IBM AIX 6 for POWER V6.1 Technology level 6100-00-02 with optional IBM Virtual I/O Server Security Target

for CAPP, LSPP, and RBACPP Compliance

Version: 1.3
Status: Release
Last Update: 2008-04-07

Document History

Version	Date	Changes	Summary	Author
1.0	2007-09-14	Entire document	Baseline from AIX 5.3H ST.	Scott Chapman, atsec
1.1	2007-09-21	Entire document	Several changes as per review comments.	Scott Chapman, atsec
1.1.1	2007-09-27	Ch 5 & 8.	Fixed bolding and underlining issues in ch 5. Updated section 8.3.3 to include new SRFs.	Scott Chapman, atsec
1.1.2	2007-09-27	Ch 6	Updated AZ.5 and MIC.1.	Scott Chapman, atsec
1.1.3	2007-10-01	Ch2	Moved Trusted Execution text within chapter 2. Updated FDP_ACF.1(4).	Scott Chapman, atsec
1.2	2007-10-09	Entire document	Removed AZ.3, AZ.5 & PV.3. Moved PV.5 to PV.3. Removed WAS. Update WPAR text. Updated FDP_IFF.1. Rename Global WPAR to Global environment. Removed App WPAR from TOE. Added clogin.	Scott Chapman, atsec
1.2.1	2007-10-15		Removed AZ.4. Renumbered AZ.2 to AZ.1. Updated SM.3.	Scott Chapman, atsec
1.2.2	2007-10-24		Removed DLPAR support. Removed LDAP auth support for LSPP mode. Updated table in AU.1. Added CID & CLiC to ch 9.	Scott Chapman, atsec
1.2.3	2007-11-08		Updated tables 7 and 14. Removed FSF_EPS from TP.9. Updated FPT_TST.1.1. Added /var/efs to TP.5 table. Added LDAP auth support for LSPP mode. Updated FDP_IFF.1.2.	Scott Chapman, atsec
1.2.4	2007-12-07		Updated section 2.2.7. Updated FDP_IFF.1.2. Modified AIX technology level. Removed modcrypt LPP from table 2. Removed LDAP auth support for LSPP mode. Updated audit records in chp 5 FAU_GEN.1 table. Added Trusted AIX to chp 1.1 Keywords.	Scott Chapman, atsec
1.2.5	2007-12-10		Updated FPT_TST.1. Added IP Filter Control Policy (i.e. FDP_IFC.1(5), FDP_IFF.1(2), FDP_MSA.1(12), FDP_MSA.3(9)).	Scott Chapman, atsec
1.2.6	2007-12-12		Remove Trusted Execution (TE).	Scott Chapman, atsec
1.2.7	2007-12-13		Changed p6 to POWER6. Clarified that CIPSO is IPv4 stds compliant & IPv6 is AIX specific in Table 1 & in TN.2. Merged 2.2.13 into 2.2.11.	Scott Chapman, atsec
1.2.8	2007-12-14		Added FMT_MTD.1(11). Modified FMT_SMF.1, SM.3, & ch 8.	Scott Chapman, atsec
1.2.9	2007-12-21		Added TDS 6.1 to section 2.2.1.	Scott Chapman, atsec
1.3	2008-04-07			Scott Chapman, atsec

Table of Contents

List of Tables

References

[ANSI-X9.31]	American National Standards Institute, Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (X9.31), 1998
[CAPP]	Controlled Access Protection Profile, Issue 1.d, 8 October 1999
[CC]	Common Criteria for Information Technology Security Evaluation, August 2005, Version 2.3, Part 1-3, CCIMB-2005-08-001, CCIMB-2005-08-002, CCIMB-2005-08-003
[CEM]	Common Methodology for Information Technology Security Evaluation, August 2005, Version 2.3, CCIMB-2005-08-004
[FIPS180-2]	FIPS PUB 180-2: Specification for the SECURE HASH STANDARD, including Change Notice to include SHA-224, August 1, 2002
[FIPS186-2]	FIPS PUB 186-2 DIGITAL SIGNATURE STANDARD (DSS), 2000 January 27
[FIPS197]	FIPS PUB 197: ADVANCED ENCRYPTION STANDARD (AES), November 26, 2001
[FIPS46-3]	FIPS PUB 46-3: DATA ENCRYPTION STANDARD (DES), October 25, 1999
[GUIDE]	ISO/IEC PDTR 15446 Title: Information technology – Security techniques – Guide for the production of protection profiles and security targets, ISO/IEC JTC 1/SC 27 N 2449, 2000-01-04
[ITSEC]	Information Technology Security Evaluation Criteria, Version 1.2, CEC, June 1991
[LSPP]	Labeled Security Protection Profile, Version 1.b, 8 October 1999
[MLOSPP]	US Government Protection Profile for Multilevel Operating Systems in Medium Robustness Environments, Version 1.91, 16 March 2007
[RBACPP]	Role-Based Access Control Protection Profile, Version 1.0, July 30, 1998
[RFC3268]	IETF RFC 3268: Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)
[RFC4120]	IETF RFC 4120: The Kerberos Network Authentication Service (V5), July 2005
[SSLv3]	The SSL Protocol Version 3.0; http://wp.netscape.com/eng/ssl3/draft302.txt

1 Introduction

A security target (ST) document provides the basis for the evaluation of an information technology (IT) product or system (i.e., the Target of Evaluation (TOE)). An ST principally defines:

- A set of assumptions about the security aspects of the environment, a list of threats that the product is intended to counter, and any known rules with which the product must comply (Section 3, TOE Security Environment).
- A set of security objectives and a set of security requirements to address security problems (Sections 4 and 5, Security Objectives and IT Security Requirements, respectively).
- The IT security functions provided by the TOE which meet the set of requirements (Section 6, TOE Summary Specification).

The ST for a TOE is a basis for agreement among developers, evaluators, and consumers on the security properties of the TOE and the scope of the evaluation. The audience for a ST may include evaluators, developers and “those responsible for managing, marketing, purchasing, installing, configuring, operating, and using the TOE.”

1.1 ST Identification

Title: IBM AIX 6 for POWER V6.1 Technology level 6100-00-02 with optional IBM Virtual I/O Server Security Target for CAPP, LSPP, and RBACPP Compliance, Version 1.3

Keywords: AIX, AIX 6.1, general-purpose operating system, POSIX, UNIX, access control, discretionary access control, information protection, labels, mandatory access control, MLS, security, Trusted AIX, trusted operating system, LPAR, VIOS

Publication Date: 2008-04-07

This document is the security target for the CC evaluation of the AIX 6.1 operating system product with optional Virtual I/O Server, and is conformant to the Common Criteria for Information Technology Security Evaluation [CC] with extensions as defined in the Controlled Access Protection Profile [CAPP], the Labeled Security Protection Profile [LSPP], and the Role-based Access Control Protection Profile [RBACPP].

In addition, this document has been extended with portions of the Multilevel Operating Systems in Medium Robustness Environments Protection Profile [MLOSPP], but conformance is not claimed against [MLOSPP].

1.2 ST Overview

This security target documents the security characteristics of the AIX 6.1 operating system.

AIX 6.1 is a highly-configurable UNIX-based operating system that meets the requirements of the Controlled Access Protection Profile and the Labeled Security Protection Profile developed by the Information Systems Security Organization within the National Security Agency to map the TCSEC C2 and B1 classes of the U.S. Department of Defense (DoD) Trusted Computer System Evaluation Criteria (TCSEC) to the Common Criteria framework. This includes the requirements for Identification and Authentication, Audit, Object Reuse, and Access Control including the use of Access Control Lists. AIX 6.1 also meets the requirements of the Role-Based Access Control Protection Profile. This Security Target claims full compliance with the requirements of these Protection Profiles and also includes additional functional and assurance packages beyond those required by [CAPP], [LSPP], and [RBACPP].

The TOE can be installed in two different modes called “CAPP mode” and “LSPP mode”. In CAPP mode, the TOE offers the capabilities of [CAPP] without enforcing the mandatory access control polices and sensitivity labels of an [LSPP] system. In LSPP mode, the TOE offers the capabilities of [LSPP]. The mode of operation is decided at installation time. In both modes, role-based access control (RBAC) conforming to [RBACPP] is enabled.

LSPP mode includes enhanced sensitivity labels, authorizations, privileges, labeled printing, network filtering, and enhanced audit features. It enforces mandatory access control (MAC), mandatory integrity control (MIC) and discretionary access control (DAC) policies and can provide network filtering on incoming and outgoing packets. It supports the Internet Protocol (IP) and Common IP Security Option (CIPSO) and provides network filtering based on network interface and host filtering rules.

Several servers running AIX 6.1 (any combination of CAPP mode systems and LSPP mode systems can be used) can be connected to form a distributed system, but not all components of such a system are components of the TOE. The communication aspects within AIX 6.1 used for this connection are also part of the evaluation. It is assumed that the communication links themselves are protected against interception and manipulation by measures which are outside the scope of this evaluation.

Additionally, the IBM Virtual I/O Server (VIOS) is included in the evaluated configuration as an optional component.

1.3 CC Conformance

This ST is CC *Part 2 extended* and *Part 3 conformant*, with a claimed Evaluation Assurance Level of EAL4 augmented by ALC_FLR.3.

The extensions to part 2 of the Common Criteria are those introduced by the Controlled Access Protection Profile [CAPP] (which are also included in the Labeled Security Protection Profile [LSPP]).

1.4 Strength of Function

The claimed strength of function for this TOE is: SOF-medium.

1.5 Structure

The structure of this document is as defined by [CC] Part 1 Annex C.

- Section 2 is the TOE Description.
- Section 3 provides the statement of TOE security environment.
- Section 4 provides the statement of security objectives.
- Section 5 provides the statement of IT security requirements.
- Section 6 provides the TOE summary specification, which includes the detailed specification of the IT Security Functions.
- Section 7 provides the Protection Profile claim
- Section 8 provides the rationale for the security objectives, security requirements, TOE summary specification and PP claims.

1.6 Terminology

This section contains definitions of technical terms that are used with a meaning specific to this document. Terms defined in the [CC] are not reiterated here, unless stated otherwise.

AIX:	This document uses the term AIX for AIX 6.1 when discussing AIX 6.1 in general terms (i.e. when it applies to both CAPP mode and LSPP mode)..
Administrative User:	This term refers to an administrator of an AIX system. Some administrative tasks require the use of authorizations, which can be assigned to one or more user accounts while other tasks can be performed by specified users only. Authorizations can determine which privileges are assigned to a task.
Authentication data:	This includes a user identifier, password and authorizations for each user of the product.
CAPP Mode:	This term refers to the evaluated version and installation method of AIX that complies with [CAPP].
LSPP Mode:	This term refers to the evaluated version and installation method of Trusted AIX that complies with [LSPP].
Object:	In AIX, objects belong to one of four categories: file system objects, other kernel objects (such as processes, programs and inter-process communication), window system objects, and miscellaneous objects.
Privileges:	A privilege is an attribute of a process that allows the process to bypass specific restrictions and limitations of the system. Privileges are used to override security constraints and are controlled by an administrator.

Product: The term product is used to define all hardware and software components that comprise the AIX system including VIOS.

Public object: A type of object for which all subjects have read access, but only the TSF or the system administrators have write access.

Security Attributes:
As defined by functional requirement FIA_ATD.1, the term 'security attributes' includes the following as a minimum: user identifier; group memberships; user authentication data; and security-relevant roles.

Subject: There are two classes of subjects in AIX:

untrusted internal subject - this is an AIX process running on behalf of some user, running outside of the TSF (for example, with no privileges).

trusted internal subject - this is an AIX process running as part of the TSF. Examples are service daemons and the process implementing the identification and authentication of users.

System: Includes the hardware, software and firmware components of the AIX product which are connected/networked together and configured to form a usable system.

Target of Evaluation (TOE):
The TOE is defined as the AIX 6.1 operating system, running and tested on the hardware and firmware specified in this Security Target. The BootPROM firmware as well as the hardware form part of the TOE Environment.

Trusted AIX: The term refers to the multi-level security version of AIX.

User: Any individual/person who has a unique user identifier and who interacts with the AIX product. (See below.)

Users can further be categorized as follows:

User	Any entity (human user or external IT entity) outside the TOE that interacts with the TOE.
Human user	Any person who interacts with the TOE.
External IT entity	Any IT product or system, untrusted or trusted, outside of the TOE that interacts with the TOE.
Role	A predefined set of rules establishing the allowed interactions between a user and the TOE.
Identity	A representation (e.g., a string) uniquely identifying an authorized user, which can either be the full or abbreviated name of that user or a pseudonym.
Authentication data	Information used to verify the claimed identity of a user.
Authorized User	A user who may, in accordance with the TSP, perform an operation.

In addition to the above general definitions, this Security Target provides the following specialized definitions:

Access	A right to interact with a system resource.
Authorization	An attribute associated with a user account that allows the user to run restricted programs or to run public programs with additional privilege.
Authorized Administrator	A user whose account and session has authorizations allowing privileged, administrative commands to be run.
Category	The non-hierarchical portion of the sensitivity label. The terms ***compartment*** and ***category*** are used interchangeably within this ST.
Classification	The component of a sensitivity label that is hierarchical.
Compartment	See *category*.
Discretionary Access Control (DAC)	A control mechanism that mediates access based on user identity and owner-controlled attributes on objects.
Dominates	Greater than or equal to, as used with labels, privileges, and authorizations.
Integrity Label (TL)	An attribute of a system resource that represents the level of trust associated with the integrity of the resource or data associated with the resource. A TL has only a hierarchical component.
Mandatory Access Control	A control mechanism that mediates access based on a label associated with the

(MAC)	subject and a label associated with the object and where such labels are not generally under the control of the user/owner associated with the object or subject.
Mandatory Integrity Control (MIC)	A control mechanism that mediates access based on the integrity label associated with the subject and the integrity label associated with the object and where such labels are not generally under the control of the user/owner associated with the object or subject.
Mediation	The act of applying rules to determine access to TOE protected objects.
Privileges	An attribute of a process that allows the process to operate outside of the security policy of the TOE.
Sensitivity Label (SL)	An attribute of system resources that represents the sensitivity of the resource or data associated with the resource. An SL has both a hierarchical and a non-hierarchical component.
Trusted Computing Base (TCB)	The software components of the TOE that enforce the TSFs and which must remain inviolate in order to enforce the system security policies.
Trusted Network (TN)	The component of the system that labels internal and external network traffic and which mediates access between processes and network resources.

2 TOE Description

The target of evaluation (TOE) is the operating system AIX Version 6.1 and optional IBM Virtual I/O Server (VIOS).

AIX is a general purpose, multi-user, multi-tasking operating system. It is compliant with all major international standards for UNIX systems, such as the POSIX standards, X/Open XPG 4, Spec 1170, and FIPS Pub 180. It provides a platform for a variety of applications in the governmental and commercial environment. AIX is available on a broad range of computer systems from IBM, ranging from departmental servers to multi-processor enterprise servers, and is capable of running in an LPAR (Logical Partitioning) environment.

In LSPP mode, the TOE enforces MAC, MIC, DAC, and TCB control policies to implement security goals, such as confidentiality, integrity, and accountability. LSPP mode can operate in a network or stand-alone configuration. In a network configuration, LSPP mode supports BSO/ESO/CIPSO/RIPSO and provides network filtering on incoming and outgoing packets, based on network interface and host filtering rules.

The AIX evaluation shall consist of a closed network of high-end, mid-range and low-end IBM System p5 and POWER6 servers running the TOE.

The TOE Security Functions (TSF) consists of those parts of AIX that run in kernel mode plus some trusted processes. These are the functions that enforce the security policy as defined in this Security Target. Tools and commands executed in user mode that are used by the system administrator need also to be trusted to manage the system in a secure way but, as with other operating system evaluations, they are not considered to be part of this TSF.

The following table provides a guide as to what‘s supported in CAPP mode and what’s supported in LSPP mode. An ‘X’ means that the mode supports the description.

Table 1: CAPP mode vs. LSPP Mode

CAPP mode	LSPP mode	TOE Description
X	X	The TOE includes installation from CD-ROM and the network.
X	X	The TOE includes the Virtual Input/Output Server (VIOS) which allows for the virtualization of SCSI drives and network adapters.
X	X	System administration tools include the *smitty* non-graphical system management tool. The WebSM administrative tool is excluded.
X		The TOE includes standard networking applications, such as ftp, rlogin, rsh, and NFS. Port filtering will be used to protect network applications which might otherwise have security exposures.
	X	The TOE includes the following networking applications: telnet and ftp. It also includes NFS as a single level file system.
X		The TOE includes the X-Window graphical interface and many X-Window applications.
	X	The TOE supports BSO/ESO/CIPSO/RIPSO for IPv4 with an AIX specific implementation for IPv6 and provides network filtering on incoming and outgoing packets, based on network interface and host filtering rules.
		IT Environment Description
X	X	The IT environment includes the hardware and the BootProm firmware.
X	X	The IT environment includes applications that are not evaluated, but are used as unprivileged tools to access public system services, for example the Mozilla web browser or the Adobe Acrobat Reader to access the supplied online documentation (which is provided in HTML and PDF formats). No HTTP server is included in the evaluated configuration.
X		The IT environment includes LDAP for maintaining TOE authentication data.
X	X	The IT environment includes Kerberos for aiding in establishing a trusted channel between NFSv4 clients and servers.

2.1 Intended Method of Use

The TOE is a UNIX-based, multi-user, multi-tasking operating system. The TOE is a multi-user system providing service to several users at the same time. After successful login, the users have access to a general computing environment, allowing the start-up of user applications, issuing user commands at shell level, creating and accessing files. The TOE provides adequate mechanisms to separate the users and protect their data. Privileged commands are restricted to the system administrator roles.

The TOE can be configured to operate in one of two modes, *CAPP mode* and *LSPP mode,* as defined in section 1.2 of this document.

In LSPP mode, the TOE uses mandatory access control together with discretionary and role-based access control. In LSPP mode, rules are defined to assign *sensitivity labels* to subjects and objects and to implement the information flow mandatory access control policy defined in [LSPP]. CAPP mode also uses role-based access control.

The TOE permits one or more processors and attached peripheral and storage devices to be used by multiple users to perform a variety of functions requiring controlled shared access to the data stored on the system. Such installations are typical of personal, workgroup, or enterprise computing systems accessed by users local to, or with otherwise protected access to, the computer systems.

The TOE provides facilities for on-line interaction with users. Networking is covered only to the extent to which the TOE can be considered to be part of a centrally-managed system that meets a common set of security requirements.

It is assumed that responsibility for the safeguarding of the data protected by the TOE can be delegated to the TOE users. All data is under the control of the TOE. The data is stored in named objects, and the TOE can associate with each controlled object a description of the access rights to that object.

All individual users are assigned a unique user identifier. This user identifier supports individual accountability. The TOE authenticates the claimed identity of the user before allowing the user to perform any further actions.

The TOE enforces controls such that access to data objects can only take place in accordance with the access restrictions placed on that object by its owner or other suitably authorized user.

Access rights (e.g.,read, write, execute) can be assigned to data objects with respect to subjects (users). Once a subject is granted access to an object, the content of that object may be freely used to influence other objects accessible to this subject. In LSPP mode, the TOE allows mandatory access control based on sensitivity labels, integrity controls based on integrity labels and TCB flags, and access control to executable files based on role/authorization.

This ST describes both a “root enabled mode” and a “root disabled mode” available in CAPP mode, but **only “root enabled mode” is allowed in the evaluated configuration of CAPP mode**. All mention of root enabled mode and root disabled mode refer to a CAPP mode system only. (In root enabled mode, the ‘root’ user has the typical ‘root’ authority found in previous versions of AIX. In root disabled mode, the ‘root’ user has its authority reduced to the equivalence of an ordinary user.) Only root disabled mode is supported/allowed in LSPP mode.

AIX 6.1 is the platform for a large amount of commercial and scientific applications.

AIX 6.1 complies with the following international standards:

- XPG4 Base 95 Profile (X/Open Portability Guide)
- XPG4 Commands and Utilities V2 (X/Open Portability Guide)
- ANSI/IEEE 1003.2:1992
- ISO/IEC 9945-2 1993
- FIPS PUB 189 (Effective date April 3, 1995)
- SPEC 1170

AIX 6.1 has significant security extensions compared to standard UNIX systems:

- Access control lists
- Integrity protection
- A journaled file system
- Integrated login framework

- Multiple administrative and user roles
- TCB integrity protection
- IP filtering

AIX 6.1 provides easy to use interfaces for users and system administrators:

- SMIT (`smitty`) for system and user administration

LSPP mode provides the following additional security mechanisms:

- MAC labels for objects and subjects
- MIC labels for objects and subjects
- User account clearances
- Mandatory access control
- Mandatory integrity control
- Least privilege
- Networking controls based on label
- Labeling of printed output

2.2 Summary of Security Features

The following sections present a summary of the security features that the TOE offers. These security features are supported by domain separation and reference mediation, which ensure that the features are always invoked and cannot be bypassed.

2.2.1 Identification and Authentication

AIX provides identification and authentication (I&A) based upon user passwords. The quality of the passwords used can be enforced through configuration options controlled by AIX. The evaluated configurations for I&A are:

- The default configuration for authentication, which uses passwords to authenticate users.
- CAPP Mode Only: The LDAP authentication method configured for UNIX-type authentication, which uses passwords to authenticate users. (In the UNIX-type configuration, LDAP only stores the data used for I&A. It does not perform I&A for AIX. AIX must communicate with the LDAP server across an SSL connection.)

Other authentication methods (e. g. Kerberos authentication) that are supported by AIX in general are not part of the evaluated configuration. Especially pluggable authentication modules that would allow e. g. to use a token based authentication process are not part of the evaluated configuration.

IBM Tivoli Directory Server (ITDS) 6.0 and 6.1 are used for the LDAP service. The ITDS client interface used by AIX uses the IBM Global Services Kit (GSKit) for performing the SSL services. The client interface, including GSKit, is part of the TOE.

2.2.2 Auditing

AIX can collect extensive auditing information about security related actions taken or attempted by users, ensuring that users are accountable for their actions.

For each event record, the audit event logger prefixes an audit header to the event-specific information. This header identifies the user and process for which this event is being audited, as well as the time of the event. The code that detects the event supplies the event type and return code or status and optionally, additional event-specific information (the event tail). Event-specific information consists of object names (for example, files refused access or tty used in failed login attempts), subroutine parameters, and other modified information.

This audit trail can be analyzed to identify attempts to compromise security and determine the extent of the compromise. The audit tools can also extract audit records of events involving objects and/or subjects having specified security attributes.

2.2.3 Discretionary Access Control

Discretionary Access Control (DAC) restricts access to objects, such as files and is based on Access Control Lists (ACLs) and the standard UNIX permissions for user, group and others. Access control mechanisms also protect IPC objects from unauthorized access. CAPP mode supports ACLs on sockets for TCP connections. LSPP mode supports ACLs on network ports and interfaces.

In addition, AIX supports the Encrypted Filesystem (EFS) which allows for the encryption and decryption of files using the Advanced Encryption Standard (AES). File encryption works as a type of access control mechanism. The user must have DAC access and have access to the file's encryption key in order to decrypt the file's content. AIX uses the IBM CryptoLite for C (CLiC) cryptographic module for EFS encryption and decryption.

Additionally, VIOS provides DAC between VIOS SCSI device drivers acting on behalf of LPAR partitions as subjects and logical/physical volumes as objects. It also provides DAC between VIOS Ethernet device drivers acting on behalf of groups of LPAR partitions sharing a virtual network and VIOS Ethernet adapter device drivers where one is the subject and the other is the object (the Ethernet packets cannot contain VLAN tags).

2.2.4 Object Reuse

All resources are protected from Object Reuse (*scavenging*) by one of three techniques: explicit initialization, explicit clearing, or storage management. Four general techniques are used to meet this requirement:

- ***Explicit Initialization:*** The resource's contents are explicitly and completely initialized to a known state before the resource is made accessible to a subject after creation.
- ***Explicit Clearing:*** The resource's contents are explicitly cleared to a known state when the resource is returned for re-use.
- ***Storage Management:*** The storage making up the resource is managed to ensure that uninitialized storage is never accessible.
- ***Erase Disk:*** AIX offers as part of its diagnostic subsystem an Erase Disc service aid that can be invoked by the administrator to overwrite all data currently stored in user-accessible blocks of a disk with pre-defined bit patterns.

2.2.5 Security Management

The management of the security critical parameters of AIX is performed by administrative users. A set of commands that require system administrator privileges are used for system management. Security parameters are stored in specific files that are protected by the access control mechanisms of the TOE against unauthorized access by users that are not administrative users.

In CAPP mode and LSPP mode, security management can be split between different roles.

VIOS defines a separate set of roles for system management than AIX. Each VIOS role has a set of commands available to it. Security parameters are stored in specific files that are protected by the access control mechanisms of the TOE against unauthorized access by users.

2.2.6 TSF Protection

While in operation, the kernel software and data are protected by the hardware memory protection mechanisms. The memory and process management components of the kernel ensure a user process cannot access kernel storage or storage belonging to other processes.

