6.4 Independent testing - sample (ATE_IND.2)

ATE_IND.2.1d	The developer shall provide the TOE for testing.
ATE_IND.2.1c	The TOE shall be suitable for testing.
ATE_IND.2.2c	The developer shall provide an equivalent set of resources to those that were used in the developer’s functional testing of the TSF.
ATE_IND.2.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ATE_IND.2.2e	The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified.
ATE_IND.2.3e	The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

5.3.7 Vulnerability assessment (AVA)

5.3.7.1 Examination of guidance (AVA_MSU.1)

AVA_MSU.1.1d	The developer shall provide guidance documentation.

AVA_MSU.1.1c	The guidance documentation shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.
AVA_MSU.1.2c	The guidance documentation shall be complete, clear, consistent and reasonable.
AVA_MSU.1.3c	The guidance documentation shall list all assumptions about the intended environment.
AVA_MSU.1.4c	The guidance documentation shall list all requirements for external security measures (including external procedural, physical and personnel controls).
AVA_MSU.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_MSU.1.2e	The evaluator shall repeat all configuration and installation procedures to confirm that the TOE can be configured and used securely using only the supplied guidance documentation.
AVA_MSU.1.3e	The evaluator shall determine that the use of the guidance documentation allows all insecure states to be detected.

5.3.7.2 Strength of TOE security function evaluation (AVA_SOF.1)

AVA_SOF.1.1d	The developer shall perform a Strength of TOE Security Function analysis for each mechanism identified in the ST as having a strength of TOE security function claim.
AVA_SOF.1.1c	For each mechanism with a Strength of TOE Security Function claim the strength of TOE security function analysis shall show that it meets or exceeds the minimum strength level defined in the PP/ST.
AVA_SOF.1.2c	For each mechanism with a specific strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the specific strength of function metric defined in the PP/ST.
AVA_SOF.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_SOF.1.2e	The evaluator shall confirm that the strength claims are correct.

5.3.7.3 Developer vulnerability analysis (AVA_VLA.1)

AVA_VLA.1.1d	The developer shall perform a vulnerability analysis.
AVA_VLA.1.2d	The developer shall provide vulnerability analysis documentation.
AVA_VLA.1.1c	The vulnerability analysis documentation shall describe the analysis of the TOE deliverables performed to search for obvious ways in which a user can violate the TSP.
AVA_VLA.1.2c	The vulnerability analysis documentation shall describe the disposition of obvious vulnerabilities.
AVA_VLA.1.3c	The vulnerability analysis documentation shall show, for all identified vulnerabilities, that the vulnerability cannot be exploited in the intended environment for the TOE.
AVA_VLA.1.1e	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
AVA_VLA.1.2e	The evaluator shall conduct penetration testing, building on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.

6. TOE Summary Specification

This chapter describes the security functions and associated assurance measures.

6.1 TOE Security Functions

6.1.1 Change Audit Assessment (EX)

The TOE monitors system files for unauthorized or unexpected modification (i.e. performs integrity checking on targeted files) and reports the occurrence to the system administrators using a configured mechanism. Integrity checking consists of comparing collected attributes of objects against stored baselines of object attributes for each object as follows:

- After configuring a Tripwire for Servers machine, an integrity check can be run at any time. During a check, the TOE compares the data snapshot in the database file to the current state of the system and creates a report of changes.
- If the TOE finds changes, authorized administrators can view the report file to decide if the changes to the system are authorized (for example, caused by an operating system update) or unauthorized (due to malicious or accidental changes).
- If the changes are authorized, authorized administrators should update the database file for that machine to reflect the current state of the system. This prevents these changes from being flagged as violations in the future. If the changes are unauthorized, authorized administrators should take appropriate measures, including restoring files from backup, or changing security procedures to prevent further intrusions.
- After resolving all of the changes, authorized administrators run another integrity check to verify the integrity of the system.
- After an integrity check, authorized administrators may want to update the policy file for a machine to monitor new files, or to change rules that are generating unwanted noise in Tripwire report files.

Authorized administrators configure the integrity checking mechanism by creating integrity check rules that specify objects and corresponding object attributes to monitor in a policy file configuration file. After making changes to the returned baseline, the administrator can use the TOE to perform integrity checks at regular intervals using, for example, *crontab* on UNIX operating systems, or scheduled checks via the Tripwire Manager.

The above operations are depicted in the figure below.

