BeyondTrust PowerBroker UNIX® + Linux® Edition V9.1

Security Target

Version 1.0
3 August 2016

Prepared for:

BeyondTrust Software, Inc.
5090 N. 40th Street
Phoenix, AZ 85018

Prepared by:

leidos

Accredited Testing & Evaluation Labs
6841 Benjamin Franklin Drive
Columbia, Maryland 21046

LIST OF TABLES

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is PowerBroker UNIX® + Linux® Edition V9.1, provided by BeyondTrust Software, Inc. BeyondTrust PowerBroker is a security management product that provides the capability to delegate access to operating system functions available to specific privileged accounts (e.g., 'root') and offer those functions in a controlled and granular fashion to other specific and suitably trusted users. The TOE provides both Enterprise Security Policy Management and Access Control functions. The focus of this evaluation is on the TOE functionality supporting the claims in the ESM Access Control and Policy Management Protection Profiles (See section 1.2 for specific version information). The security functionality specified in [pp_esm_ac_v2.1] and [pp_esm_pm_v2.1] includes access control policy management and enforcement, protection of communication channels, reliance on enterprise authentication, and auditing of security-relevant events.

The Security Target contains the following additional sections:

-

UID	Also referred to as uid: User ID or User Identity

- TOE Description (Section 2)
- Security Problem Definition (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
- TOE Summary Specification (Section 6)
- Protection Profile Claims (Section 7)
- Rationale (Section 8).

1.1 Security Target, TOE and CC Identification

ST Title – BeyondTrust PowerBroker UNIX® + Linux® Edition Security Target

ST Version – Version 1.0

ST Date – 3 August 2016

TOE Identification – BeyondTrust PowerBroker ® UNIX® + Linux® Edition V9.1

TOE Developer – BeyondTrust Software, Inc.

Evaluation Sponsor – BeyondTrust Software, Inc.

CC Identification – *Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 4, September 2012*

1.2 Conformance Claims

This TOE is conformant to the following CC specifications:

- This ST is conformant to the
  - *Standard Protection Profile for Enterprise Security Management Access Control*, Version 2.1, 24 October 2013 (pp_esm_ac_v2.1) with no additional optional SFRs.
  - *Standard Protection Profile for Enterprise Security Management Policy Management*, Version 2.1, 24 October 2013 (pp_esm_pm_v2.1) and includes the additional optional SFRs: FAU_SEL.1, and FMT_MTD.1.
- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 4, September 2012.
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1 Revision 4, September 2012.
  - Part 3 Conformant

1.3 Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
  - Iteration: allows a component to be used more than once with varying operations. In the ST, iteration is indicated by a number in parentheses placed at the end of the component. For example FDP_ACC.1(1) and FDP_ACC.1(2) indicate that the ST includes two iterations of the FDP_ACC.1 requirement, (1) and (2).

  - Assignment: allows the specification of an identified parameter. Assignments are indicated using bold and are surrounded by brackets (e.g., [**assignment**]). Note that an assignment within a selection would be identified in italics and with embedded bold brackets (e.g., [***[selected-assignment]***]).
  - Selection: allows the specification of one or more elements from a list. Selections are indicated using bold italics and are surrounded by brackets (e.g., [***selection***]).
  - Refinement: allows the addition of details. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …"). Note that 'cases' that are not applicable in a given SFR have simply been removed without any explicit identification.

- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions; and unique font to identify specific TOE commands or entries in policy files (e.g. “`accept”`)

1.3.1 Terminology

This section identifies TOE-specific terminology.

administrator In the context of this ST and the TOE it describes, an administrator is a user, defined in the underlying operating system, that has been authorized to perform administrative functions on the underlying operating system and the TOE, by virtue of being granted 'root' or similar privileged access.

AES Advanced Encryption Standard—a symmetric cryptographic algorithm, defined in FIPS-197.

ANSI American National Standards Institute—a private, non-profit organization that oversees the development of voluntary consensus standards for products, services, processes, systems, and personnel in the United States.

authorized user In the context of this ST and the TOE it describes, an authorized user is a user defined in the TOE's operational environment and whose requests to invoke controlled commands are mediated by the TOE.

Computername The attribute that contains the name of the computer derived from LDAP, RADIUS sources.

FIPS Federal Information Processing Standard(s)—a series of publicly announced standards developed by the United States Federal government.

HMAC-SHA1 A keyed-Hash Message Authentication Code (HMAC) using the SHA-1 secure hash algorithm. SHA-1 is defined in FIPS 180-1, while HMAC-SHA1 is defined in FIPS 198.

inetd A super-server daemon on many Unix systems that manages Internet services. It has been replaced by ***xinetd*** in many systems, and by ***launchd*** in Mac OS X v10.4.

Log Server A component in the TOE architecture responsible for managing event logs and I/O logs.

Master Host A component in the TOE architecture responsible for determining if requests to invoke controlled commands will be accepted or rejected.

OpenLDAP A free, open source implementation of the Lightweight Directory Access Protocol (LDAP).

OpenSSL A free, open source implementation of the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols.

PAM Pluggable Authentication Module—a mechanism to integrate multiple low-level authentication schemes into a high-level application programming interface (API). It allows programs that rely on authentication to be written independently of the underlying authentication scheme.

PRNG Pseudo-Random Number Generator—specifications for PRNGs that can be used in FIPS validated cryptographic modules are defined in ANSI X9.31.

RADIUS Remote Authentication Dial-In User Service

RFC Request for Comments—a memorandum published by the Internet Engineering Task Force (IETF) describing methods, behaviors, research, or innovations applicable to the working of the Internet and Internet-connected systems.

RSA Rivest-Shamir-Adleman—an asymmetric cryptographic algorithm that supports public key cryptography.

Run Host A component in the TOE architecture on which an accepted controlled command will be executed.

secured task A request to invoke a controlled command, submitted to the TOE by an authorized user.

setuid Short for **set u**ser **ID** upon execution, it is a Unix access rights flag that allows users to run an executable file with the permissions of the file's owner.

SMF Service Management Facility—a feature of the Solaris operating system that creates a supported, unified model for services and service management.

Submit Host A component in the TOE architecture on which an authorized user submits a secured task.

TLS Transport Layer Security—a cryptographic protocol that provides confidentiality and integrity of data communicated over a computer network.

1.3.2 Abbreviations

This section identifies abbreviations and acronyms used in this ST.

AC	Access Control
API	Application Programming Interface
CC	Common Criteria for Information Technology Security Evaluation
ESM	Enterprise Security Management
ESM AC	Enterprise Security Management Access Control
ESM PM	Enterprise Security Management Policy Management
ESMPPs	The ESM AC and ESM PM Protection Profiles
GID	Also referred to as gid: Group ID or Group Identity
GUI	Graphical User Interface
HMAC	Hashed Message Authentication Code
HTTP(S)	Hypertext Transfer Protocol (Secure)
LDAP	Lightweight Directory Access Protocol
OpenLDAP	A free, open source implementation of the Lightweight Directory Access Protocol (LDAP).
OS	Operating System
PB	PowerBroker
PM	Policy Management
PP	Protection Profile
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SMF	Service Management Facility
ST	Security Target

TOE	Target of Evaluation
TSF	TOE Security Functions
UID	Also referred to as uid: User ID or User Identity

2. TOE Description

The Target of Evaluation (TOE) is BeyondTrust PowerBroker ® UNIX® + Linux® Edition V9.1 (PBUL). PBUL is a security management product that provides the capability to delegate access to operating system functions available to specific privileged accounts (e.g., 'root') and offer those functions in a controlled and granular fashion to other specific and suitably trusted users. The TOE provides both Enterprise Security Policy Management and Access Control functions.

2.1 TOE Overview

A characteristic of Unix/Linux operating systems is the existence of a single administrative account (e.g., 'root') that has complete administrative access to the operating system. Any user that has a requirement to access system resources or run a privileged command needs to be given the root password. This can result in many users having more privileges than they necessarily require for performing their work.

PowerBroker addresses this problem by providing the capability to selectively delegate Unix/Linux administrative privileges to trusted users without divulging the root password.

PowerBroker is a security management product that provides the capability to partition the functions available to specific privileged accounts (such as 'root') and offer those functions in a granular fashion to other specific and trusted users. PowerBroker provides granular delegation of administrative privileges on Unix and Linux hosts (e.g., those associated with the 'root' account), with an audit trail of attempts to exercise functions associated with those privileges. PowerBroker allows administrative capabilities of 'root' and other privileged accounts to be accessed by authorized users without having to provide direct access to those privileged accounts. PowerBroker policies are written to selectively allow specific users access to specific commands on specific hosts. Access to privileged functionality can be allowed or revoked at any time without concern for the status of the underlying privileged account. Users and administrators are required to authenticate in order to access the TOE and subsequently run a privileged command. The operational environment authenticates the user and administrators and the TOE enforces the results. Authentication attempts and attempts to exercise privileged functions are always audited. In addition, the input and output stream of a privileged function can be logged. In summary, PowerBroker allows the administrator to grant or deny other authorized users access to privileged functions of the managed operating system and audit the use of those functions.

The TOE uses FIPS 140-2 validated OpenSSL cryptographic modules provided in the operational environment. In the evaluated configuration, the FIPS 140 mode of operation will be required.

2.2 TOE Architecture

PowerBroker is a software-only product suite that runs on numerous Unix and Linux operating systems without modifying the kernel. The purpose of the product is to act as the "broker" between the user and the privileged operations on the system. To achieve this, the PowerBroker security policy is consulted each time the user attempts to run a privileged command through PowerBroker. The product provides two mechanisms through which this can be accomplished: the *pbrun* command and the PB Shells.

The *pbrun* command is used in a standard Unix shell just like any other command. A user wishing to execute a privileged command invokes the desired privileged command through *pbrun*. For example, if the command *mount* is a privileged command delegated by PowerBroker, a user wishing to run *mount* would execute the command '`pbrun mount <mount options>`' from the regular shell. PBRun sends the secured task request to a policy server for processing. The TOE determines whether or not the user has permission to execute the *mount* command on the target host. If permission is granted, the command is executed on behalf of the user. Privileged commands requested by a user and authorized and executed by PowerBroker are known as 'secured tasks'.

The PB Shells are customized versions of the public domain pdksh '88 Korn shell (*pbksh*) and Bourne shell (*pbsh*). These modified shells contain the full functionality and features of the standard public domain shells, but they have been modified to verify all command operations through PowerBroker before allowing execution. Any user running *pbsh* or *pbksh* as the shell will be under the control of the PowerBroker access control mechanisms.

All attempted actions mediated by PowerBroker are logged in a detailed audit log. The administrator has control over whether or not the keystrokes and output of a particular action are audited. Security audit data is stored in the following:

- Event Log—this PowerBroker audit file records when each requested task was accepted or rejected. For tasks not run in local mode, it also logs when the task terminated, and any configured keystroke-monitoring events that were triggered by that task attempt. These events are known as ACCEPT, REJECT, FINISH, and KEYSTROKE events. The Event Log is a binary file that can be encrypted, but is not encrypted by default.
- IO Logs—optional logs that record I/O (i.e., keystrokes and output) information for specific secured tasks. Auditing of this type of data is not within the scope of the evaluation.
- Configuration Database—this database is a version controlled database that stores key configuration, settings and policy files, including auditing of activities such as the creation of new files and version changes within controlled files.

A typical PowerBroker configuration consists of the following primary components:

- *pbrun* (or *pbsh, pbksh*)—requests that a secured task is run in a controlled environment
- *pbmasterd*—receives secured task requests from *pbrun*, *pbksh*, and *pbsh* and evaluates them according to the current security policies. If the request is accepted, it directs *pblocald* to run the secured task
- *pblocald*—the daemon that runs secured tasks on behalf of the user, when instructed to do so by the master daemon (*pbmasterd*)
- *pblogd*—the log server daemon records event logs and I/O logs as directed by other PB programs.