TSF software and data, files and directories, kernel objects, IPC and networks sockets/packets are protected by TCB, DAC, and process isolation mechanisms. LSPP mode provides additional mechanisms of MAC and MIC.

The TOE and the hardware and firmware components are required to be physically protected from unauthorized access. The system kernel mediates all access to the hardware mechanisms themselves, other than program visible CPU instruction functions.

The system administrator has the ability to start a program that checks the hardware for correct operation.

LSPP Mode Only: The operational mode of AIX is intended to be the standard operating mode of the machine. The restrictions associated with operational mode cannot be overridden or bypassed by any mechanism. These restrictions are:

1. the kernel security flags cannot be modified
2. FSF_TLIB and FSF_TLIB_PROC flagged objects cannot be created, modified, or deleted

2.2.7 Privileges, Authorizations, Roles, and Superuser Emulation

The TOE implements a privilege mechanism within the kernel that allows users to implement the *least privilege principle*. A privilege is an attribute of a process that allows the process to bypass specific restrictions and limitations of the system. Privileges are associated only with processes, not user accounts. Privileges are used to override security constraints, to permit expanded use of certain system resources such as memory and disk space, and to adjust the performance and priority of the process. Restricting privileges on a process limits the damage that can result if an operation is improperly performed. Untrusted programs must not have any privileges assigned to them.

The TOE least privilege mechanism can take the place of the traditional user ID 0 (superuser/root) mechanism of UNIX. In LSPP mode, user ID 0 is treated exactly like any other system user ID unless superuser emulation is in effect for the process. In CAPP mode with root enabled mode enabled, user ID 0 supports the traditional superuser mechanism.

Privileges can be associated with executable files and assigned to an executing process, similar to the way the setuid bit on a file modifies the executing process's user ID. A process can also be prevented from acquiring privileges via the exec mechanism. Privileges can be used directly within a user-level program that is responsible for mediating or enforcing security by having the program retrieve its privilege set from the kernel and to make decisions based on the presence or absence of specific privileges. A process can temporarily disable one or more of its privileges if the process needs to perform an action on the system without bypassing the system security policy.

The TOE supports the policy of separation of duties, which provides for the compartmentalization of responsibility reducing the potential damage from a corrupt user or administrator, and places limits on the authority of the user or administrator. Authorizations provide a mechanism to grant rights to users to perform particular actions and run particular programs, such as programs that will run with privileges to bypass MAC, MIC, or DAC limitations. Each authorization has a well-defined set of functions that can be performed by users who are granted that authorization. There are two types of authorized users: administrative role users and ordinary users. An administrative user is any authorized user that has one or more of the RBAC related authorizations (see the next paragraph for a discussion on RBAC). An ordinary user has no RBAC authorizations.

A role-based access control (RBAC) mechanism is implemented in the TOE. Roles are predefined collections of authorizations that can be assigned to users. AIX comes with a set of predefined roles. It also allows system administrators to create new roles for their environment. AIX has two types of RBAC: Legacy RBAC and Enhanced RBAC. The evaluated configuration uses Enhanced RBAC only. All references to RBAC in this document imply Enhanced RBAC unless otherwise specified.

A program has the ability to query the active authorizations associated with the user running the program, and the program can behave differently and use different privileges based on the authorization set of the user running the program. For the evaluated configuration, administrators (or, administrative users) are defined as all users that have any authorization assigned to them. All user IDs below 205 are considered system IDs; they are typically used for daemons and other trusted applications.

Additionally, AIX provides a Privileged Commands (privcmds) database for granting privileges and setuid/setgid capabilities to trusted executables at runtime when a user has the proper authorizations. When the kernel invokes a program, it checks the database for the existence of the program. If the program exists and the user has the proper authorizations, the discretionary access control on the program is ignored and the program is invoked with the privileges and/or setuid/setgid specified in the privcmds database.

The TOE provides a superuser emulation mechanism that allows the system to operate similar to a standard UNIX system. Superuser emulation can be enabled for specific processes while leaving all other processes running under the standard TOE least privilege and authorization mechanisms. There are several ways in which a process can emulate superuser:

1) A process can be granted all privileges on the system, regardless of its user ID.
2) Using the PV_SU_ROOT privilege, a process can be granted all privileges associated with standard AIX/UNIX superuser regardless of its user ID, such as the privileges to bypass any DAC restrictions and to management the auditing mechanism, but not privileges that are specific to the TOE-provided augmentation of standard AIX/UNIX security functionality, such as the privileges to modify kernel authorization tables, override MAC checks, etc.

3) Alternatively, the PV_SU_EMUL privilege can be set to grant processes all privileges associated with standard AIX/UNIX superuser when their process user ID is 0.
4) A process can be granted all authorizations/roles regardless of its user ID.
5) A process can be granted a "virtual user ID" of 0 so that queries to the kernel for its user ID will return 0 even regardless of the actual user ID associated with the process.

2.2.8 Mandatory Access Control (LSPP Mode Only)

LSPP mode provides full mandatory access control (MAC) for all objects on the system. Every file, directory, IPC object, and process on the system is given a sensitivity label (SL) which cannot be modified by an unprivileged process. Each user account is assigned a range of valid SLs, and the user can only operate on the TOE within that range. A process (or user) can only create objects at its current SL, and can only read and write objects subject to the MAC restrictions imposed by the system. It is not possible for unauthorized users to "downgrade" information or to bypass MAC restrictions by any utility or application on the system. Copies of a file, or portions of a file, created by any possible means, will always be protected at an SL at least as high as the original file.

2.2.9 Mandatory Integrity Control (LSPP Mode Only)

LSPP mode provides full mandatory integrity control (MIC) for all objects on the system. Every file, directory, IPC object, and process on the system is given an integrity label (TL) which cannot be modified by an unprivileged process. Each user account is assigned a range of valid TLs, and the user can only operate on the TOE within that range. A process (or user) can only create objects at its current TL, and can only read and write objects subject to the MIC restrictions imposed by the system. It is not possible for unauthorized users to "upgrade" integrity levels associated with data or to bypass MIC restrictions by any utility of application on the system. Copies of a file, or portions of a file, created by any possible means, will always be protected at a TL no greater than that of the original file.

2.2.10 TCB Protection

The TOE provides the concept of a Trusted Computing Base (TCB). Kernel, device drivers, system administration utilities, and other critical software that is used to enforce and administer the security of the system are part of this TCB. In addition, any file system object in the TOE (file, directory, device, etc.) can be marked with a TCB flag: FSF_TLIB. Alternatively, executables can be marked with the FSF_TLIB_PROC flag. The TCB is subject to several bypass control mechanisms enforced by the TOE, such as additional access control and integrity protection. Changes to objects being flagged as TCB objects can only be made when the system is in configuration mode or when the kernel security flag *trustedlib_enabled* is disabled.

The integrity of objects in the TCB database is verified at every system startup and at the request of an authorized administrator.

2.2.11 Trusted Network (TN – LSPP Mode Only) & IP Filtering

In LSPP mode only, the TOE provides export and import of labeled data via network interfaces and enforces mandatory access control for network traffic by means of Trusted Network (TN). TN provides two sets of networking rules: network interface and host filtering. Both types of networking rules determine what processing occurs on a packet before its transmission or when it is received. These rules apply sensitivity labels to packets and enforce MAC restrictions on packets according to those labels.

TN network interface rules enforce packet label processing based on the physical network interface of the host. Host rules enforce packet label processing based on the source and destination IP addresses (with network masking allowed) of the packet, the source and destination ports of the request, and the protocol being used. Both types of rules provide several criteria for determining which packets to drop and which to pass.

In both CAPP and LSPP mode, AIX supports IP filtering of packets flowing to and through the AIX system. IP packet flow can be permitted or denied based on several criteria/rules including presumed source address, destination address, and destination ports. IP packet filtering includes time-based rules where packet flow can be permitted or denied for a limited period of time after which the rules change.

2.2.12 Workload Partitions (WPAR)

AIX supports virtual environments called Workload Partitions (WPARs) which provide virtual AIX environments within AIX. WPARs provide process isolation so that applications can be installed and tested in a virtual environment.

AIX supports two types of WPARs: System WPARs and Application WPARs. Only System WPARs are evaluated as part of the TOE (i.e. Application WPARs are not part of the evaluation).

A System WPAR is a virtual AIX system with its own set of users, administrators, hostname, network addresses, process isolation, IPC isolation, and file system isolation. An Application WPAR is similar to a System WPAR except without file system isolation. With the advent of WPARs, the main AIX environment is now called the Global environment. Multiple WPARs can be created and executed within the Global environment by a system administrator.

2.3 Software

The Target of Evaluation is based on the following system software:

- IBM AIX 6 for POWER V6.1, Program Number 5765-G62, with Recommended Technology Package 6100-00-02 (6.1).
- The Virtual I/O Server (VIOS) contained in IBM Advanced Power Virtualization Version 1.5, Program Number 5765-G30.

The TOE documentation is supplied on CD-ROM.

The following table contains a list of LPPs / File Sets that make up the TOE. For each of these “LPP Names” there may be multiple actual installable components with that prefix. An ‘X’ means that the mode supports the LPP.

Table 2: List of LPPs / File Sets

CAPP mode	LSPP mode	LPP Name	Description
X	X	bos	AIX Base Operating System
X	X	devices	AIX supported devices
X	X	printers	AIX printer drivers and control files
X	X	sysmgt	System management tools
X		X11	X Windows server, libraries, and applications.
X	X	krb5.client	Kerberos client (optional)
X		ldap.client	ITDS (LDAP) client (optional)
X	X	clic	CLiC cryptographic module (optional)

The following PTFs are required on the evaluated system:

Table 3: List of PTFs

PTF	LPP
n/a	n/a

2.4 Configurations

The evaluated configurations are defined as follows.

- Either the CAPP installation mode or the LSPP installation mode must be selected during installation time.
- If CAPP mode is selected, RBAC must also be selected. (LSPP mode includes RBAC.)
- AIX 6.1 supports the use of IPv4 and IPv6.
- Only 64 bit architectures are included.
- Web Based Systems Management (WebSM) is not included.
- Both network (NIM, Network Install Manager) and CD installations are supported.

- Only the default mechanism for identification and authentication and, in CAPP mode only, the LDAP authentication method configured for “UNIX-type” authentication with an SSL connection are included. Support for other authentication options, e.g.,smartcard authentication, is not included in the evaluation configuration.
- If the system console is used, it must be connect directly to the workstation and afforded the same physical protection as the workstation.
- AIX 6.1 provides both a native and a Sys5 print system. In LSPP mode, only Sys5 is supported in the evaluated configuration, as it implements the labeling requirements from LSPP, and only single-level printers are supported.
- LSPP Mode Only: System security flags (a.k.a. kernel security flags) need to be configured as identified in section 6.2.14.1).
- The system must be configured to disable remote access for an individual user after five consecutively failed login attempts have occurred for this user.
- If in CAPP mode and if a windowing environment is used, the CDE file set must be selected at installation time.
- CLiC is included in the evaluated configuration.
- Dynamic Partitioning (Dynamic LPAR, DLPAR) is not supported in the evaluated configuration, i.e. the dynamic (de-) allocation of resources to a partition during operations is not allowed and must be prevented by organizational means in the IT environment.

The TOE comprises one of the server machines (and optional peripherals) listed in section 2.4.2 running the system software listed in Table 2 (a server running the above listed software is referred to as a “TOE server” below).

If the product is configured with more than one TOE server, they are linked by LANs, which may be joined by bridges/routers or by TOE workstations which act as routers/gateways or they connect using the Virtual Input/Output Server (VIOS).

If other systems are connected to the network they need to be configured and managed by the same authority using an appropriate security policy not conflicting with the security policy of the TOE.

2.4.1 File systems

The following file system types are supported:

- the AIX journaled filesystem *jfs2*
- the High Sierra filesystem for CD-ROM drives, *cdrfs*
- the DVD-ROM file system, *udfs*
- The process file system, *procfs (*/proc*)*, provides access to the process image of each process on the machine as if the process was a “file”. Process access decisions are enforced by MAC (LSPP mode only), MIC (LSPP mode only), and DAC attributes inferred from the underlying process’s and user security attributes.
- the Network File System, *nfs* (V3, V4)
- the Encrypted Filesystem (EFS)

LSPP Mode Note: cdrfs, udfs, procfs and (client-side) nfs are single level file systems. For mandatory access control, the labels of their mount point apply to all objects in the mounted file system. Single level file systems are not subject to mandatory integrity control and TCB policies, and their objects cannot be associated with privileges.

2.4.2 Technical Environment for Use

The following assumptions about the technical environment the TOE is intended to be used in are made:

a) The TOE is running on the following hardware platforms:
   - The TOE is running in an LPAR on an IBM System p5 POWER5 or POWER5+ server.
   - The TOE is running in an LPAR on an IBM System p5 POWER6 server.

b) The following peripherals can be run with the TOE preserving the security functionality:
   - all terminals supported by the TOE

- all storage devices and backup devices supported by the TOE (hard disks, CDROM drives, streamer drives, floppy disk drives)[1]
- all printer devices supported by the TOE

c) Network connectors supported by the TOE (e.g., Ethernet) supporting TCP/IP services over the TCP/IP protocol stack.

d) NFSv4 supports the use of NAS v1.4 (Kerberos Version 5) for aiding in establishing a trusted channel between NFSv4 clients and servers. NAS v1.4 is part of the TOE environment. NAS v1.4 must be configured to use LDAP for its database.

2.4.2.1 LPAR Environment

The logical partitioning capable System p5 and POWER6 eServers that represent the underlying hardware for the TOE support a logical partitioned environment that enables the System p5 and POWER6 systems to run multiple logical partitions concurrently. In a logical partition, an operating system instance runs with dedicated resources: processors, memory, and I/O slots. These resources are statically assigned to the logical partition. The total amount of assignable resources is limited by the physically installed resources in the system. Because the implementation of logical partitioning is static, one has to shut down every operating system instance in all logical partitions to change the resource assignment of running logical partitions.

From a functional point of view, applications on top of an operating system are running inside partitions in the same way they run on a stand-alone System p5 machine. There are no issues when moving an application from a stand-alone server to a partition. Operating system software needs to be modified in some areas to call Hypervisor functions instead of native code. The design of partitioning-capable System p5 and POWER6 servers is such that one partition is isolated from software running in the other partitions, including protection against natural software defects and even deliberate software attempts to break the partition barriers.

The logical resources of the underlying hardware that can be assigned to a partition are:

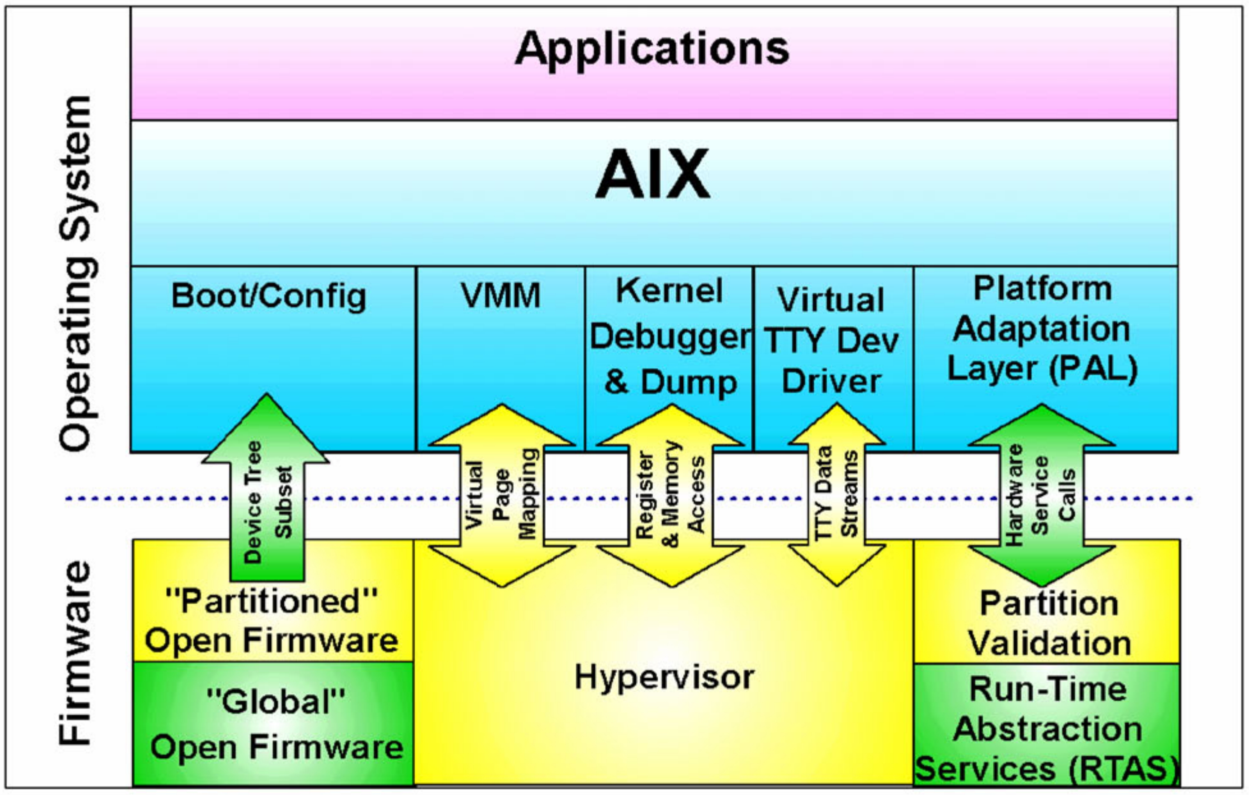
- Processors
- Main memory regions
- I/O slots

The assignment of those resources to the individual logical partitions is stored in non-volatile RAM. This part of the NVRAM is maintained by a “Service Processor” and cannot be read or modified directly by the TOE running in a logical partition. The assignment itself is performed by a System Administrator, who uses a “Hardware Management Console” (HMC) to define those assignments. The HMC communicates with the service processor that accepts the commands from the HMC and sets the values to define the logical partitions in the non-volatile RAM (NVRAM) accordingly. A Run-Time Abstraction Layer (RTAS) provides an abstraction mechanism for platform service calls.

The functions of the underlying LPAR architecture need to be used by different parts of the TOE. The following figure shows the parts of AIX that interact with the functions of the IT environment. Adaptations in AIX have been made to enable the TOE to interact in an LPAR specific way with the VMM, virtual TTY console, RTAS and kernel debugger.

Please note that the support of static LPARs does not introduce any additional security functionality for the TOE - the separation between partitions and protection of the TOE from operating systems running in other logical partitions on the same underlying machine is completely enforced by the underlying machine.

[1] The system distinguishes between storage and backup devices. Storage devices are hardware devices holding parts of the AIX file system, such as hard disks and CD ROMs. Backup devices are devices used for archiving data like floppy disks and streamer tapes that do not have a file system. Note that the distinction depends on the actual usage.


Operating System
Firmware
Applications
AIX
Boot/Config
VMM
Kernel Debugger & Dump
Virtual TTY Dev Driver
Platform Adaptation Layer (PAL)
Device Tree Subset
Virtual Page Mapping
Register & Memory Access
TTY Data Streams
Hardware Service Calls
"Partitioned" Open Firmware
"Global" Open Firmware
Hypervisor
Partition Validation
Run-Time Abstraction Services (RTAS)

3 TOE Security Environment

3.1 Introduction

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

To this end, the statement of TOE security environment identifies the list of assumptions made on the operational environment (including physical and procedural measures) and the intended method of use of the product, defines the threats that the product is designed to counter, and the organizational security policies with which the product is designed to comply.

3.2 Threats

The assumed security threats are listed below.

The **IT assets** to be protected comprise the information stored, processed or transmitted by the TOE. The term “information” is used here to refer to all data held within a server, including data in transit between workstations.

The TOE counters the general threat of unauthorized access to information, where “access” includes disclosure, modification and destruction.

The **threat agents** can be categorized as either:

- unauthorized users of the TOE, i.e. individuals who have not been granted the right to access the system; or
- authorized users of the TOE, i.e. individuals who have been granted the right to access the system.

The threat agents are assumed to originate from a well managed user community in a non-hostile working environment, and hence the product protects against threats of inadvertent or casual attempts to breach the system security. The TOE is not intended to be applicable to circumstances in which protection is required against determined attempts by hostile and well funded attackers to breach system security.

The threats listed below are grouped according to whether or not they are countered by the TOE. Those that are not countered by the TOE are countered by environmental or external mechanisms.

3.2.1 Threats countered by the TOE

T.ACCESS	A user may gain access to resources or perform operations for which no access rights have been granted.
T.OPERATE	Compromise of the IT assets may occur because of improper administration and operation of the TOE.
T.ROLEDEV	The development and assignment of user roles may be done in a manner that undermines security.
T.UAACTION	An undetected violation of the security policy may be caused as a result of an attacker (possibly, but not necessarily, an unauthorized user of the TOE) attempting to perform actions that the individual is not authorized to do.
T.UAUSER	An attacker (possibly, but not necessarily, an unauthorized user of the TOE) may impersonate an authorized user of the TOE. This includes the threat of an authorized user A that tries to impersonate as another authorized user without knowing the authentication information.
T.VIOS	A VIOS SCSI device driver acting on behalf of an LPAR partition may try to access logical volumes or physical volumes that are not assigned to device driver. A VIOS Ethernet device driver acting on behalf of a group of LPAR partitions may try to access a VIOS Ethernet adapter device driver intended for a different VIOS Ethernet device driver (or vice versa).
T.IP_PASSTHRU	A user may gain access to resources or perform operations on a system other than the TOE via the TOE’s network interface for which no access rights have been granted on the other system.

3.2.2 Threats to be countered by measures within the TOE environment

The following threats are to be countered by the TOE environment.

TE.COR_FILE	An attacker (possibly, but not necessarily, an unauthorized user of the TOE) or environmental conditions like a hardware malfunction may intentionally or accidentally modify or corrupt security enforcing or relevant files of the TOE without an administrative user being able to detect this. An attacker may corrupt such files either by having physical access to the hardware the TOE is running on, by booting other software than the TOE in its evaluated configuration or by modifying or corrupting files on backup media. Note that such a corruption may be caused accidentally without malicious intent by persons having legitimate access to media where such data is stored.
TE.HW_SEP	An attacker (possibly, but not necessarily, an unauthorized user of the TOE) with legitimate physical access to the hardware the TOE is running on or environmental conditions may cause the underlying hardware functions of the hardware the TOE is running on to not provide sufficient capabilities to support the self-protection of the TSF from unauthorized programs. Note that this also covers persons with legitimate access to the TOE hardware causing such a problem accidentally without malicious intent.
TE.HWMF	An attacker with legitimate physical access to the hardware of the TOE (examples are maintenance personnel or legitimate users) or environmental conditions may cause a hardware malfunction with the effect that a user (normal or administrative) is losing stored data due to this hardware malfunction. An attacker may cause such a hardware malfunction either by having physical access to the hardware the TOE is running on or by executing software that is capable of causing hardware malfunction. Note that such a hardware malfunction may be caused accidentally without malicious intent by persons having physical access to the TOE.
TE.KERB_BIND	An attacker (possibly, but not necessarily, an unauthorized user of the TOE) may attempt to guess the password of a Kerberos account through repeated bind attempts to Kerberos.
TE.LPAR	When running in a logical partition, software running in a different partition than the TOE is able to access resources that are assigned to the TOE (i.e. resources that belong to the partition the TOE is running in).

3.3 Organizational Security Policies

The TOE complies with the following organizational security policies:

P.ACCESS	Access rights to specific data objects are determined by the owner of the object, the role of the subject attempting access, and the implicit and explicit access rights to the object granted to the role by the object owner.
P.ACCOUNTABILITY	The users of the system shall be held accountable for their actions within the system.
P.AUTHORIZED_USERS	Only those users who have been authorized to access the information within the system may access the system.
P.CLASSIFICATION	**LSPP Mode Only**: Access to information is based on sensitivity, as represented by a label, associated with that information, and the formal sensitivity clearance of users, to access that information. The access rules enforced prevent a user from accessing information that is of higher sensitivity than the user's sensitivity clearance allows. Furthermore, unauthorized users can not downgrade information to a lower sensitivity.
P.COMPROT	When the TOE is used in a distributed environment, the administrator may create a trusted communications path between NFSv4 clients and servers and, for LDAP-based authentication, between the TOE and LDAP server.

P.DIST_USERS	When the TOE is used in a distributed environment, the administrators shall ensure that the user databases on each TOE are consistent with each other.
P.ERASE	Administrators shall be able to support information compartmentalization by preventing recovery of logically deleted information from physically and logically intact SCSI hard disk drives before they are re-used within the same system. Such hard disk drives will remain within the physical and logical protection domain of the TOE and will reside within the TSC.
P.INTEGRITY	**LSPP Mode Only**: The TOE shall be capable of distinguishing between levels of trustworthiness in terms of integrity, and the TOE shall prevent data from being modified by users operating at a lower level of trust.
P.NEED_TO_KNOW	The right to access specific data objects is determined on the basis of: • the owner of the object; and • the identity of the subject attempting the access; and • the implicit and explicit access rights to the object granted to the subject by the object owner.
P.STATIC	Dynamic partitioning (DLPAR) must not be used for the allocation and de-allocation of resources to the TOE's partition during operation of the TOE. Only "static" partitioning may be performed while the TOE is in a non-operating phase.
P.TCBINTEGRITY	The TOE shall be able to ensure that components of the TCB shall be modified only by authorized administrators. The TOE shall be able to check the integrity of the TCB.