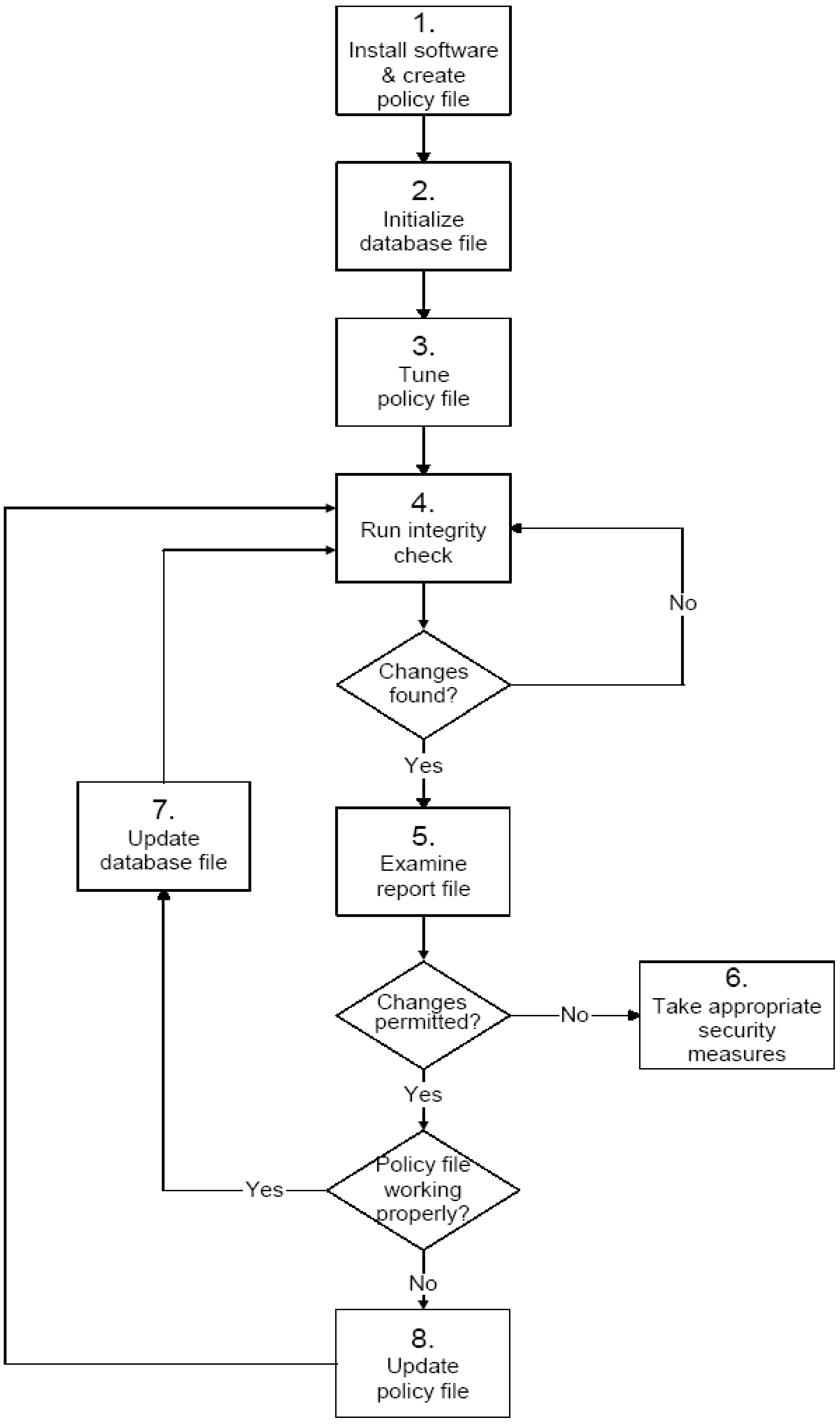

Figure 2: Change audit assessment

The TOE can monitor the following objects and object attributes:

- UNIX file object attributes monitored:
    - The access control list for a file or directory

  - The last date and time when a file or directory was accessed
  - The last date and time when file or directory metadata was modified (or created)
  - The UNIX user group that owns a file or directory
  - The MD5 hash for a file
  - The last time file or directory content was changed by a user
  - Permission and file mode bits
  - The SHA hash for a file
  - The size of a file
  - The owner of the file or directory
- Windows file object attributes monitored:
  - The last time a file or directory was accessed by a user
  - Archive flag
  - A flag that indicates whether the file or directory is compressed
  - The date and time when a file or directory was created
  - A list that specifies the level of file or directory access granted to Windows users or user groups
  - The Windows user group that owns a file or directory
  - Hide flag
  - The MD5 hash of a file
  - Offline flag
  - The owner of the file
  - Read-only flag
  - A list that controls the generation of audit log entries for attempts to access a securable object.
  - The SHA-1 hash of a file
  - The size of a file
  - The number of alternate data streams on a file or directory
  - The MD5 hash for the file or directory alternate data stream(s)
  - The SHA-1 hash for the file or directory alternate data stream(s)
  - System flag
  - Temp flag
  - The date and time when file or directory content was last changed
- Windows registry key and value objects attributes monitored:
  - A list that specifies the level of access granted to Windows users or user groups
  - Type of value (e.g., string, binary)
  - The Windows user or user group that owns a registry key or value
  - The MD5 hash of data in a registry value
  - The owner of a registry key
  - A list that controls the generation of audit log entries for attempts to access a registry key

- The SHA-1 hash of data in a value
- The size of data in a value
- The date and time when a key was last changed

Authorized administrators also configure the integrity checking mechanism by specifying actions to take in response to integrity checks in the policy file configuration file, as well.

- Display integrity check results to the console
- Write integrity check results to a file
- Send integrity check results to administrators using email
- Send integrity check results to administrators using SNMP
- Generate operating system audit events containing integrity check results and TOE management actions
- Execute an operating system shell command

For more information about TOE administration (including more information about integrity check policies), see the user data protection and security management descriptions below.

The Change audit assessment function is designed to satisfy the following security functional requirements:

- CHG_COL.1(EX): The TOE can monitor files, and registry keys and values of a targeted IT system resource by collecting object attribute information and comparing it against stored object attribute baselines.
- CHG_ASM.1(EX): The TOE can compare collected attribute information from a monitored IT system resource using administrator-configured rules.
- CHG_REP.1(EX): The TOE can perform actions in response to object attribute comparisons, specifically: display integrity check results to the console, write integrity check results to a file, send integrity check results to administrators using email, send integrity check results to administrators using SNMP, generate operating system audit events containing integrity check results and TOE management actions, execute a command.

6.1.2 Security audit

The TOE generates audit records containing integrity check results and TOE management actions. The TOE does not maintain its own audit trail however – audit records are sent by the TOE to the underlying operating system audit mechanism to add to its audit trail. Auditing in the TOE is not enabled by default and must be enabled by an authorized administrator in the evaluated configuration.

The Tripwire for Servers component generates audit records that are sent to the operating system audit trail includes date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event. For the subject identity, an individual user is not recorded. The identification of the TOE component is recorded as the source and the OS account that the component runs as is recorded as the user. When a command comes from the Manager to the TSF, the tripwire_sec identity is recorded in the audit trail. The auditable events include:

- Success and failure of digital signature generation/verification operations on attribute baselines for objects, as well as configuration files and reports written to files.
- Success and failure of generating site and local key files.
- Unsuccessful attempts to access key files using a passphrase.
- Use of the management functions:
  - Specification of integrity check rules,
  - Specification of integrity check reporting actions,

  - Integrity check violations,
  - Promotion of collected object attributes to baselines,
  - Review of integrity check reports.

The Security audit function is designed to satisfy the following security functional requirements:

- FAU_GEN.1(EX): The TOE generates audit events for TOE management events.

6.1.3 Cryptographic support

The TOE provides its own crypto module that can generate cryptographic keys and can digitally sign/verify stored attribute baselines for objects, as well as configuration files and reports written to files. The crypto module can also perform cryptographic operations to support SSL connections between TOE components, specifically to protect communication between Tripwire for Servers and Tripwire Manager components.

The TOE uses El Gamal asymmetric cryptography with a 1024 bit signature. The El Gamal signature process uses a paired set of keys: one public and one private key. The TOE generates and stores two sets of public and private keys:

- The site key– An El Gamal key pair used to sign policy and configuration files. This key is so called because it protects files that can be distributed to multiple machines across the entire site.
- The local key– Contains the El Gamal key pair used to sign database and (optionally) report files. This key is so called because it protects files that are specific to each particular (local) system.