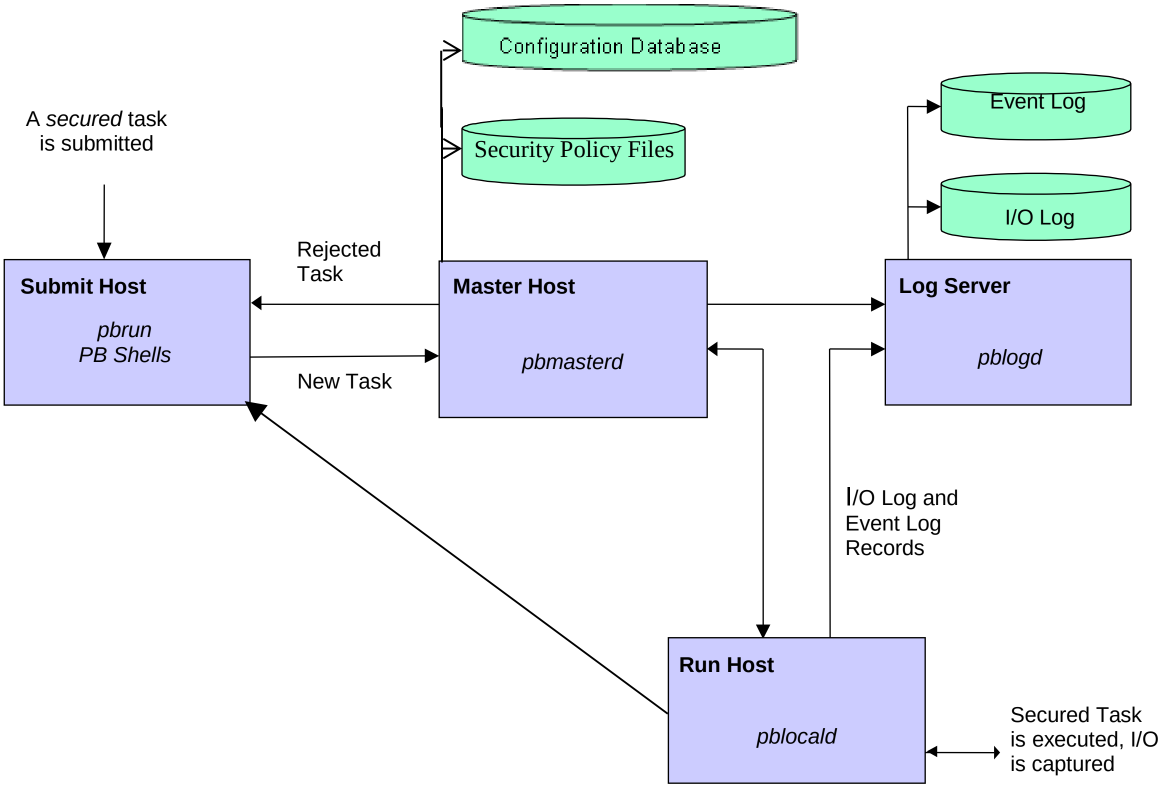

Figure 1: PowerBroker Component Interactions

Figure 1 depicts the interactions between the primary TOE components. Each blue box represents a logical operating environment (e.g., 'Submit Host') for the listed TOE components (identified by italics, e.g., '*pbrun*'). The policy files used by the TOE and the logs generated by the TOE are stored in files in the operational environment (the green 'disk drives').

The machine from which a task is submitted is referred to as the Submit Host. The machine on which the Configuration Database resides and on which the Security Policy File processing takes place is referred to as the Master Host. The machine on which a task is actually executed is referred to as the Run Host. The machine on which Event Log records and I/O log records are written is referred to as the Log Server (or Host). It is possible to install any or all of these components on a single machine, or to distribute them between different machines. Use of a separate log server and *pblogd* daemon is optional, but highly recommended. When *pblogd* is not used, *pbmasterd* logs the audit records. For optimal security, the master hosts and log servers should be separate machines that are isolated from normal user activity. When the TOE components are deployed on separate machines, the TOE must be configured to encrypt communications between the separate components. The TOE uses TLS and FIPS validated algorithms provided by OpenSSL in the operational environment.

The typical sequence of PowerBroker processing is as follows:

- A user (or administrator) establishes a session with the Unix/Linux machine running the Submit Host
- The user is authenticated by an authentication server in the operational environment
- From a normal shell on the Submit Host, a user submits a request via *pbrun*
- *pbmasterd* on the Master Host processes the security policy and either accepts or rejects the request
- The request acceptance or rejection is audited and an event sent to the Log Server. For rejected requests, processing ends here
- An accepted request is executed via *pblocald* on the Run Host
- If I/O logging was designated by the security policy, this data is sent to the Log Server.

Common variations to this processing sequence are as follows:

- If the Submit Host is the same server as the Run Host, "local mode" or "Optimized Run Mode" can be enabled. In these cases, if *pbmasterd* accepts the command, it is executed from the *pbrun* process rather than launching *pblocald*
- If there is no separate Log Server, *pbmasterd* performs the logging services
- *pbmasterd*, *pblocald* and *pblogd* can be configured to run continuously as daemons, or alternately can be configured to launch on a per-use basis by *inetd* or equivalent (e.g., *xinetd*, *SMF*, *launchd*).

The *pbmasterd*, *pblocald*, and *pblogd* components all run as 'root' (or equivalent, depending on the operational environment). The *pbrun*, *pbsh*, and *pbksh* components run as the invoking user but with setuid root.

As indicated above, all TOE components can be installed on a single machine, or can be deployed across a number of machines. Any machine that is to be used as a Submit Host requires *pbrun*, *pbsh*, or *pbksh* to be installed on it. Each Submit Host will have (in its configuration file) a list of one or more Master Hosts. Each Master Host requires *pbmasterd* to be installed on it to process secured task requests. Any machine that will be used as a Run Host requires *pblocald* to be installed on it. Use of a Log Host is optional—in the absence of a Log Host, *pbmasterd* is responsible for logging activities. Any machine that will be used as a Log Host requires *pblogd* to be installed on it.

In summary, in the TOE model, an access request originates at a network host (Submit Host) and is transmitted to the central Policy Manager (Master Host), which also acts as the policy decision point. If the Policy Manager determines the access control request complies with the defined policy, it forwards the access request (secured task) to the target host (Run Host) for action. The Run Host is part of the Access Control portion of the TOE that performs the requested operation and communicates the results back to the Submit Host. If the access request does not conform with policy, it is rejected and the originator (Submit Host) is notified. As such, the TOE model inverts the ESM model for PM and AC presented in the PP in that the ESM model assumes a central point where access control policies are created and managed and then distributed as appropriate to other computers on the network where the policy is enforced.

Other PowerBroker components comprise:

- PB Shells (*pbsh* and *pbksh*)—as indicated above, the PB shells function similarly to *pbrun* in Figure 1. The PB shells obtain approval from *pbmasterd* for every command issued at the PB shell prompt. The PB shells provide transparent authorization and event logging for every command, shell built-in, and shell I/O redirection, and control of shell scripts. Once accepted by *pbmasterd*, the commands are executed by either *pblocald* or by the PB shell
- PB GUIs (*pbguid* and *pbsguid*)—the PB GUI programs provide an HTTP (*pbguid*) and an HTTPS (*pbsguid*) server for browser-based administration of PowerBroker. The administrator accesses the GUI by starting a browser (in the operational environment) on their local machine and connecting to a host and port where *pbguid* or *pbsguid* is hosted. Administrators using the GUI are authenticated by the underlying OS on which the GUI programs are installed. The GUI allows an authorized user to change settings files, view events and keystroke logs, edit policy configuration files, run reports, and update the GUI configuration. *pbguid* provides unencrypted GUI access, and *pbsguid* provides TLS-protected GUI access. Administrators must use HTTPS; HTTP is not permitted in the evaluated configuration.
- PB Administrative Utilities—used to administer PowerBroker. They provide the following capabilities:
    - *pbbench*—a diagnostic tool that helps solve configuration, file permission and network problems. It reads the information in the PowerBroker settings file on the local machine and uses system information to verify the information in the settings file
    - *pbcall*—allows a PowerBroker policy language function to be executed from the command line, allowing the administrator to test the effects of that function on the local machine
    - *pbcheck*—used to check the PowerBroker configuration file for errors
    - *pbencode*—encrypyts a file using a key specified in the command line or in the settings file
    - *pbhostid*—used to display a computer's unique, hardware-dependent identifier, which is subsequently used in generating a product license string. This utility is used only during TOE installation
    - *pbkey*—used to generate symmetric encryption keys for protecting files and network traffic
    - *pblicense*—displays current licensing information and retires licenses
    - *pblog*—used to display entries from a PowerBroker event log (the PB GUIs also provide this capability)
    - *pbpasswd*—generates an encrypted password that can then be used in the policy file to provide password protection to secured tasks
    - *pbprint*—produces a formatted display of a PowerBroker policy file
    - *pbreplay*—used to display the contents of a PowerBroker keystroke log (the PB GUIs also provide this capability)
    - *pbreport*—used to extract data from PowerBroker event logs and generate reports (the PB GUIs also provide this capability)
    - *pbsum*—prints the checksum of one or more files, which can then be used in the policy file to check the requested program's integrity
    - *pbsync*—starts the log synchronization process
    - *pbsyncd*—a server that listens for log synchronization requests from one or more clients
    - *pbuvqrpg*—works with *pbreport* to generate text based reports.

  The *pbbench*, *pbcall*, *pbencode*, and *pbsum* utilities can be run on any host (i.e., Submit, Master, Log or Run). The *pbcheck*, *pbhostid*, *pbkey*, *pblicense*, *pbpasswd*, and *pbprint* utilities can be run only on the Master host, while the *pblog*, *pbreplay*, *pbsync, pbsyncd*, *pbreport*, and *pbuvqrpg* utilities can be run on the Master and Log hosts.

- PB User Utilities (*pbvi*, *pbnvi*, *pbless*, *pbumacs*, and *pbmg*)—the PB User Utilities are similar to standard Unix *vi*, *emacs*, and *less* commands, except that they are 'hardened' such that they do not include the standard functions to allow access to other files, run other commands from inside the utility, or to access sub-shells from which other commands could be run. Note that these utilities would be run on the Run Host under the control of the rest of the TOE.
- PB REST API—the PB REST API developed for PowerBroker Servers Unix/Linux to allow other software to configure, customize and retrieve data from PBUL. The API is web based and uses industry standard modern components, connectors and data elements within a distributed and secure enterprise environment. The REST API is not included in the evaluated configuration and should not be enabled.

2.2.1 Physical Boundaries

The TOE is a series of software applications hosted on the following supported operating system platforms (supported hardware architectures are in parentheses):

- AIX:
  - v6.1 (Power5 64-bit)
  - v7.1 (POWER 64-bit)
- HP-UX:
  - 11i v3 (B.11.31) (PA-RISC 64-bit, Itanium 64-bit)
- Solaris:
  - 11 (Sparc, x86 64-bit)
- Red Hat Enterprise Linux:
  - v6.x (x86 64-bit)
  - v7.x (x86 64-bit)
- Ubuntu:
  - Ubuntu 13.4 (x86 64-bit)
  - Ubuntu 14.4 (x86 64-bit)

The PB GUI is compatible with the following browsers (which are in the operational environment): Opera, Firefox, Netscape, and Internet Explorer. The browser must have Javascript, cascading style sheets, and pop-ups enabled.

The TOE comprises the components described above, i.e., *pbrun*, *pbmasterd*, *pblocald*, *pblogd*, *pbsh*, *pbksh*, *pbguid*, *pbsguid*, *pbkey*, *pbcheck*, *pblog*, *pbreport*, *pbreplay*, *pbvi*, *pbnvi*, *pbless*, *pbumacs*, *pbbench*, *pbmg*, *pbcall*, *pbencode*, *pbhostid*, *pblicense*, *pbpasswd*, *pbprint*, *pbsum*, *pbsync*, *pbsyncd*, and *pbuvqrpg*. Each of these components is either security enforcing or security relevant.

The TOE includes the following third party libraries: OpenSSL v1.0.2a; and OpenLDAP v2.4.40. The OpenSSL libraries are used in distributed configurations and must be configured to use Transport Layer Security (TLS). OpenLDAP 2.4.40 implements LDAPv3, which is defined in RFC 4511. The TOE uses OpenLDAP within the PowerBroker policy language to perform LDAP queries to help determine if a command request should be accepted or rejected. Note there is no requirement for an LDAP server in the operational environment (for use in policies), but if the customer's installation uses one for managing user information, the TOE allows policy decisions to be made based on that information.

The TOE can be configured to record events in the Master Host's syslog system, in addition to the TOE's event logs. The syslog system is provided by the operational environment and is outside the physical boundary of the TOE.