3.4 Assumptions

This section indicates the minimum physical and procedural measures required to maintain security of the TOE.

3.4.1 Physical Assumptions

A.ASSET	It is assumed that the value of the stored assets merits moderately intensive penetration or masquerading attacks.
A.LOCATE	The processing resources of the TOE will be located within controlled access facilities which will prevent unauthorized physical access.
A.PROTECT	The TOE hardware and software critical to security policy enforcement will be protected from unauthorized physical modification.

3.4.2 Personnel Assumptions

A.ACCESS	Rights for users to gain access and perform operations on information are basd on their membership in one or more roles. These roles are granted to the users by the TOE Administrator. These roles accurately reflect the users job function, responsibilities, qualifications, and/or competencies within the enterprise.
A.COOP	Authorized users possess the necessary authorization to access at least some of the information managed by the TOE and are expected to act in a cooperating manner in a benign environment.
A.MANAGE	It is assumed that there are one or more competent individuals who are assigned to manage the TOE and the security of the information it contains.
A.NO_EVIL_ADM	The system administrative personnel are not careless, willfully negligent, or hostile, and will follow and abide by the instructions provided by the administrator documentation.

A.OWNER	A limited set of users is given the rights to "create new data objects" and they become owners for those data objects. The organization is the owner of the rest of the information under the control of TOE.
A.UTRAIN	Users are trained well enough to use the security functionality provided by the system appropriately.
A.UTRUST	Users are trusted to accomplish some task or group of tasks within a secure IT environment by exercising complete control over their data.

3.4.3 Connectivity Assumptions

A.CONNECT	All connections to peripheral devices and all network connections reside within the controlled access facilities. Internal communication paths to access points such as terminals or other systems are assumed to be protected against unauthorized access and the loss of integrity and confidentiality of the information transmitted.
A.KERB_KEY	The Kerberos KDC generates encryption keys used for encrypting the data communications between an NFSv4 client and server.
A.KERB_PROTECT	The Kerberos Key Distribution Center (KDC) used by the TOE provides protection mechanisms against unauthorized access to TSF data stored in the Kerberos database. This includes the requirement for user-subject binding when communicating to Kerberos and the use of the Kerberos protocol to protect the communication link between Kerberos and a Kerberos client.
A.LDAP_PROTECT	**CAPP Mode Only:** The LDAP server used by the TOE provides protection mechanisms against unauthorized access to TSF data stored in the LDAP server. This includes the requirement for authentication when accessing user entries and the configuration to use SSL v3 as the preferred protocol to protect the communication links.
A.NET_COMP	All network components (like bridges and routers) are assumed to correctly pass data without modification.
A.PEER	Any other systems with which the TOE communicates are assumed to be under the same management control and operate under the same security policy constraints. There are no security requirements which address the need to trust external systems or the communications links to such systems.
A.RSA_KEY	The environment generates RSA encryption keys used by the SSL communications.

3.4.4 Procedural Assumptions

A. CLEARANCE	**LSPP Mode Only**: Procedures exist for granting users authorization for access to specific security levels.
A. SENSITIVITY	**LSPP Mode Only**: Procedures exist for establishing the security level of all information imported into the system, for establishing the security level for all peripheral devices (e.g., printers, tape drives, disk drives) attached to the TOE, and marking a sensitivity label on all output generated.

4 Security Objectives

4.1 Security Objectives for the TOE

O.AUDITING	The TSF must record the security relevant actions of users of the TOE and security relevant events. The TSF must present this information to authorized administrators. The information recorded with security relevant events must be in sufficient detail to help an administrator of the TOE detect attempted security violations or potential misconfiguration of the TOE security features that would leave the IT assets open to compromise.
O.AUTHORIZATION	The TOE must ensure that only authorized users gain access to the TOE and its resources.
O.COMPROT	The TSF must be designed and implemented in a manner that allows for establishing a trusted channel between the TOE and another trusted IT product that protect the user data transferred over this channel from disclosure and undetected modification.
O.CONFINE	The TSF shall offer a mechanism to isolate and/or confine the execution of one or more processes to their own execution environment so that they cannot affect the security of other processes outside of this execution environment.
O.DISCRETIONARY_ACCESS	The TSF must control accessed to resources based on identity of users. The TSF must allow authorized users to specify which resources may be accessed by which users.
O.DUTY	The TOE must provide the capability of enforcing 'separation of duties', so that no single user has to be granted the right to perform all operations on important information.
O.ENFORCEMENT	The TSF must be designed and implemented in a manner which ensures that the organizational policies are enforced in the target environment The TOE security policy is enforced in a manner which ensures that the organizational policies are enforced in the target environment, i.e. the integrity of the TSF is protected.
O.ERASE	The TOE shall offer a mechanism to overwrite user-accessible blocks of SCSI hard disk drives with pre-defined bit patterns.
O.HIERARCHICAL	The TOE must allow hierarchical definitions of roles. Hierarchical definition of roles means the ability to define roles in terms of other roles. This saves time and allows for more convenient administration of the TOE.
O.IP_FILTER	The TOE shall mediate the flow of information between sets of TOE network interfaces or between a network interface and the TOE itself in accordance with its security policy.
O.MANAGE	The TSF must provide all the functions and facilities necessary to support the authorized administrators that are responsible for the management of TOE security and must ensure that only authorized administrators are able to access such functionality.
O.MANDATORY_ACCESS	**LSPP Mode Only**: The TSF must control access to resources based upon the sensitivity and categories of the information being accessed and the sensitivity clearance of the subject attempting to access that information.
O.MANDATORY_INTEGRITY	**LSPP Mode Only**: The TOE must control access to resources based on the integrity level of the information being accessed and the integrity level of the subject attempting to access that information.
O.NETWORK_ACCESS	**LSPP Mode Only**: The TOE must control access between the TOE and other systems based on host security attributes and the network interface on which packets are sent or received.

O.RESIDUAL_INFORMATION	The TOE must ensure that any information contained in a protected resource is not released when the resource is recycled.
O.ROLE	The TOE must prevent users from gaining access to and performing operations on its resources/objects unless they have been granted access by the resource/object owner or they have been assigned to a role (by an authorized administrator) which permits those operations.
O.STACK	The TOE shall offer a mechanism to prevent the execution of code on the stack of selected processes.
O.TCB_ACCESS	The TOE must control write and/or execute access to and provide integrity checks for resources protected as part of the trusted computing base as specified by an authorized administrator.
O.VIOS	The TSF must control access between LPAR partitions and logical/physical volumes and VIOS SCSI device drivers acting on behalf of a group of LPAR partitions. The TSF must allow authorized users to specify which logical/physical volumes may be accessed by the VIOS SCSI device drivers. The TSF must control access between VIOS Ethernet adapter device drivers and VIOS Ethernet device drivers acting on behalf of groups of LPAR partitions sharing a virtual network. The TSF must allow authorized users to specify which VIOS Ethernet adapter device drivers may be accessed by which VIOS Ethernet device driver acting on behalf of a group of LPAR partitions sharing a virtual network.

4.2 Security Objectives for the TOE Environment

All security requirements listed in this section are targeted at the TOE environment.

OE.ADMIN	Those responsible for the TOE are competent and trustworthy individuals, capable of managing the TOE and the security of the information it contains.
OE.CREDEN	Those responsible for the TOE must ensure that user authentication data is stored securely and not disclosed to unauthorized individuals. In particular: • Procedures must be established to ensure that user passwords generated by an administrator during user account creation or modification are distributed in a secure manner, as appropriate for the sensitivity clearance of the system. • The media on which authentication data is stored must not be physically removable from the system by unauthorized users. • Users must not disclose their passwords to other individuals.
OE.HW_SEP	The underlying hardware must provide separation mechanism that can be used by the TOE to protect the TSF and TSF data from unauthorized access and modification.
OE.INFO_PROTECT	Those responsible for the TOE must establish and implement procedures to ensure that information is protected in an appropriate manner. In particular: • All network and peripheral cabling must be approved for the transmittal of the most sensitive data held by the system. Such physical links are assumed to be adequately protected against threats to the confidentiality and integrity of the data transmitted. • DAC protections on security critical files (such as audit trails and authentication databases) shall always be set up correctly.

	• **LSPP Mode Only**: User sensitivity clearance and MAC protection must always be set up correctly by specifying appropriate security levels for all information imported into the system, peripheral devices and output generated and granting users authorization to access only specific security levels.
OE.INSTALL	Those responsible for the TOE must establish and implement procedures to ensure that the hardware, software and firmware components that comprise the system are distributed, installed and configured in a secure manner. This includes the configuration of logical partitions of the LPAR feature of the TOE environment.
OE.MAINTENANCE	Administrators of the TOE must ensure that the comprehensive diagnostics facilities provided by the product are invoked at every scheduled preventative maintenance period.
OE.PHYSICAL	Those responsible for the TOE must ensure that those parts of the TOE critical to security policy are protected from physical attack which might compromise IT security objectives.
OE.RECOVER	Those responsible for the TOE must ensure that procedures and/or mechanisms are provided to assure that, after system failure or other discontinuity, recovery without a protection (i.e., security) compromise is obtained.
OE.SERIAL_LOGIN	Those responsible for the TOE shall implement procedures to ensure that users clear the screen before logging off where serial login devices (e.g., IBM 3151 terminals) are used.
OE.SOFTWARE_IN	Those responsible for the TOE shall ensure that the system shall be configured so that only an administrator can introduce new trusted software into the system.

The following security objectives apply in environments where specific threats to networked systems need to be countered. (Either physical protection measures or cryptographic controls may be applied to achieve this objective, but they are **not** part of the TOE defined in this Security Target.)

OE.KERB_BIND	The Kerberos KDC must provide password complexity support and failed login attempt abatement for Kerberos accounts that meets or exceeds the password complexity requirements of the TOE.
OE.KERB_KEY	The Kerberos KDC must perform key generation for the trusted communications channel between the NFSv4 client and server.
OE.KERB_PROTECT	The Kerberos Key Distribution Center (KDC) supports the protection of TSF data from unauthorized access.
OE.LDAP_PROTECT	**CAPP Mode Only**: LDAP server must protect TSF data from unauthorized access.
OE.PROTECT	Those responsible for the TOE must ensure that procedures and/or mechanisms exist to ensure that data transferred between workstations is secured from disclosure, interruption or tampering.
OE.RSA_KEY	The environment must provide a mechanism to generate RSA keys for use by the TOE for SSL communications.

The following security objective applies when the TOE is running on underlying machines that have more than one logical partition configured:

OE.LPAR	The underlying hardware must protect the resources assigned to the logical partition the TOE is running in against access from software running in a different logical partition.

5 Security Requirements

Selection and assignment operations on IT security requirements are marked **bold**. Refinements are marked underlined. Iterations are identified by additional identifiers in parentheses as part of the component reference. All substitutions of security functional requirements to [CAPP], [LSPP], [RBACPP], and all additional security functional requirements beyond [CAPP], [LSPP], and [RBACPP] are described in section 7.2.

Application notes have been added as necessary. Please note that application notes are by nature informative and supposed to aid the reader’s comprehension of the TSP and their implementation by the TOE.

5.1 Extended Component Definitions

5.1.1 FDP_RIP.3-AIX

The Security Target defines an extended component FDP_RIP.3-AIX as part of the FDP_RIP family in CC Part 2 for usage within this ST.

Component leveling

FDP_RIP.3-AIX Hard disk drive residual information protection requires that the TSF ensure that any residual information content of a hard disk drive that is being formatted is made unavailable for logical recovery (“erased”) upon administrator-invoked de-allocation, or formatting, of the hard disk drive.

Management: FDP_RIP.3-AIX

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

a. The choice of when to erase a hard disk drive, and which hard disk drive, should be made configurable within in the TOE.

The choice of bit patterns used to overwrite the blocks of the hard disk drive, and how often to overwrite the blocks, could be made configurable within the TOE.

Audit: FDP_RIP.3-AIX

There are no events identified that should be auditable if FAU_GEN Security audit data generation is included in the ST.

FDP_RIP.3-AIX Hard disk drive residual information protection

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_RIP.3-AIX.1	The TSF shall overwrite all data stored in currently user-accessible blocks of a hard disk drive with pre-defined bit patterns upon request of the authorized administrator.

5.1.2 FPT_RVM.2-AIX

The Security Target defines an extended component FPT_RVM.2-AIX as part of the FPT_RVM family in CC Part 2 for usage within this ST.

Component leveling

FPT_RVM.2-AIX Stack execution reference mediation requires that the TSF ensure that code on the stack of selected processes cannot be executed on the stack.

Management: FPT_RVM.2-AIX

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

a. The choice of which processes are monitored and which are not monitored should be made configurable within the TOE.

Audit: FPT_RVM.2-AIX

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

a. Minimal: disabling of the monitoring of one or more processes.

FPT_RVM.2-AIX Stack execution reference mediation

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_RVM.2-AIX.1	The TSF shall provide the ability to allow/deny the execution of code residing on the stack of a process created from the executable to anyone who can write to the executable and allow an authorized administrator to override this setting.

5.2 TOE Security Functional Requirements

The following is a table identifying the SFRs for the TOE used in this ST, their origin, and the performed operations. An ‘X’ under CAPP mode and/or LSPP mode means that the mode supports the SFR. CAPP mode and LSPP mode apply to AIX only (i.e., VIOS does not have a CAPP mode or LSPP mode). ECD means Extended Component Definition. “CC Part 2” means that it’s from Common Criteria Part 2, not from a protection profile.

Table 4: SFRs for the TOE

SFR	Origin	VIOS	AIX	CAPP mode	LSPP mode
FAU_GEN.1	CAPP, LSPP, RBACPP		X	X	X
FAU_GEN.2	CAPP, LSPP, RBACPP		X	X	X
FAU_SAR.1	CAPP, LSPP, RBACPP		X	X	X
FAU_SAR.2	CAPP, LSPP, RBACPP		X	X	X
FAU_SAR.3	CAPP, LSPP, RBACPP		X	X	X
FAU_SEL.1	CAPP, LSPP, RBACPP		X	X	X
FAU_STG.1	CAPP, LSPP,RBACPP		X	X	X
FAU_STG.3	CAPP, LSPP		X	X	X
FAU_STG.4	CAPP, LSPP		X	X	X
FCS_CKM.1(1)	CC Part 2		X	X	
FCS_CKM.1(2)	CC Part 2		X	X	
FCS_CKM.1(3)	CC Part 2		X	X	X
FCS_CKM.2(1)	CC Part 2		X	X	X
FCS_CKM.2(2)	CC Part 2		X	X	X
FCS_COP.1(1)	CC Part 2		X	X	
FCS_COP.1(2)	CC Part 2		X	X	
FCS_COP.1(3)	CC Part 2		X	X	X
FCS_COP.1(4)	CC Part 2		X	X	X
FCS_COP.1(5)	CC Part 2		X	X	X
FCS_COP.1(6)	CC Part 2		X	X	X
FCS_COP.1(7)	CC Part 2		X	X	X
FCS_COP.1(8)	CC Part 2		X	X	X
FDP_ACC.1(1)	CAPP, LSPP		X	X	X
FDP_ACC.1(2)	CC Part 2		X		X
FDP_ACC.1(3)	CC Part 2		X	X	X
FDP_ACC.1(4)	RBACPP		X	X	X
FDP_ACC.1(5)	CC Part 2	X		▨	▨
FDP_ACF.1(1)	CAPP, LSPP		X	X	X
FDP_ACF.1(2)	CC Part 2		X		X
FDP_ACF.1(3)	CC Part 2		X	X	X
FDP_ACF.1(4)	RBACPP		X	X	X
FDP_ACF.1(5)	CC Part 2	X		▨	▨
FDP_ETC.1	LSPP		X		X
FDP_ETC.2	LSPP		X		X
FDP_IFC.1(1)	LSPP		X		X
FDP_IFC.1(2)	CC Part 2		X		X

SFR	Origin	VIOS	AIX	CAPP mode	LSPP mode
FDP_IFC.1(3)	CC Part 2		X		X
FDP_IFC.1(4)	CC Part 2		X	X	X
FDP_IFC.1(5)	CC Part 2		X	X	X
FDP_IFF.1(1)	CC Part 2		X	X	X
FDP_IFF.1(2)	CC Part 2		X	X	X
FDP_IFF.2(1)	LSPP		X		X
FDP_IFF.2(2)	CC Part 2		X		X
FDP_IFF.2(3)	CC Part 2		X		X
FDP_ITC.1	LSPP		X		X
FDP_ITC.2	LSPP		X		X
FDP_RIP.2	CAPP, LSPP		X	X	X
Note 1	CAPP, LSPP		X	X	X
FDP_RIP.3-AIX	ECD (section 5.1.1)		X	X	X
FIA_ATD.1(1)	CAPP, LSPP, RBACPP		X	X	X
FIA_ATD.1(2)	CC Part 2	X			
FIA_SOS.1	CAPP, LSPP	X	X	X	X
FIA_UAU.2	CAPP*, LSPP*, RBACPP	X	X	X	X
FIA_UAU.7	CAPP, LSPP	X	X	X	X
FIA_UID.2	CAPP*, LSPP*, RBACPP	X	X	X	X
FIA_USB.1(1)	CAPP, LSPP, RBACPP		X	X	X
FIA_USB.1(2)	CC Part 2	X			
FMT_MOF.1(1)	MLOSPP		X	X	X
FMT_MOF.1(2)	MLOSPP		X	X	X
FMT_MSA.1(1)	CAPP, LSPP		X	X	X
FMT_MSA.1(2)	CC Part 2		X		X
FMT_MSA.1(3)	CC Part 2		X	X	X
FMT_MSA.1(4)	CC Part 2		X		X
FMT_MSA.1(5)	CC Part 2		X	X	X
FMT_MSA.1(6)	RBACPP		X	X	X
FMT_MSA.1(7)	RBACPP		X	X	X
FMT_MSA.1(8)	RBACPP		X	X	X
FMT_MSA.1(9)	RBACPP		X	X	X
FMT_MSA.1(10)	CC Part 2	X			
FMT_MSA.1(11)	CC Part 2		X	X	X
FMT_MSA.1(12)	CC Part 2		X	X	X
FMT_MSA.2	RBACPP, CC Part 2		X	X	X
FMT_MSA.3(1)	CAPP, LSPP		X	X	X
FMT_MSA.3(2)	CC Part 2		X		X
FMT_MSA.3(3)	CC Part 2		X	X	X
FMT_MSA.3(4)	CC Part 2		X		X
FMT_MSA.3(5)	CC Part 2		X	X	X
FMT_MSA.3(6)	RBACPP		X	X	X
FMT_MSA.3(7)	CC Part 2	X			
FMT_MSA.3(8)	CC Part 2		X	X	X
FMT_MSA.3(9)	CC Part 2		X	X	X
FMT_MTD.1(1)	CAPP, LSPP		X	X	X
FMT_MTD.1(2)	CAPP, LSPP		X	X	X
FMT_MTD.1(3)	CC Part 2		X	X	X
FMT_MTD.1(4)	CAPP, LSPP	X	X	X	X
FMT_MTD.1(5)	CAPP, LSPP	X	X	X	X
FMT_MTD.1(6)	CC Part 2		X	X	X
FMT_MTD.1(7)	RBACPP		X	X	X
FMT_MTD.1(8)	CC Part 2	X			
FMT_MTD.1(9)	CC Part 2		X	X	X
FMT_MTD.1(10)	CC Part 2		X	X	X

SFR	Origin	VIOS	AIX	CAPP mode	LSPP mode
FMT_MTD.1(11)	CC Part 2		X	X	X
FMT_MTD.3	RBACPP		X	X	X
FMT_REV.1(1)	CAPP, LSPP, RBACPP		X	X	X
FMT_REV.1(2)	CAPP, LSPP, RBACPP		X	X	X
FMT_REV.1(3)	CC part 2	X			
FMT_SMF.1	CC Part 2	X	X	X	X
FMT_SMR.1	CC Part 2	X			
FMT_SMR.2	CAPP*, LSPP*, RBACPP		X	X	X
FPT_AMT.1	CAPP, LSPP, RBACPP		X	X	X
FPT_FLS.1	RBACPP		X	X	X
FPT_RCV.1	RBACPP		X	X	X
FPT_RCV.4	RBACPP		X	X	X
FPT_RVM.1	CAPP, LSPP, RBACPP		X	X	X
FPT_RVM.2-AIX	ECD (section 5.1.2)		X	X	X
FPT_SEP.1	CAPP, LSPP, RBACPP		X	X	X
FPT_STM.1	CAPP, LSPP, RBACPP		X	X	X
FPT_TDC.1	CC Part 2		X		X
FPT_TST.1	RBACPP		X	X	X
FTA_LSA.1	RBACPP		X	X	X
FTA_TSE.1	RBACPP		X	X	X
FTP_ITC.1	CC Part 2		X	X	X

* Hierarchically higher components than required by [CAPP] and [LSPP].

5.2.1 Security Audit (FAU)

5.2.1.1 Audit Data Generation (FAU_GEN.1)

The TSF shall be able to generate an audit record of the auditable events listed in column “Event” of Table 5 (Auditable Events). This includes **the start-up and shutdown of the audit functions and** all auditable events for the basic level of audit, except FIA_UID.2’s user identity during failures. **This also includes the:** FAU_GEN.1.1/NOTE 4

a. Assignment of Users, Roles and Privileges to Roles

b. Deletion of Users, Roles and Privileges from Roles

c. Creation and Deletion of Roles

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

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

b. LSPP Mode Only: The sensitivity labels of subjects, objects, or information involved; and

c. The additional information specified in the “Details” column of Table 5 (Auditable Events)

d. For each audit event type, based on the auditable event definitions of the functional components included in this ST the following information:

  i. For each invocation of a security function, the RBAC Administrator role that made invocation of that security function possible.

  ii. For each access control action on the user data, the role that made possible the invocation of that action.

Application Note for AIX: The execution of tests of the underlying machine and the results of the test are not stored in the normal audit trail but in a separate Diagnostic Error Log file. This file is protected by the MAC (LSPP mode only), MIC (LSPP mode only), and DAC such that only the System Administrator can access the file. When an installation requires auditing the use of the diag command, object auditing can be used where the diag command file is the object and execution is the access mode. In LSPP mode, in order to record/add an audit record to the audit log, a process needs the PV_AU_ADD or PV_AU privilege.

Table 5: Auditable Events

Component	**Event**	**Details (Event Names)**
FAU_GEN.1	Start-up and shutdown of the audit functions.	**start_up: AUD_It (cmd=1)** **shutdown: AUD_It (cmd=4)**
FAU_GEN.2	None.	
FAU_SAR.1	Reading of information from the audit records.	**See below**[1]
FAU_SAR.2	Unsuccessful attempts to read information from the audit records.	**FILE_Open**
FAU_SAR.3	None.	
FAU_SEL.1	All modifications to the audit configuration that occur while the audit collection functions are operating.	**AUD_Bin_Def, AUD_Events, AUD_Objects, AUD_Proc**
FAU_STG.1	None.	
FAU_STG.3	Actions taken due to exceeding of a threshold.	**AUD_Lost_Recs**
FAU_STG.4	Actions taken due to the audit storage failure.	**AUD_Lost_Recs**
FCS_CKM.1(1)	**None.**	
FCS_CMK.1(2)	**None.**	
FCS_CMK.1(3)	**None.**	
FCS_CKM.2(1)	**None.**	
FCS_CKM.2(2)	**None.**	
FCS_COP.1(1)	**None.**	
FCS_COP.1(2)	**None.**	
FCS_COP.1(3)	**None.**	
FCS_COP.1(4)	**None.**	
FCS_COP.1(5)	**None**	
FCS_COP.1(6)	**None**	
FCS_COP.1(7)	**None**	
FCS_COP.1(8)	**None**	
FDP_ACC.1(1)	None.	
FDP_ACC.1(2)	**None.**	
FDP_ACC.1(3)	**None.**	
FDP_ACC.1(4)	**None.**	
FDP_ACC.1(5)	**None.**	
FDP_ACF.1(1)	All requests to perform an operation on an object covered by the SFP.	**FILE_Mode** **FILE_Owner** **FILE_Chpriv** **FILE_Facl** **FILE_Fchmod** **FILE_Fchown** **FILE_Fchpriv** **FILE_Link** **FILE_Mknod**

[1] Object auditing can be used to define the events to be audited specifically for the audit files. This feature allows to define the whole set of events that should be audited for those files. Object auditing allows to define all actions on the audit file that should generate an audit record.