The private parts of these key pairs are encrypted using 3DES (DES_EDE3) and accessed using passphrases (each file has a passphrase). A private key is required to sign a file after it is modified so that readers can verify it with the corresponding public key. If it is not signed with the correct key, it cannot be read and will not change the configuration. The private keys are stored in a file called *console.dat*. The Triple-DES key used to protect each private key is derived from that key's passphrase. If a user enters an incorrect passphrase for a given key, the decryption of that key will fail, and the private key will never be derived.

The site private key is used to sign policy and configuration files (i.e., by *tripwire* and *twagent*). The local private key is used to sign database files and reports (if requested). If an incorrect site passphrase is provided during an attempt to write a configuration or policy file, TFS will fail when it tries to decrypt the site.key file. If an incorrect local passphrase is provided during an attempt to write to the database, then TFS will fail when it tries to decrypt the local key file.

The site.key file contains the public site key, and the local.key file contains the public local key. These keys are stored unencrypted. The public keys are used to check the signatures of the files that are signed by their private counterparts.

The TOE supports either signed data files or unsigned data files. By default, data files are signed. The use of signing of the data files is necessary in order for the product to conform to the security requirements set forth in this Security Target. Disabling the signing of the data files removes the product from the evaluated configuration and invalidates any claim that it meets the security assurance rating specified herein.

The Tripwire Manager also contains its own console.dat file where it stores its keys. Tripwire Manager acts as a user to the TFS machine, so it must be able to provide the requested TFS passphrase. The TFS passphrase is stored in the console.dat file.

The TOE uses SSL to protect communication between Tripwire for Servers and Tripwire Manager components. Both Tripwire Manager and Tripwire for Servers are configured to only accept the TLSv1 protocol, to require mutual (two-way) authentication, and to only accept the cipher suite RSA-RC4-MD5. Mutual authentication with RSA requires Tripwire Manager and Tripwire for Servers to each have their own private RSA key and a copy of the other's public RSA key. Tripwire Manager can generate a set of certificates containing new instances of all of these keys at the user's request. The RSA keys that Tripwire Manager generates are 2048-bit keys. Every installation of Tripwire Manager in a deployment must use the same RSA private key, and every installation of Tripwire for

Servers must use the same RSA key. It is the user's responsibility to distribute the correct certificate files to each installation of Tripwire for Servers and Tripwire Manager. Without them, Tripwire for Servers and Tripwire Manager will refuse to communicate.

For more information about using key files to protect communication between Tripwire for Servers and Tripwire Manager components, see the identification and authentication description below.

The Cryptographic support function is designed to satisfy the following security functional requirements:

- FCS_COP.1: The TOE digitally signs stored attribute baselines for objects, as well as configuration files and reports written to files. The TOE also uses SSL when managing one or more Tripwire for Server installations remotely using Tripwire Manager.
- FCS_CKM.1: The TOE generates passphrase protected files with sets of asymmetric keys to sign stored attribute baselines for objects, configuration files and reports written to files. The TOE also uses the keys that it generates to authenticate Tripwire for Servers to Tripwire Managers.
- FCS_CKM.4: The TOE overwrites existing operating system key files when new keys are generated.

6.1.4 Identification and authentication

Administrators (the only role defined by the TOE) can manage the Tripwire Servers individually using each server's Command Line Interface (CLI), or manage them collectively using the Tripwire Manager Graphical User Interface (GUI). The TOE supports only role authentication. Users are not identified.

In order to issue a command that modifies any of the critical files on the Tripwire Server, the administrator must include the proper passphrase as part of the command. The Tripwire Server does not remember the passphrase from one command to the next (i.e., there is no concept of a logon session). This is how role authentication is accomplished through the Tripwire Server CLI.

Tripwire Manager allows the administration of multiple Tripwire Servers. It provides a GUI to users, but it uses the Tripwire Server's CLI to invoke Server commands on behalf of its GUI users. It therefore must also include the correct Server passphrase on each CLI command. To accommodate this, Tripwire Server passphrases are passed to the Tripwire Manager as each Tripwire Server is registered to it. The Tripwire Manager stores the passphrases for its registered Tripwire Servers in its own file called *console.dat.* And, like on Tripwire Servers, the Tripwire Manager's *console.dat* also stores public keys that protect the Tripwire Manager's critical configuration files, and it is also encrypted with Triple-DES whose key is generated by a passphrase.

Administrators using the Tripwire Manager GUI must know and enter the Tripwire Manager's passphrase (when requested) to modify any Tripwire Manager files or to allow Tripwire Manager to access the Tripwire Server passphrases that allow modification of the Tripwire Server files. The Tripwire Manager provides a grace period after a user has entered the passphrase before it requires it to be re-entered. This is 5 minutes by default, but can be changed. Tripwire Manager does not store the passphrase on disk during the grace period, only in memory. This is how role authentication is accomplished through the Tripwire Manager GUI.

These role authentication mechanisms prevent changes to the configuration of the Tripwire Manager and Tripwire Servers by people who do not know the correct passphrases. Passphrases can be changed.

The passphrase mechanism is implemented in the Tripwire Servers by the *twadmin* component. It is also implemented in the Tripwire Manager, which has no components.

A passphrase must adhere to the following rules:

- It must contain at least 8 characters.
- It must contain at least one alphabetic character (a-z, A-Z) or a space character.
- It must contain at least one other printable character that is not alphabetic.

Authentication keys may be changed periodically to avoid brute force attacks, or if administrators suspect that some of the keys have been compromised. To generate new authentication keys for Tripwire for Servers machines, administrators must un-register and re-register each machine with the Tripwire Manager. To generate new Manager keys, administrators must un-register all Tripwire for Servers machines, and delete the “console.key” file on the Tripwire Manager machine.

The Identification and authentication function is designed to satisfy the following security functional requirements:

- FIA_SOS.1: The TOE requires passphrases of a minimum length that meet authentication secret composition rules.
- FIA_RAU.2 (EX): TOE users must enter a valid passphrase before any TSF-mediated functions can be performed.