Use of the external authentication methods in the operational environment require LDAP, and RADIUS servers. The TOE can optionally be configured to use Pluggable Authentication Modules (PAMs) on operating systems where they are available. In this case, the TOE can use PAM password authentication services, account management services, and session start/end services. The PAM and its services are in the operational environment.

The TOE can be configured to integrate with the SafeNet Luna SA Hardware Security Module (HSM), to provide hardware-based storage of TOE TLS encryption keys. The HSM is outside the TOE boundary.

The TOE relies on 3rd party FIPS capable OpenSSL 1.0.2a in conjunction with the TOEs FIPS mode (that disables non FIPS algorithms). Customers are instructed to choose their own validated FIPS validated Object Module and link that with the provided FIPS capable OpenSSL v1.0.2a. The combination of the FIPS validated Object Module linked with the FIPS capable OpenSSl provide key management, random bit generation, encryption/decryption, digital signature and cryptographic hashing and keyed-hash message authentication features in support of higher level cryptographic protocols, including TLS and HTTP over TLS.

The operational environment authenticates the administrator (and users) and as such the TOE relies on the operational environment to also provide an advisory warning message regarding unauthorized use of the TOE. The banner is displayed prior to users and administrators being permitted to establish interactive sessions.

2.2.2 Logical Boundaries

This section summarizes the security functions provided by the TOE:

- Enterprise Security Management
- Security audit
- Communication
- User data protection
- Identification and authentication
- Security management
- Protection of the TSF
- Resource Utilization
- Trusted path/channels

2.2.2.1 Enterprise Security Management

The TOE provides the ability to define access control policies for consumption by a compatible Access Control product: i.e. the TOE itself. Access control policies consist of subject, object, and attributes; policies are uniquely identified. The TOE ensures that policies are available to the TOE's Access Control component immediately following creation of a new or updated policy.

The TOE relies on LINUX/UNIX host, LDAP, RADIUS, and optionally PAM in the operational environment for subject identification and authentication; and requires each subject to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that subject.

2.2.2.2 Security audit

The TOE is designed to be able to generate logs for security relevant events including the events specified in ESMPPs. The TOE can be configured to store the logs locally. The audit records identify the date/time, event type, outcome of the event, responsible subject/user, as well as the additional event-specific content indicated in Table 2.

The TOE is capable of selective auditing based upon Administrator defined policy variables and by object identity.

The TOE transmits audit records to TOE internal storage and uses TLS for distributed communications. The TOE protects the stored audit records in the TOE-internal audit trail from unauthorized deletion and modification. The cryptographic algorithms used in TLS are provided by the OpenSSL FIPS validated modules in the operational environment.

2.2.2.3 Communication

The TOE is both a Policy Management and Access Control product where policies are centralized and never transmitted. Policies are defined on a Master Host and available immediately as soon as it is saved. The policy files never leave this location or otherwise traverse across the TOE or outside the TOE. The administrator can verify the existence of the policy by performing a policy lookup using the policy file name; and can verify the location (Master Host) of the policy by viewing the Master Host field/attribute.

2.2.2.4 User data protection

The TOE controls access to commands that have been defined to be controlled on target hosts. The TOE's self-protection Security Function Policy restricts access to objects that reside in the Operational Environment that impact the TOE's behavior.

2.2.2.5 Identification and authentication

The TOE associates the uid and gid user security attributes with subjects acting on the behalf of a user. The TOE uses an external LDAP or RADIUS server to authenticate users and enforces the result. The TOE determines the uid from the credentials presented at authentication and associates the gid retrieved from the authentication server with the corresponding uid.

2.2.2.6 Security management

The TOE provides administrative functions available from a CLI and a GUI to access the management functions and for administrators to change their own passwords. Security management commands are limited to authenticated users with root access. The TOE provides the AdminUser role which provides root access.

The TOE also provides the ability for the Policy Management components to manage the Access Control components of the TOE. The TOE components must be configured to communicate with one another using TLS or HTTPS and as such can trust one another. The default values for security attributes used in the access control policies are restrictive and the Policy Management component can change these defaults. The TOE's policy management engine defines an unambiguous hierarchical method of implementing a policy such that no contradictions occur.

2.2.2.7 Protection of the TSF

The TOE uses external Identity and Credential Management products to define its administrator authentication data, the TOE does not store or cache the data. The TOE does not offer any functions that will disclose to any users a stored cryptographic key; and all keys are stored encrypted using AES-256.

Should the TOE or a TOE component encounter a failure state; all access control requests are denied. The TOE is both an Access Control and Policy Management product. If the TOE is in a failed state then no access control requests or decisions can be made. Policies are defined in a central location and are never transmitted. The TOE detects replay attacks for secured and rejects the secured task when replay is detected. The TOE relies on the implementation of TLS in the operational environment to provide secure transmission, including replay detection, of secured tasks.

2.2.2.8 Resource Utilization

The TOE is both an Access Control and Policy Management product. The most recent policy will always be enforced even in the event of a TOE failure. Should the TOE experience a failure, no access control is permitted until the system comes back up. Policies are defined and enforced on the same component and therefore it is not possible to lose communication during a policy transmission.

2.2.2.9 Trusted path/channels

The TOE protects interactive communication with remote administrators using HTTP over TLS. TLS ensures both integrity and disclosure protection.

The TOE protects communication with external LDAP servers and internal distributed TOE components using TLS connections to prevent unintended disclosure or modification of the transferred data.

The TOE uses FIPS capable OpenSSL v1.0.2a and requires FIPS mode to disable non FIPS algorithms. Customers are instructed to choose their own validated FIPS Object Module and link that with the provided FIPS capable OpenSSL v1.0.2a. The validated Object Module and FIPS capable OpenSSL are in the operational environment.

2.3 TOE Documentation

BeyondTrust Software, Inc. offers a series of documents that describe the installation process for the TOE, as well as guidance for subsequent use and administration of the system security features.

- PowerBroker for Unix & Linux Common Criteria Supplementary Information Document
- PowerBroker Servers UNIX + LINUX Edition Browser Interface Guide, Version 9.1, July 2015
- PowerBroker Servers UNIX + LINUX Edition System Administrators Guide, Version 9.1, July 2015
- PowerBroker Servers UNIX + LINUX Edition Installation Guide, Version 9.1, July 2015
- PowerBroker Servers UNIX + LINUX Edition Policy Language Guide, Version 9.1, July 2015

3. Security Problem Definition

This security target includes by reference the Security Problem Definitions (composed of organizational policies, threat statements, and assumptions) from the ESMPPs.

In general, the ESMPPs have presented Security Problem Definitions appropriate for Enterprise Security Management Access Control and Policy Management products, and as such are applicable to the BeyondTrust TOE.

4. Security Objectives

Like the Security Problem Definition, this security target includes by reference the Security Objectives from the ESMPPs with the exclusion of O.BANNER as allowed by TD055. Only the applicable optional security objectives for the operational environment from the ESMPPs are reproduced below, since these objectives characterize technical and procedural measures each consumer must implement in their operational environment. TD055 adds OE.BANNER.

In general, the ESMPPs have presented Security Objectives statements appropriate for Enterprise Security Management Access Control and Policy Management products, and as such is applicable to the BeyondTrust TOE.

4.1 Security Objectives for the Operational Environment

OE.BANNER
The Operational Environment will display an advisory warning regarding use of the TOE

OE.CRYPTO
The Operational Environment will provide cryptographic primitives that can be used by the TOE to provide services such as ensuring the confidentiality and integrity of communications.

OE.PROTECT
The Operational Environment will protect the TOE from unauthorized modifications and access to its functions and data.

OE.ROBUST
The Operational Environment will provide mechanisms to reduce the ability for an attacker to impersonate a legitimate user during authentication.

OE.SYSTIME
The Operational Environment will provide reliable time data to the TOE.

5. IT Security Requirements

This section defines the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) that serve to represent the security functional claims for the Target of Evaluation (TOE) and to scope the evaluation effort.

The SFRs have all been drawn from the Protection Profiles (PP): *Standard Protection Profile for Enterprise Security Management Access Control*, Version 2.1, 24 October 2013 (pp_esm_ac_v2.1) and *Standard Protection Profile for Enterprise Security Management Policy Management*, Version 2.1, 24 October 2013 (pp_esm_pm_v2.1).

As a result, refinements and operations already performed in that PP are not identified (e.g., highlighted) here, rather the requirements have been copied from that PP and any residual operations have been completed herein. Of particular note, the ESMPPs made a number of refinements and completed some of the SFR operations defined in the CC and the PPs should be consulted to identify those changes if necessary.

In the case where the PPs contained duplicate SFRs, this ST combines the SFRs into one and ensures that each individual copy of the SFR is satisfied on its own. Where the PPs contain different SFRs that have identical names, both are included in this ST with the original source clearly referenced for each (e.g. [pp_esm_pm_v2.1] or [pp_esm_ac_v2.1]).

The SARs are the set of SARs specified in ESMPPs.

5.1 Extended Requirements

All of the extended requirements in this ST have been drawn from the ESMPPs. The ESMPPs define the following extended SFRs and since they are not redefined in this ST, the PPs should be consulted for more information in regard to those CC extensions.

- FAU_SEL_EXT.1: External Selective Audit
- FAU_STG_EXT.1: External Audit Trail Storage
- FMT_MOF_EXT.1: External Management of Functions Behavior
- FMT_MSA_EXT.5: Consistent Security Attributes
- FPT_APW_EXT.1: Protection of Stored Credentials
- FPT_FLS_EXT.1: Failure of Communications
- FPT_SKP_EXT.1: Protection of Secret Key Parameters

5.2 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by the BeyondTrust PBUL.

Requirement Class	Requirement Component
ESM: Enterprise Security Management	ESM_ACD.1: Access Control Policy Definition
	ESM_ACT.1: Access Control Policy Transmission
	ESM_EAU.2(1): Reliance on Enterprise Authentication
	ESM_EAU.2(2): Reliance on Enterprise Authentication
	ESM_EID.2(1): Reliance on Enterprise Identification
	ESM_EID.2(2): Reliance on Enterprise Identification
FAU: Security audit	FAU_GEN.1: Audit Data Generation
	FAU_SEL.1: Selective Audit
	FAU_SEL_EXT.1: External Selective Audit
	FAU_STG.1: Protected Audit Trail Storage (Local Storage)
	FAU_STG_EXT.1: External Audit Trail Storage
FCO: Communication	FCO_NRR.2: Enforced proof of receipt
FDP: User data protection	FDP_ACC.1(1): Access Control Policy
	FDP_ACC.1(2): Access Control Policy (Self-Protection)
	FDP_ACF.1(1): Access Control Functions
	FDP_ACF.1(2): Access Control Functions (Self-Protection)
FIA: Identification and authentication	FIA_USB.1: User-Subject Binding
FMT: Security management	FMT_MOF.1: Management of Functions Behavior
	FMT_MOF.1(1): Management of Functions Behavior
	FMT_MOF.1(2): Management of Functions Behavior
	FMT_MOF_EXT.1: External Management of Functions Behavior
	FMT_MSA.1: Management of Security Attributes
	FMT_MSA.3: Static Attribute Initialization
	FMT_MSA_EXT.5: Consistent Security Attributes
	FMT_MTD.1: Management of TSF Data
	FMT_SMF.1: Specification of Management Functions [pp_esm_ac_v2.1]
	FMT_SMF.1: Specification of Management Functions [pp_esm_pm_v2.1]
	FMT_SMR.1: Security Management Roles [pp_esm_pm_v2.1]
	FMT_SMR.1: Security Roles [pp_esm_ac_v2.1]
FPT: Protection of the TSF	FPT_APW_EXT.1: Protection of Stored Credentials
	FPT_FLS_EXT.1: Failure of Communications
	FPT_RPL.1: Replay Detection
	FPT_SKP_EXT.1: Protection of Secret Key Parameters
FRU: Resource Utilization	FRU_FLT.1: Degraded Fault Tolerance
FTP: Trusted path/channels	FTP_ITC.1: Inter-TSF Trusted Channel
	FTP_TRP.1: Trusted Path

Table 1 TOE Security Functional Components

5.2.1 Enterprise Security Management (ESM)

5.2.1.1 Access Control Policy Definition (ESM_ACD.1) [ESM_PM]

ESM_ACD.1.1 The TSF shall provide the ability to define access control policies for consumption by one or more compatible Access Control products.