Component	Event	Details (Event Names)
		FILE_Open (for create) **FILE_Rename** **FILE_Truncate** **FILE_Unlink** **FS_Rmdir** **FS_Mount** **FS_Umount** **MSG_Owner** **MSG_Mode** **MSG_Delete** **MSG_Create** **PROC_Execute** **PROC_LPExecute** **SEM_Owner** **SEM_Create** **SEM_Delete** **SEM_Mode** **SHM_Create** **SHM_Delete** **SHM_LPCreate** **SHM_Owner** **SHM_Mode**
FDP_ACF.1(2)	**All requests to perform an operation on an object covered by the SFP.**	**MLS_GetSecconf** **MLS_SetSecconf** **MLS_SetTLIBMode** **MLS_SetPTLIBMode** **MLS_TLibPath** **RFM_Fsf** **RFM_ApdFsf**
FDP_ACF.1(3)	**All requests to perform an operation on an object covered by the SFP.**	**AUTH_Change** **AUTH_Create** **AUTH_Remove** **KST_Change** **SEC_ChkAuth** **SEC_ChkAuthId** **SEC_GetKat** **SEC_GetKatIds** **SEC_GetUserKat** **SEC_SetKst**
FDP_ACF.1(4)	**All requests to perform an operation on an object covered by the SFP.**	**ROLE_Change** **ROLE_Create** **ROLE_Remove** **SEC_GetRoleById** **SEC_GetRoleByName** **SEC_RoleNameToId** **SEC_RoleIdToName** **SEC_SetKst**
FDP_ACF.1(5)	**None.**	
FDP_ETC.1	All attempts to export information.	**BACKUP_Export** **ENQUE_admin** **ENQUE_exec** **TCP_kconnect** **TCP_ksend**
FDP_ETC.2	All attempts to export information.	**BACKUP_Export** **ENQUE_admin** **ENQUE_exec**

Component	Event	Details (Event Names)
		TCP_kconnect **TCP_ksend**
FDP_ETC.2	Overriding of human-readable output marking. (Additional)	**(not applicable)**
FDP_IFC.1(1)	None.	
FDP_IFC.1(2)	**None.**	
FDP_IFC.1(3)	**None.**	
FDP_IFC.1(4)	**None.**	
FDP_IFC.1(5)	**None.**	
FDP_IFF.1(1)	**All decisions on requests for information flow.**	**MLS_GetWAR** **MLS_SetWAR** **SEC_GetWps** **SEC_GetWpsCid** **SEC_SetWpsCid** **WM_CreateWPAR** **WM_ModifyConfig** **WM_ModifyWPAR** **WM_RemoveWPAR** **WM_ResetConfig** **WM_SetInitConf** **WM_StartWPAR** **WM_StopWPAR**
FDP_IFF.1(2)	**None.**	
FDP_IFF.2(1)	All decisions on requests for information flow.	**RFM_MacFs** **RFM_MacProc** **RFM_MacStr** **RFM_MacFifo**
FDP_IFF.2(2)	**All decisions on requests for information flow.**	**TCP_ksend**
FDP_IFF.2(3)	**All decisions on requests for information flow.**	**RFM_Mic**
FDP_ITC.1	All attempts to import user data, including any security attributes.	**RESTORE_Import** **TCP_kreceive**
FDP_ITC.2	All attempts to import user data, including any security attributes.	**RESTORE_Import** **TCP_kreceive**
FDP_RIP.2	None.	
Note 1	None.	
FDP_RIP.3-AIX	**None.**	
FIA_ATD.1(1)	None.	
FIA_ATD.1(2)	**None.**	
FIA_SOS.1	Rejection or acceptance by the TSF of any tested secret.	**USER_Login** **PASSWORD_Change** **USER_SU**
FIA_UAU.**2**	All use of the authentication mechanism.	**USER_Login** **PASSWORD_Change** **USER_SU**
FIA_UAU.7	None.	
FIA_UID.**2**	All use of the user identification mechanism, including the identity provided during successful attempts.	**USER_Login** **PASSWORD_Change** **USER_SU**
FIA_USB.1(1)	Success and failure of binding user security attributes to a subject (e.g.,success	**PROC_Execute** **PROC_LPExecute**

Component	**Event**	**Details** **(Event Names)**
	and failure to create a subject).	**PROC_RealUID** **PROC_AuditID** **PROC_SetUserIDs** **PROC_RealGID** **PROC_SetGroups** **PROC_Environ**
FIA_USB.1(2)	**None.**	
FMT_MOF.1(1)	**None.**	
FMT_MOF.1(2)	**None.**	
FMT_MSA.1(1) **through (9)**	All modifications of the values of security attributes.	**PROC_Environ** **PROC_Privilege** **PROC_Execute** **PROC_LPExecute** **PROC_RealUID** **PROC_AuditID** **PROC_SetUserIDs** **PROC_RealGID** **PROC_SetGroups** **AUTH_Change** **AUTH_Create** **AUTH_Remove** **KST_Change** **ROLE_Change** **ROLE_Create** **ROLE_Remove** **MLS_GetSecconf** **MLS_SetSecconf** **MLS_SetPTLIBMode** **MLS_SetTLIBMode** **MLS_TLibPath** **RFM_ApdFsf** **RFM_Fsf** **SEC_ChkAuth** **SEC_ChkAuthId** **SEC_GetKat** **SEC_GetKatIds** **SEC_GetRoleById** **SEC_GetRoleByName** **SEC_GetUserKat** **SEC_RoleIdToName** **SEC_RoleNameToId** **SEC_SetKst**
FMT_MSA.1(10) **through (11)**	**All modifications of the values of security attributes.**	**SEC_SetWpsCid**
FMT_MSA.1(12)	**None.**	
FMT_MSA.2	**None.**	
FMT_MSA.3(1) **through (6)**	Modifications of the default setting of permissive or restrictive rules. All modifications of the initial value of security attributes.	**Administrator-defined object write auditing events for /etc/security/user, /etc/security/rules.host and /etc/security/rules.int**
FMT_MSA.3(7) **through (9)**	**None.**	
FMT_MTD.1(1)	All modifications to the values of TSF	**Object write auditing**

Component	Event	Details (Event Names)
	data.	**events for the TSF files containing the TSF data**
FMT_MTD.1(2)	All modifications to the values of TSF data.	**PROC_Sysconfig AUD_it AUD_Bin_Def**
FMT_MTD.1(3)	**All modifications to the values of TSF data.**	**Object write auditing events for the /etc/security/audit/config file,**
FMT_MTD.1(4)	All modifications to the values of TSF data.	**USER_Change**
FMT_MTD.1(5)	All modifications to the values of TSF data.	**USER_Change PASSWORD_Change PASSWORD_Flags**
FMT_MTD.1(6)	**All modifications to the values of TSF data.**	**CMD_Change CMD_Remove SEC_SetKst**
FMT_MTD.1(7)	**None.**	
FMT_MTD.1(8)	**None.**	
FMT_MTD.1(9)	**None.**	
FMT_MTD.1(10)	**None.**	
FMT_MTD.1(11)	**None.**	
FMT_MTD.3	**None.**	
FMT_REV.1(1)	All attempts to revoke security attributes.	**USER_Change**
FMT_REV.1(2)	All modifications to the values of TSF data.	**FILE_Acl**
FMT_REV.1(3)	**None.**	
FMT_SMF.1	**None.**	
FMT_SMR.1	**None.**	
FMT_SMR.**2**	Modifications to the group of users that are part of a role. Every use of the rights of a role. (Additional / Detailed)	**USER_Change PROC_Privilege AUTH_Change AUTH_Create AUTH_Remove CMD_Change CMD_Remove DEV_Change DEV_Remove RFM_ChkPriv RFM_SetPPriv ROLE_Create ROLE_Change ROLE_Remove SEC_SetKst**
FPT_AMT.1	Execution of the tests of the underlying machine and the results of the test.	**Diagnostic Error Log or object auditing (see Application Note)**
FPT_FLS.1	**None.**	
FPT_RCV.1	**None.**	
FPT_RCV.4	**None.**	
FPT_RVM.1	None.	
FPT_RVM.2-AIX	**Disabling of the stack execution**	**SEDMGR_File,**

Component	Event	Details (Event Names)
	detection.	**SEDMGR_Odm**
FPT_SEP.1	None.	
FPT_STM.1	Changes to the time.	**PROC_Adjtime**
FPT_TDC.1	**None.**	
FPT_TST.1	**None.**	
FTA_LSA.1	**None.**	
FTA_TSE.1	**None.**	
FTP_ITC.1	**None.**	

5.2.1.2 User Identity Association (FAU_GEN.2)

The TSF shall be able to associate each auditable event with the identity of the user that caused the event. FAU_GEN.2.1

Application Note for AIX: AIX stores the identity of the user in the header field "ah_ruid" and "ah_luid" of each audit record. For a description of the difference between the "real user ID" and the "login user ID" see chapter 6.

5.2.1.3 Audit Review (FAU_SAR.1)

The TSF shall provide authorized administrator roles with the capability to read all audit information from the audit records including: FAU_SAR.1.1

a. Date and Time of Audit Event

b. The UserID responsible for the Event and optionally the role membership which enabled the user to perform the event successfully

c. The access control operation and the object on which it was performed.

d. The outcome of the event (success or failure)

e. The User Session Identifier or Terminal Type

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

Application Note for AIX: The access control to audit files within AIX is regulated by the MAC (LSPP mode only), MIC (LSPP mode only), and DAC of AIX, as well as the authorization subsystem. Furthermore, in LSPP mode in order to read files marked as audit files, a process needs the PV_AU_READ or PV_AU privilege. It is the task of the administrator to ensure that the audit files as well as the audit configuration files are protected appropriately. Tools are provided to the administrator to read and format the audit records. For a more detailed description see chapter 6.

5.2.1.4 Restricted Audit Review (FAU_SAR.2)

The TSF shall prohibit all users read access to the audit records, except those users that have been granted explicit read-access. FAU_SAR.2.1

Application Note for AIX: This requirement is satisfied by the access control facility of AIX. It is the task of the authorized administrators to manage the read access right to the audit files appropriately. See Application Note for FAU_SAR.1.

5.2.1.5 Selectable Audit Review (FAU_SAR.3)

The TSF shall provide the ability to perform **searches, sorting and ordering** of audit data based on the following attributes: FAU_SAR.3.1

a. User identity;

b. <u>LSPP Mode Only: </u>Subject sensitivity label;

c. <u>LSPP Mode Only: </u>Object sensitivity label;

d. **LSPP Mode Only: MAC success or failures;**

e. **LSPP Mode Only: MIC success or failures;**

f. **DAC success or failures;**

g. **PRIV success or failures;**

h. **FSF success or failures;**

i. **AUTH success;**

j. **Audit event;**

k. **Event status;**

l. **Command name;**

m. **Process ID;**

n. **ID of the parent process;**

o. **Kernel thread ID;**

p. **Name of the host that generated the audit event;**

q. Date and Time of Audit event;

r. Object name & type of access;

s. Role that enabled the access;

t. Any combination of the above items (a), (q), (r), or (s).

Application Note for AIX: AIX provides two commands for audit data processing: *auditpr* and *auditselect*. Details are described in chapter 6.

5.2.1.6 Selective Audit (FAU_SEL.1)

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

a. User identity;

b. LSPP Mode Only: Subject sensitivity label;

c. LSPP Mode Only: Object sensitivity label;

d. Object identity;

e. Subject identity (process ID);

f. Host identity;

g. Event type;

h. Users belonging to a specified Role;

i. Access types (e.g. delete, insert) on a particular object;

j. **File name**

Application Note for AIX: The main configuration data for the audit is stored in the file */etc/security/audit/config*. This file together with */etc/security/user* or equivalent user database allows the administrator to include or exclude auditable events based on the identity of the user. In addition, the audit configuration file */etc/security/audit/objects* allows to include or exclude auditable events based on the file name. In order to modify the audit system configurations of the TOE in LSPP mode, a process needs the PV_AU_ADMIN or PV_AU privilege.

5.2.1.7 Guarantees of Audit Data Availability (FAU_STG.1)

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

The TSF shall be able to prevent modifications to the audit records. FAU_STG.1.2

Application Note for AIX: Protection of stored audit records from deletion and modifications is performed using the access control mechanisms of AIX. In LSPP mode in order to allow a process to write or delete a file marked as an audit file, or to mark a file as an audit file, this process needs the PV_AU_WRITE or PV_AU privilege.

5.2.1.8 Action in Case of Possible Audit Data Loss (FAU_STG.3)

The TSF shall generate an alarm to the authorized administrator if the file system holding the audit trail falls below **a configurable limit of free blocks**. FAU_STG.3.1 / NOTE 3

Application Note for AIX: AIX implements a configurable variable for this.

5.2.1.9 Prevention of Audit Data Loss (FAU_STG.4)

The TSF shall be able to prevent auditable events, except those taken by the authorized administrator, and **either stop the system in panic mode or count the number of audit records lost** if the audit trail is full. FAU_STG.4.1 / NOTE 5

Application Note for AIX: In the case all audit bins are full, AIX can be configured to stop execution (panic) or count the number of audit records lost. Normal execution can only be resumed after space for the audit bin is available. This has to be achieved by an authorized administrator, that starts the system in single-user mode and perform the necessary actions to make disk space available for auditing.

5.2.2 Cryptographic Support (FCS)

5.2.2.1 Cryptographic Key Generation (SSL: Symmetric Algorithms) (FCS_CKM.1(1))

The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **as defined in the SSL v3 standard** and specified cryptographic key sizes **128 bit (RC4), 168 bit (TDEA), 128 bit (AES), 256 bit (AES)** that meet the following: FCS_CKM.1.1

- **generation of session keys as defined in the SSL v3 standard with the cipher suites defined in FCS_COP.1(1).**

Application Note for AIX: Generation of symmetric keys is defined in section 6.2 of the SSL v3 standard. The library used by the TOE also supports SSL v2, but SSL v2 is not supported as part of the evaluated configuration. The evaluation will assess that the keys are generated in accordance with the requirements defined in the SSL v3 standard. With respect to the strength of function, no assessment of the strength of the cryptographic algorithm itself and no analysis for potential weaknesses of keys with respect to the algorithm are performed.

5.2.2.2 Cryptographic Key Generation (CLiC: Symmetric Algorithms) (FCS_CKM.1(2))

The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **none** and specified cryptographic key sizes **128 bit, 192 bit, and 256 bit** that meet the following: FCS_CKM.1.1

- **compliant with FIPS 186-2 [FIPS186-2] random number generation.**

Application Note for AIX: Generation of symmetric keys by CLiC for the use with EFS.

5.2.2.3 Cryptographic Key Generation (CLiC: Asymmetric Algorithms) (FCS_CKM.1(3))

The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **RSA** and specified cryptographic key sizes **1024 bit, 2048 bit, and 4096 bit** that meet the following: FCS_CKM.1.1

- **compliant with ANSI X9.31 [ANSI-X9.31] key generation.**

Application Note for AIX: Generation of asymmetric keys by CLiC for the use with EFS.

5.2.2.4 Cryptographic Key Distribution (SSL: Symmetric Keys) (FCS_CKM.2(1))

The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **Secure Socket Layer handshake using RSA encrypted exchange of session keys** that meets the following: FCS_CKM.2.1

- **SSL Version 3 [SSLv3]**
- **TLS Version 1.0 [RFC3268] (for SSL support of AES)**

Application Note for AIX: This requirement addresses the exchange of SSL session keys as part of the SSL handshake protocols.

5.2.2.5 Cryptographic Key Distribution (Kerberos) (FCS_CKM.2(2))

The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **Kerberos Version 5** that meets the following: FCS_CKM.2.1

- **conformant to RFC 4120 [RFC4120]**

Application Note for AIX: This requirement addresses the keys distributed by Kerberos and used by NFSv4 client/server communication.

5.2.2.6 Cryptographic Operation (SSL: Symmetric Operations) (FCS_COP.1(1))

The TSF shall perform **encryption and decryption** in accordance with a specified cryptographic algorithm **RC4, DES, TDEA, and AES** and cryptographic key sizes **128 bit (RC4), 168 bit (TDEA), 128 bit (AES), and 256 bit (AES)** that meet the following: FCS_COP.1.1

- **SSL Version 3 [SSLv3] and the following cipher suites as defined in the SSL v3 standard:**
    - **SSL_RSA_WITH_RC4_128_MD5**
    - **SSL_RSA_WITH_RC4_128_SHA**
    - **SSL_RSA_WITH_3DES_EDE_CBC_SHA**
- **SSL Version 3 and the following cipher suites as defined in [RFC3268]:**
    - **TLS_RSA_WITH_AES_128_CBC_SHA**
    - **TLS_RSA_WITH_AES_256_CBC_SHA**

Application Note for AIX: This requirement applies to the LDAP communications connection which uses the IBM Global Security Kit (GSKit) in the LDAP client.

5.2.2.7 Cryptographic Operation (SSL: RSA) (FCS_COP.1(2))

The TSF shall perform **digital signature generation and digital signature verification** in accordance with a specified cryptographic algorithm **RSA** and cryptographic key sizes **1024 bit** that meet the following: FCS_COP.1.1

- **SSL Version 3 [SSLv3]**

Application Note for AIX: This requirement addresses the RSA digital signature generation and verification operations using the RSA algorithm as required by the SSL session establishment protocols (provided a cipher suite including RSA is used). Note that the details of the signature format such as the use of the PKCS#1 block type 1 and block type 2 are defined in the SSL Version 3. This requirement applies to the LDAP communications connection which uses GSKit.

5.2.2.8 Cryptographic Operation (NFSv4) (FCS_COP.1(3))

The TSF shall perform

- **symmetric encryption, symmetric decryption, digest generation, and digest verification**

in accordance with a specified cryptographic algorithm

- **AES for symmetric encryption and symmetric decryption using CTS mode**
- **TDEA for symmetric encryption and symmetric decryption with 3 independent keys (CBC mode)**

- **SHA-1 for digest generation and digest verification**

and cryptographic key sizes

- **128 bit (AES)**
- **256 bit (AES)**
- **168 bit (TDEA)**
- **none for SHA-1**

that meet the following: FCS_COP.1.1

- **conformant to FIPS 197 [FIPS197] Advanced Encryption Standard (AES) and conformant to RFC 4120 [RFC4120] for CTS mode**
- **conformant to FIPS 46-3 [FIPS46-3] (TDEA)**
- **conformant to the Secure Hash Standard (SHS) as defined in FIPS 180-2 [FIPS180-2] (SHA-1)**

Application Note for AIX: This requirement is for AES, TDEA, and SHA-1 used in the NFSv4 client/server communications. NFSv4 uses the CLiC library to perform these cryptographic operations.

5.2.2.9 Cryptographic Operation (Kerberos) (FCS_COP.1(4))

The TSF shall perform **symmetric encryption and symmetric decryption** in accordance with a specified cryptographic algorithm

- **AES**
- **TDEA with 3 independent keys (CBC mode)**

and cryptographic key sizes

- **128 bit (AES)**
- **256 bit (AES)**
- **168 bit (TDEA)**

that meet the following: FCS_COP.1.1

- **conformant to Kerberos Version 5 standard [RFC4120].**

Application Note for AIX: This requirement is for AES and TDEA used when communicating with Kerberos (e.g. during Kerberos authentication and other Kerberos communication tasks). AIX uses the Kerberos encryption library for these cryptographic operations.

5.2.2.10 Cryptographic Operation (CLiC: Symmetric) (FCS_COP.1(5))

The TSF shall perform **symmetric encryption and decryption** in accordance with a specified cryptographic algorithm**:** FCS_COP.1.1

- **AES (CBC and ECB modes)**

and cryptographic key sizes

- **128 bit**
- **192 bit**
- **256 bit**

that meet the following:

- **FIPS 197 [FIPS197] Advanced Encryption Standard (AES).**

Application Note for AIX: This requirement applies to the CLiC cryptographic library when used by EFS.

5.2.2.11 Cryptographic Operation (CLiC: Asymmetric) (FCS_COP.1(6))

The TSF shall perform **asymmetric encryption and decryption** in accordance with a specified cryptographic algorithm **RSA** and cryptographic key sizes **1024 bit, 2048 bit, and 4096 bit** that meet the following: FCS_COP.1.1

- **compliant with ANSI X9.31 [ANSI-X9.31].**

Application Note for AIX: This requirement applies to the CLiC cryptographic library when used by Encrypted File System (EFS) and File Integrity Verification (FIV).

5.2.2.12 Cryptographic Operation (CLiC: Digest) (FCS_COP.1(7))

The TSF shall perform **digest generation and verification** in accordance with a specified cryptographic algorithm**:** FCS_COP.1.1

- **SHA-1**
- **SHA-256**
- **SHA-512**

and cryptographic key sizes **none** that meet the following:

- **compliant with the Secure Hash Standard (SHS) as defined in FIPS 180-2 [FIPS180-2].**

Application Note for AIX: This requirement applies to the CLiC cryptographic library when used by EFS and FIV.

5.2.2.13 Cryptographic Operation (CLiC: RNG) (FCS_COP.1(8))

The TSF shall perform **random number generation** in accordance with a specified cryptographic algorithm **none** and cryptographic key sizes **none** that meet the following: FCS_COP.1.1

- **compliant with FIPS 186-2 [FIPS186-2] random number generation.**

Application Note for AIX: This requirement applies to the CLiC cryptographic library when used by EFS.

5.2.3 User Data Protection (FDP)

5.2.3.1 Discretionary Access Control Policy (FDP_ACC.1(1))

The TSF shall enforce the Discretionary Access Control (DAC) Policy on **processes** acting on the behalf of users **as subjects and file system objects (ordinary files, directories, device special files, UNIX Domain socket special files, named pipes), IPC objects (message queues, SysV semaphores, shared memory segments) and TCP ports as objects (CAPP mode only)** and all operations among subjects and objects covered by the DAC policy. FDP_ACC.1.1

Application Note for AIX: See chapter 6 for details of the Discretionary Access Control capabilities for the different types of subjects.

5.2.3.2 TCB Access Control Policy (FDP_ACC.1(2))

The TSF shall enforce the **TCB Policy** on **processes acting on the behalf of users as subjects and jfs2 file system objects (ordinary files, directories, device special files, UNIX Domain socket special files, named pipes) and all operations among subjects and objects covered by the TCB Policy**. FDP_ACC.1.1

5.2.3.3 Authorization Policy (FDP_ACC.1(3))

The TSF shall enforce the **Authorization Policy** on **processes acting on the behalf of users as subjects, functions implemented in executable files as objects, and the attempts of processes to invoke such functions as operations**. FDP_ACC.1.1

5.2.3.4 Role-Based Access Control Policy (FDP_ACC.1(4))

The TSF shall enforce the Role-Based Access Control (RBAC) Policy on **processes acting on behalf of users as subjects and file system objects (ordinary files, directories, device special files, UNIX Domain socket special files,**

named pipes), IPC objects (message queues, SysV semaphores, shared memory segments) and TCP ports as objects and all operations among subjects and objects covered by the RBAC policy. FDP_ACC.1.1

5.2.3.5 VIOS Access Control Policy (FDP_ACC.1(5))

The TSF shall enforce the **VIOS Access Control Policy** on**:** FDP_ACC.1.1

- **Volumes: VIOS SCSI device drivers acting on behalf of LPAR partitions as subjects with Logical Volumes and Physical Volumes as objects and the operations among subjects and objects as covered by the policy**
- **Network: VIOS Ethernet device drivers acting on behalf of a group of LPAR partitions sharing a virtual network and VIOS Ethernet adapter device drivers (where either one can be the subject and the other the object) and the operations among subjects and objects as covered by the policy**.

Application Note for VIOS: This requirement is taken directly from CC part 2, not from any PP requirements; thus, PP audit requirements do not apply.

5.2.3.6 Discretionary Access Control Functions (FDP_ACF.1(1))

The TSF shall enforce the Discretionary Access Control Policy to objects based on the following: FDP_ACF.1.1

a. The user identity and group membership(s) associated with a subject; and

b. The following access control attributes associated with an object:

File system objects:

AIXC policy:

permission bits, extended permissions, privilege sets, and encryption attributes. (Permission bits are the standard UNIX permission bits for user, group, world. Extended permissions can be used to grant or deny access to the granularity of a single user or group using Access Control Lists).

Access rights for file system objects are:
- read
- write
- execute (ordinary files)
- search (directories)

NFSv4 policy:

permission bits or fine grained permissions. (NFSv4 policy where the access rights listed below apply to the following entities: owner, group, everyone. The access rights can be used to allow or deny access to the granularity of a single entity.)

Access rights for system objects are:

- read data (ordinary files)

- list contents (directories)

- write file data (ordinary files)

- add a file (directories)

- append data (ordinary files)

- add subdirectory (directories)

- read extended attributes

- write extended attributes

- execute (ordinary files)

- search (directories)

- delete an object within a directory

- delete the associated object

- read core object attributes (size, time, etc.)