6.1.5 Security management

The TOE provides two sets of interfaces to control how it operates: a graphical user interface (GUI) provided by Tripwire Manager, and command-line interfaces provided by Tripwire for Servers executables. GUI interfaces provide the same administrative functions and restrictions as command-line interfaces. Administrators can use these interfaces to perform the following:

- Starting and stopping the audit function
- Specification of integrity check rules
- Specification of integrity check actions
- Promotion of collected object attributes to baselines
- Review of integrity check reports
- Perform an integrity check

The *twadmin*, *twprint*, and *tripwire* executable command-line interfaces provide the capability to create a baseline of a target, perform integrity checking, and produce reports of the integrity check results. When the Tripwire Manager is used to perform administrative activities, the *twagent* executable is present. It is responsible for all communications between the Tripwire Manager and the local TFS instance.

The *twadmin* executable is used to create and sign the configuration and policy files used by TFS. The *twprint* executable is used for reporting when management is performed locally.

The *tripwire* executable is responsible for creating a baseline of the target server and for performing integrity checking. To accomplish its tasks, the tripwire executable uses the following files:

- Tripwire Database – This contains the baseline for the target server. It contains the baseline versions of the designated files. This file is stored in a proprietary format and is digitally signed to prevent undetected tampering.
- Tripwire Policy - It contains pathnames to objects that have been identified as requiring integrity monitoring. It is digitally signed to prevent undetected tampering.
- Tripwire Configuration File – Contains processing information such as the paths to the local TFS files, where to send alerts, etc. It is digitally signed to prevent undetected tampering.

The *twagent* executable provides a wrapper for the other three TFS executables when a Tripwire Manager is present. It interprets commands from the Tripwire Manager and invokes the proper TFS component. The *twagent* has one configuration file. It contains basic network information such as port number and network card identification. The *twagent* configuration file points to a local file that contains scheduling information if the Tripwire Manager has configured the TFS to run periodically (this is IT Environment independent). These files are signed to prevent undetected tampering. When the tripwire executable completes, the *twagent* receives a notification. It can let the Tripwire Manager know that a check has completed, its status, and provide any requested reports.

The policy file contains policies or rules for specific objects (such as files, directories, and registry keys/values) on a target machine. Using Tripwire Manager interfaces, authorized administrators specify which system objects Tripwire for Servers scans during integrity checks. By modifying policy file rules, administrators change how

Tripwire for Servers scans objects during integrity checks. The policy file performs two functions. Initially, it acts as a blueprint for the Tripwire database file. When administrators initialize a database file, Tripwire for Servers reads the policy file to determine which objects and properties to include in the database file's baseline data. Later, Tripwire for Servers reads the policy file each time it performs an integrity check. It then scans the system according to the policy file's rules and compares the scan against the baseline data in the database file. Inconsistencies between the two sets of data are reported as violations or errors in the integrity check's report file.

When the TOE is installed, the Tripwire for Servers installation includes a minimal default policy file for the underlying operating system. This default policy file monitors basic components common to all versions of the operating system. It does not monitor the applications or files specific to the installed machine (authorized administrators must add rules for these). Policy files support the following policy file language components:

- Comments – Used to exclude (comment out) text from functional parsing.
- Rules – Used to specify object such as files, directories, and registry keys/values to scan during integrity checks. Rule attributes assign names or severity levels to rules, specify e-mail addresses for e-mailed reports, specify commands to execute if a rule is violated, and specify recursion levels for scanned directories and registry keys/values.
- Variables – Used to substitute for other items in the policy file. Tripwire for Servers provides some predefined variables and allows administrators to define their own.
- Exclusions – Used to exclude files, directories and registry objects from an integrity check.
- Directive – Used to organize rules into major sections and allow conditional logic.

When the TOE reporting mechanism is configured, authorized administrators can specify reports in general with varying levels of detail about the results of an integrity check as follows:

- Level 0 – Contains a single line summary of total adds, removes and changes
- Level 1 – Contains a parsable list of all violated objects
- Level 2 – Contains a summary report listing violations by section and rule name
- Level 3 – Is the default report level; contains expected and observed properties for each violation; more concise than level 4
- Level 4 – Contains full report; maximum level of detail

The Security management function is designed to satisfy the following security functional requirements:

- FMT_SMF.1: The TOE provides administrators with the ability to perform all management functions, including: starting/stopping of the audit function, specification of integrity check rules and reporting actions (including which machines and attributes will be checked), promoting object attribute snapshots to baselines, reviewing and clearing integrity check reports written to files, changing the passphrase (see I&A discussion), and performing an integrity check.

6.1.6 Protection of the TSF

The TOE uses its SSL mechanism to protect TSF data from disclosure and modification when it is being transmitted between Tripwire for Manager and Tripwire for Servers components. All administrators are required to log into the TOE before performing any administrative functions.

The Protection of the TSF function is designed to satisfy the following security functional requirements:

- FCS_COP.1: The TOE signs its configuration files, policy files, and baseline files before storing them in the host operating system's file system. The TOE digitally signs its scanning policy files, reporting files, and baseline files before storing them in the host operating system's file system.
- FPT_ITT.1: The TOE uses SSL when managing one or more Tripwire for Servers remotely using Tripwire Manager.

- FPT_RVM.1a: The TOE provides a well-defined interface that administrators use to access the TOE and that the TOE uses for performing integrity checks.

6.2 TOE Security Assurance Measures

6.2.1 Configuration management

The configuration management measures applied by Tripwire ensure that configuration items are uniquely identified, and that documented procedures are used to control and track changes that are made to the TOE. Tripwire ensures changes to the implementation representation are controlled. Tripwire performs configuration management on the TOE implementation representation, design documentation, tests and test documentation, user and administrator guidance, delivery and operation documentation, life-cycle documentation, vulnerability analysis documentation, and configuration management documentation.

These activities are documented in:

- Tripwire, Inc. Tripwire Enterprise 5.2, Tripwire for Servers 4.6, Tripwire Manager 4.6.1 Configuration Management Plan, TW-ACM1-04, Version 0.4, April 3, 2009

The Configuration management assurance measure satisfies the following EAL 3 augmented with ALC_FLR.2 assurance requirements:

- ACM_CAP.3
- ACM_SCP.1

6.2.2 Delivery and operation

Tripwire provides delivery documentation and procedures to identify the TOE, secure the TOE during delivery, and provide necessary installation and generation instructions. Tripwire's delivery procedures describe all applicable procedures to be used to prevent in inappropriate access to the TOE. Tripwire also provides documentation that describes the steps necessary to install Tripwire for Servers/Tripwire Manager in accordance with the evaluated configuration.