ESM_ACD.1.2 Access control policies defined by the TSF shall be capable of containing the following:

Subjects: [**users derived from LDAP, RADIUS sources**]; and

Objects: [**programs, files, host configuration, authentication function derived from the operating system of the host source from which the objects reside**]; and

Operations: [**ability to create, read, modify, execute, delete, terminate, or change permissions of objects, ability to use authentication function derived from the operating system of the host on which those objects reside**]; and

Attributes: [**uid, gid, computername derived from LDAP, RADIUS sources**].

ESM_ACD.1.3 The TSF shall associate unique identifying information with each policy.

5.2.1.2 Access Control Policy Transmission (ESM_ACT.1) [ESM_PM]

ESM_ACT.1.1 The TSF shall transmit policies to compatible and authorized Access Control products under the following circumstances: [***immediately following creation of a new or updated policy***].

Application Note: *In this TOE policies are centralized and do not get transmitted or pushed. The TOE is both an esm access control product and an esm policy management product and as such 'Transmit" should be interpreted as "made available". The policy is immediately available.*

5.2.1.3 Reliance on Enterprise Authentication (ESM_EAU.2(1)) [ESM_PM]

ESM_EAU.2.1(1) The TSF rely on [***[LDAP, RADIUS]***] for subject authentication.

ESM_EAU.2.2(1) The TSF shall require each subject to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that subject.

5.2.1.4 Reliance on Enterprise Authentication (ESM_EAU.2(2)) [ESM_PM]

ESM_EAU.2.1(2) The TSF shall rely on [***[LINUX/UNIX host]***] for subject authentication.

ESM_EAU.2.2(2) The TSF shall require each subject to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that subject.

5.2.1.5 Reliance on Enterprise Identification (ESM_EID.2(1)) [ESM_PM], [ESM_AC]

ESM_EID.2.1(1) The TSF shall rely on [***[LDAP, RADIUS]***] for subject identification.

ESM_EID.2.2(1) The TSF shall require each subject to be successfully identified before allowing any other TSF-mediated actions on behalf of that subject

5.2.1.6 Reliance on Enterprise Identification (ESM_EID.2(2)) [ESM_PM], [ESM_AC]

ESM_EID.2.1(2) The TSF shall rely on [***[LINUX/UNIX host]***] for subject identification.

ESM_EID.2.2(2) The TSF shall require each subject to be successfully identified before allowing any other TSF-mediated actions on behalf of that subject.

5.2.2 Security audit (FAU)

5.2.2.1 Audit Data Generation (FAU_GEN.1)

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

a) Start-up and shut-down of the audit functions; and

b) All auditable events identified in **Table 2** for the [not specified] level of audit; and

c) [**no other auditable events**].

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

a) Date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [**information specified in column three of Table 2**].

Component	Event	Additional Information
ESM_ACD.1	Creation or modification of policy	Unique policy identifier
ESM_ACT.1 [ESM_PM]	Transmission of policy to Access Control products	Destination of policy
ESM_EAU.2 [ESM_PM]	All use of the authentication mechanism	None
FAU_SEL.1 [ESM_AC]	All modifications to audit configuration	None
FAU_SEL_EXT.1 [ESM_PM]	All modifications to audit configuration	None
FAU_STG_EXT.1 [ESM_PM], [ESM_AC]	Establishment and disestablishment of communications with audit server	Identification of audit server
FCO_NRR.2 [ESM_AC]	The invocation of the non-repudiation service	Identification of the information, the destination, and a copy of the evidence provided
FDP_ACC.1(1), (2)[ESM_AC]	Any changes to the enforced policy or policies	Identification of Policy Management product making the change
FDP_ACF.1(1), (2) [ESM_AC]	All requests to perform an operation on an object covered by the SFP	Subject identity, object identity, requested operation
FMT_MOF.1 [ESM_AC]	All modifications to TSF behavior	None
FMT_SMF.1 [ESM_PM]	Use of the management functions	Management function performed
FMT_SMR.1 [ESM_PM]	Modifications to the members of the management roles	None
FPT_FLS_EXT.1 [ESM_AC]	Failure of communication between the TOE and Policy Management product	Identity of the Policy Management product, reason for the failure
FPT_RPL.1 [ESM_AC]	Detection of replay	Action to be taken based on the specific actions
FTP_ITC.1 [ESM_AC], [ESM_PM]	All use of trusted channel functions	Identity of the initiator and target of the trusted channel
FTP_TRP.1 [ESM_PM]	All attempted uses of the trusted path functions	Identification of user associated with all trusted path functions, if available

Table 2 Auditable Events

5.2.2.2 Selective Audit (FAU_SEL.1) [ESM_AC]

FAU_SEL.1.1 The TSF shall be able to select the set of events to be audited from the set of all auditable events based on the following attributes:

a) [***object identity***]

b) [**Administrator defined policy variables**].

5.2.2.3 External Selective Audit (FAU_SEL_EXT.1) [ESM_PM]

FAU_SEL_EXT.1.1 The TSF shall be able to select the set of events to be audited by an ESM Access Control product from the set of all auditable events based on the following attributes:
a) [***object identity***]; and
b) [**Administrator defined policy variables**].

5.2.2.4 Protected Audit Trail Storage (Local Storage) [ESM_AC]

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

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

5.2.2.5 External Audit Trail Storage (FAU_STG_EXT.1) [ESM_PM]. [ESM_AC]

FAU_STG_EXT.1.1 The TSF shall be able transmit the generated audit data to [**TOE-internal storage**].

FAU_STG_EXT.1.2 The TSF shall ensure that transmission of generated audit data to any external IT entity uses a trusted channel defined in FTP_ITC.1.

FAU_STG_EXT.1.3 The TSF shall ensure that any TOE-internal storage of generated audit data:
a) protects the stored audit records in the TOE-internal audit trail from unauthorized deletion; and
b) prevents unauthorized modifications to the stored audit records in the TOE-internal audit trail.

5.2.3 Communication (FCO)

5.2.3.1 Enforced proof of receipt (FCO_NRR.2) [ESM_AC]

FCO_NRR.2.1 The TSF shall enforce the generation of evidence of receipt for received [policies] at all times.

FCO_NRR.2.2 The TSF shall be able to relate the [**Master host name**] of the recipient of the information, and the [**Submit Host Name, uid**] of the information to which the evidence applies.

FCO_NRR.2.3 The TSF shall provide a capability to verify the evidence of receipt of information to [originator] given [**policies are immediately available**].

Application Note: *The TOE is both a Policy Management and Access Control product. Policies are centralized and never transmitted. The execution of every pbrun command is captured in the event log. The event log identifies the Master Host Name and Policy File Name for each command executed. The event log entry confirms that the policy is in effect.*

5.2.4 Access Control Policy (FDP)

5.2.4.1 Access Control Policy (FDP_ACC.1(1)) [ESM_AC]

FDP_ACC.1.1(1) The TSF shall enforce the [access control Security Function Policy (SFP)] on [

- subjects: subset of users from an organizational data store, [**no additional subjects**]; and
- objects: programs, files, host configuration, authentication function, [**no additional objects**]; and
- operations: ability to create, read, modify, execute, delete, terminate, or change permissions of objects, ability to use authentication function, [**no additional operations**]].

5.2.4.2 Access Control Policy (FDP_ACC.1(2)) (Self-Protection) [ESM_AC]

FDP_ACC.1.1(2) The TSF shall enforce the [self-protection Security Function Policy (SFP)] on [

- subjects: subset of users from an organizational data store, [**no additional subjects**]; and
- objects: programs, files, and configuration values that comprise or contain TOE data [**no additional objects**]; and
- operations: ability to create, read, modify, execute, delete, terminate, or change permissions of objects, [**no additional operations**]]

5.2.4.3 Access Control Functions (FDP_ACF.1(1))

FDP_ACF.1.1(1) The TSF shall enforce the [access control SFP] to objects based on the following: [all operations between subjects and objects defined in **Table 3** below based upon some set of organizational attributes].

FDP_ACF.1.2(1) The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [**rules based on administrator-configured policy which specifies the conditions under which the request will be accepted or rejected**].

FDP_ACF.1.3(1) The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: [**no other additional rules**].

FDP_ACF.1.4(1) The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [**no other additional rules**].

Subject	**Object**	**Operation**
User	Processes	Execute | Delete | Terminate
		Change Permissions
	Files	Create | Read | Modify | Delete
		Change Permissions
	Host Configuration	Read | Modify | Delete
	Authentication Function	Login

Table 3: FDP Requirement Table for Host-Based Access Control

5.2.4.4 Access Control Functions (FDP_ACF.1(2))

FDP_ACF.1.1(2) The TSF shall enforce the [self-protection SFP] to objects based on the following: [all operations between subjects and objects based upon some set of organizational attributes].

FDP_ACF.1.2(2) The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [the TOE will not permit requested operations against objects that are defined to be protected unless the acting subject is the individual that was responsible for the TOE's installation and initial configuration].

FDP_ACF.1.3(2) The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: [none**].**

FDP_ACF.1.4(2) The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [none].

5.2.5 Identification and authentication (FIA)

5.2.5.1 User-Subject Binding (FIA_USB.1) [ESM_PM]

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: [**uid, gid**].

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: [**none**].

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: [**none**].

5.2.6 Security management (FMT)

5.2.6.1 Management of Functions Behavior (FMT_MOF.1) [ESM_PM]

FMT_MOF.1 The TSF shall restrict the ability to [**determine the behavior of, disable, enable, modify the behavior of**] the functions: [**functions identified in Table 4**] to [**users with root access**].

5.2.6.2 Management of Functions Behavior (FMT_MOF.1(1)) [ESM_AC]

FMT_MOF.1.1(1) The TSF shall restrict the ability to [***determine the behavior of, disable, enable, modify the behavior of***] the functions[: audited events, repository for trusted audit storage, access control SFP, policy being implemented by the TSF, access control SFP behavior to enforce in the event of communications outage, [**no other functions**] to [an authorized and compatible Policy Management product].

5.2.6.3 Management of Functions Behavior (FMT_MOF.1(2)) [ESM_AC]

FMT_MOF.1(2) The TSF shall restrict the ability to [determine the behavior of] the functions[: policy being implemented by the TSF, [**no other function**]] to [an authorized and compatible Enterprise Security Management product].

5.2.6.4 External Management of Functions Behavior (FMT_MOF_EXT.1) [ESM_PM]

FMT_MOF_EXT.1.1 The TSF shall restrict the ability to query the behavior of, modify the functions of Access Control products: audited events, repository for audit storage, Access Control SFP, policy version being implemented, Access Control SFP behavior to enforce in the event of communications outage, [**no other functions**] to [**no roles**].

5.2.6.5 Management of Security Attributes (FMT_MSA.1) [ESM_AC]

FMT_MSA.1.1 The TSF shall enforce the [access control SFP] to restrict the ability to [***change_default, query, modify, delete, [no other operations]***] the security attributes [access control policies, access control policy attributes, implementation status of access control policies] to [an authorized and compatible Policy Management product].

5.2.6.6 Static Attribute Initialization (FMT_MSA.3) [ESM_AC]

FMT_MSA.3.1 The TSF shall enforce the [access control SFP] to provide [restrictive] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the [authorized and compatible Policy Management product] to specify alternative initial values to override the default values when an object or information is created.

5.2.6.7 Consistent Security Attributes (FMT_MSA_EXT.5) [ESM_PM]

FMT_MSA_EXT.5.1 The TSF shall [***only permit definition of unambiguous policies***].

FMT_MSA_EXT.5.2 The TSF shall take the following action when an inconsistency is detected: [***[no action]***].

Application Note: *Since the TOE's policy management engine defines an unambiguous hierarchical method of implementing a policy such that no contradictions occur, FMT_MSA_EXT.5.2 is vacuously satisfied as it is impossible to have inconsistencies to detect.*

5.2.6.8 Management of TSF Data (for general TSF data) (FMT_MTD.1) [ESM_PM]

FMT_MTD.1.1 The TSF shall restrict the ability to [***[manage]***] the [**an administrative user's own password**] to the [**users with root access**].

5.2.6.9 Specification of Management Functions (FMT_SMF.1) [ESM_PM]

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: **[management functions listed in Table 4**].