- write core object attributes

- read ACL contents

- write ACL contents

- change ownership (user or group)

- synchronize

IPC objects:

permission bits

Access rights for IPC objects are:
- read
- write

TCP ports (CAPP mode only):

Access control lists with entries of the following form:

user@host

user@subnet

group@host

group@subnet

The only access right is the right to set up a connection on the specified port

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: FDP_ACF.1.2

File system objects:

AIXC Policy:

A subject must have search permission for every element of the pathname and the requested access for the object. A subject has a specific type access to an object if the type of access is within the union of all permission rights (grant entries) defined in the access control list of the object for the subject and is not within the logical union of all restrictions (deny entries) defined in the access control list of the object for the subject. If no entry in the extended permissions either allows or denies access, the access right defined in the permission bits apply. In any other case access is denied.

NFSv4 Policy:

A subject must have search permission for every element of the pathname and the requested access for the object. A subject has the requested type access to an object if all requested access types are specifically allowed before reaching an entry that denies one or more requested types or before reaching the end of the ACL. Otherwise, the requested access is denied.

CAPP Mode Only: A subject with an effective UID other than 0 and with WRITE_OWNER access specified in the ACL can change the object owner to himself, otherwise the request is denied.

IPC objects:

Access permissions are defined by permission bits of the IPC object. The process creating the object defines the creator, owner and group based on the user ID of the current process. Access of a process to an IPC object is allowed, if

the user ID of the of the current process is equal to the user ID of the IPC object creator or owner and the “owner” permission bit for the requested type of access is set or

the group ID of the current process is equal to the group ID of the IPC object and the “group” permission bit for the requested type of access is set or

The "world" permission bit for the requested type of access is set

TCP ports (CAPP mode only):

Setting up a connection from another system to a TCP port on a given system can be regulated by access control lists. A connection can only be established to a TCP port if the connection is coming from a user and a host where the ACL for the TCP port has entry either of the form user@host, group@host, user@subnet or group@subnet matches the userid or groupid and the host or subnet.

In addition ports with numbers larger than 1024 can be turned into privileged ports, i. e. a local user that wants to start a server process listening on such a port must have root privileges.

The TSF shall explicitly authorize access of subjects to objects based in the following additional rules: FDP_ACF.1.3

Privilege sets override DAC decisions in order to allow a process to:

1. **Change its GID (PV_DAC_GID);**
2. **Bypass DAC ownership restrictions except on IPC objects (PV_DAC_O);**
3. **Bypass DAC read restrictions except on IPC objects (PV_DAC_R);**
4. **Change its role ID (PV_DAC_RID);**
5. **Change its UID (PV_DAC_UID);**
6. **Bypass DAC write restrictions except on IPC objects (PV_DAC_W);**
7. **Bypass DAC execute/search restrictions (PV_DAC_X);**
8. **Obtain all of the privileges listed in 1. through 7. (PV_DAC);**
9. **Bypass DAC ownership restrictions on IPC objects (PV_KER_IPC_O);**
10. **Bypass DAC read restrictions on IPC objects (PV_KER_IPC_R);**
11. **Bypass DAC write restrictions on IPC objects (PV_KER_IPC_W).**

File System Objects:

The process has the appropriate privileges to perform the specific access request. In addition for the NFSv4 policy, the object owner is always allowed to read/write the ACL contents and read/write the core object attributes.

The attributes for a privileged command (an aspect of RBAC) defined in the Privileged Commands database (privcmds) override the file system DAC attributes.

CAPP Mode Only: A process with a user ID of 0 is known as a root user process. These processes are generally allowed all access permissions. But if a root user process requests execute permission for a program (as a file system object), access is granted only if execute permission is granted to at least one user. In addition for NFSv4 policy, the object owner is always allowed to read/write the ACL contents and read/write the core object attributes.

The TSF shall explicitly deny access of subjects to objects based on the**:** FDP_ACF.1.4

a. **write requests to objects that reside on a file system that is mounted read-only**
b. **inability of a subject to decrypt a file encrypted by the EFS file system**

Application Note for AIX: The Discretionary Access Control mechanism is explained in more detail in chapter 6. The details of the handling of discretionary access control for the different types of objects are explained there.

5.2.3.7 TCB Access Control Functions (FDP_ACF.1(2))

The TSF shall enforce the **Trusted Computing Base (TCB) Policy** to objects based on the following: **the trustedlib_enabled kernel security flag, TCB flag attribute for file system objects and the PV_TCB privilege for processes**. FDP_ACF.1.1

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: FDP_ACF.1.2

a) **If kernel security flag trustedlib_enabled is set to *enabled* and the system is in operational mode**

A process may not write to a file with the FSF_TLIB flag set.

If the FSF_TLIB_PROC flag is set for an executable, a corresponding process may only load shared libraries that have the FSF_TLIB flag set.

A process may not set or clear the FSF_TLIB or FSF_TLIB_PROC flag on an object.

b) **If kernel security flag trustedlib_enabled is set to *enabled* and the system is in maintenance mode**

A process with the PV_TCB privilege will ignore the file's FSF_TLIB or FSF_TLIB_PROC flag when attempting to access files.

A process with the PV_TCB privilege may set or clear the FSF_TLIB or FSF_TLIB_PROC flag on an object.

The TSF shall explicitly authorize access of subject to objects based on the following additional rules: **none**. FDP_ACF.1.3

The TSF shall explicitly deny access of subject to objects based on **no additional rules**. FDP_ACF.1.4

Application Note for AIX: As spelled out in section 6.2.14.1, the evaluated configuration mandates that *trustedlib_enabled* be enabled, therefore no policy for the TCB being disabled is specified here.

5.2.3.8 Authorization Functions (FDP_ACF.1(3))

The TSF shall enforce the **Authorization Policy** to objects based on the following: **the user identity, authorizations as the attributes that are assigned to users, and TOE-defined functions associated with the privilege**. FDP_ACF.1.1

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: FDP_ACF.1.2

If a function within an executable requires a specific authorization, the function will only be executed if the executing user has been assigned the proper authorization.

The TSF shall explicitly authorize access of subject to objects based on the following additional rules: FDP_ACF.1.3

Privilege sets override Authorization decisions in order to allow a process to

1. **Modify the kernel security table (PV_AZ_ADMIN);**
2. **Bypass all authorization checks (PV_AZ_CHECK);**
3. **Retrieve the entire kernel security table (PV_AZ_READ);**
4. **Pass all authorization checks during exec() (PV_AZ_ROOT).**

The TSF shall explicitly deny access of subject to objects based on **no additional rules**. FDP_ACF.1.4

5.2.3.9 Role-Based Access Control Functions (FDP_ACF.1(4))

The TSF shall enforce the RBAC Policy to objects based on the following user attributes: FDP_ACF.1.1

a. User Identity
b. Authorized Roles for the User

The TSF shall enforce the RBAC Policy to objects based on the following subject attributes:

a. Subject Identity
b. Role(s) which can invoke the subject

The TSF shall enforce the RBAC Policy to objects based on the following object attributes:

a. Object Identity
b. Operations permitted on the objects for various Roles

The TSF shall enforce the following rules to determine if any operation among controlled subjects and controlled objects is allowed: FDP_ACF.1.2

- The subject invoking the operation on an object is assigned to a role whose privilege set includes the operation on the object.

The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: FDP_ACF.1.3

- Allow an access operation by a subject on an object only if the user associated with the subject belongs to a role that permits the access operation on the object.

The TSF shall explicitly deny access of subjects to objects based on the user associated with the subject not belonging to any role that permits the requested access operation on the object. FDP_ACF.1.4

5.2.3.10 VIOS Access Control Functions (FDP_ACF.1(5))

The TSF shall enforce the **VIOS Access Control Policy** to objects based on the following: FDP_ACF.1.1

- **Volumes: A logical volume or physical volume (object) can only be mapped to (accessed by) one VIOS SCSI device driver acting on behalf of an LPAR partition (subject) and this mapping is the access control rule**
- **Network: A VIOS Ethernet device driver acting on behalf of a group of LPAR partitions sharing a virtual network and a VIOS Ethernet adapter device driver (where either one can be the subject and the other the object) can only be mapped to each other in a one-to-one mapping and this mapping is the access control rule**.

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: FDP_ACF.1.2

- **Volumes: If the logical volume or physical volume is mapped to a VIOS SCSI device driver acting on behalf of an LPAR partition, then the device driver can access the logical volume or physical volume, respectively; otherwise, access is denied**
- **Network: If a VIOS Ethernet device driver acting on behalf of a group of LPAR partitions sharing a virtual network is mapped to a VIOS Ethernet adapter device driver, then the device drivers can exchange untagged packets; otherwise, access is denied**.

The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: FDP_ACF.1.3

- **none**

The TSF shall explicitly deny access of subjects to objects based on **no additional rules**. FDP_ACF.1.4

Application Note for VIOS: This requirement is taken directly from CC part 2, not from any PP requirements; thus, the PP audit requirements do not apply. An untagged packet is a packet that does not contain a VLAN (Virtual LAN) identifier/tag.

5.2.3.11 Export of Unlabeled User Data (FDP_ETC.1)

The TSF shall enforce the Mandatory Access Control (MAC) Policy when exporting unlabeled user data, controlled under the MAC policy, outside the TSC. FDP_ETC.1.1

The TSF shall export the unlabeled user data without the user data's associated security attributes. FDP_ETC.1.2

The TSF shall enforce the following rules when unlabeled user data is exported from the TSC: NOTE 6

a) Devices used to export data without security attributes cannot be used to export data with security attributes unless the change in device state is performed manually and is auditable;

b) the subject exporting data on the behalf of a user must have appropriate privileges as specified in FDP_IFF.2(1) and FDP_IFF.2(2).

5.2.3.12 Export of Labeled User Data (FDP_ETC.2)

The TSF shall enforce the Mandatory Access Control Policy when exporting labeled user data, controlled under the MAC policy, outside the TSC. FDP_ETC.2.1

The TSF shall export the labeled user data with the user data's associated security attributes. FDP_ETC.2.2

The TSF shall ensure that the security attributes, when exported outside the TSC, are unambiguously associated with the exported labeled user data. FDP_ETC.2.3

The TSF shall enforce the following rules when labeled user data is exported from the TSC: FDP_ETC.2.4

a) When data is exported in a human-readable or printable form:

- The authorized administrator shall be able to specify the printable label, which is assigned to the sensitivity label associated with the data.
- Each print job shall be marked at the beginning and end with the printable label assigned to the "least upper bound" sensitivity label of all the data exported in the print job.
- Each page of printed output shall be marked with the printable label assigned to the "least upper bound" sensitivity label of all the data exported to the page. By default this marking shall appear on both the top and bottom of each printed page.

b) Devices used to export data with security attributes cannot be used to export data without security attributes unless the change in device state is performed manually and is auditable;

c) Devices used to export data with security attributes shall completely and unambiguously associate the security attributes with the corresponding data;

d) No additional labeled data export rules.

Application Note for AIX: When data is exported via FTP, the label of the user initiating the transaction is associated with the FTP transmission. When data is exported via a printer, the backend filtering mechanism of the print system implements the required labeling functionality for single-level printers. The printer hardware and software (printer engine) are outside the evaluated TOE boundary.

5.2.3.13 Mandatory Access Control Policy (FDP_IFC.1(1))

The TSF shall enforce the Mandatory Access Control (MAC) Policy on FDP_IFC.1.1

a) the subjects listed in Table 6 acting on the behalf of users;

b) the named objects in Table 6;

and all operations among subjects and objects covered by the MAC policy <u>as defined in Table 6</u>.

Table 6: MAC SFP Subjects, Objects, and Operations

Subject	Object	Named Object	Operations between Subject/Named Object
processes acting on behalf of a specific user	FSO	device special files – block and character, TCB	Read/Write/Exec
		directory – regular	
		file – regular, system, audit	
		symbolic link	
	IPC	message	Read/Write/Exec
		semaphore	
		shared memory	
	Miscellaneous	signal vector	Read/Write
		STREAMS message block	
		pipe – unnamed (FIFO)	

5.2.3.14 Mandatory Integrity Control Policy (FDP_IFC.1(2))

The TSF shall enforce the **Mandatory Integrity Control (MIC) Policy** on FDP_IFC.1.1

a) the subjects listed in Table 7 acting on the behalf of users;

b) the named objects in Table 7;

and all operations among subjects and objects covered by the MIC policy as defined in Table 7.

Table 7: MIC SFP Subjects, Objects, and Operations

Subject	Object	Named Object	Operations between Subject/Named Object
processes acting on behalf of a specific user	FSO	device special files – block and character, TCB	Write
		directory – regular	
		file – regular, system, audit	
		symbolic link	
	IPC	message	Write
		semaphore	
		shared memory	
	Miscellaneous	signal vector	Write
		STREAMS message block	

5.2.3.15 Mandatory Trusted Network Policy (FDP_IFC.1(3))

The TSF shall enforce the **Trusted Network (TN) Policy** on FDP_IFC.1.1

a) **hosts identified by IP addresses as the subjects;**

b) **data packets to be transferred between hosts as objects;**

and the transfer of data packets from and to hosts via network connections.

5.2.3.16 Workload Partition Control Policy (FDP_IFC.1(4))

The TSF shall enforce the **Workload Partition (WPAR) Control Policy** on **processes in a WPAR as subjects, networks, file systems, IPCs, and signals as information, and operations that cause controlled information to flow to and from controlled subjects covered by the Workload Partition Control Policy**. FDP_IFC.1.1

5.2.3.17 IP Filter Control Policy (FDP_IFC.1(5))

The TSF shall enforce the **IP Filter Control Policy** on: FDP_IFC.1.1

- **source subject: TOE interface on which information is received**
- **destination subject: TOE interface to which information is destined**
- **information: network packets**
- **operations: accept or reject network packets**

5.2.3.18 Workload Partition Control Functions (FDP_IFF.1(1))

The TSF shall enforce the **Workload Partition (WPAR) Control Policy** based on the following types of subject and information security attributes: FDP_IFF.1.1

- **processes in a System WPAR as subjects, networks, file systems, IPCs, and signals as information, and the corral ID (CID) as the security attribute**.

The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: FDP_IFF.1.2

a) **If a process has a CID = 0 (i.e. the process is in the Global environment), then:**

   1) **the process can see processes in all WPARs via /dev/kmem**

   2) **the process with appropriate privileges can send signals to a process in any WPAR.**

b) **If a process has a CID ≠ 0 (i.e. the process is in a System WPAR), then:**

   1) **the process can only send signals to other processes with the same CID value**

   2) **the process cannot load kernel extensions, configure devices, or create/mount device special files**

   3) **the process cannot perform operations on system hardware (e.g. reboot, power off).**

c) **A process can only create and share IPC objects (message, semaphore, shared memory) with other processes with the same CID value.**

d) **A process can only bind to and receive packets from the network address(es) associated with its WPAR.**

e) **A process in a System WPAR has access to a sub-tree of the Global environment's file system.**

The TSF shall enforce the **rules:** FDP_IFF.1.3

a) **Processes in a System WPAR have read-only access to the Global environment's system files (/usr and /opt) unless other access is allowed by the Global environment's administrator.**

b) **Processes in a System WPAR have an isolated file system inaccessible by processes in any other System WPAR unless access is allowed by the Global environment administrator.**

c) **Processes in a System WPAR do not have access to the system's raw devices (e.g. /dev/rhdisk0) unless access is allowed by the Global environment's administrator.**

The TSF shall provide **no additional WPAR capabilities**. FDP_IFF.1.4

The TSF shall explicitly authorize an information flow based on the following rules: FDP_IFF.1.5

a) **by sharing file systems with one or more System WPARs**

b) **by granting WPARs privileges to load kernel extensions, configure devices and create/mount device special files**

c) **by granting WPARs privileges to access kernel data not associated with the WPAR**

d) **by granting WPARs privileges to perform operations on system hardware**

e) **by granting WPARs privileges to perform raw socket accesses**

The TSF shall explicitly deny an information flow based on the following rules: **none**. FDP_IFF.1.6

Application Note: Only System WPARs are a part of the evaluation.

5.2.3.19 IP Filter Control Functions (FDP_IFF.1(2))

The TSF shall enforce the **IP Filter Control Policy** based on the following types of subject and information security attributes: FDP_IFF.1.1

- **source subject security attributes:**
  - **set of source subject addresses**
- **destination subject security attributes:**
  - **set of destination subject addresses**
- **information security attributes:**
  - **presumed address of the source subject**
  - **address of the destination subject**
  - **transport layer protocol**
  - **source subject service address**
  - **destination subject service address (e.g., TCP or UDP destination port number)**
  - **ICMP message type and code as specified in RFC 792**
  - **IP version**

The TSF shall permit an information flow <u>(indefinitely or for an authorized administrator specified fixed time period)</u> between a <u>source</u> and <u>destination</u> via a controlled operation if the following rules hold: FDP_IFF.1.2

- **the presumed address of the source subject is in the set of source subject addresses**
- **the address of the destination subject is the set of destination subject addresses**
- **the information security attributes match the attributes in an information flow control policy specified by an authorized administrator**

The TSF shall enforce the **following rules:** FDP_IFF.1.3

- **allow source subjects to access TOE services ICMP without authenticating those source subjects**

- **allow the list of services specified immediately above to be enabled (become available to unauthenticated users) or disabled (become unavailable to unauthenticated users)**

The TSF shall provide the following**:** FDP_IFF.1.4

- **an authorized administrator shall have the capability to view all information flow allowed by this information flow control policy before the policy is applied.**

The TSF shall explicitly authorize an information flow based on the following rules: **none.** FDP_IFF.1.5

The TSF shall explicitly deny (indefinitely or for an authorized administrator specified time period) an information flow based on the following rules: FDP_IFF.1.6

- **requests for access or services where the presumed source identity of the information received by the TOE is not included in the set of source addresses for the source subject.**
- **requests for access or services where the presumed source address of the information received by the TOE specifies a broadcast address.**
- **requests in which the information received by the TOE contains the route (set of host network addresses) by which information shall flow from the source subject to the TOE.**

5.2.3.20 Mandatory Access Control Functions (FDP_IFF.2(1))

The TSF shall enforce the Mandatory Access Control Policy based on the following types of subject and information security attributes: FDP_IFF.2.1

a) The sensitivity label of the subject; and

b) The sensitivity label of the object containing the information.

Sensitivity label of subjects and objects shall consist of the following:

- A hierarchical level; and
- A set of non-hierarchical categories.

The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules, based on the ordering relationships between security attributes hold: FDP_IFF.2.2

a) If the sensitivity label of the subject is greater than or equal to the sensitivity label of the object, then the flow of information from the object to the subject is permitted (a read operation);

b) If the sensitivity label of the object is equal to the sensitivity label of the subject; then the flow of information from the subject to the object is permitted (a write operation);

c) If the sensitivity label of subject A is greater than or equal to the sensitivity label of subject B; then the flow of information from subject B to subject A is permitted.

The TSF shall enforce **no additional MAC rules**. FDP_IFF.2.3

The TSF shall provide **no additional MAC capabilities**. FDP_IFF.2.4

The TSF shall explicitly authorize an information flow based on the following rules: FDP_IFF.2.5

1. Privilege sets override MAC decisions in order to bypass

a) sensitivity clearance restrictions (PV_MAC_CL);

b) MAC restrictions when

1. **files flagged as being exempt from MAC (PV_MAC_OVRRD);**
2. **sending a signal (PV_MAC_W_PROC) or when getting information about a process (PV_MAC_R_PROC), provided that the target process's label is within the acting process's sensitivity clearance;**

c) all MAC READ restrictions (PV_MAC_R);

d) MAC READ restrictions when

1. **the object's label is within the process's sensitivity clearance (PV_MAC_R_CL);**
2. **reading a STREAM message block, provided that the message's label is within the process's sensitivity clearance (PV_MAC_R_STR);**

e) **all MAC WRITE restrictions (PV_MAC_W);**

f) **MAC WRITE restrictions when**

1. **the process label is greater than or equal to the object's label and the object's label is within the process's sensitivity clearance (PV_MAC_W_DN);**
2. **the process label is less then or equal to the object's label and the object's label is within the process's sensitivity clearance (PV_MAC_W_UP);**
3. **the object's label is within the process's sensitivity clearance (PV_MAC_W_CL);**

g) **a combination of all other MAC privileges (PV_MAC).**

2. SIGCHILD is exempt from the checks in FDP_IFF.2.2.

3. For partitioned directories, signals, and network streams: if the sensitivity label of the object is greater than the sensitivity label of the subject, then the flow of information from the subject to the object is permitted (write operation).

The TSF shall explicitly deny an information flow based on the following rules: **no additional rules**. FDP_IFF.2.6

The TSF shall enforce the following relationships for any two valid sensitivity labels: FDP_IFF.2.7

a) There exists an ordering function that, given two valid sensitivity labels, determines if the sensitivity labels are equal, if one sensitivity label is greater than the other, or if the sensitivity labels are incomparable, and

- Sensitivity labels are equal if the hierarchical level of both labels are equal and the non-hierarchically category sets are equal.
- Sensitivity label A is greater than sensitivity label B if one of the following conditions exists:
  - If the hierarchical level of A is greater than the hierarchical level of B, and the non-hierarchical category set of A is equal to the non-hierarchical category set of B.
  - If the hierarchical level of A is equal to the hierarchical level of B, and the non-hierarchical category set of A is a proper super-set of the non-hierarchical category set of B.
  - If the hierarchical level of A is greater than the hierarchical level of B, and the non-hierarchical category set of A is a proper super-set of the non-hierarchical category set of B.
- Sensitivity labels are incomparable if they are not equal and neither label is greater than the other.

b) There exists a "least upper bound" in the set of sensitivity labels, such that, given any two valid sensitivity labels, there is a valid sensitivity label that is greater than or equal to the two valid sensitivity labels; and

c) There exists a "greatest lower bound" in the set of the sensitivity labels, such that, given any two valid sensitivity labels, there is a valid sensitivity label that is not greater than the two valid sensitivity labels

Application Note for AIX: The AIX security policy model mandates equality when writing, which has been reflected by refining FDP_IFF.2.2 b).

Application Note for AIX: AIX allows to assign label ranges to devices and directories (see also section 6.1.6). Effectively, in such cases the MAC policy is applied to each of the labels in the label range and permits information flow if the subject's label matches any one of the object's labels as required by the policy.

5.2.3.21 Trusted Network (TN) Policy (FDP_IFF.2(2))

The TSF shall enforce the **TN Policy** based on the following types of subject and information security attributes: FDP_IFF.2.1

a) **Subject(s):**

1. **IP address;**

b) **information security attributes associated with objects:**

1. **IP address packet source and destination;**
2. **protocol;**
3. **port (source and destination);**
4. **network interface;**
5. **IPSO labels;**
6. **IPSO security attributes;**
7. **minimum and maximum SL (only when using CIPSO).**

The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules, based on the sequential ordering relationships between security attributes hold: FDP_IFF.2.2

a) **if the source/destination IP address of the packet is equal to the source/destination IP address specified in the rule;**
b) **if the IP address of the packet is within the network mask specified for the rule;**
c) **if the direction of the packet flow corresponds to the direction of the rule (IN/OUT);**
d) **if the protocol of the packet is equal to the protocol specified in the rule;**
e) **if the source/destination port is within the source/destination port range specified in the rule;**
f) **if the network interface of the packet is equal to the network interface specified in the rule;**
g) **if the IPSO labels are within the range defined by the rule, and rule set to allow IPSO labels;**
h) **if the packet's SL is within the minimum and maximum SL specified for the rule.**

The TSF shall enforce **no additional TN SFP rules**. FDP_IFF.2.3

The TSF shall provide **no additional TN SFP capabilities**. FDP_IFF.2.4

The TSF shall explicitly authorize an information flow based on the following rules: FDP_IFF.2.5

Privilege sets override TN decisions in order to allow a process to

a) **perform restricted ioctl calls to drivers (PV_NET_CNTL);**
b) **modify network configuration (PV_NET_CONFIG);**
c) **open a restricted port (PV_NET_PORT);**
d) **access raw sockets (PV_NET_RAWSOCK);**
e) **obtain the privileges in a) to c) (PV_NET).**

The TSF shall explicitly deny an information flow based on **no additional rules**. FDP_IFF.2.6

The TSF shall enforce the following relationships for any two valid information flow control security attributes: FDP_IFF.2.7

a) There exists an ordering function that, given two valid security attributes, determines the security attributes are equal, if one security attribute is greater than the other, or if the security attributes are incomparable; and
b) There exists a "least upper bound" in the set of security attributes, such that, given any two valid security attributes, there is a valid security attribute that is greater than or equal to the two valid security attributes; and
c) There exists a "greatest lower bound" in the set of security attributes, such that, given any two valid security attributes, there is a valid security attribute that is not greater than the two valid security attributes.

Application Note for AIX: RIPSO header are only able to apply hierarchical labels to the data transferred, RIPSO packets do not contain information about any compartments.