These activities are documented in:

- Tripwire Manager and Tripwire for Servers Delivery Procedures Delivery Procedures, Rev 0.8, September 25, 2007, TW-TFSADO1-08
- Tripwire® for Servers Installation Guide 4.6, TW1002-12
- Tripwire Manager Quick Start 4.6, TW1052-07
- Release notes:
    - Tripwire for Servers version 4.6.1 Release Notes Addendum December 2008
    - Tripwire for Servers 4.6.1 README December 2008
    - Tripwire Manager 4.6.1 README December 2008
    - Tripwire for Servers version 4.6.1 for Windows December 2008
    - Tripwire for Servers version 4.6.1 for UNIX Operating Systems December 2008
    - Tripwire Manager version 4.6.1 December 2008

The Delivery and operation assurance measure satisfies the following EAL 3 augmented with ALC_FLR.2 assurance requirements:

- ADO_DEL.1
- ADO_IGS.1

6.2.3 Development

Tripwire has numerous documents describing all facets of the design of the TOE. In particular, they have a functional specification that describes the accessible TOE interfaces; a high-level design that decomposes the TOE architecture into subsystems and describes each subsystem and their interfaces; and, correspondence documentation that explains how each of the design abstractions correspond from the TOE summary specification in the Security Target to the subsystems.

These activities are documented in:

- Tripwire Manager and Tripwire for Servers Version 4.6 Design Document (HLD, FSP, and RCR), Version 2.4, March 9, 2009
- Tripwire for Servers 4.6 User Guide, TW1005-08
- Tripwire Manager 4.6 User Guide, TW1004-10
- Tripwire Manager & Tripwire for Servers 4.6 Reference Guide, TW1003-13
- Tripwire for Servers for UNIX Quick Reference Card, TW1007-06
- Tripwire for Servers for Windows Quick Reference Card, TW1008-07

The Development assurance measure satisfies the following EAL 3 augmented with ALC_FLR.2 assurance requirements:

- ADV_FSP.1
- ADV_HLD.2
- ADV_RCR.1

6.2.4 Guidance documents

Tripwire provides administrator and user guidance on how to utilize the TOE security functions and warnings to administrators and users about actions that can compromise the security of the TOE.

These activities are documented in:

- Tripwire for Servers 4.6 User Guide, TW1005-08
- Tripwire Manager 4.6 User Guide, TW1004-10
- Tripwire Manager 4.6 Quick Start, TW1052-07
- Tripwire Manager & Tripwire for Servers 4.6 Reference Guide, TW1003-13
- Tripwire for Servers for UNIX Quick Reference Card, TW1007-06
- Tripwire for Servers for Windows Quick Reference Card, TW1008-07
- Tripwire® for Servers Installation Guide 4.6, TW1002-12
- Release notes:
    - Tripwire for Servers version 4.6.1 Release Notes Addendum December 2008
    - Tripwire for Servers 4.6.1 README December 2008
    - Tripwire Manager 4.6.1 README December 2008
    - Tripwire for Servers version 4.6.1 for Windows December 2008
    - Tripwire for Servers version 4.6.1 for UNIX Operating Systems December 2008
    - Tripwire Manager version 4.6.1 December 2008

The Guidance documents assurance measure satisfies the following EAL 3 augmented with ALC_FLR.2 assurance requirements:

- AGD_ADM.1
- AGD_USR.1

6.2.5 Life cycle support

Tripwire ensures the adequacy of the procedures used during the development and maintenance of the TOE through its life-cycle. Tripwire includes security controls on the development environment that are adequate to provide the confidentiality and integrity of the TOE design and implementation that is necessary to ensure the secure operation of the TOE. In addition, Tripwire identifies and tracks reported flaws, ensuring that they are addressed and corrections and corrective measures are made available as applicable.

These activities are documented in:

- Tripwire, Inc. Tripwire Enterprise 5.2, Tripwire for Servers 4.6, Tripwire Manager 4.6 Lifecycle, TW-ALC1-03, Version 0.3b, September 21, 2007

The Life cycle support assurance measure satisfies the following EAL 3 augmented with ALC_FLR.2 assurance requirements:

- ALC_DVS.1
- ALC_FLR.2

6.2.6 Tests

Tripwire has a test plan that describes how each of the necessary security functions is tested, along with the expected test results. Tripwire has documented each test as well as an analysis of test coverage and depth demonstrating that the security aspects of the design evident from the functional specification and high-level design are appropriately tested. Actual test results are created on a regular basis to demonstrate that the tests have been applied and that the TOE operates as designed.

These activities are documented in:

- Test case spreadsheet
- Test case zip (The table in ATE_COV.2-1 identifies the test cases that evaluation team found applicable.)
- Tripwire, Inc Tripwire Manager and Tripwire for Servers Common Criteria Test Plan, Revision 0.4, January 8, 2008, TW-TFSATE1-01

The Tests assurance measure satisfies the following EAL 3 augmented with ALC_FLR.2 assurance requirements:

- ATE_COV.2
- ATE_DPT.1
- ATE_FUN.1
- ATE_IND.2

6.2.7 Vulnerability assessment

The TOE administrator and user guidance documents describe the operation of Tripwire for Servers/Tripwire Manager and how to maintain a secure state. These guides also describe all necessary operating assumptions and security requirements outside the scope of control of the TOE. They have been developed to serve as complete, clear, consistent, and reasonable administrator and user references.

Tripwire has conducted a strength of function analysis wherein all permutational or probabilistic security mechanisms have been identified and analyzed resulting in a demonstration that all of the relevant mechanisms fulfill the minimum strength of function claim, SOF-Medium.

Tripwire performs regular vulnerability analyses of the entire TOE (including documentation) to identify obvious weaknesses that can be exploited in the TOE.

These activities are documented in:

- Tripwire, Inc. Tripwire for Servers 4.6, Tripwire Manager 4.6 Strength of Function Analysis
- Tripwire Manager and Tripwire for Servers version 4.6, Vulnerabilities Assessment
- Guidance Documentation

The Vulnerability assessment assurance measure satisfies the following EAL 3 augmented with ALC_FLR.2 assurance requirements:

- AVA_MSU.1
- AVA_SOF.1
- AVA_VLA.1

7. Protection Profile Claims

This ST makes no Protection Profile conformance claim.

8. Rationale

This section provides the rationale for completeness and consistency of the Security Target. The rationale addresses the following areas:

- Security Objectives;
- Security Functional Requirements;
- Security Assurance Requirements;
- Strength of Functions;
- Requirement Dependencies;
- TOE Summary Specification; and,
- PP Claims.