Table 4: Management Functions within the TOE

Requirement	Management Activities
ESM_ACD.1	Creation of policies
ESM_EAU.2	Management of authentication data for administrative users
FAU_SEL.1	Configuration of auditable events
FAU_SEL_EXT.1	Configuration of auditable events for defined external entities
FMT_MOF_EXT.1	Configuration of the behavior of other ESM products
FTP_ITC.1	Configuration of actions that require trusted channel (if applicable)
FTP_TRP.1	Configuration of actions that require trusted path (if applicable)

5.2.6.10 Specification of Management Functions (FMT_SMF.1) [ESM_AC]

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [configuration of audited events, configuration of repository for trusted audit storage, configuration of Access Control SFP, querying of policy being implemented by the TSF, management of Access Control SFP behavior to enforce in the event of communications outage, [**no other management functions]].**

5.2.6.11 Security Management Roles (FMT_SMR.1) [ESM_PM] [ESM_AC]

FMT_SMR.1.1 The TSF shall maintain the roles [**AdminUsers**].

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

5.2.7 Protection of the TSF (FPT)

5.2.7.1 Extended: Protection of Stored Credentials (FPT_APW_EXT.1) [ESM_PM], [ESM_AC]

FPT_APW_EXT.1.1 The TSF shall store credentials in non-plaintext form.
FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext credentials.

Application Note: *User authentication data is stored on a LDAP or RADIUS server and administrator authentication data is stored in the UNIX/LINUX system user database. The TOE does not store or cache the data.*

5.2.7.2 Failure of Communications (FPT_FLS_EXT.1) [ESM_AC]

FPT_FLS_EXT.1.1 The TSF shall maintain policy enforcement in the following manner when the communication between the TSF and the Policy Management product encounters a failure state: [***deny all requests***].

5.2.7.3 Replay Detection (FPT_RPL.1) [ESM_AC]

FPT_RPL.1.1 The TSF shall detect replay for the following entities: [**secured tasks**].

FPT_RPL.1.2 The TSF shall perform [**reject the secured task**] when replay is detected.

Application Note: *The TOE relies on the implementation of TLS in the operational environment to provide secure transmission, including replay detection, of secured tasks.*

5.2.7.4 Extended: Protection of Secret Key Parameters (FPT_SKP_EXT.1) [ESM_PM], [ESM_AC]

FPT_SKP_EXT.1.1	The TSF shall prevent reading of all pre-shared keys, symmetric key, and private keys.

5.2.8 Resource Utilization (FRU_FLT.1)

5.2.8.1 Degraded Fault Tolerance (FRU_FLT.1)

FRU_FLT.1.1	The TSF shall ensure the operation of [enforcing the most recent policy] when the following failures occur: [restoration of communications with the Policy Management product after an outage].

5.2.9 Trusted path/channels (FTP)

5.2.9.1 Inter-TSF Trusted Channel (FTP_ITC.1) [ESM_PM], [ESM_AC]

FTP_ITC.1.1	Refinement: The TSF shall use [***TLS***] to provide a trusted communication channel between itself and authorized IT entities that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification and disclosure.
FTP_ITC.1.2	The TSF shall permit [***the TSF***] to initiate communication via the trusted channel.
FTP_ITC.1.3	Refinement: The TSF shall initiate communication via the trusted channel for transfer of policy data, [**external LDAP server, internal TOE component communications**].

5.2.9.2 Trusted Path (FTP_TRP.1) [ESM_PM]

FTP_TRP.1.1	Refinement: The TSF shall use [***TLS/HTTPS***] **to** provide a communication path between itself and remote administrators that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from disclosure and detection of modification of the communicated data.
FTP_TRP.1.2	The TSF shall permit [remote users] to initiate communication via the trusted path.
FTP_TRP.1.3	Refinement: The TSF shall require the use of the trusted path for initial user authentication, execution of management functions.

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are included by reference from the ESMPPs.

Requirement Class	Requirement Component
ADV: Development	ADV_FSP.1 Basic functional specification
AGD: Guidance documents	AGD_OPE.1: Operational user guidance
	AGD_PRE.1: Preparative procedures
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE
	ALC_CMS.1 TOE CM coverage
ATE: Tests	ATE_IND.1 Independent testing - conformance
AVA: Vulnerability assessment	AVA_VAN.1 Vulnerability survey

Table 5 Assurance Components

Consequently, the assurance activities specified in [ESMAC] and [ESMPM] apply to the TOE evaluation.

6. TOE Summary Specification

This chapter describes the security functions:

- Enterprise Security Management
- Security audit
- Communication
- User data protection
- Identification and authentication
- Protection of the TSF
- Security management
- Resource Utilization
- Trusted path/channels

6.1 Enterprise security management

The TOE provides a means to grant controlled access to functions on Unix/Linux type operating systems that otherwise would require the user to have full administrative (i.e., 'root') privileges. The TOE does not disable or otherwise replace the root account—instead, it provides a means whereby normal users can invoke commands that require 'root' or 'admin' privilege without having to be logged on as 'root' or 'admin'. Such commands are also referred to in the TOE documentation as 'secured tasks'.

Authorized administrators can create and manage policies that allow users access to protected commands and functions. Policies can be configured for the following object types: programs, files, host configuration, and authentication function. Policies are consumed and enforced by the Master Host component of the TOE. All types of policies that can be defined can also be enforced. Defined policies are uniquely identified by absolute path (directory and file name). The PowerBroker Policy File Selection Page enables you to specify the file name of the policy file to create or edit with the Policy Editor. To edit an existing policy file or create a new one, an administrator specifies the absolute path (directory and file name) for the policy file that you want to create or edit. Once the policy file has been saved it is immediately available and enforced. You can Browse/sort existing policy files using the File Browser Page in the GUI.

The objects and operations used in policies are derived from the operating system of the host source on which the objects reside. The operations on the objects which can be delegated within policies include: ability to create, read, modify, execute, delete, terminate, or change permissions of objects, and the ability to use authentication function. The attributes which can be used in the policies are uid, gid, and computername. Policies can be configured for users and administrators. The policy subject and attribute data is derived from LDAP or RADIUS sources for users and from UNIX/LINUX of the host the TOE is installed on for administrators. As such, user authentication data is stored on the corresponding LDAP or RADIUS server and administrator authentication data is stored in the corresponding UNIX/LINUX system user database. Identification and authentication of the users is performed by the corresponding LDAP or RADIUS server in the operational environment. Identification and authentication of the administrators is performed by checking the entered user name and password against the Unix/Linux passwords on the host that is running pbguid (Submit Host). Identification and authentication of users and administrators can optionally be performed using PAM in the operational environment.

Policies are defined and stored in, and used from the UNIX/LINUX host underlying the Master Host TOE component. The policies are never transmitted among TOE components or outside of the TOE. In this TOE, policies are centralized and do not get transmitted or pushed. The TOE is both an ESM access control product and an ESM policy management product and upon saving newly created or modified policies, the policy is immediately available.

The security policy can be structured, using the instructions in the policy scripting language, to ensure commands are accepted only from authorized users, or members of authorized groups, submitted from authorized hosts, on authorized days and at authorized periods of time. The security policy can also be structured to ensure commands are rejected from specified users, members of specified groups, submitted from specified hosts, submitted on specified days or at specified times. The scripting language also provides specialized forms of the `accept` and `reject` commands that allow all of these conditions to be specified in a single statement. The following examples are drawn from the Policy Language Manual.

- To accept all commands from user1: `accept from user1;`
- To accept all commands from user1 when they are submitted from host1: `accept from user1, host1;`
- To accept the “date” command from user1 submitted from any host: `accept from user1, , “date”;`
- To accept all commands from user3 when the time on the submitting host is between 9AM and 5PM: `accept from user3 when timebetween (“0900”, “1700”);`
- To reject all commands from user4: `reject from user4;`
- To reject all commands when the time on the submitting host is between 5PM and 9AM: `reject when timebetween (“1700”, “0900”);`
- To reject all commands from user5 when they are submitted from host5, and to provide a custom rejection message: `reject “Permission denied.” From user5, host5;`

The security policy can also be used to specify the host on which the command will actually be run and to specify the context in which the command will run (e.g., run as ‘root’, run as ‘dba’).

The TOE stores the policy files in the operational environment and therefore relies on the operational environment to prevent unauthorized access to those files.

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

- ESM_ACD.1: The TOE provides the ability to define access control policies for consumption by a compatible Access Control product: e.g. the TOE itself. Access control policies consist of subject, object, attributes and policies are uniquely identified.
- ESM_ACT.1: The TOE ensures that policies are available to the Access Control component immediately following creation of a new or updated policy.
- ESM_EAU.2(1), (2): The TOE relies on LINUX/UNIX host, LDAP and RADIUS in the operational environment for subject authentication; and requires each subject to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that subject.
- ESM_EID.2(1), (2): The TOE relies on LINUX/UNIX host, LDAP and RADIUS in the operational environment for subject identification; and requires each subject to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that subject.

6.2 Security audit

The TOE includes an internal log implementation that can be used to store and review audit records locally on the machine hosting the TOE. The TOE is designed to be able to generate log records for security relevant events as they occur. The events that can cause an audit record to be logged include starting and stopping the audit function, and all of the events identified in the table below and in **Table 2**.

The logged audit records identify the date and time (obtained from the underlying operating system), the nature or type of the triggering event, an indication of whether the event succeeded or failed, the identity of the user responsible for the event (retrieved from the underlying operating system), as well as the completion status of the applicable function. Note that the success or failure of an audited event is implied by the event type. The logged audit records also include event-specific content that includes at least all of the content required in **Table 2**. More details are provided in the table below.

The Master Host generates audit records for each attempt to submit a secured task to the TOE—ACCEPT for accepted tasks and REJECT for rejected tasks. Note the TOE always audits these attempts and as such auditing start-up and shutdown of the audit function is irrelevant since the TOE offers no means to stop the audit function. For secured tasks that are not run in local mode, the TOE also audits when the task terminates (FINISH event) and any keystroke monitoring events configured for the task in the security policy that were triggered by the task (KEYSTROKE events). The contents of each audit record that can be defined by the policy is fully customizable and can contain hundreds of objects and administrator defined variables.

The audit records identified in the table below identify the generically-specified auditable events in the SFRs and translate them into TOE-specific auditable events. The information that pertains to the SFR requirement is extracted from each audit record to clearly identify the required audit information.

Component	Event	Additional Information	Example Audit
ESM_ACD.1	Creation or modification of policy	Unique policy identifier	The audit record entry records the creation or modification of the policy. The policy is identified as /etc/pb/pbul_functions.conf". "hostname":"pbul-qa-aix61-01.unix.symark.com", "evtname": "file_import", "service":"pbdbutil9.1.0-08", "who":"root", "severity":16, "utc":"2015-12-07 14:59:11", "progname":"pbdbutil9.1.0-08", "version":"9.1.0-08", "arch":"rs6000_aixC", "data":{ "fname":"/etc/pb/pbul_functions.conf", "msg":"Innitial import", "version":1, "sid":8978524, "pid":10420340, "uid":0} **Audit Record Location: Configuration Database**
ESM_ACT.1 [ESM_PM]	Transmission of policy to Access Control products	Destination of policy	Policies are not transmitted, instead policies are stored centrally and requests are made against the central policy. Requests from the Submit Host (the Access control portion of the TOE) are transmitted to the Master Host (the Policy Management portion of the TOE). If the task

Component	Event	Additional Information	Example Audit
			is ACCECPTED by the policy, the Master Host transmits the secure task to the Run Host (the Access control portion of the TOE). The event log captures the entire process in the Event Log Accept record. The ACCECPT record captures the identification of the requesting user and each TOE component is identified. Portions of the ACCEPT Event Log entry is provided below. The information in the Event Log entry provides the identification of the information ('Accept' command), the destination (submithostip '10.0.2.20', runhost 'CC-PowerBroker-RunHost', and Master Host masterhost '10.0.2.11'). The Name of the policy in effect (lineinfile '/etc/pb/pbul_functions.conf') verifies that the latest and correct policy is in effect. Name of User Requesting the Privileged Command 'SUDO_USER=cctester' cwd '/home/cctester' Submit Host Identification TargetSubmitHostShortName 'CC-PowerBroker-Client' submithost 'CC-PowerBroker-Client' submithostip '10.0.2.20' clienthost '10.0.2.20' Run Host Identification pblocaldnodename 'CC-PowerBroker-RunHost' runhost 'CC-PowerBroker-RunHost' Master Host Identification pbmasterdnodename 'CC-PowerBroker-Master2' masterhost '10.0.2.11' masterhostip '10.0.2.11' Type of Command event 'Accept' Requested Elevated Command command 'whoami' Successful Execution of the Command event 'Finish' exitdate '2016/06/27' exitstatus 'Command finished with exit status 0' Location of the Audit Record eventlog '/var/log/pb.eventlog' Name of the Policy in Effect lineinfile '/etc/pb/pbul_functions.conf' **Audit Record Location: Event Log**
ESM_EAU.2 [ESM_PM]	All use of the authentication	None	The ACCEPT Event Log record below captures the successful authentication of “root” via the browser