5.2.3.22 Mandatory Integrity Control Policy (FDP_IFF.2(3))

The TSF shall enforce the **Mandatory Integrity Control (MIC) Policy** based on the following types of subject and information security attributes: FDP_IFF.2.1

a) **The integrity label (TL) of the object containing the information.**

b) **Integrity labels of subjects and objects shall consist of a hierarchical level.**

The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules, based on the ordering relationships between security attributes hold: FDP_IFF.2.2

a) **If the integrity label of the subject is greater than or equal to the integrity label of the object; then the flow of information from the subject to the object is permitted (a write operation).**

The TSF shall enforce **no additional MIC SFP rules**. FDP_IFF.2.3

The TSF shall provide **no additional MIC SFP capabilities**. FDP_IFF.2.4

The TSF shall explicitly authorize an information flow based on the following rules: FDP_IFF.2.5

Privilege sets override MIC decisions in order to

a) **bypass integrity clearance restrictions (PV_MIC_CL);**

b) **bypass integrity restrictions (PV_MIC).**

The TSF shall explicitly deny an information flow based on **no additional rules**. FDP_IFF.2.6

The TSF shall enforce the following relationships for any two valid information flow control security attributes: FDP_IFF.2.7

a) There exists an ordering function that, given two valid security attributes, determines the security attributes are equal, if one security attribute is greater than the other, or if the security attributes are incomparable; and

b) There exists a "least upper bound" in the set of security attributes, such that, given any two valid security attributes, there is a valid security attribute that is greater than or equal to the two valid security attributes; and

c) There exists a "greatest lower bound" in the set of security attributes, such that, given any two valid security attributes, there is a valid security attribute that is not greater than the two valid security attributes.

Application Note for AIX: Read enforcement for mandatory integrity control is disabled in the evaluated configuration, and consequently no corresponding rule is specified in FDP_IFF.2.2.

5.2.3.23 Import of Unlabeled User Data (FDP_ITC.1)

The TSF shall enforce the Mandatory Access Control Policy when importing unlabeled user data, controlled under the MAC Policy, from outside the TSC. FDP_ITC.1.1

The TSF shall ignore any security attributes associated with the unlabeled user data when imported from outside the TSC. FDP_ITC.1.2

The TSF shall enforce the following rules when importing unlabeled user data controlled under the MAC policy from outside the TSC: FDP_ITC.1.3

a) Devices used to import data without security attributes cannot be used to import data with security attributes unless the change in device state is performed manually and is auditable;

b) **Only users with authorization can import unlabeled data from outside the TSC.**

c) **Only authorized administrators can specify the default label for imported data.**

5.2.3.24 Import of Labeled User Data (FDP_ITC.2)

The TSF shall enforce the Mandatory Access Control Policy when importing labeled user data, controlled under the MAC SFP from outside the TSC. FDP_ITC.2.1

The TSF shall use the security attributes associated with the imported labeled user data. FDP_ITC.2.2

The TSF shall ensure that the protocol used provides for the unambiguous association between security attributes and the labeled user data received. FDP_ITC.2.3

The TSF shall ensure that interpretation of the security attributes of the imported labeled user data is as intended by the source of the user data. FDP_ITC.2.4

The TSF shall enforce the following rules when importing labeled user data controlled under the MAC SFP from outside the TSC: FDP_ITC.2.5

a) Devices used to import data with security attributes cannot be used to import data without security attributes unless the change in device state is performed manually and is auditable;

b) no additional importation control rules.

c) Sensitivity label, consisting of the following:

- A hierarchical level; and
- A set of non-hierarchical categories.

5.2.3.25 Object Residual Information Protection (FDP_RIP.2)

The TSF shall ensure that any previous information content of a resource is made unavailable upon the allocation of the resource to all objects. FDP_RIP.2

Application Note for AIX: Chapter 6 describes for each object type how object reuse is handled.

5.2.3.26 Subject Residual Information Protection (Note 1)

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

Application Note for AIX: This requirement was added in the Labeled Security Protection Profile to address resources that are not directly allocated to objects. Chapter 6 explains in detail how object reuse is handled for many kinds of resources, also those that are not objects defined in this Security Target.

Application Note for AIX: Chapter 6 describes how residual information protection for processes is handled.

5.2.3.27 Hard disk drive residual information protection (FDP_RIP.3-AIX)

The TSF shall overwrite all data stored in currently user-accessible blocks of a hard disk drive with pre-defined bit patterns upon request of the authorized administrator. FDP_RIP.3-AIX.1

Application note for AIX: This requirement applies to SCSI drives only. Chapter 6 describes further how residual information protection for SCSI hard disk drives is implemented.

5.2.4 Identification and Authentication (FIA)

5.2.4.1 User Attribute Definition (FIA_ATD.1(1))

The TSF shall maintain the following list of security attributes belonging to individual users: FIA_ATD.1.1

a) User Identifier;

b) Group Memberships;

c) Authentication Data;

d) <u>LSPP Mode Only: </u>User <u>Sensitivity and Integrity </u>Clearances;

e) Security-relevant Roles<u> (User Authorizations)</u>; and

f) LSPP Mode Only: Default SL;

g) LSPP Mode Only: Default TL;

h) Audit Classes;

i) Password Aging Data;

j) Principle Name (Kerberos); and

k) Kerberos Tickets.

Application Note for AIX: The “Security-relevant Roles” correspond to the “Security-relevant Roles” of [CAPP] and [LSPP] and the “List of Authorized Roles” of [RBACPP].

5.2.4.2 User Attribute Definition (FIA_ATD.1(2))

The TSF shall maintain the following list of security attributes belonging to individual users: FIA_ATD.1.1

a) User identifier;

b) Group Memberships;

c) Authentication Data;

d) Security-relevant Roles.

Application Note for VIOS: The roles that are referenced here are the VIOS relevant roles.

5.2.4.3 Strength of Authentication Data (FIA_SOS.1)

The TSF shall provide a mechanism to verify that secrets meet the following: FIA_SOS.1

a) For each attempt to use the authentication mechanism, the probability that a random attempt will succeed is less than one in 1,000,000;

b) For multiple attempts to use the authentication mechanism during a one minute period, the probability that a random attempt during that minute will succeed is less than one in 100,000; and

c) Any feedback given during an attempt to use the authentication mechanism will not reduce the probability below the above metrics.

Application Note for AIX and VIOS: The TOE supports a number of configuration parameters that allow a system administrator to define a specific password policy. With a well-defined password policy and a clear guideline for users how to select passwords that are hard to guess the requirement can be satisfied. Additionally, the TOE supports the ability to restrict the number of failed attempts. The claimed strength of function for this mechanism is: SOF-medium.

5.2.4.4 Authentication (FIA_UAU.2)

The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user. FIA_UAU.2.1

5.2.4.5 Protected Authentication Feedback (FIA_UAU.7)

The TSF shall provide only obscured feedback to the user while the authentication is in progress. FIA_UAU.7

5.2.4.6 Identification (FIA_UID.2)

The TSF shall require each user to identify itself before allowing any other TSF-mediated actions on behalf of that user. FIA_UID.2.1

5.2.4.7 User-Subject Binding (FIA_USB.1(1))

The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: FIA_USB.1.1 / NOTE 2

a) The user identity which is associated with auditable events;

b) The user identity or identities which are used to enforce the Discretionary Access Control Policy;

c) The group membership or memberships used to enforce the Discretionary Access Control Policy;

d) LSPP Mode Only: The sensitivity label used to enforce the MAC SFP, which consists of the following:

- A hierarchical level; and
- A set of non-hierarchical categories.

e) LSPP Mode Only: The integrity label used to enforce the MIC SFP;

f) LSPP Mode Only: User sensitivity clearance;

g) **LSPP Mode Only: User integrity clearance;**

h) **Security relevant roles (user authorizations);**

i) **Privilege sets and privilege authorization sets associated with the subject being activated;**

j) **Audit Classes.**

The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of a user: FIA_USB.1.2 / NOTE 2

a) The sensitivity label associated with a subject shall be within the sensitivity clearance range of the user;

b) **Upon successful identification and authentication, the real user identifier, the effective user identifier and audit user identifier shall be those specified in the user entry for the user that has authenticated successfully;**

c) **Upon successful identification and authentication, the real group identifier and the effective group identifier shall be those specified via the group membership attribute in the user entry.**

The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of a user: FIA_USB.1.3 / NOTE 2

a) **The effective user ID of a user can be changed by the use of an executable with the setuid bit set. In this case the program is executed with the effective user ID of the program owner. The effective user ID can also be changed if the executable is listed in the privcmds table and the table includes an effective user ID value. In this case, the program is executed with the effective user ID in the table when the program is executed as a Privileged Command. Access rights are then evaluated using the effective user ID. The login user ID is not changed with this process, so all audit records can be traced to the real user that executes the program.**

b) **The effective user ID of a user can be changed by the su command. In this case the effective user ID of the user is changed to the user specified in the su command (provided authentication is successful). The login user ID remains unchanged, so all audit records can be traced to the real user that executes the program.**

c) **The effective group ID of a user can be changed by the use of an executable with the setguid bit set. In this case the program is executed with the effective group ID of the program owner. The effective group ID can also be changed if the executable is listed in the privcmds table and the table includes an effective group ID value. In this case, the program is executed with the effective group ID in the table when the program is executed as a Privileged Command. Access rights are then evaluated using the effective group ID. The login user ID is not changed with this process, so all audit records can be traced to the real user that executes the program.**

Application Note for AIX: While privilege sets and privilege authorization sets are actually associated with subjects (i.e. executables being activated on the behalf of a user as processes) rather than being user security attributes in the narrower sense, they have been added to this requirement to emphasize the fact that they become part of the subject's security attributes relevant for its execution.

5.2.4.8 VIOS User-Subject Binding (FIA_USB.1(2))

The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: FIA_USB.1.1

a) **User identity;**

b) **Group memberships;**

c) **Security-relevant roles.**

The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: FIA_USB.1.2

a) **Upon successful identification and authentication, the real user identifier, the effective user identifier and login user identifier shall be those specified in the user entry for the user that has authenticated successfully.**

b) **Upon successful identification and authentication, the real group identifier, and the effective group identifier shall be those specified via the group membership attribute in the user entry.**

The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: FIA_USB.1.3

a) **The effective userID of a user can be changed by the use of an executable with the setuid bit set. In this case the program is executed with the effective userID of the program owner. Access rights are then evaluated using the effective userID of the program owner. The login userID is not changed with this process.**

b) **The effective userID of a user can be changed by the su command. In this case the effective userID of the user is changed to the user specified in the su command (provided authentication is successful). The login userID remains unchanged.**

c) **The effective groupID of a user can be changed by the use of an executable with the setgid bit set. In this case the program is executed with the effective groupID of the program owning group. Access rights are then evaluated using the effective groupID of the program owner. The login userID is not changed with this process.**

5.2.5 Security Management (FMT)

5.2.5.1 Management of Security Functions Behavior (for specification of auditable events) (FMT_MOF.1(1))

The TSF shall restrict the ability to **disable and enable** the **audit** functions **and to specify which events are to be audited (see FAU_SEL.1.1)** to **the authorized administrators**. FMT_MOF.1.1

5.2.5.2 Management of Security Functions Behavior (for authentication data) (FMT_MOF.1(2))

The TSF shall restrict the ability to **manage the values of security attributes associated with user authentication data** to **authorized administrators**. FMT_MOF.1.1

5.2.5.3 Management of Object Security Attributes (FMT_MSA.1(1))

The TSF shall enforce the Discretionary Access Control Policy to restrict the ability to modify the access control attributes associated with a named object to **authorized administrators and the owner of the object. In the case of objects with NFSv4 ACLs, the system administrator and the owner can modify the access control attributes, plus other users can be granted permission within the ACL to modify the access control attributes of the object. In the case of TCP ports (CAPP mode only), modification of access control lists is restricted to system administrators only.** FMT_MSA.1.1

LSPP Mode Only: The TSF shall enforce the Mandatory Access Control Policy to restrict the ability to modify the sensitivity label associated with an object to **authorized administrators**. FMT_MSA.1.1

5.2.5.4 Management of TN Object Security Attributes (FMT_MSA.1(2))

The TSF shall enforce the **TN SFP** to restrict the ability to **modify** the security attributes **information flow control attributes representing the TN rules** to **authorized administrators**. FMT_MSA.1.1

5.2.5.5 Management of TCB Object Security Attributes (FMT_MSA.1(3))

The TSF shall enforce the **TCB SFP** to restrict the ability to **change_default** the security attributes **FSF_TLIB and FSF_TLIB_PROC** to **authorized administrators**. FMT_MSA.1.1

5.2.5.6 Management of MIC Object Security Attributes (FMT_MSA.1(4))

The TSF shall enforce the **MIC SFP** to restrict the ability to **modify** the security attributes **integrity labels** to **authorized administrators**. FMT_MSA.1.1

5.2.5.7 Management of Authorization Object Security Attributes (FMT_MSA.1(5))

The TSF shall enforce the **Authorization SFP** to restrict the ability to **modify** the security attributes **authorizations** to **authorized administrators**. FMT_MSA.1.1

5.2.5.8 Management of RBAC Authorization Object Security Attributes (FMT_MSA.1(6))

The TSF shall enforce the RBAC Policy to restrict the ability to modify, delete, and create instances of the following user security attribute to a set of RBAC Administrative Roles: FMT_MSA.1.1

a) User Role Authorization

5.2.5.9 Management of RBAC Default Object Security Attributes (FMT_MSA.1(7))

The TSF shall enforce the RBAC Policy to restrict the ability to create, modify the composition of the following user security attribute to a set of RBAC Administrative Roles: FMT_MSA.1.1

a) Default Active Role Set

5.2.5.10 Management of RBAC User Object Security Attributes (FMT_MSA.1(8))

The TSF shall enforce the RBAC Policy to restrict the ability to modify the composition of the following session security attribute to session owner: FMT_MSA.1.1

a) Active Role set for a user

5.2.5.11 Management of RBAC Administrative Object Security Attributes (FMT_MSA.1(9))

The TSF shall enforce the RBAC Policy to restrict the ability to modify the object security attributes to**:** FMT_MSA.1.1

i. Object Owners and

ii. Set of RBAC administrative roles.

5.2.5.12 Management of VIOS Object Security Attributes (FMT_MSA.1(10))

The TSF shall enforce the **VIOS Access Control Policy** to restrict the ability to **modify** the security attributes**:** FMT_MSA.1.1

- **For Volumes: mapping SCSI device drivers acting on behalf of LPAR partitions to logical volumes and physical volumes**
- **For Network: mapping of Ethernet device drivers acting on behalf of a group of LPAR partitions sharing a virtual network to Ethernet adapter device drivers**

to **system administrators only**.

Application Note for VIOS: This requirement is taken directly from CC part 2, not from any PP requirements; thus, PP audit requirements do not apply.

5.2.5.13 Management of WPAR Object Security Attributes (FMT_MSA.1(11))

The TSF shall enforce the **Workload Partition Control Policy** to restrict the ability to **modify** the security attributes **information flow control attributes representing the WPAR rules** to **authorized Global environment administrators**. FMT_MSA.1.1

5.2.5.14 Management of IP Filter Security Attributes (FMT_MSA.1(12))

The TSF shall enforce the IP Filter Control Policy to restrict the ability to **modify** the security attributes **information flow control attributes representing the IP Filter rules** to **authorized administrators**. FMT_MSA.1.1

5.2.5.15 Secure Security Attributes (FMT_MSA.2)

The TSF shall ensure that only secure values are accepted for security attributes. FMT_MSA.2.1

Application Note for AIX: This SFR fulfills a dependency from the security functional requirements FCS_CKM.1, FCS_CKM.2, FCS_COP.1, and for [RBACPP]. The assessment with respect to this requirement in the evaluation of this TOE does not include any assessment of the cryptographic strength of the keys generated or used. Instead the assessment with respect to this requirement just includes an assessment that the TOE protects those keys from unauthorized access, disclosure or tampering.

5.2.5.16 Static Attribute Initialization (FMT_MSA.3(1))

The TSF shall enforce the Discretionary Access Control Policy to provide restrictive default values for security attributes that are used to enforce the Discretionary Access Control Policy. FMT_MSA.3.1

LSPP Mode Only: The TSF shall enforce the Mandatory Access Control Policy to provide restrictive default values for security attributes that are used to enforce the Mandatory Access Control Policy. FMT_MSA.3.1

The TSF shall allow the **authorized administrators and the owner of the object for Discretionary Access Control and, in LSPP mode, authorized administrators for MAC** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

5.2.5.17 TN Static Attribute Initialization (FMT_MSA.3(2))

The TSF shall enforce **TN SFP** to provide **permissive** default values for security attributes that are used to enforce the **TN SFP**. FMT_MSA.3.1

The TSF shall allow the **authorized administrators** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

5.2.5.18 TCB Static Attribute Initialization (FMT_MSA.3(3))

The TSF shall enforce the **TCB SFP** to provide **permissive** default values for security attributes that are used to enforce the **TCB SFP**. FMT_MSA.3.1

The TSF shall allow the **authorized administrators** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

5.2.5.19 MIC Static Attribute Initialization (FMT_MSA.3(4))

The TSF shall enforce the **MIC SFP** to provide **restrictive** default values for security attributes that are used to enforce the **MIC SFP**. FMT_MSA.3.1

The TSF shall allow the **authorized administrators** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

5.2.5.20 Authorization Static Attribute Initialization (FMT_MSA.3(5))

The TSF shall enforce the **Authorization SFP** to provide **restrictive** default values for security attributes that are used to enforce the **Authorization SFP**. FMT_MSA.3.1

The TSF shall allow the **authorized administrators** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

5.2.5.21 RBAC Static Attribute Initialization (FMT_MSA.3(6))

The TSF shall enforce the RBAC Policy to provide **administrative user defined** default values for security attributes that are used to enforce the RBAC Policy. FMT_MSA.3.1

The TSF shall allow the following roles to specify alternative initial values to override the default values when an object or information is created: FMT_MSA.3.2

a) Set of RBAC Administrative Roles

5.2.5.22 VIOS Static Attribute Initialization (FMT_MSA.3(7))

The TSF shall enforce the **VIOS Access Control Policy** to provide **restrictive** default values for security attributes that are used to enforce the SFP. FMT_MSA.3.1

The TSF shall allow **no one** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

Application Note for VIOS: This requirement is taken directly from CC part 2, not from any PP requirements; thus, PP audit requirements do not apply.

5.2.5.23 WPAR Static Attribute Initialization (FMT_MSA.3(8))

The TSF shall enforce the **Workload Partition Control Policy** to provide **restrictive** default values for security attributes that are used to enforce the SFP. FMT_MSA.3.1

The TSF shall allow the **authorized Global environment administrators** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

5.2.5.24 IP Filter Static Attribute Initialization (FMT_MSA.3(9))

The TSF shall enforce the **IP Filter Control Policy** to provide **permissive** default values for security attributes that are used to enforce the SFP. FMT_MSA.3.1

The TSF shall allow the **authorized administrators** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

5.2.5.25 Management of the Audit Trail (FMT_MTD.1(1))

The TSF shall restrict the ability to create, delete, and clear the audit trail to authorized administrators. FMT_MTD.1.1

5.2.5.26 Management of Audited Events (FMT_MTD.1(2))

The TSF shall restrict the ability to modify or observe the set of audited events to authorized administrators. FMT_MTD.1.1

Application Note for AIX: DAC and, in LSPP mode, MAC are used to protect the information from unauthorized access.

5.2.5.27 Management of Audit Threshold (FMT_MTD.1(3))

The TSF shall restrict the ability to **modify** the **audit threshold** to **authorized administrators**. FMT_MTD.1.1

5.2.5.28 Management of User Attributes (FMT_MTD.1(4))

The TSF shall restrict the ability to initialize and modify the user security attributes, other than authentication data, to authorized administrators. FMT_MTD.1.1

5.2.5.29 Management of Authentication Data (FMT_MTD.1(5))

The TSF shall restrict the ability to initialize the authentication data to authorized administrators. FMT_MTD.1.1

The TSF shall restrict the ability to modify the authentication data to the following: FMT_MTD.1.1

a) authorized administrators; and

b) users authorized to modify their own authentication data.

5.2.5.30 Management of Privileges (FMT_MTD.1(6))

The TSF shall restrict the ability to **modify** the **privileges for applications** to **authorized administrators**. FMT_MTD.1.1

5.2.5.31 Management of RBAC (FMT_MTD.1(7))

The TSF shall restrict the ability to modify, create the following list of TSF Data to a set of RBAC Administrative Roles: FMT_MTD.1.1

a) All User Passwords

b) Role Definitions & Role Attributes

c) Role Hierarchies (by assigning one or more roles to other roles)

d) Constraints among Role Relationships

e) List of Auditable Events

5.2.5.32 Management of VIOS Mappings (FMT_MTD.1(8))

The TSF shall restrict the ability to **create, modify, and delete** the**:** FMT_MTD.1.1

- **For Volumes: mappings of logical volumes and physical volumes to VIOS SCSI device drivers acting on behalf of LPAR partitions**
- **For Network: mapping of VIOS Ethernet adapter device drivers to VIOS Ethernet device drivers acting on behalf of groups of LPAR partitions sharing virtual networks**

to **authorized administrators**.

Application Note for VIOS: This requirement is taken directly from CC part 2, not from any PP requirements; thus, PP audit requirements do not apply.

5.2.5.33 Management of WPAR Privileges (FMT_MTD.1(9))

The TSF shall restrict the ability to **modify** the **privileges of a WPAR** to **authorized Global environment administrators**. FMT_MTD.1.1

5.2.5.34 Management of WPAR CIDs (FMT_MTD.1(10))

The TSF shall restrict the ability to **modify** the **corral ID (CID) of a WPAR** to **none**. FMT_MTD.1.1

5.2.5.35 Management of IP Filter Rules (FMT_MTD.1(11))

The TSF shall restrict the ability to **create, modify, delete, activate, de-activate, and query** the **IP filter rules** to **authorized administrators**. FMT_MTD.1.1

5.2.5.36 Secure RBAC TSF Data (FMT_MTD.3)

The TSF shall ensure that only secure values are accepted for TSF data. FMT_MTD.3.1

5.2.5.37 Revocation of User Attributes (FMT_REV.1(1))

The TSF shall restrict the ability to revoke security attributes associated with the users within the TSC to a set of RBAC Administrative Roles. FMT_REV.1.1

The TSF shall enforce the rules: FMT_REV.1.2

a) The immediate revocation of security-relevant authorizations; and

b) **Revocations/modifications made by an administrator to security attributes of a user , such as the user identifier, user name, user group(s), user password or user login shell, shall be effective the next time the user logs in. Authorization changes take effect immediately upon reloading the kernel authorization table.**

c) **Kerberos tickets, once granted by Kerberos, are valid until they expire.**

Application Note for AIX: The immediate revocation method that can be used in AIX is the one described in the application note from the PP: Make the modifications to the users profile and then force the user to log off. Kerberos ticket revocation is outside the scope of the TOE, thus, Kerberos tickets are assumed to be valid until they expire.

5.2.5.38 Revocation of Object Attributes (FMT_REV.1(2))

The TSF shall restrict the ability to revoke security attributes associated with objects within the TSC to users authorized to modify the security attributes by the Discretionary Access Control, Role-Based Access Control Policy, and, in LSPP mode, Mandatory Access Control policy. FMT_REV.1.1

The TSF shall enforce the rules: FMT_REV.1.2

a) The access rights associated with an object shall be enforced when an access check is made;

b) LSPP Mode Only: The rules of the Mandatory Access Control policy are enforced on all future operations; and

c) Access rights to file system and IPC objects are checked when the object is opened. Revocations of access rights for file system objects become effective the next time a user affected by the revocation tries to open a file system object.

Application Note for AIX: Immediate revocation for file system objects is not implemented in AIX. AIX uses delayed revocation as described in the application note from the PP.

5.2.5.39 Revocation of VIOS User Attributes (FMT_REV.1(3))

The TSF shall restrict the ability to revoke security attributes associated with the **users** within the TSC to **authorized administrators**. FMT_REV.1.1

The TSF shall enforce the rules: FMT_REV.1.2

a) The immediate revocation of security-relevant authorizations; and

b) Revocations/modifications made by an administrator to security attributes of a user , such as the user identifier, user name, user group(s), user password or user login shell, shall be effective the next time the user logs in.

5.2.5.40 Specification of Management Functions (FMT_SMF.1)

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

- **Object security attributes management**
- **User attribute management**
- **Authentication data management**
- **Audit trail management**
- **Audit event management**
- **WPAR management**
- **VIOS mapping management**
- **IP filter rules management**

Application Note for AIX: This security functional requirement has been added as a result of AIS 32, Final Interpretation 065. This security functional requirement is not included in [CAPP] or [LSPP], because [CAPP] and [LSPP] were developed before AIS 32, Final Interpretation 065 was published. The security functional requirement was added because a dependency from FMT_MSA.1 and FMT_MTD.1 to this new component has been defined in AIS 32, Final Interpretation 065.

5.2.5.41 VIOS Security Roles (FMT_SMR.1)

The TSF shall maintain the roles**:** FMT_SMR.1.1

a) Prime Administrator

b) System Administrator

c) Development Engineer

d) Service Representative.

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

Application Note for AIX: This requirement is taken directly from CC part 2, not from any PP requirements; thus, PP audit requirements do not apply.