8.1 Security Objectives Rationale

This section shows that all secure usage assumptions, organizational security policies, and threats are completely covered by security objectives. In addition, each objective counters or addresses at least one assumption, organizational security policy, or threat.

8.1.1 Complete Coverage – Threats

This section shows that all secure usage assumptions, organizational security policies, and threats are completely covered by security objectives. In addition, each objective counters or addresses at least one assumption, organizational security policy, or threat.

	TOE						**IT Env.**				**Non-IT Env.**				
	O.AUDITING	**O.AUTHENT**	**O.COLLECT**	**O.COMPARE**	**O.MANAGE**	**O.PROTECT**	**OE.AUDRPT**	**OE.PROTECT**	**OE.STORAGE**	**OE.TIME**	**OE.CREDEN**	**OE.INSTAL**	**OE.INTEROP**	**OE.PERSON**	**OE.PHYCAL**
T.AUTHENT		X			X							X		X	
T.COLLECT	X		X	X			X		X	X			X		
T.MANAGE	X				X						X	X		X	
T.PROTECT						X		X							X

8.1.1.1 T.AUTHENT

An authorized user may incorrectly change TOE data or functions they are authorized to modify.

This Threat is satisfied by ensuring that:

- O.MANAGE: The TOE provides management functions for use by its administrators.
- O.AUTHENT: The TOE allows access to management functions only by authorized administrators.
- OE.INSTAL: The TOE will be operated in a manner that is consistent with IT security.

- OE.PERSON: Authorized administrators are carefully selected and trained for proper operation of the system.

8.1.1.2 T.COLLECT

An attacker may be able to change attribute information for targeted objects and have that change go undetected.

This Threat is satisfied by ensuring that:

- O.AUDITING: The TOE can create records containing integrity check results.
- O.COLLECT: The TOE collects attribute information for targeted objects, thus allowing unauthorized changes in the targeted object to be detected quickly, so that recovery, isolation, or restoration actions can be initiated. The TOE cannot prevent the changes from occurring, but it provides notification when things are changed, something the targeted object may not be able to do in either its normal or compromised state.
- O.COMPARE: The TOE compares collected attribute information for targeted objects against stored baselines. This is another step in monitoring targeted objects for unauthorized modification.
- OE.INTEROP: The TOE is interoperable with the IT Systems it monitors.
- OE.STORAGE: The IT environment will provide storage for audit records.
- OE.AUDRPT: The IT environment will provide functions to select and display audit records.
- OE.TIME: The TOE includes time stamps on all collected records.

8.1.1.3 T.MANAGE

An administrator may incorrectly install or configure the TOE resulting in ineffective security mechanisms.

This Threat is satisfied by ensuring that:

- O.AUDITING: The TOE can create records containing security relevant events.
- O.MANAGE: The TOE provides administrative interfaces that can be used to administer its security functions.
- OE.CREDEN: The credentials for administering the TOE will be protected in a manner that is consistent with IT security.
- OE.INSTAL: The TOE will be managed in a manner that is consistent with IT security.
- OE.PERSON: Authorized administrators are carefully selected and trained for proper operation of the system.

8.1.1.4 T.PROTECT

An attacker may be able to gain unauthorized access to data collected from targeted objects.

This Threat is satisfied by ensuring that:

- O.PROTECT: The TOE protects collected attribute information for targeted objects.
- OE.PHYCAL: The parts of the TOE critical to security policy are protected from physical attack.
- OE.PROTECT: The TOE environment provides domain separation for the TOE that protects it from external interference and tampering by untrusted users. This is provided by the process separation mechanism of the host operating system.

8.1.2 Complete Coverage – Policy

There are no organization security policies.

8.1.3 Complete Coverage – Environmental Assumptions

This section provides evidence that coverage of the Non-IT security objectives by the environmental assumptions. The following table shows this assumption to objective mapping.

Assumption Class	Assumption	OE.INSTAL	OE.PHYCAL	OE.CREDEN	OE.PERSON	OE.INTROP
Intended usage assumptions	**A.ACCESS**					X
	A.ASCOPE					X
	A.DYNMIC				X	X
Physical assumptions	**A.LOCATE**		X			
	A.PROTCT		X			
Personnel assumptions	**A.MANAGE**				X	
	A.NOEVIL	X			X	
	A.NOTRST		X	X		

8.1.3.1 A.ACCESS

The TOE has access to all the IT System data it needs to perform its functions.

This Assumption is satisfied by ensuring that:

- OE.INTROP: The OE.INTROP objective ensures the TOE has the needed access.

8.1.3.2 A.ASCOPE

The TOE will be configured to monitor products that it is compatible with and in quantities it can handle.

This Assumption is satisfied by ensuring that:

- OE.INTROP: The OE.INTROP objective ensures the TOE compatible with the IT system it is monitoring.

8.1.3.3 A.DYNMIC

The TOE will be managed in a manner that allows it to appropriately address changes in the IT System the TOE monitors.

This Assumption is satisfied by ensuring that:

- OE.INTROP: The OE.INTROP objective ensures the TOE has the proper access to the IT System.
- OE.PERSON: The OE.PERSON objective ensures that the TOE will be managed appropriately.

8.1.3.4 A.LOCATE

The processing resources of the TOE will be located within controlled access facilities, which will prevent unauthorized physical access.

This Assumption is satisfied by ensuring that:

- OE.PHYCAL: Ensures that those responsible for the TOE protect the security critical parts of the TOE from physical attack.

8.1.3.5 A.PROTCT

The TOE software critical to security policy enforcement will be protected from unauthorized physical modification.

This Assumption is satisfied by ensuring that:

- OE.PHYCAL: The OE.PHYCAL provides for the physical protection of the TOE.

8.1.3.6 A.MANAGE

There will be one or more competent individuals assigned to manage the TOE and its supporting platforms and the security of the information they contain.

This Assumption is satisfied by ensuring that:

- OE.PERSON: The OE.PERSON objective ensures all authorized administrators are qualified and trained to manage the TOE.

8.1.3.7 A.NOEVIL

The authorized administrators are not careless, willfully negligent, or hostile, and will follow and abide by the instructions provided by the TOE and its supporting platforms documentation.

This Assumption is satisfied by ensuring that:

- OE.INSTAL: Ensures that the TOE is properly installed and operated.
- OE.PERSON: Ensures that personnel serving as TOE administrators will be carefully selected and well trained for proper operation of the system.

8.1.3.8 A.NOTRST

The TOE and its supporting platforms can only be accessed by authorized users.