Component	Event	Additional Information	Example Audit
	mechanism		interface GUI. Accept 2015/12/07 15:50:04 root pbul-qa-hpux11v3-01.unix.symark.com root 172.20.31.66 pbul-qa-hpux11v3-01.unix.symark.com /usr/sbin/pbguid log Authorized **Audit Record Location: Event Log**
FAU_GEN.1	Start-up of the audit functions	None	Dec 8 11:33:08 pbul-qa-hpux11v3-01 inetd[3821]: pblogd/tcp: Added service, server /usr/sbin/pblogd **Audit Record Location** **/var/log/syslog (on Linux)** **/var/adm/syslog (on Unix)**
FAU_GEN.1	Shut-down of the audit functions	None	Dec 8 11:39:18 pbul-qa-hpux11v3-01 inetd[3821]: Going down on signal 15 **Audit Record Location** **/var/log/syslog (on Linux)** **/var/adm/syslog (on Unix)**
FAU_SEL.1 [ESM_AC]	All modifications to audit configuration	None	The audit record below captures the audit configuration modified by the “logomit” command. "hostname": "pbul-qa-aix61-01.unix.symark.com", "evtname": "file_import", "service": "pbdbutil9.2.0-08", "who": "root", "severity": 16, "utc": "2016-05-24 17:17:48", "progname": "pbdbutil9.1.0-08", "version": "9.1.0-08", "arch": "rs6000_aixC", "data": { "fname": "/etc/pb.conf", "msg": "Logomit Added", "version": 3, "sid": 6226020, "pid": 4718624, "uid": 0} **Audit Record Location: Configuration Database**
FAU_SEL_EXT.1 [ESM_PM]	All modifications to audit configuration	None	The audit record below captures the audit configuration modified by the “logomit” command. "hostname": "pbul-qa-aix61-01.unix.symark.com", "evtname": "file_import", "service": "pbdbutil9.2.0-08", "who": "root", "severity": 16, "utc": "2016-05-24 17:17:48", "progname": "pbdbutil9.1.0-08", "version": "9.1.0-08", "arch": "rs6000_aixC", "data": {

Component	Event	Additional Information	Example Audit
			"fname": "/etc/pb.conf", "msg": "Logomit Added", "version": 3, "sid": 6226020, "pid": 4718624, "uid": 0} **Audit Record Location: Configuration Database**
FAU_STG_EXT.1 [ESM_PM], [ESM_AC]	Establishment and disestablishment of communications with audit server	Identification of audit server	The audit record captures the establishment of communication with the pblogd audit server. Dec 8 11:33:08 pbul-qa-hpux11v3-01 inetd[3821]: pblogd/tcp: Added service, server /usr/sbin/pblogd **Audit Record Location: Configuration Database**
FCO_NRR.2 [ESM_AC]	The invocation of the non-repudiation service	Identification of the information, the destination, and a copy of the evidence provided	Policies are not transmitted, instead policies are stored centrally and requests are made against the central policy. Requests from the Submit Host (the Access control portion of the TOE) are transmitted to the Master Host (the Policy Management portion of the TOE). If the task is ACCECPTED by the policy, the Master Host transmits the secure task to the Run Host (the Access control portion of the TOE). The event log captures the entire process in the Event Log Accept record. The ACCECPT record captures the identification of the requesting user and each TOE component is identified. Portions of the ACCEPT Event Log entry is provided below. The information in the Event Log entry provides the identification of the information ('Accept' command), the destination (submithostip '10.0.2.20', runhost 'CC-PowerBroker-RunHost', and Master Host masterhost '10.0.2.11'). A copy of the evidence provided is verified by the successful execution of the command (event 'Finish', exitdate '2016/06/27' exitstatus 'Command finished with exit status 0'). <u>Name of User Requesting the Privileged Command</u> 'SUDO_USER=cctester' cwd '/home/cctester' <u>Submit Host Identification</u> TargetSubmitHostShortName 'CC-PowerBroker-Client' submithost 'CC-PowerBroker-Client' submithostip '10.0.2.20' clienthost '10.0.2.20' <u>Run Host Identification</u> pblocaldnodename 'CC-PowerBroker-RunHost' runhost 'CC-PowerBroker-RunHost' <u>Master Host Identification</u> pbmasterdnodename 'CC-PowerBroker-Master2' masterhost '10.0.2.11'

Component	Event	Additional Information	Example Audit
			masterhostip '10.0.2.11' Type of Command event 'Accept' Requested Elevated Command command 'whoami' Successful Execution of the Command event 'Finish' exitdate '2016/06/27' exitstatus 'Command finished with exit status 0' Location of the Audit Record eventlog '/var/log/pb.eventlog' Name of the Policy in Effect lineinfile '/etc/pb/pbul_functions.conf' **Audit Record Location: Event Log**
FDP_ACC.1(1), (2)[ESM_AC]	Any changes to the enforced policy or policies	Identification of Policy Management product making the change	The audit record captures the policy "/etc/pb/pbul_functions.conf" modification. "hostname":"pbul-qa-spsol11-01.unix.symark.com", "evtname":"file_import", "service":"pbdbutil9.1.0-08", "who":"root", "severity":16, "utc":"2015-12-07 15:21:17", "progname":"pbdbutil9.1.0-08", "version":"9.1.0-08", "arch":"sparc_solarisD", "data":{ "version":2, "fname":"/etc/pb/pbul_functions.conf", "msg":"Policy Changed", "sid":15438, "pid":15484, "uid":0} **Audit Record Location: Configuration Database**
FDP_ACF.1(1), (2) [ESM_AC]	All requests to perform an operation on an object covered by the SFP	Subject identity, object identity, requested operation	Portions of the ACCEPT Event Log entry is provided below. The information in the Event Log entry provides the identification of the information ('Accept' command), the destination (submithostip '10.0.2.20', runhost 'CC-PowerBroker-RunHost', and Master Host masterhost '10.0.2.11'). The Name of the policy in effect (lineinfile '/etc/pb/pbul_functions.conf') verifies that the latest and correct policy is in effect. The subject "cctester" is requesting access to run the elevated command 'whoami'. Name of User Requesting the Privileged Command 'SUDO_USER=cctester' cwd '/home/cctester'

Component	Event	Additional Information	Example Audit
			Submit Host Identification TargetSubmitHostShortName 'CC-PowerBroker-Client' submithost 'CC-PowerBroker-Client' submithostip '10.0.2.20' clienthost '10.0.2.20' Run Host Identification pblocaldnodename 'CC-PowerBroker-RunHost' runhost 'CC-PowerBroker-RunHost' Master Host Identification pbmasterdnodename 'CC-PowerBroker-Master2' masterhost '10.0.2.11' masterhostip '10.0.2.11' Type of Command event 'Accept' Requested Elevated Command command 'whoami' Successful Execution of the Command event 'Finish' exitdate '2016/06/27' exitstatus 'Command finished with exit status 0' Name of the Policy in Effect lineinfile '/etc/pb/pbul_functions.conf' **Audit Record Location: Event Log**
FMT_MOF.1 [ESM_AC]	All modifications to TSF behavior	None	The audit record captures the policy "/etc/pb/pbul_functions.conf" modification. “hostname":"pbul-qa-hpux11v3-01.unix.symark.com", "evtname": "file_import", "service": "pbdbutil9.1.0-08", "who": "root", "severity": 16, "utc":"2015-12-07 16:09:25", "progname": "pbdbutil9.1.0-08", "version": "9.1.0-08", "arch": "ia64_hpuxA", "data":{ "version" :1, "fname": "/etc/pb/pbul_functions.conf", "msg": "Policy Modified", "sid": 23198, "pid":24697, "uid": 0} **Audit Record Location: Configuration Database**
FMT_SMF.1 [ESM_PM]	Use of the management functions	Management function performed	The audit record captures the management function of the creation of the "/etc/pb/pbul_functions.conf" policy.

Component	Event	Additional Information	Example Audit
			"hostname":"pbul-qa-spsol11-01.unix.symark.com", "evtname":"file_import", "service":"pbdbutil9.1.0-08", "who":"root", "severity":16, "utc":"2015-12-07 15:15:21", "progname":"pbdbutil9.1.0-08", "version":"9.1.0-08", "arch":"sparc_solarisD", "data":{ "msg":"New Policy Created", "fname":"/etc/pb/pbul_functions.conf", "version":7 ,"sid":15438, "pid":15469, "uid":0} **Audit Record Location: Configuration Database**
FMT_SMR.1 [ESM_PM]	Modifications to the members of the management roles	None	This is an audit record from importing the policy file, thus applying the policy. The policy file is what controls who can perform the management functions. "hostname":"pbul-qa-aix61-01.unix.symark.com", "evtname":"file_import", "service":"pbdbutil9.1.0-08", "who":"root", "severity":16, "utc":"2015-12-07 14:59:11", "progname":"pbdbutil9.1.0-08", "version":"9.1.0-08", "arch":"rs6000_aixC", "data":{ "fname":"/etc/pb/pbul_functions.conf", "msg":"Innitial import", "version":1, "sid":8978524, "pid":10420340, "uid":0} **Audit Record Location: Configuration Database**
FPT_FLS_EXT.1 [ESM_AC]	Failure of communication between the TOE and Policy Management product	Identity of the Policy Management product, reason for the failure	Dec 4 12:34:36 pbul-qa-spsol11-01 pbmasterd9.1.0-08: [ID 702911 auth.error] [14388] 8540.2 client on pbul-qa-hpux11v3-01.unix.symark.com is not SSL enabled **Audit Record Location:** **/var/log/pbmasterd.log (on Linux)** **/var/adm/pbmasterd.log (on Unix)**
FTP_ITC.1 [ESM_AC], [ESM_PM]	All use of trusted channel functions	Identity of the initiator and target of the trusted channel	The ACCEPT Event Log entry captures the use of the trusted channel functions. Portions of the ACCEPT Event Log entry are provided below that are applicable to this audit requirement. These two fields are in the Event Log entry identifies the

Component	Event	Additional Information	Example Audit
			initiator and target of the trusted channel. The IP address of the remote LDAP server and the user attempting to authenticate over the trusted channel to LDAP are recorded. LDAPServer “10.42.215.74” LDAPUser “tester” Portions of the ACCEPT Event Log entry are provided below that are applicable to this audit requirement. The fields are in the Event Log entry identifies the internal TOE component communications. The identity of the initiator and the targets for the trusted channel are recorded. Name of User Requesting the Privileged Command 'SUDO_USER=cctester' cwd '/home/cctester' Submit Host Identification TargetSubmitHostShortName 'CC-PowerBroker-Client' submithost 'CC-PowerBroker-Client' submithostip '10.0.2.20' clienthost '10.0.2.20' Run Host Identification pblocaldnodename 'CC-PowerBroker-RunHost' runhost 'CC-PowerBroker-RunHost' Master Host Identification pbmasterdnodename 'CC-PowerBroker-Master2' masterhost '10.0.2.11' masterhostip '10.0.2.11' Type of Command event 'Accept' Successful Execution of the Command event 'Finish' exitdate '2016/06/27' exitstatus 'Command finished with exit status 0' Location of the Audit Record eventlog '/var/log/pb.eventlog' Name of the Policy in Effect lineinfile '/etc/pb/pbul_functions.conf' **Audit Record Location: Event Log**