5.2.5.42 Security Management Roles (FMT_SMR.2)

The TSF shall maintain the roles: FMT_SMR.2.1

a) Set of RBAC administrative roles;

b) users authorized by the Discretionary Access Control Policy to modify object security attributes;

c) LSPP Mode Only: users authorized by the Mandatory Access Control Policy to modify object security attributes;

d) users authorized to modify their own authentication data; and

e) LSPP Mode Only: users authorized by the Mandatory Integrity Control Policy to modify object security attributes;

f) users authorized by the TCB Policy to modify object TCB attributes.

The TSF shall be able to associate users with roles. FMT_SMR.2.2

The TSF shall ensure that the following conditions for (a) Roles of Object Owners and (b) the set of RBAC administrative roles are satisfied: FMT_SMR.2.3

a) Object Owners can modify security attributes for only the objects they own (except for the sensitivity label)

b) The set of RBAC administrative roles can modify security attributes for all objects under the control of TOE (since they automatically inherit the privileges of all Objects Owners).

Application Note: A Global environment administrator is a special name given to an administrator of the Global environment. It is not a special type of administrator.

5.2.6 Protection of the TOE Security Functions (FPT)

5.2.6.1 Abstract Machine Testing (FPT_AMT.1)

The TSF shall run a suite of tests at the request of an authorized administrator to demonstrate the correct operation of the security assumptions provided by the abstract machine that underlies the TSF. FPT_AMT.1.1

Application Note for AIX: Such a test suite is provided as a separate program that an administrator may execute under controlled conditions.

5.2.6.2 Failure with preservation of Secure State (FPT_FLS.1)

The TSF shall preserve a secure state when the following failures occur: FPT_FLS.1.1

a) The entire RBAC database containing data on Privileges assigned to a role, Users authorized for a role, Role constraints and relationships or some specific tables containing subsets of these data are off-line, corrupt or inaccessible.

5.2.6.3 Manual Recovery (FPT_RCV.1)

After a failure or service discontinuity, the TSF shall enter a maintenance mode where the ability to return the TOE to a secure state is provided. FPT_RCV.1.1

5.2.6.4 Function Recovery (FPT_RCV.4)

The TSF shall ensure that the following SFs and failure scenarios have the property that the SF either completes successfully, or for the indicated failure scenarios, recovers to a consistent and secure state**:** FPT_RCV.4.1

a) The SF that checks whether a specified privilege is assigned to any role but the database containing the privilege data is not on-line or the particular data table is inaccessible.

b) The SF that checks whether a specified role has been assigned to a particular user but the database containing the role membership information is not on-line or the particular data table is inaccessible.

5.2.6.5 Reference Mediation (FPT_RVM.1)

The TSF shall ensure that the TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed. FPT_RVM.1.1

Application Note for AIX: AIX implements a reference monitor as single point of decision within its kernel.

5.2.6.6 Stack Execution Reference Mediation (FPT_RVM.2-AIX)

The TSF shall provide the ability to allow/deny the execution of code residing on the stack of a process created from the executable to anyone who can write to the executable and allow an authorized administrator to override this setting. FPT_RVM.2-AIX.1

5.2.6.7 Domain Separation (FPT_SEP.1)

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

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

5.2.6.8 Reliable Time Stamps (FPT_STM.1)

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

Application Note for AIX: The reliability of the time stamp is provided by the monotonic increasing time within AIX and the fact that all changes to the time are audited. This allows the determination of the exact sequence of audit events (which according to the application note within [CAPP] and [LSPP] is the source for this requirement). The accuracy of the internal clock is on the level of nanoseconds, which allows a precise sorting of audit records according to the time they have been generated.

5.2.6.9 Inter-TSF Basic TSF Data Consistency (FPT_TDC.1)

The TSF shall provide the capability to consistently interpret **sensitivity labels** when shared between the TSF and another trusted IT product. FPT_TDC.1.1

The TSF shall use **label encoding rules** when interpreting the TSF data from another trusted IT product. FPT_TDC.1.2

5.2.6.10 TSF Testing (FPT_TST.1)

The TSF shall run a suite of self tests periodically during normal operation **and** at the request of the authorized administrator to demonstrate the correct operation of the TSF. FPT_TST.1.1

The TSF shall provide authorized users with the capability to verify the integrity of TSF data. FPT_TST.1.2

The TSF shall provide authorized users with the capability to verify the integrity of stored TSF executable code. FPT_TST.1.3

Application Note for AIX: During boot-time and upon request of authorized administrators, integrity checks are performed to verify that the security attributes of each TCB file have not been modified.

5.2.7 Protection of the TOE Security Functions (FTA)

5.2.7.1 Limitation on Scope of Selectable Attributes (FTA_LSA.1)

The TSF shall restrict the scope of the session security attributes (Active Role Set for the User) based on the set of Authorized Roles for the User. FTA_LSA.1.1

5.2.7.2 TOE Session Establishment (FTA_TSE.1)

The TSF shall be able to deny session establishment based on the default active role set for the user being empty. FTA_TSE.1.1

5.2.8 Trusted path/channels (FTP)

5.2.8.1 Inter-TSF Trusted Channel (FTP_ITC.1)

The TSF shall provide a communication channel between itself and a remote trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure. FTP_ITC.1.1

The TSF shall permit **the TSF** to initiate communications via the trusted channel. FTP_ITC.1.2

The TSF shall initiate communication via the trusted channel for **communication with an NFSv4 server and for communications with an LDAP server**. FTP_ITC.1.3

Application Note for AIX: This requirement applies to the NFSv4 client/server communications. It also applies to LDAP when LDAP is used to serve authentication data.

5.3 Strength of Function

The claimed minimum strength of function is *SOF-medium*.

The security function within the TOE that uses a statistical or probabilistic mechanism is the authentication function that uses passwords.

5.4 TOE Security Assurance Requirements

The target evaluation assurance level for the product is EAL4 [CC] augmented by ALC_FLR.3.

5.5 Security Requirements for the IT Environment

The only IT environment where requirements are stated is the underlying processor, which has to provide the mechanism to protect the TSF and TSF data from unauthorized access and tampering. This is expressed with the following security functional requirement for the processor used to execute TOE software:

The following is a table identifying the SFRs for the IT environment used in this ST and their origin.

Table 8: SFRs for the IT environment

SFR	Origin
FCS_CKM.1(a)	CC Part 2
FCS_CKM.1(b)	CC Part 2
FDP_ACC.1(a)	CC Part 2
FDP_ACC.1(b)	CC Part 2
FDP_ACF.1(a)	CC Part 2
FDP_ACF.1(b)	CC Part 2
FIA_SOS.1(a)	CC Part 2
FIA_UID.2(a)	CC Part 2
FMT_MSA.3(a)	CC Part 2

Note: Section 4.2 mentions that OE.PROTECT can be implemented with cryptographic controls as one possible security function to meet this objective. But it is also mentioned there that this objective can be fully met by physical protection features, which are then part of the non-IT environment. Therefore it is not mandatory to address this security objective by a security function in the IT environment.

5.5.1 RSA Cryptographic Key Generation (FCS_CKM.1(a))

The <u>IT environment</u> shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **product specific** and specified cryptographic key sizes **1024 bit** that meet the following: **not specified.** FCS_CKM.1.1

Application Note: The SSL v3 specification does not define how the RSA key pair is generated. This is up to the implementation. Almost all implementations of the SSL v3 standard have their own algorithm for RSA key pair generation (if they support cipher suites that use RSA). Therefore the key generation and algorithm and the standard to follow are not defined here. Only the required key size is specified. The evaluation will assess that the keys generated form a correct RSA key pair. No assessment on the strength of the keys generated will be performed as part of this evaluation. The only assessment made is with respect to the probability of the numbers used to be prime.

5.5.2 Kerberos Cryptographic Key Generation (FCS_CKM.1(b))

The IT environment shall generate **symmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm**:** FCS_CKM.1.1

- **as defined in the Kerberos Version 5 standard**

and specified cryptographic key sizes

- **128 bit (AES)**
- **256 bit (AES)**
- **168 bit (TDEA)**

that meet the following:

- **generation as defined in the Kerberos Version 5 standard [RFC4120] with cipher suites defined in FCS_COP.1(3) and FCS_COP.1(4).**

Application Note: These keys are used with Kerberos communication and NFSv4 encrypted client/server communications.

5.5.3 Memory Subset Access Control (FDP_ACC.1(a))

The IT environment shall enforce the **Memory Access Control Policy** on **instructions as subjects and memory locations and processor register as objects**. FDP_ACC.1.1

5.5.4 LPAR Subset Access Control (FDP_ACC.1(b))

The IT environment shall enforce the **LPAR Resource Access Control Policy** on **processors, memory regions and I/O slots as objects and partitions as subjects and all access to those resources by a partition**. FDP_ACC.1.1

5.5.5 Memory Security Attribute Based Access Control (FDP_ACF.1(a))

The IT environment shall enforce the **Memory Access Control Policy** to objects based on the **processor state (user or supervisor)**. FDP_ACF.1.1

The IT environment shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **access to memory locations and special registers is based on the processor state and the state of the memory management unit. Access to dedicated processor registers is allowed only if the processor is in supervisor state when the instruction accessing the register is executed**. FDP_ACF.1.2

The IT environment shall explicitly authorize access of subjects to objects based on the following additional rules: **some dedicated processor registers may be read but not modified when the instruction accessing the register is in user mode**. FDP_ACF.1.3

The IT environment shall explicitly deny access of subjects to objects based on **no additional rules**. FDP_ACF.1.4

Application Note: The precise definition of the objects and the rules for the access control policy differ slightly depending on the processor type. For this security requirement on the IT environment the definition is detailed enough, since the implementation is not checked in this evaluation. When used for the hardware evaluation of a real processor those rules have to be stated precisely.

5.5.6 LPAR Security Attribute Based Access Control (FDP_ACF.1(b))

The IT environment shall enforce the **LPAR Resource Access Control Policy** to objects based on **the partition number**. FDP_ACF.1.1

The IT environment shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **a partition shall have access to a processor, a memory region or an I/O slot only if the resource is allocated to the partition by the table in the NVRAM**. FDP_ACF.1.2

The IT environment shall explicitly authorize access of subjects to objects based on the following additional rules: **none**. FDP_ACF.1.3

The IT environment shall explicitly deny access of subjects to objects based on **no other rules**. FDP_ACF.1.4

5.5.7 Kerberos Verification of Secrets (FIA_SOS.1(a))

The IT environment shall provide a mechanism to verify that secrets meet **the following:** FIA_SOS.1.1

a) **For each attempt to use the authentication mechanism, the probability that a random attempt will succeed is less than one in 1,000,000;**

b) **For multiple attempts to use the authentication mechanism during a one minute period, the probability that a random attempt during that minute will succeed is less than one in 100,000; and**

c) **Any feedback given during an attempt to use the authentication mechanism will not reduce the probability below the above metrics.**

Application Note: This requirement applies only to principle passwords in Kerberos.

5.5.8 LPAR User Identification Before Any Action (FIA_UID.2(a))

The IT environment shall require each user to identify itself before allowing any other IT environment-mediation actions on behalf of that user. FIA_UID.2.1

Application Note: This identification requirement applies to LPAR partitions being identified by the hypervisor.

5.5.9 Memory Static Attribute Initialization (FMT_MSA.3(a))

The IT environment shall enforce the **Memory Access Control Policy** to provide **permissive** default values for security attributes that are used to enforce the SFP. FMT_MSA.3.1

The IT environment shall allow **no role** to specify alternative initial values to override the default values when an object or information is created. FMT_MSA.3.2

Application Note: The “default” values in this case are seen as the values the processor has after start-up. They have to be “permissive”, since the initialization routine needs to set up the memory management unit and the device register etc. With respect to the hardware there is no “role” model implemented but the access control policy is purely based on a single attribute (“user” or “supervisor” state) that cannot be managed or assigned to a “user”. The attribute changes under well defined conditions (when the processor encounters an exception, an interrupt or when the sc instruction is executed (which effectively causes an interrupt to occur). The security requirement FMT_MSA.1 was therefore not applicable because the security attribute cannot be “managed”. For this reason there is also no security requirement FMT_SMR.1 included, because there are no “roles” that need to be managed or assigned to “users”. The dependency of FMT_MSA.3 to FMT_MSA.1 and FMT_SMR.1 is therefore unresolved.

5.6 Security Requirements for the Non-IT Environment

All the security objectives for the TOE environment address physical protection of the TOE or procedures that need to be obeyed by system administrators.

6 TOE Summary Specification

6.1 Security Enforcing Components Overview

6.1.1 Introduction

AIX provides a multi-user, multitasking, and multi-virtualized environment, where users interact with the operating system through commands issued to a command interpreter. The command interpreter invokes command programs, which in turn function by making system calls to the operating system kernel. The TSF is comprised of the kernel and trusted processes (trusted programs that are not part of the kernel). All operations performed by users are mediated by the TSF in accordance with the policies defined in Chapter 5.

Within AIX a user can LOGIN to the console of any host computer, request local services at that computer, as well as request network services from any other host in the system.

Processes perform all activity. A process may be started by a user issuing a command, may be created automatically to service a network request, or may be part of the running system created at system initialization. Each process is running a program. A process may begin running a new program (via the exec system call), or create a copy of itself (via the fork system call).

Some activities, such as responding to network requests, are performed directly by the kernel.

The following sections discuss services provided by the kernel, by non-kernel trusted software and the network services. Network services are discussed separately because their implementation is split between kernel and non-kernel components.

As long as those functions just provide a user interface (e. g. System Management tools) they are not considered to be part of the TSF. But if they directly implement part of a security function (e. g. the trusted processes that reads identification and authentication data) they are considered to be part of the TSF.

6.1.2 Kernel Services

The AIX kernel includes the base kernel and kernel extensions. The base kernel includes support for system initialization, memory management, file and I/O management, process control, audit services and Inter-Process Communications (IPC) services. Kernel extensions and device drivers are separate kernel software modules that perform specific functions within the operating system.

Device drivers are implemented as kernel extensions.

The base kernel has the following key characteristics:

- Can be paged out: Portions of the kernel code and data can be paged out, permitting the kernel to run using less memory than would be required for the whole kernel.
- Pinned: Part of the kernel is always resident or “pinned” into memory and cannot be paged. Pinned code cannot call kernel services that may result in a page fault.
- Can be preempted: The AIX kernel can be preempted. Higher priority threads may interrupt kernel threads, providing support for time critical functions.
- Dynamic and extendible: In standard AIX, kernel extensions can be loaded and unloaded while the system is running to allow a dynamic, extendible kernel without requiring a rebuild and reboot. In the evaluated configuration, dynamic changes to the kernel are prohibited through warnings described in the Security Guide. At system start up, only the kernel extensions that are part of the evaluated product may be loaded. As an example, the administrator can add pieces of hardware (as long as they are part of the hardware configuration listed in this Security Target) to a specific configuration and reboot the system. This will cause the kernel extensions that support the needed device drivers for the new hardware to be loaded. The ability to load/unload kernel extensions is restricted to the processes having the PV_DEV_LOAD privilege.

The AIX kernel implements a virtual memory manager (VMM) that allocates a large, contiguous address space to each process running on the system. This address space is spread across physical memory and paging space on a secondary storage device. The VMM manages the paging space used by the AIX file system and provides memory buffers for use within the kernel. The file system and VMM are tightly coupled. Disk pages, whether for file I/O or paging space, are faulted into free pages in memory. The VMM does not maintain a separate pool of pages solely for file system I/O.

The process management component includes the software that is responsible for creating, scheduling, and terminating processes and process threads. Process management allows multiple processes to exist simultaneously on a computer and to share usage of the computer's processor(s). A process is defined as a program in execution, that is, it consists of the program and the execution state of the program.

Process management also provides services such as inter-process communications (IPC) and event notification. The base kernel implements

- named pipes
- unnamed pipes
- signals
- semaphores
- shared memory
- message queues
- Internet domain sockets
- UNIX domain sockets
- Audit event generation

The file and I/O software provides access to files and devices. The AIX Logical File System (LFS) provides a consistent view of multiple physical file system implementations. The following file system types are included in the evaluated configuration:

- Journaled File System 2 (JFS2)
- Encrypted File System (EFS)
- CDROM File System (CDRFS)
- Universal Disk Format file system (UDFS)
- Network File System (NFS)
- Special File File System (SPECFS)

JFS2, EFS, CDRFS and UDFS work off of a physical medium (disk, CDROM, DVD) and NFS works across the network. SPECFS is a file system used internally by the kernel to support disk and other physical and virtual device I/O. The process file system, PROCFS, provides access to the process image of each process on the machine as if the process were a "file". Process access decisions are enforced by DAC, MAC (LSPP mode only), MIC (LSPP mode only), and TCB attributes inferred from the underlying process's security attributes.

LSPP Mode Only: cdrfs, udfs, procfs and (client-side) nfs are single level file systems: For mandatory access control, the labels of their mount point apply to all objects in the mounted file system. Single level file systems are not subject to mandatory integrity control, TCB and file security flag policies, and their objects cannot be associated with privileges. This is to be taken into account when reading the following sections of the TSS.

6.1.3 Non-Kernel TSF Services

The non-kernel TSF services are:

- Identification and Authentication services
- Auditing journaling and post-processing services
- Network application layer services
- System integrity checking

Those services support the security functions implemented within the kernel and use the kernel interface for this purpose, but they are not running themselves in kernel mode. Those functions are included in the TSF as far as they are required for the security services of the TOE (Identification and Authentication services), while other services that are implemented as tools or commands for the use of the system administrator and where the kernel prohibits the use/misuse of those tools or commands since they use kernel functions restricted to the system administrator and attempted use by normal users is prohibited by the kernel.

6.1.4 Network Services

Each computer is capable of providing the following types of services:

- Local services to the user currently logged in to the local computer console.
- Local services to previous users via deferred jobs.
- Local services to users who have accessed the local host via the network using protocols such as telnet.
- Network services to clients on either the local host or on remote hosts.

Network services are provided to clients via a client-server architecture. This client-server architecture refers to the division of the software that provides a service into a client portion, which makes requests, and a server portion, which carries out client requests (usually on a different computer). A service protocol acts as the interface between the client and server.

The primary low-level protocols are Internet Protocol (IP), Transmission Control Protocol (TCP), and User Datagram Protocol (UDP). IP is not user visible, but non-TSF processes may communicate with other hosts using a reliable byte stream or unreliable datagrams, TCP and UDP respectively.

The higher-level network services are built on TCP or UDP. The TOE supports the TCP application protocols listed below:

- Internet remote login and file transfer services (telnet and ftp) are supported within the evaluated product, as are similar BSD interfaces, including remote command execution (rlogin, rcp, rsh, rexec).
- The Hyper-Text Transfer Protocol (HTTP) is used by the WebInfo document display system (docsearch) for the presentation of public data. The HTTP server is not security relevant and therefore not part of the TSF.
- The Network File System (NFS) protocol is supported for remote file access. This includes some subsidiary protocols, such as the Remote Procedure Call (RPC), portmap protocols, and the mountd protocol for file system import and export.

CAPP Mode Only: AIX includes multiple X Windows clients in addition to an X Windows server on each host. Each server accepts connections from local clients using UNIX domain sockets.

LSPP Mode Only: In addition to the base connectivity provided, all network connections can be configured to support labeling as specified in the BSO/ESO/CIPSO/RIPSO protocols.

6.1.5 Workload Partitions

AIX supports virtual environments called Workload Partitions (WPARs) which provide virtual AIX environments within AIX. It provides the following distinct environments:

- **Global environment** – This is the main or top-level environment which provides the standard, full functionality of AIX. From this level, the other types of WPARs can be created, controlled, and destroyed. Only one Global environment exists within an LPAR.
- **System WPAR** – This is a virtual AIX environment created from the Global environment. It contains most of the functionality and programs that the Global environment contains and, in short, has the look and feel of a separate AIX system. The Global environment can create multiple, concurrent System WPARs.
- **Application WPAR** (Not part of the evaluated configuration.) – This is a very limited virtual AIX environment created from a Global environment. This WPAR contains only the applications specified during the creation of this WPAR. It allows for the running of one or more programs with process-space isolation from the Global environment. In short, the processes in an Application WPAR only know about the other processes in the same Application WPAR. The Global environment can create multiple, concurrent Application WPARs.

The install type (i.e. LSPP mode or CAPP mode) of the Global environment determines the type of System WPARs and Application WPARs. Thus, if the Global environment is in LSPP mode, then all WPARs created by the Global environment will be in LSPP mode. Similarly, if the Global environment is in CAPP mode, then all WPARs created by the Global environment will be in CAPP mode.

6.1.6 Security Policy Overview

The TOE is distributed across multiple host computers, each running a semiautonomous instance of the AIX operating system optionally using the NFSv4 distributed file system. The policy is described as follows:

- There is not a single kernel; rather, there is an AIX kernel running on each host computer in the system.
- The system does not have a common memory space; rather, each host in the system has its own memory space. Memory management, segmentation and paging are all managed locally, without respect to other hosts.
- The systems are maintained using a consistent user management policy across all systems.
- Identification and authentication (I&A) is performed locally by each host computer, but can use a common database. Each user is required to LOGIN with a valid password and user identifier combination at the local workstation and also at any remote computer where the user can enter commands to a shell program (e.g., remote login, and telnet sessions).
- Neither the process ID, nor the associated thread IDs, are unique within the system; rather, a PID, and its associated TIDs, are unique on each host within the system. Process and thread management is performed locally, without respect to other hosts.
- The names of objects may not be unique within the system; rather, object names are unique on each host. For example, each host maintains its own local file system, but may mount NFS exported file systems at various locations in the local directory tree.
- Discretionary access control (DAC) is performed locally by each of the host computers and is based on user identity and group membership. Each process has an identity (the user on whose behalf it is operating) and belongs to one or more groups. All named objects have an owning user, an owning group and a DAC attribute, which is a set of permission bits. In addition, file system objects optionally have an extended permission list also known as an AIXC Access Control List (ACL) or, in lieu of enforced permission bits, an NFSv4 ACL. Both the extended permissions mechanism and NFSv4 ACL are significant enhancements beyond traditional UNIX systems, and permits control of access based on lists of users and/or groups to whom specific permissions may be individually granted or denied. For TCP based services in CAPP mode, an additional type of ACL is provided that can be used to restrict user access to specific services.
- LSPP Mode Only: The system supports mandatory access control (MAC) based on sensitivity labels. From a defined set of hierarchical sensitivity levels (SLs), each named object is assigned a dedicated SL, and each user is assigned a range of SLs that he is allowed to access. Users (or, processes acting on behalf of users) can create new objects only with the SL they are currently operating under, cannot read objects that have a higher SL than the user and not write objects that have another SL than themselves. Directories and devices can optionally be assigned a label range – in this case, users can write to an object if their SL is within the SL range of the object.
- LSPP Mode Only: The system supports mandatory integrity control (MIC) based on hierarchical integrity labels (TLs). Every named object is assigned a dedicated TL, and each user is assigned a range of valid TLs. Users (or, processes acting on behalf of users) can create objects with the TL that they are currently operating under and cannot write objects that have a higher TL than themselves.
- The system protects trusted computing base (TCB) objects from modification during normal multi-user operation. Objects tagged with the FSF_TLIB flag can only be modified when the system is in configuration mode and the user (or, process acting on behalf of a user) has the appropriate privileges to apply changes to the TCB.
- The system uses authorizations and privileges to implement administrative roles (RBAC) and to allow the controlled by-passing of the security policies enforced by the TOE.
- Object reuse is performed locally, without respect to other hosts.
- Audit is performed locally by each host computer. The audit facility generates audit records for activities performed directly by untrusted processes (e.g., the system calls that perform file I/O) as well as trusted process activities (e.g., requests for batch jobs). Audit tools are available to merge audit files from the various hosts.
- Interrupt handling is performed locally, without respect to other hosts.

- Root Enabled Mode Only: Privilege is based on the root identity. All privileged processes (setuid root programs and programs run under the root identity) start as processes with all privileges enabled. Unprivileged processes, which include setgid trusted processes, start and end with no privileges enabled.
- VIOS discretionary access control is performed by VIOS to provide access control between VIOS SCSI device drivers acting on behalf of LPAR partitions and logical/physical volumes. It also provides access control between VIOS Ethernet device drivers acting on behalf of groups of LPAR partitions sharing virtual networks and VIOS Ethernet adapter device drivers.

6.1.7 TSF Structure

The TSF is the portion of the system that is responsible for enforcing the system's security policy. The TSFs of AIX are distributed across each System p5 and POWER6 host computer and consist of three major components: kernel software, kernel extension software, and trusted processes. All these components must operate correctly for the system to be trusted. Those functions are supported by the mechanisms of the underlying hardware which are used to protect the TSF from tampering by untrusted processes.

The hardware components support two execution states where kernel mode or supervisor state, software runs with hardware privilege and user mode or problem state software runs without hardware privilege. AIX also provides two types of memory protection: segmentation and page protection. The memory protection features isolate critical parts of the kernel from user processes and ensure that segments in use by one process are not available to other processes. The two-state architecture and the memory protections form the basis of the argument for process isolation and protection of the TSF.