This Assumption is satisfied by ensuring that:

- OE.PHYCAL: The OE.PHYCAL objective provides for physical protection of the TOE to protect against unauthorized access.
- OE.CREDEN: The OE.CREDEN objective supports this assumption by requiring protection of all authentication data.

8.2 Security Requirements Rationale

This section provides evidence supporting the internal consistency and completeness of the components (requirements) in the Security Target. **Table 5** indicates the requirements that effectively satisfy the individual objectives.

8.2.1 Security Functional Requirements Rationale

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

	O.AUDITING	O.AUTHENT	O.COLLECT	O.COMPARE	O.MANAGE	O.PROTECT	OE.AUDRPT	OE.PROTECT	OE.TIME	OE.STORAGE
CHG_COL.1(EX)	X		X							
CHG_ASM.1(EX)				X						
CHG_REP.1(EX)			X	X						
FAU_GEN.1(EX)	X									
FCS_CKM.1						X				
FCS_CKM.4						X				
FCS_COP.1						X				
FIA_RAU.2(EX)		X								
FIA_SOS.1		X								
FMT_SMF.1					X					
FPT_ITT.1						X				
FPT_RVM.1a						X				
IT Environment SFRs										
FAU_SAR.1							X			
FAU_SAR.3							X			
FAU_STG.1										X
FMT_SMF.1							X			
FPT_RVM.1b								X		
FPT_SEP.1								X		
FPT_STM.1									X	

Table 5 Objective to Requirement Correspondence

8.2.1.1 O.AUDITING

The TOE shall provide the capability to create records containing integrity check results and security-relevant events associated with roles.

This TOE Security Objective is satisfied by ensuring that:

- CHG_COL.1(EX): The TOE generates audit events for integrity check results.
- FAU_GEN.1(EX): The TOE generates audit events for TOE management events.

8.2.1.2 O.AUTHENT

The TOE shall verify the claimed identity of users.

This TOE Security Objective is satisfied by ensuring that:

- FIA_RAU.2(EX): The TOE requires authorized administrators to possess pass phrase-protected key files containing El Gamal secret keys that are used to sign challenges when a Tripwire Manager component authenticates to a Tripwire for Servers component.
- FIA_SOS.1: The TOE provides a user authentication mechanism with only a one in over a 1,000,000,000 chance of successful guessing at random.

8.2.1.3 O.COLLECT

The TOE shall collect attribute information for targeted objects and maintain a baseline of attributes for each.

This TOE Security Objective is satisfied by ensuring that:

- CHG_COL.1(EX): The TOE can monitor files, and registry keys and values of a targeted IT system resource by collecting object attribute information and comparing it against stored object attribute baselines.
- CHG_REP.1(EX): The TOE creates and updates baselines of object elements.

8.2.1.4 O.COMPARE

The TOE shall perform integrity checks on targeted objects by comparing collected attributes of each object against its stored baseline and generating a report containing integrity check results.

This TOE Security Objective is satisfied by ensuring that:

- CHG_ASM.1(EX): The TOE can compare collected attribute information from a monitored IT system resource using administrator-configured rules.
- CHG_REP.1(EX): The TOE can perform actions in response to object attribute comparisons, specifically: display integrity check results to the console, write integrity check results to a file, send integrity check results to administrators using email, send integrity check results to administrators using SNMP, generate operating system audit events containing integrity check results and TOE management actions, execute a command.

8.2.1.5 O.MANAGE

The TOE shall provide functions such that it can be managed by authorized users.

This TOE Security Objective is satisfied by ensuring that:

- FMT_SMF.1: The TOE provides administrators with the ability to perform all management functions, including: starting/stopping of the audit function, specification of integrity check rules and reporting actions (including which machines and attributes will be checked), promoting object attribute snapshots to baselines, and reviewing and clearing integrity check reports written to files.

8.2.1.6 O.PROTECT

The TOE shall protect collected attribute information for targeted objects from external interference or tampering.

This TOE Security Objective is satisfied by ensuring that:

- FCS_COP.1: The TOE digitally signs stored attribute baselines for objects, as well as configuration files and reports written to files. This prevents the object monitoring mechanism from generating a erroneous change alarm caused by using a baseline that has been tampered with, which could result in object values being restored to invalid values from the tampered baseline. The TOE also uses SSL when communicating with remote instances of Tripwire for Server.
- FCS_CKM.1: The TOE generates passphrase protected files with sets of asymmetric keys to sign stored attribute baselines for objects, configuration files and reports written to files. The TOE also uses the keys that it generates to authenticate Tripwire for Servers to Tripwire Managers.
- FCS_CKM.4: The TOE securely overwrites keys when they are no longer needed.

- FPT_ITT.1: The TOE uses SSL when managing one or more Tripwire for Server installations remotely using Tripwire Manager.
- FPT_RVM.1a: The TOE ensures that the administrative interface cannot be bypassed and that when performing integrity checks, the interfaces cannot be bypassed.

8.2.1.7 OE.AUDRPT

The IT environment will provide functions to select and display audit records.

This TOE Security Objective is satisfied by ensuring that:

- FAU_SAR.1: The TOE provides administrators the ability to read from the audit trail using administrator console interfaces.
- FAU_SAR.3: The TOE provides administrators the ability to search and sort audit data using administrator console interfaces based on date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event.
- FMT_SMF.1: The TOE environment supports the selection and display of security audit records.

8.2.1.8 OE.PROTECT

The IT environment will provide domain separation for the TOE that protects it from external interference and tampering by untrusted users.

This TOE Security Objective is satisfied by ensuring that:

- FPT_SEP.1: The TOE environment protects the TOE from interference or tampering by untrusted subjects.
- FPT_RVM.1b: The TOE environment protects the TOE by ensuring its security polices are always enforced.

8.2.1.9 OE.TIME

The IT environment will provide a time source that provides reliable time stamps.

This TOE Security Objective is satisfied by ensuring that:

- FPT_STM.1: The operating system provides reliable time stamps to the TOE.

8.2.1.10 OE.STORAGE

The IT environment will provide storage for audit records.

This TOE Security Objective is satisfied by ensuring that:

- FPT_STG.1: The TOE environment provides audit record storage.

8.3 Security Assurance Requirements Rationale

The base assurance level was augmented to EAL3 augmented with ALC_FLR.2, because flaw remediation procedures provide greater assurance that security-related bugs will be fixed in a widely distributed commercial product.