Component	Event	Additional Information	Example Audit
FTP_TRP.1 [ESM_PM]	All attempted uses of the trusted path functions	Identification of user associated with all trusted path functions, if available	Portions of the ACCEPT Event Log entry are provided below that are applicable to this audit requirement. The Event Log entry records the identification of the user associated with the trusted path function. Accept 2015/12/07 15:50:04 root CC-PowerBroker-Client root CC-PowerBroker-Master CC-PowerBroker-Master /usr/sbin/pbguid log Authorized **Audit Record Location: Event Log**

The generated records are sent to an internal Log Server for storage (note that if no Log Server (i.e., pblogd) is configured, the Master Host (i.e., pbmasterd) will serve as the Log Server). The Master Host also provides the version controlled Configuration Database that provides storage of key configuration, settings and policy files, and auditing of activities such as the creation of new files and version changes within controlled files. If the Log Server storage location is not the Master Host, the audit records are sent to the Log Server using TLS. The algorithms used by TLS are provided by an openSSL FIPS validated module in the operational environment. No pre-allocation of storage is performed by the TOE. Physical storage for log records (internal and external) is provided by the operational environment. The amount of audit data which can be stored is dependent upon on the amount of disk space available on the server hosting pblogd. The same applies for logs exported to external log servers. The TOE includes options for log file management, i.e. log file rotation and archiving based on time and/or size. Additionally, to help prevent loss of space on the file system for audit logs; space on the log host can be controlled and the system can be configured to fail over to the next log server with the logreservedfilesystems and logreservedblocks settings. The logreservedfilesystems and logreservedblocks settings enable the administrator to control free space on the logreservedfilesystems file systems, and cause an immediate failover if the log host's free space falls below logreservedblocks. If the number of free 1-KB blocks falls below logreservedblocks on any of the file systems that are specified in any of the logreservedfilesystems on the log host, then the log daemon immediately refuses any new requests, causing an immediate failover. The same happens on the Policy Server host if you are not using a log server. If the free space in any of the file systems containing /var/log or /usr/log falls below 10,000 blocks, then new requests are rejected. Requests that are already in progress are allowed to continue. If there are no Log Servers (including the Master Host) capable of recording an event (e.g., no disk space is available), the TOE itself would fail and therefore stop.

The TOE does not provide any interfaces to modify or delete the log files.

The Administrator can define variables inside the policy. The value of the policy variables is recorded in the event log by default immediately after the policy is saved and enforced. An administrator may implement selective auditing in order to disable certain items being entered into the event log by using the *logomit* function anywhere within the policy file. If the function is used globally, then selected items will be excluded from all event log records. In addition, the *logomit* function can be used inside of certain rules allowing item level omissions to occur only when certain conditions are met, i.e. for certain users, certain commands or certain hosts. Additionally, a more selective method allows for the eventlog to be disabled based on the statements inside the policy file. Each object type: Processes, Files, Host Configuration, and Authentication Function can be individually selected to be recorded in the event log or not. For example, to disable the eventlog for the whoami command, but still allow the command to run, the following policy code will disable the eventlog recording for this command only:

```
If (basename(command)==”/usr/bin/whoami”)
{ eventlog = “/dev/null”;
accept; }
```

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

- FAU_GEN.1: The TOE can generate audit records for events including starting and stopping the audit function and all events identified in **Table 2**. Furthermore, each audit record identifies the date/time, event type, outcome of the event, responsible subject/user, as well as the additional event-specific content indicated in **Table 2**.
- FAU_SEL.1: Selective audit capability is exercised by the Policy Management portion of the TOE by Administrator defined policy variables and by object identity.
- FAU_SEL_EXT.1: the Policy Management portion of the TOE configures the access-control related auditing functions by Administrator defined policy variables and by object identity.
- FAU_STG.1: The TOE protects the stored audit records in the audit trail from unauthorized deletion and modification.
- FAU_STG_EXT.1: The TOE transmits audit records to TOE internal storage and uses TLS for distributed communications. The TOE protects the stored audit records in the TOE-internal audit trail from unauthorized deletion and modification.

6.3 Communication

The TOE is both a Policy Management and Access Control product where policies are centralized and never transmitted. Policies are defined on a Master Host and available immediately as soon as it is saved. Once defined, the policy files never leave this location or otherwise traverse across the TOE or outside the TOE. Policies are defined by administrators using secured task requests sent to the Master Controller. The Submit Host identifies where to submit the Secured Task using Master host name. The TOE identifies the submitter of a Secured Task using Submit Host Name, and uid. The administrator who has defined the policy or any authorized administrator can immediately verify the existence of the policy by performing a policy lookup using the policy file name. The administrator can also verify the location (Master Host) of the policy by viewing the Master Host field/attribute which contains the name of the Master Host the policy file is located on.

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

FCO_NRR.2: The TOE shall enforce the generation of evidence of receipt for received policies at all times; relates attributes of the recipient of the information with fields of the information to which the evidence applies; and provides a capability to verify the evidence of receipt of information to originator given policies are immediately available.

6.4 User data protection

The TOE provides a means to grant controlled access to functions on Unix/Linux type operating systems that otherwise would require the user to have full administrative (i.e., ‘root’) privileges. The TOE does not disable or otherwise replace the root account—instead, it provides a means whereby normal users can invoke commands that require ‘root’ or ‘admin’ privilege without having to be logged on as ‘root’ or ‘admin’. Such commands are also referred to in the TOE documentation as ‘secured tasks’.

The TOE’s Access Control Policy function located on the Master Host enforces defined policies that allow users access to protected commands and functions. Policies can be configured and enforced for the following object types: programs, files, host configuration, and authentication function. Policies are consumed and enforced by the Master Host component of the TOE. All types of policies that can be defined can also be enforced. The operations allowed between subjects and objects are defined in **Table 3**.

In order to invoke a controlled command, the user uses one of the clients provided with the TOE: *pbrun*; *pbsh*; or *pbksh*. The client submits the command and its parameters, along with the user identity, user group, and the hostname of the computer from which the command was invoked, to *pbmasterd,* which evaluates the request against a policy file to determine whether the request will be accepted and executed, or rejected. The policy file is a collection of instructions that define the security rules the TOE applies during task verification processing. The instructions are written using PowerBroker’s security policy scripting language, a C-like interpreted language.

The critical instructions in the policy file are `accept` and `reject`. As soon as processing of the policy file reaches an `accept` statement, policy file processing ceases and the TOE attempts to execute the requested command. As soon as processing of the policy file reaches a `reject` statement, policy file processing ceases and the TOE notifies the user that the requested command has been rejected. If policy processing reaches the end of the file without encountering an `accept` or `reject` statement, the TOE will reject the request.

See Section 6.1 for a description of security policies can be structured examples of policy statements.

The TOE's self-protection Security Function Policy restricts access to protected objects that reside in the Operational Environment that impact the TOE's behavior, such as executable processes and/or configuration files. The TOE enforces policy that will not permit requested operations against objects that are defined to be protected unless the acting subject is the individual that was responsible for the TOE's installation and initial configuration. By default these objects are restricted to the root administrator of the UNIX/LINUX Operating System. Users are not located on the servers where log files and policy files reside, wouldn't be able to alter them even if they had rights. Optionally, the configuration and log files can be encrypted, thus securing sensitive information should the disks themselves become compromised. Encryption of these files has not been evaluated.

The PowerBroker Servers settings and configuration files contains settings that control PowerBroker Servers operation. The Settings and Configuration Policy Files are located in the /etc/pb.settings and /etc/pb.conf directories. Examples of Settings Files are enforcehighsecurity, keyfile, eventlog, and identification of which TOE Client and Server Programs are installed on the different TOE host components.

Installation and initial configuration of the TOE requires root access, but the policies enforced by the TOE can be configured so that it mediates access to its own configuration files. The users with root access can modify the TOE's configuration files directly (e.g., using any text-based editor available on the host system). The administrators can configure policies to restrict use of the tools necessary to manage the TOE and to control which files can be accessed using those tools, but the TOE does not define such restrictions by default. The administrative guidance instructs the system administrator (users with root access) not to define any policies allowing access to these protected files.

The TOE stores the settings and configuration files in the operational environment and therefore relies on the operational environment to prevent unauthorized access to those files.

The User data protection function is designed to satisfy the following security functional requirements:

- FDP_ACC.1(1) and FDP_ACF.1(1): The TOE controls access to commands that have been defined to be controlled on target hosts.
- FDP_ACC.1(2) and FDP_ACF.1(2): The TOE's self-protection Security Function Policy restricts access to objects that reside in the Operational Environment that impact the TOE's behavior.

6.5 Identification and authentication

The TOE associates the uid and gid user security attributes with subjects acting on the behalf of that user. The TOE uses an external LDAP or RADIUS server to authenticate users and enforces the result. The TOE determines the uid from the credentials presented at authentication and associates the gid retrieved from the authentication server with the corresponding uid.

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

- FIA_USB.1: The TOE associates user security attributes with subjects acting on the behalf of that user.

6.6 Security management

The TOE comprises both Access Control components and a Policy Management component. The components can be collocated or distributed. In a distributed configuration, the Policy Management component (Master Host) accepts connections/commands from configured Access Control components via TLS (e.g. Submit Host). In both configurations, the TOE components can be configured and managed by configuring text Policy files and text TOE configuration files using the Unix/Linux command line; and/or by using an administrative GUI accessible via HTTPS. The TOE is able to trust data originating from its Access Control components since the TOE components communicate with each other using trusted channels TLS and HTTPS.

The Protection Profile AC components are spread across the three TOE components. Management functions occur locally on each device with the Master Host providing both Policy Management and Access Control functionality. Configuration of each node is done locally on that node, through the use of locally defined configuration files. The management functions include managing the audited events, the repository for trusted audit storage, the access control SFP security attributes consisting of access control policies, access control policy attributes, implementation status of access control policies, and the policy being implemented by the TSF.

Note that it is not necessary to be able to manage access control behavior to enforce in the event of a communications outage because in the event of an outage no access is permitted. The design of the TOE requires all components be operational in order for requests for access to succeed. The TOE restricts queries to the defined policy details of the Policy Management component of the TOE to the Submit Host (CLI) and GUI components of the Access Control function of the TOE. Queries are executed at the request of an authorized administrator.

The administrative commands are restricted to authenticated users with root access. The TOE includes a pre-defined administrative role with root access: the Admin role (also referred to as AdminUsers). The role is enabled by default and allows users in the AdminUsers (by default 'root') Role to run any command on Run Hosts (by default only Submit Hosts). Users must be assigned to the role through definition of a policy. Once assigned to this role, the authenticated user can access the administrative commands from both the command line of the Submit Host and from the GUI. The TOE restricts the ability for administrative users to change their own passwords. All administrative commands including the ability for administrators to change their own passwords must occur over HTTPS or via local connection to CLI on the Submit Host which sends the request to the Master Host over TLS.

The superset of users with root access consists of the user(s) assigned to AdminUsers Role and the System Administrators with root access to the underlying Linux/Unix operating system. All administrative users are defined and maintained in the operational environment. The TOE relies on the operational environment to protect the management utilities from unauthorized use. The TOE restricts access by default in that the users are not permitted access without a policy definition specifically allowing access to that user or to a group that the user is a member of. The Policy Management component of the TOE can specify alternative initial values to override the default values when an object or information is created (at the request of an authorized administrator). Administrators can define additional roles using policies for users to manage the TOE or portions of the TOE in addition to the AdminUsers role; however this is not within the scope of the evaluation.

The TOE provides the management functions necessary to manage the TOE; specifically those identified in Section 5.2 **Table 4**. In particular, the security management functions provided by the TOE are as follows:

- Definition of the security policy that determines if submitted secured tasks are accepted or rejected—this can be done by editing the policy file using any text editor available in the operating environment, or by using either of the PB GUI programs (i.e., *pbguid*, *pbsguid*), each of which provides a policy file editor. This provides the capability to manage the PowerBroker Access Control Policy.
- Configuration of TLS, HTTPS, changing administrative passwords, and configuration of auditable events—this is done by setting the appropriate values in the TOE configuration files, using any text editor available in the operating environment, or by using either of the PB GUI programs (i.e., *pbguid*, *pbsguid*), each of which provides a configuration file editor.

The TOE is implemented in a manner that prevents inconsistent policy definitions by implementing a data driven policy which processes policy rules in the order defined. The administrator controls how the rules are processed by controlling the ordering of the rules. More restrictive policies are defined first to ensure a more restrictive access control. When the processing of the policy file reaches an accept or reject statement, then pbmasterd logs the result (possibly through the log server daemon); and either terminates the request (if processing stopped at a reject statement); or permits the request (if processing stopped at an accept statement). It is in this manner that the TOE's policy management engine defines an unambiguous hierarchical method of implementing a policy such that no contradictions can occur.

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

- FMT_MOF.1: The TOE restricts the ability to manage the TOE and the TOE functions to users with root access.
- FMT_MOF.1(1): The Access Control component of the TOE can manage the behavior of the Policy Management functions (e.g. a Policy Management product).
- FMT_MOF.1(2): The TOE restricts queries to the defined policy details of the Policy Management component of the TOE to the Submit Host (CLI) and GUI components of the Access Control function of the TOE. Queries are executed at the request of an authorized administrator.
- FMT_MOF_EXT.1: The TOE restricts the ability to manage the Access Control functions of the TOE to the Submit Host component or GUI components of the TOE. No roles are associated with the components but rather a secure TLS or HTTPS connection is configured.
- FMT_MSA.1: The TOE restricts the ability to manage the security attributes of the Access Control SFP to the Policy Management component of the TOE.
- FMT_MSA.3: The TOE provides restrictive default values for security attributes that are used to enforce the Access Control SFP. The Policy Management component of the TOE is authorized to specify alternative initial values to override the default values when an object or information is created (at the request of an authorized administrator).
- FMT_MSA_EXT.5: The TOE only permits definition of unambiguous policies.
- FMT_MTD.1: The TOE restricts the ability for administrative user's to change their own passwords.
- FMT_SMF.1 [ESM_PM]: The TOE includes the functions necessary to manage the TOE and the TOE functions.
- FMT_SMF.1 [ESM_AC]: The Access Control functions of the TOE managed by the Policy Management component of the TOE.
- FMT_SMR.1 [ESM_PM]: The TOE maintains management roles for the Policy Management component of the TOE and associates users with roles.

6.7 Protection of the TSF

User authentication data is stored on a LDAP or RADIUS server and administrator authentication data is stored in the UNIX/LINUX system user database. The TOE does not store or cache the data. The TOE does not offer any functions that will disclose to any users a stored cryptographic key; and all keys are stored encrypted using AES-256. The AES key used to encrypt/decrypt the key file is obfuscated within the binary code. Proprietary BeyondTrust code performs de-obfuscation only when the key is needed.

Should the TOE or a TOE component encounter a failure state; all access control requests are denied. The TOE is both an Access Control and Policy Management product; therefore if the TOE is in a failed state then no access control requests or decisions can be made.

Communications between distributed components of the TOE are protected using TLS. The use of TLS provides confidentiality of data (including secured tasks) transmitted between TOE components, protects against undetected modification of such data, and prevents successful attempts to replay such data.

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

- FPT_APW_EXT.1: The TOE does not offer any functions that will disclose to any user a plain text password. Passwords are stored in cryptographically protected from within the TOE.
- FPT_FLS_EXT.1.1: The TOE maintains policy enforcement by denying all requests when the TOE encounters a failure state.

- FPT_RPL.1: Communications between distributed components of the TOE are protected using TLS. The use of TLS provides confidentiality of data transmitted between TOE components, protects against undetected modification of data, and prevents successful attempts to replay data.
- FPT_SKP_EXT.1: The TOE does not offer any functions that will disclose to any users a stored cryptographic key.

6.8 Resource Utilization

The TOE is both an Access Control and Policy Management product. Should the TOE experience a failure, no access control is permitted until the system comes back up. Policies are defined and enforced on the same component: the Master Host. In order to define or enforce policy the Master Host must be operational. Communication between the Submit Host and the Master Host or between the Mast Host and the Run Host cannot occur when the communication is lost or down due to a network problem or a hardware failure. The communication channel from the Submit Host to the Master Host must be operational in order for a user request such as a policy definition to be transmitted. Likewise, the communication channel between the Master Host and the Run Host must be operational in order for an accepted processed request to be sent to the Run Host. This ensures that the most recent policy will always be enforced even after the recovery of a TOE failure..

The Resource Utilization function is designed to satisfy the following security functional requirements:

FRU_FLT.1: The TSF shall ensure the enforcement of the most recent policy when communications with the Policy Management product is restored after an outage.

6.9 Trusted path/channels

TOE components may be configured such that all components reside on the same machine or they can be located on separate machines. If the components are distributed, all communication channels are protected using TLS connections to prevent unintended disclosure or modification of the transferred data. Internal TOE data traversing the internal components may include audit data being sent to a dedicated audit host; or management commands sent from the Submit Host to the Master Host.

The TOE optionally can be configured to use an external LDAP Server to provide user and object attributes for use in policies. If an external LDAP server is configured for this purpose, the communication channel is protected using TLS.

The TOE is architected in such a manner as so the policies are never transmitted therefore there is no communication channel for which to protect policy traversals.

A remote administrator can establish secure remote connections with the TOE using HTTP over TLS. HTTPS ensures both integrity and disclosure protection. To successfully establish an interactive administrative session, the administrator must be able to provide acceptable user credentials (e.g., user id and password), after which they will be able to access the administrative GUI features.

The secure protocols are provided by NIST-validated cryptographic mechanisms included in the operational environment. The TOE uses FIPS capable OpenSSL v1.0.2a and uses a FIPS mode to disable non FIPS algorithms. Customers are instructed to choose their own validated FIPS Object Module and link that with the provided FIPS capable OpenSSL v1.0.2a. The validated Object Module and FIPS capable OpenSSL are in the operational environment.

The Trusted path/channels function is designed to satisfy the following security functional requirements:

- FTP_ITC.1: The TOE uses TLS to ensure that communication channels between the TOE and an external LDAP Server, and communications between internal distributed TOE components are so they are not subject to inappropriate disclosure or modification.

- FTP_TRP.1: The TOE provides HTTP over TLS to support secure remote administration. Administrators can initiate a remote session to the TOE using HTTPS that is secured (from disclosure and modification) using NIST-validated cryptographic operations. The use of all remote security management functions requires the use of this secure channel.

7. Protection Profile Claims

The ST conforms to the *Standard Protection Profile for Enterprise Security Management Access Control, Version 2.1, 24 October 2013 (pp_esm_ac_v2.1) and Standard Protection Profile for Enterprise Security Management Policy Management, Version 2.1, 24 October 2013 (pp_esm_pm_v2.1).*

As explained in Section 3, Security Problem Definition, the Security Problem Definition of the ESMPPs has been included by reference into this ST.

As explained in Section 4, Security Objectives, the Security Objectives of the ESMPPs have been included by reference into this ST.

The following table identifies all the Security Functional Requirements (SFRs) in this ST. Each SFR is reproduced from the ESMPPs and operations completed as appropriate.

Requirement Class	Requirement Component	Source
ESM: Enterprise Security Management	ESM_ACD.1: Access Control Policy Definition	ESM_PM
	ESM_ACT.1: Access Control Policy Transmission	ESM_PM
	ESM_EAU.2(1): Reliance on Enterprise Authentication	ESM_PM
	ESM_EAU.2(2): Reliance on Enterprise Authentication	ESM_PM
	ESM_EID.2(1): Reliance on Enterprise Identification	ESM_PM, ESM_AC
	ESM_EID.2(2): Reliance on Enterprise Identification	ESM_PM, ESM_AC
FAU: Security audit	FAU_GEN.1: Audit Data Generation	ESM_PM, ESM_AC
	FAU_SEL.1: Selective Audit	ESM_PM, ESM_AC
	FAU_SEL_EXT.1: External Selective Audit	ESM_AC
	FAU_STG.1: Protected Audit Trail Storage (Local Storage)	ESM_AC
	FAU_STG_EXT.1: External Audit Trail Storage	ESM_PM, ESM_AC
FCO: Communication	FCO_NRR.2: Enforced proof of receipt	ESM_AC
FDP: User data protection	FDP_ACC.1(1): Access Control Policy	ESM_AC
	FDP_ACC.1(2): Access Control Policy (Self-Protection)	ESM_AC
	FDP_ACF.1(1): Access Control Functions	ESM_AC
	FDP_ACF.1(2): Access Control Functions (Self-Protection)	ESM_AC
FIA: Identification and authentication	FIA_USB.1: User-Subject Binding	ESM_PM
FMT: Security management	FMT_MOF.1: Management of Functions Behavior	ESM_PM
	FMT_MOF.1(1): Management of Functions Behavior	ESM_AC
	FMT_MOF.1(2): Management of Functions Behavior	ESM_AC
	FMT_MOF_EXT.1: External Management of Functions Behavior	ESM_PM
	FMT_MSA.1: Management of Security Attributes	ESM_AC
	FMT_MSA.3: Static Attribute Initialization	ESM_AC
	FMT_MSA_EXT.5: Consistent Security Attributes	ESM_PM
	FMT_MTD.1: Management of TSF Data	ESM_PM
	FMT_SMF.1: Specification of Management Functions	ESM_AC
	FMT_SMF.1: Specification of Management Functions	ESM_PM
	FMT_SMR.1: Security Management Roles	ESM_PM, ESM_AC

Requirement Class	Requirement Component	Source
FPT: Protection of the TSF	FPT_APW_EXT.1: Protection of Stored Credentials	ESM_PM, ESM_AC
	FPT_FLS_EXT.1: Failure of Communications	ESM_AC
	FPT_RPL.1: Replay Detection	ESM_AC
	FPT_SKP_EXT.1: Protection of Secret Key Parameters	ESM_PM, ESM_AC
FRU: Resource Utilization	FRU_FLT.1: Degraded Fault Tolerance	ESM_AC
FTP: Trusted path/channels	FTP_ITC.1: Inter-TSF Trusted Channel	ESM_PM, ESM_AC
	FTP_TRP.1: Trusted Path	ESM_AC

Table 6 SFR Protection Profile Sources

8. Rationale

This security target includes by reference the ESMPPs Security Problem Definition, Security Objectives, and Security Assurance Requirements. The security target makes no additions to the ESMPP assumptions. ESMPP security functional requirements have been reproduced with the Protection Profile operations completed. TD055 allows FTA_TAB.1 to be placed on the operational environment which this ST does. Therefore the corresponding objective is identified as an objective for the environment and identified in Section 4.1. Operations on the security requirements follow ESMPP application notes and assurance activities. Consequently, ESMPP rationale applies but is incomplete. The TOE Summary Specification rationale below serves to complete the rationale required for the security target.

8.1 TOE Summary Specification Rationale

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

This Section in conjunction with Section 6, the TOE Summary Specification, provides evidence that the security functions are suitable to meet the TOE security requirements. The collection of security functions work together to provide all of the security requirements. The security functions described in the TOE summary specification are all necessary for the required security functionality in the TSF. **Table 7 Security Functions vs. Requirements Mapping** demonstrates the relationship between security requirements and security functions.

	Enterprise Security Management	Security audit	Communication	User data protection	Identification and authentication	Security management	Protection of the TSF	Resource Utilization	Trusted path/channels
ESM_ACD.1	X								
ESM_ACT.1	X								
ESM_EAU.2(1)	X								
ESM_EAU.2(2)	X								
ESM_EID.2(1)	X								
ESM_EID.2(2)	X								
FAU_GEN.1		X							
FAU_SEL.1		X							
FAU_SEL_EXT.1		X							
FAU_STG.1:		X							
FAU_STG_EXT.1		X							
FCO_NRR.2			X						
FDP_ACC.1(1)				X					
FDP_ACC.1(2)				X					
FDP_ACF.1(1)				X					
FDP_ACF.1(2)				X					
FIA_USB.1					X				
FMT_MOF.1						X			
FMT_MOF.1(1)						X			
FMT_MOF.1(2)						X			
FMT_MOF_EXT.1						X			
FMT_MSA.1						X			
FMT_MSA.3						X			
FMT_MSA_EXT.5						X			
FMT_MTD.1						X			
FMT_SMF.1						X			
FMT_SMF.1						X			
FMT_SMR.1						X			
FRU_FLT.1								X	
FPT_APW_EXT.1							X		
FPT_FLS_EXT.1							X		
FPT_RPL.1							X		
FPT_SKP_EXT.1							X		
FTA_TAB.1									
FTP_ITC.1									X
FTP_TRP.1									X

Table 7 Security Functions vs. Requirements Mapping