EAL3 was selected as the assurance level because the TOE is a commercial product whose users require a moderate to high level of independently assured security. Tripwire Manager/Tripwire for Servers is targeted at a relatively benign environment with good physical access security and competent administrators. Within such environments it is assumed that attackers will have little attack potential. As such, EAL3 is appropriate to provide the assurance necessary to counter the limited potential for attack.

8.4 Strength of Functions Rationale

The TOE is targeted at a generalized IT environment with good physical access security and competent administrators. Within such environments it is assumed that attackers will have a moderate attack potential. As such, a Strength of Function of 'medium' is appropriate for the intended environment.

The only applicable mechanisms (i.e., those that are probabilistic or permutational) are related to identification and authentication (FIA_RAU.2(EX) and FIA_SOS.1).

8.5 Requirement Dependency Rationale

The following table demonstrates that all dependencies among the claimed security requirements are satisfied and therefore the requirements work together to accomplish the overall objectives defined for the TOE.

ST Requirement	CC Dependencies	ST Dependencies
CHG_COL.1(EX)	FPT_STM.1	FPT_STM.1
CHG_ASM.1(EX)	CHG_COL.1(EX), FPT_STM.1	CHG_COL.1(EX), FPT_STM.1
CHG_REP.1(EX)	CHG_ASM.1(EX), FPT_STM.1	CHG_ASM.1(EX), FPT_STM.1
FAU_GEN.1(EX)	FPT_STM.1	FPT_STM.1
FAU_STG.1	FAU_GEN.1	FAU_GEN.1
FCS_CKM.1	[FCS_CKM.2 or FCS_COP.1] and FCS_CKM.4 and FMT_MSA.2	FCS_COP.1 and FCS_CKM.4. See Note 1.
FCS_CKM.4	[FDP_ITC.1 or FCS_CKM.1] and FMT_MSA.2	FCS_CKM.1. See Note 1.
FCS_COP.1	[FDP_ITC.1 or FCS_CKM.1] and FCS_CKM.4 and FMT_MSA.2	FCS_CKM.1 and FCS_CKM.4. See Note 1.
FIA_RAU.2(EX)	none	none
FIA_SOS.1	none	none
FMT_SMF.1	none	none
FPT_ITT.1	none	none
FPT_RVM.1a,b	none	none
FPT_SEP.1	none	none
FPT_STM.1	none	none
ACM_CAP.3	ALC_DVS.1	ALC_DVS.1
ACM_SCP.1	ACM_CAP.3	ACM_CAP.3
ADO_DEL.1	none	none
ADO_IGS.1	AGD_ADM.1	AGD_ADM.1
ADV_FSP.1	ADV_RCR.1	ADV_RCR.1
ADV_HLD.2	ADV_FSP.1 and ADV_RCR.1	ADV_FSP.1 and ADV_RCR.1
ADV_RCR.1	none	none
AGD_ADM.1	ADV_FSP.1	ADV_FSP.1
AGD_USR.1	ADV_FSP.1	ADV_FSP.1
ALC_DVS.1	none	none
ALC_FLR.2	none	none
ATE_COV.2	ADV_FSP.1 and ATE_FUN.1	ADV_FSP.1 and ATE_FUN.1
ATE_DPT.1	ADV_HLD.1 and ATE_FUN.1	ADV_HLD.2 and ATE_FUN.1
ATE_FUN.1	none	none
ATE_IND.2	ADV_FSP.1 and AGD_ADM.1 and AGD_USR.1 and ATE_FUN.1	ADV_FSP.1 and AGD_ADM.1 and AGD_USR.1 and ATE_FUN.1
AVA_MSU.1	ADO_IGS.1 and ADV_FSP.1 and	ADO_IGS.1 and ADV_FSP.1 and

	AGD_ADM.1 and AGD_USR.1	AGD_ADM.1 and AGD_USR.1
AVA_SOF.1	ADV_FSP.1 and ADV_HLD.1	ADV_FSP.1 and ADV_HLD.2
AVA_VLA.1	ADV_FSP.1 and ADV_HLD.1 and AGD_ADM.1 and AGD_USR.1	ADV_FSP.1 and ADV_HLD.2 and AGD_ADM.1 and AGD_USR.1

Notes:

1. The FCS_CKM.1, FCS_CKM.4, and FCS_COP.1 dependency on FMT_MSA.2 is not satisfied because the TOE does not provide functionality to ensure that only secure values are accepted for security attributes. Cryptographic keys are provided at installation and the TOE cannot ensure that they are secure. The TOE's key files are encrypted at installation time and only the administrators know the encryption passphrase. The TOE attempts to use any passphrase a user presents at login to decrypt the key file.

8.6 Explicitly Stated Requirements Rationale

CHG requirements were created to specifically address the object attribute information collected and analyzed by the TOE. The audit family of the CC (FAU) was used as a model for creating these requirements. The purpose of this family of requirements is to address the unique nature of object attribute data and provide for requirements about collecting, reviewing and managing the data. These requirements depend on FPT_STM.1 when generating audit events for TOE management actions, when time stamping collected object attribute information and baselines, and when generating reports. The audit events for these requirements were also modeled after the FAU class of requirements. As such, only the collection function (like the audit generation function) is audited.

The FIA_GEN.1(EX) requirement is explicitly stated because the TOE does not audit the starting and stopping of the audit function. There is an audit event that captures a configuration file change but no specific event is generated to show that audit is started or stopped. The FPT_STM.1 dependency remains from the original CC SFR as a timestamp is needed in the audit trail.

The FIA_RAU.2(EX) requirement was created to address the use of a passphrase mechanism that does not individually identify users. The I&A family of the CC (FIA) was used as a model for creating this requirement. The FIA_RAU.2 requirement has no dependencies.

8.7 TOE Summary Specification Rationale

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

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

	Change audit assessment	Security audit	Cryptographic support	Identification and authentication	Security management	Protection of the TSF
CHG_COL.1(EX)	X					
CHG_ASM.1(EX)	X					
CHG_REP.1(EX)	X					
FAU_GEN.1(EX)		X				
FCS_CKM.1			X			
FCS_CKM.4			X			
FCS_COP.1			X			
FIA_RAU.2(EX)				X		
FIA_SOS.1				X		
FMT_SMF.1					X	
FPT_ITT.1						X
FPT_RVM.1a						X

Table 6 Security Functions vs. Requirements Mapping

8.8 PP Claims Rationale

This ST makes no Protection Profile conformance claim.