The trusted processes include programs such as AIX administrative programs, shells, and standard UNIX utilities that run with administrative privilege, as a consequence of being invoked by a user with the root identity, if in the root enabled mode, or proper authorization, if in LSPP more or in root disabled mode, and, in the case of LSPP mode, with the appropriate MAC privileges. Non-kernel TSF software also includes daemons that provide system services, such as networking and managing audit data, as well as setuid, setgid, and privileged programs that can be executed by untrusted users and privileged commands which are defined in the privcmds table.

6.1.8 TSF Interfaces

Each sub-section here summarizes a class of interfaces in the AIX distributed system, and characterizes them in terms of the TSF boundary. The TSF boundary includes some interfaces, such as commands implemented by privileged processes, which are similar in style to other interfaces that are not part of the TSF boundary and thus not trusted. Some interfaces are part of the TSF boundary only when used in a privileged environment, such as an administrator's process, but not when used in a non-privileged environment, such as a normal user process. All interface classes are described in further detail in the next chapter, and the mechanisms in subsequent chapters. As this is only an introduction, no explicit forward references are provided.

6.1.8.1 User Interfaces

The typical interface presented to a user is the command interpreter, or shell. The user types commands to the interpreter, and in turn, the interpreter invokes programs. The programs execute hardware instructions and invoke the kernel to perform services, such as file access or I/O to the user's terminal. A program may also invoke other programs, or request services using an IPC mechanism. Before using the command interpreter, a user must log in.

The command interpreter or shell as well as other programs operating on behalf of a user have the following interfaces:

- CPU instructions, which a process uses to perform computations within the processor's registers and a process's memory areas. CPU instructions are interpreted by the hardware, which is part of the TOE environment; CPU instructions are therefore not a TSF interface.
- System calls (e.g.,open, fork), through which a process requests services from the kernel, and are invoked using a special CPU instruction; System calls are the primary way for a program operating on behalf of a user to request services of the TOE including the security services. System calls related to security functions are therefore part of the TSF interface.
- Directly-invoked trusted processes (e.g., passwd) which perform higher-level services, and are invoked with an exec system call that names an appropriate program which is part of the TSF, and replaces the current process's content with it; a limited number of those processes exist that perform security functions and are therefore part of the TSF interface.

- Daemons, which accept requests stored in files or communicated via other IPC mechanisms, generally created through use of directly invoked processes (some trusted, some untrusted). A few daemons perform security functions and therefore present part of the TSF interface.
- Distributed Services (e.g., telnet) - The distributed services interface operates at many different levels of abstraction. At the highest level, it provides a means for users on one host within the system to request a virtual terminal connection on another host within the system. At a lower level, it allows a system to request a specific service from another system on behalf of a user. Examples of requested services include, executing a command line (e.g.,rsh) or transferring whole files (e.g.,FTP). At the lowest level, it allows a subject on one host in the system to request a connection (e.g.,TCP), or deliver data (e.g.,UDP) to a listening subject. Distributed services usually consist of a client on the requestor's side and a server (usually a daemon) running on the server's side. Authentication (if required by the service) and access control use dedicated interfaces to the functions on the server side which are therefore part of the TSF interface.

6.1.8.2 Operation and Administrator Interface

The primary administrative interfaces to AIX are the same as the interfaces for ordinary users. In an LSPP mode system or in a root disabled mode system, authorized administrators log into the system using their user ID and password, and perform administrative tasks they have been authorized to perform. Additionally, in a root enabled mode system, an administrator can log into the system with a standard, untrusted identity and password and, after assuming the root identity, uses standard AIX commands to perform administrative tasks.

The system is composed of one or more System p5 and/or POWER6 computer systems. Each of these host computers may be in one of the following states: shut down, initialization, single-user mode, or multi-user secure state. Administration entails the configuration of multiple computers and the interactions of those computers, as well as the administration of users, groups, files, printers, and other resources within the system. It also includes administering WPARs within a host.

AIX provides a general purposes, menu-based utility for system administration: *smitty*. Other programs (e.g., */usr/bin/acledit*, */usr/bin/chuser*, */usr/bin/rm*) and scripts are used for system administration, but *smitty* is significant because it provides comprehensive system administration capabilities.

smitty is required for the administration of the AIX distributed system, but the decision as to which administrative utility to use depends upon whether or not the system is in a secure state:

- *smitty* (a cursor-based ASCII version of the System Management Interface Tool (SMIT)) is a text menu interface and dispatcher for a collection of administrative programs.
  smitty is used to administer the local host, i.e., the computer where it is run.

There are other tools for system administration (e. g., msmit) that provide a graphical user interface for system administration. Those tools are not part of the evaluated configuration.

The part of the administrative database that is used to configure and manage TSF is seen as part of the TSF interface. The administrative database is protected by the access control mechanisms of the TOE. It is therefore very important to set the access rights to the files of the administrative database such that non-administrative users are prohibited from modifying those files and have read access on a need to know basis only.

6.1.9 Secure and Non-Secure States

The secure state for the AIX distributed system is defined as a host's entry into multi-user mode with auditing fully operational. At this point, the host accepts user logins and services network requests. If these facilities are not available, the host is considered to be in a non-secure state. Although it may be operational in a limited sense and available for an administrative user to perform system repair, maintenance, and diagnostic activity, the TSF are not in full operation and is not necessarily protecting all system resources according to the security policy. The non-secure state is also a specific configuration state of the system where kernel security flags and the trusted library path can be modified and FSF_TLIB or FSF_TLIB_PROC tagged objects can be created, modified or deleted. This functionality is not available in the operational / multiuser mode.

With respect to auditing this Security Target does not define a minimum level of events that need to be audited. But it's required that the system administrator is able to configure all the events mentioned in this Security Target to be included in the audit trail. A system administrator may then - according to his requirements - define the events that are audited. He is able to change those events using the audit configuration functions during system operation.

6.2 Description of the Security Enforcing Functions

6.2.1 Introduction

This chapter describes how the Security Enforcing components of the TOE provide the Security Requirements identified in chapter 5.

A high level description is provided for each group of security enforcing functions (SEF) providing a common feature or service, and stating how the functionality specified by the Security Enforcing Function group is provided by the security enforcing components identified in this chapter.

The security enforcing function groups identified in this chapter follow the description given in chapter 2.

The TOE Security Functions (TSF) are described with sufficient detail to provide a general understanding of those functions and how they work. A more detailed description of those functions and a mapping of the TSF to TOE subsystems is provided in the high level design documentation.

References to components given in *italics* can be traced to manual pages or TOE sources for further information. Note also that some commands initiate trusted processes or are a local front end to a trusted process (e.g.,*ftp* and the *ftpd* daemon, *telnet* and the *telnetd* daemon). In these instances, a generic reference to the command is made.

6.2.2 Identification and Authentication (IA)

User identification and authentication in the AIX system includes all forms of interactive login (e.g., using the Telnet or FTP protocols) as well as identity changes through the *su* command. These all rely on explicit authentication information provided interactively by a user.

Identification and authentication of users is performed either from a terminal where no user is logged on or when a user that is logged on starts a service that requires additional authentication. All those services use a common mechanism for authentication described as function IA.2 in this section. They all use the administrative databases described in function IA.1 in this section. The administrative database is managed by system administrators but users are allowed to modify their own password using the *passwd* command. Function IA.3 describes the authentication process for those network services that require authentication. Function IA.4 describes the change of the user’s identity using the *su* command. Function IA.5 described the login process when a user logs in at a terminal. Function IA.6 describes the logoff process.

6.2.2.1 User Identification and Authentication Data Management (IA.1)

The TOE supports the following identification and authentication (I&A) administrative database types for both the Global environment and System WPAR:

- File-based
- LDAP-based (CAPP mode only).

Only remote logins and logins using the *clogin* command are possible for System WPARs.

VIOS is limited to supporting File-based authentication only.

6.2.2.1.1 File-based I&A

AIX by default maintains a local administrative database. This database is used to manage identification and authentication data used by the operating system.

Administrators, through the SMIT (*smitty*) administrative interface or via command line tools, perform changes to the files that constitute the administrative database.

Users are allowed to change their passwords by using the *passwd* command, which is a *privileged program*. This configuration allows a process running the *passwd* program to read the contents of */etc/security/user* and to modify the */etc/security/passwd* file for the user’s password entry, both which would ordinarily be inaccessible to a non-privileged user process. Users are also forced to change their passwords at login time, if the password has expired.

The file */etc/passwd* contains the user’s name, the ID of the user, an indicator, if the password of the user is valid, the principal group ID of the user, the clearance label (MAC) of the user (in LSPP mode only), and a few other, not security relevant information. The encrypted password of the user itself is not stored in this file but in the file */etc/security/passwd* which is protected against read access for ordinary users. This prohibits dictionary attacks on passwords in the *passwd* file as for example described in the paper of Ken Thomson and Bob Morris "Password Security - A Case History".

The file */etc/security/passwd* contains the encrypted password, the time the password was last changed and some other information that are not subject to the security functions as defined in this Security Target.

For a complete list of user attributes see the description of the function SM.4.

The system administrator defines restrictions on authentication data like minimum and maximum size, the minimum number of alphabetic characters, the minimum number of characters that are different from the old password, the minimum number of non-alphabetic characters as well as the maximum life time of a password, the number of unsuccessful login attempts allowed before the account is locked and the times and days the user is allowed to log into the system. Those restrictions can be defined on a per user basis and are stored in the file */etc/security/user*. The system administrator can use those parameters to define a password policy.

VIOS supports a subset of these authentication restrictions. See SM.4 for more details.

The file */etc/security/lastlog* contains the time since the last successful login, the time of the last unsuccessful login and the number of unsuccessful login attempts since the last successful login.

6.2.2.1.2 LDAP-based I&A

LDAP-based I&A is supported in CAPP mode only.

The TOE includes LDAP-based I&A where the LDAP-base I&A is configured in the "UNIX-type" authentication mode. (The LDAP server is part of the TOE environment, not the TOE.) In this mode, the administrative data (including user names, IDs, passwords, and, in the case LSPP mode, clearance labels (MAC)) are stored in LDAP where access to the data is limited to the LDAP administrator. When a user logs into the TOE, the TOE binds to the LDAP server using the LDAP administrator account over an SSL connection, retrieves the necessary data for the user (including the password) from LDAP, and then performs authentication using the data retrieved from LDAP.

The system maintains an administrative database on an LDAP server. The remaining hosts import the administrative data from the same LDAP server through the same mechanism described in the previous paragraph.

The system maintains a consistent administrative database by making all administrative changes on the designated LDAP server. A user ID on any computer refers to the same individual on all other computers. In addition, the password configuration, name-to-UID mappings, and other data are identical on all hosts in the distributed system.

Administrators, through the SMIT administrative interface, perform changes to the LDAP data that constitute the administrative database.

Users are allowed to change their passwords by using the *passwd* command, which is a privileged program. This configuration allows a process running the *passwd* program to communicate to the local trusted process LDAP authentication daemon (which is also a privileged program) over a privileged socket, and request the LDAP authentication daemon to retrieve and modify the user's password entry. Users are also forced to change their passwords at login time, if the password has expired.

LDAP contains the user's name, the id of the user, the encrypted password, the time the password was last changed, an indicator, if the password of the user is valid, the principal group id of the user, and a few other not security relevant attributes.

For a complete list of user attributes see the description of the function SM.4.

The system administrator defines restrictions on authentication data like minimum and maximum size, the minimum number of alphabetic characters, the minimum number of characters that are different from the old password, the minimum number of non-alphabetic characters as well as the maximum life time of a password, the number of unsuccessful login attempts allowed before the account is locked and the times and days the user is allowed to log into the system. Those restrictions can be defined on a per user basis and are stored in LDAP. The system administrator can use those parameters to define a password policy such that the passwords satisfy the requirements defined in FIA_SOS.1.

LDAP also contains the time since the last successful login, the time of the last unsuccessful login and the number of unsuccessful login attempts since the last successful login.

The SSL v3 communication protocol supports the following encryption methods:

- AES 128 bit and 256 bit

- TDES 168 bit
- RC4 128 bit
- RSA 1024 bit keys.

It also supports SHA-1 and MD5 hash digests. The TOE only validates RSA keys (i.e., it does not generate or distribute them). The generation and distribution of RSA keys is performed outside of the TOE.

Note that the cryptography used in this product has not been FIPS certified nor has it been analyzed or tested to conform to cryptographic standards during this evaluation. All cryptography has only been asserted as tested by the vendor.

6.2.2.2 Common Authentication Mechanism (IA.2)

AIX includes a common authentication mechanism which is a subroutine used for all activities that create a user session, including all the interactive login activities, batch jobs, and authentication for the *su* command.

The common mechanism includes the following checks and operations:

- Check password authentication
- Check password expiration
- Check whether access should be denied due to too many consecutive authentication failures
- Get user security characteristics (e.g., user, groups, clearances (LSPP mode only), authorizations)

The common I&A mechanism identifies the user based on the supplied user name, gets that user's security attributes, and performs authentication against the user's password. A result of success indicated by a 1, or a failure indicated by a 0, is returned to the Terminal State Manager (TSM) program which continues the login process.

LSPP Mode Only: When accessing the TOE via its local system console, the ISSO and SO users are exempt from checks pertaining to consecutive authentication failures.

6.2.2.3 Interactive Login and Related Mechanisms (IA.3)

There are multiple mechanisms for interactive login and similar activities:

- the standard *login* program for interactive login sessions on the console of a user's local host;
- the *telnet* protocol and the *rlogin* protocol for ordinary interactive login sessions on any host in the system;
- the *rsh*, *rcp*, and the *rexec* protocols for remote shell, copy, and single command executions;
- the *FTP* protocol for interactive file transfer;
- CAPP Mode Only: the *xlock* program that is used to lock active X window sessions;
- the *swrole* command for changing roles within a login session;
- and the *su* command for changing user identity during a session

All of these mechanisms use the common authentication mechanism described above, but only those that create normal interactive sessions use the standard *login* program; others implement special-purpose types of sessions.

VIOS supports a subset of these mechanisms:

- the standard *login* program
- the *Telnet* protocol
- the *su* command

All those mechanisms will not display a password that is entered via a keyboard for authentication but provide obscured feedback.

Note: *xlock* is not a full login mechanism but uses the same authentication mechanism to re-authenticate a user who has locked an X window session.

6.2.2.3.1 The Login Program

The *login* program establishes interactive user sessions. In AIX and VIOS, *login* is part of the Terminal State Manager (TSM) program. This program prompts for a user identity and authentication (e.g., password), and validates them using the common authentication mechanism described above.

In LSPP mode only, during login, the user can append the option " –e" to his user name, which allows him to specify a label within his clearance to be used during this session (rather than the default label defined for the user).

Authentication prompting may also be suppressed when appropriate (e.g., *rsh*). If the validation fails, the prompts are repeated until the limits on successive authentication failures are exceeded. Each failure is considered an event that may be audited.

Login establishes a user session as follows:

1. Assigns a session identifier
2. Sets exclusive access for the controlling terminal to the process logging in
3. Calls the common authentication mechanism to check validity of the password provided for the account being accessed, and gains the session security attributes
4. Sets up the user environment
5. Checks for password expiration and if so, prompts for password change
6. The process's user and group identities are changed to those of the user
7. The process's authorizations and privileges are set to those of the user
8. In LSPP mode only, the user's sensitivity clearance is set according to the users entry in the sensitivity clearance database
9. In LSPP mode only, the process's sensitivity label is set to what was specified by the user, provided that the SL is within the user's sensitivity clearance, or to the user's default SL if no SL was specified
10. In LSPP mode only, the process's integrity label is set to the user's default TL
11. In LSPP mode only, the user's integrity clearance is set according to the users entry in the integrity clearance database
12. User is changed to his or her home directory
13. Invokes the user's default shell

The *login* program is always invoked with open file descriptors for the controlling terminal, used when prompting for identity and authentication information, and passes control to the user's shell when the session is created. At this point, the user session is established, the user environment is set up, and the program replaces itself, using the exec system call, with the user's shell).

6.2.2.3.2 Network Login

After an initial login on the console or a terminal, access to other hosts within the same security domain may occur through the following network protocols: telnet, rlogin, rsh, rcp, rexec, and FTP (refer to section 6.2.2.3.2.6 for FTP). In LSPP mode, connections are restricted to work only on the same sensitivity clearance. Connection sensitivity clearance labels are not enforced by VIOS.

6.2.2.3.2.1 *Login with telnet*

The telnet protocol always requests user identity and authentication by invoking the *login* program, which uses the common authentication mechanism. A user can change identity across a telnet connection if the password for another account is known.

6.2.2.3.2.2 *Login with rlogin*

The rlogin protocol includes user identity as part of the protocol information passed from host to host. User is not permitted to switch identity between hosts using -l option. See the description of rsh command execution below for details on the enforcement mechanism.

6.2.2.3.2.3 *Command execution using rsh*

The rsh protocol includes user identity as part of the protocol information passed from host to host. User is not permitted to switch identity between hosts using -l option. rshd checks to see that the remote and local user names are the same. Remember that remote user ID information flows with the rsh connection request as part of the protocol. The requirement that a privileged/reserved port be used as part of the setup insures that the information in the protocol flow was created by a trusted process.

6.2.2.3.2.4 *Command execution using rcp*

The RCP protocol includes user identity as part of the protocol information passed from host to host. User is not permitted to switch identity between hosts using -l option. See the description of command execution using rsh for details of the enforcement mechanism.

6.2.2.3.2.5 *Command execution using rexec*

The rexec protocol always requires the user to enter a valid user identity and password. The authentication is performed by invoking the common authentication mechanism directly rather than by invoking login. User can change identity if password is known.

6.2.2.3.2.6 *File transfer using FTP*

The FTP protocol is used to create a special type of interactive session that only permits file transfer activities. An FTP session is validated and created directly by the FTP server, which then executes all the user requests directly, as opposed to invoking a user-specified program.

The FTP server invokes the authenticate() function that uses the common authentication mechanism to validate the user identity and password supplied through FTP protocol transactions. User can change identity if password is known.

6.2.2.4 User Identity Changing (IA.4)

Users can change identity (i.e., switch to another identity) using the *su* command. When switching identities, the login UID is not changed, so all actions are ultimately traceable in the audit trail to the originating user. The primary use of the *su* command within AIX is to allow appropriately authorized individuals the ability to assume the root or other administrative identities. In both LSPP mode and root disabled mode systems, the capability to login as the root identity and to *su* to the root user have been eliminated. In CAPP mode in the */etc/security/user* file, login to root is set to false for all users and *su* is set to true for administrators. This allows an administrative user to login under his/her real identity, then *su* to the root or other administrative identities.

a) The *su* command invokes the common authentication mechanism to validate the supplied authentication.

b) When using *su* to change the id, the authorizations associated with the ID are also changed.

VIOS contains the *su* command, but it doesn't allow users to directly execute it. Instead, the command-line interface will execute a subset of the commands available to a role by using the *su* command under the covers. Commands that would normally allow a user to escape to a shell (i.e., vi) have been modified to disable the shell escape feature. Thus, users cannot directly change their identities during a session.

6.2.2.5 Login Processing (IA.5)

Permissions on the device special files control access to exclusively used public devices. When a user successfully logs in at the local direct attached console, the TSM program changes the ownership of */dev/lft0, /dev/kbd0, /dev/rcm0,* and, in system using X windows, */dev/mouse0* to the login UID of the user and sets the permissions on these devices to be readable and writable by this user. */dev/lft0* is a logical device that provides the users interface to the keyboard and graphics adapter. At system initialization, */dev/lft0* grabs the keyboard and graphics adapter devices. In case of a serially attached ASCII terminal, the tty device associated with the terminal changes ownership to the user that is logged in (for example */dev/tty0*)

The */dev/kbd0* device contains two channels for communication between the keyboard and the device driver. Only one channel is active at any given time. The */dev/lft0* device registers for the first channel when the system boots. The second channel is reserved for the X server (which is not supported by all configurations of the TOE). The permissions on the */dev/kbd0* device restrict that only the user who is logged in on the console can access this device. The logged in user could open the second channel, because he/she has permissions. This would redirect the users own keyboard device. This would pose no threat to the operation of the system. The worst thing that would happen is that the login process would not be able to regain access to the */dev/kbd0* device and no other users would be able to login on the console device until the host was rebooted.

The */dev/mouse0* device contains only one channel, which is grabbed by the */dev/lft0* device on system startup. Attempts to open additional instances of the */dev/mouse0* device will result in an error message.

The login process executes a revoke to invalidate any open file descriptors for */dev/lft0* or the appropriate */dev/ttyN* device held by a previous user. The revoke call modifies the file descriptors entry in the system open file table, causing further attempts to access the device special file based on that file descriptor to return “bad file descriptor”. This ensures that the new login session is isolated from any previous login sessions.

For LSPP mode only, users are assigned a default login SL and TL which is the effective SL and effective TL of the user's process after a successful login. If the user does not want to log in at his/her default login SL, the user may choose to supply a different SL at login time by using the -e option of the login command. The SL supplied by the user must be dominated by the user's clearance and contained in the system accreditation range. The TL cannot be specified by the user at login time. The default login SL and TL are defined in the file /etc/security/clear along with the username and clearance for each user.

6.2.2.6 Logoff Processing (IA.6)

When a user logs off, all files that were opened by the login shell are closed. Files and devices that were opened by background tasks remain open. However, a background job that had access to the console loses that access prior to the next user’s login as stated above.

The ownership of */dev/ttyN*, */dev/lft0*, */dev/kbd0*, */dev/mouse0* (on evaluated configurations supporting X windows), and */dev/rcm0* is returned to root when the logoff occurs.

6.2.3 Auditing (AU)

This section discusses the implementation of auditing in the evaluated configuration. The data structures and formats are discussed first, followed by how audit is controlled, a description of bin mode auditing, the programs used to post process the audit data, the programs used to review audit data, audit file protection, the potential for audit data loss, and finally the audit privileges.

6.2.3.1 Audit Record Format (AU.1)

The audit record consists of a header that contains information identifying the user and process who generated the record, the status of the event (success or failure), and the CPU ID for the system. The CPU ID field allows the administrator to differentiate between individual machines when merging the contents of multiple audit trails. An optional variable length tail contains extra information about the event, as defined in */etc/security/audit/events*. The audit records in the Global environment are also tagged with the WPAR CID so that the trail can also distinguish between WPARs.

The audit record is a fixed length record that contains information about the user who caused the event and whether the event was created due to a success or failure. The audit record is defined in */usr/include/sys/audit.h.*

Table 9: Audit Record Format

Magic number for audit record.	
The length of the tail portion of the audit record.	
The name of the event and a null terminator.	
An indication of whether the event describes a successful	
operation. The values for this field are:	
0	Indicates successful completion.
1	Indicates a failure.
>1	An errno value describing the failure.
The effective user ID.	
The effective group ID.	
The real user ID; that is, the ID number of the user who created the process that wrote this record.	
The login ID of the user who created the process that wrote this record.	
The program name of the process, along with a null terminator.	
The process ID of the process that wrote this record.	
The process ID of the parent of this process.	
The thread ID.	
The time in seconds at which this audit record was written.	
The nanoseconds offset from time. (used during bin recovery and trail merging to ensure proper record ordering)	
CPU identifier.	
Active role IDs.	
WPAR corral ID (CID).	
Effective privileges.	
Sensitivity label.	
Integrity label.	

6.2.3.2 Audit Record Generation (AU.2)

Audit record generation begins with the detection of an event, and follows the record as it advances to storage.

Event detection is distributed throughout the TSF, both in kernel and user mode. Programs and kernel modules that detect events that may be audited are responsible for reporting these events to the system audit logger. The system audit logger is part of the kernel, and can be accessed via a system call for trusted program auditing, or via a kernel procedure call for supervisor state auditing.

The audit logger is responsible for constructing the complete audit record, including the identity and state information and the event specific information. The audit logger appends the record to the active bin. A bin is a file that is used to store raw audit records before they are processed and stored in the audit trail.

6.2.3.3 Audit Record Processing (AU.3)

Audit record processing includes a description of bin mode auditing and the backend processors that are utilized by the audit subsystem.

6.2.3.3.1 Bin Mode Auditing

When Bin mode auditing starts, two separate bin files are allocated to store raw audit records by the auditbin daemon. When one bin file fills, the daemon switches to the other bin file and invokes the processing command specified in */etc/security/audit/bincmds* to empty the full cache file.

When that operation is complete, auditbin notifies the kernel that it is permitted to reuse the cache file. This mechanism of switching and emptying audit bins continues so long as auditing is enabled. The size a bin file may reach before being considered full is defined in */etc/security/audit/config*.

A bin file begins with a header. The tail is written when the audit bin is switched or when auditing is shut down.

6.2.3.3.2 Backend Audit Processors

There are two backend utilities available for use: *auditcat* and *auditselect*. The backend processor writes the raw audit records to the system audit trail or to a specified file after manipulating them.

Bin mode auditing makes use of *auditcat* and *auditselect*. The result of *auditcat* or *auditselect* can be directed to a file for permanent audit storage.

6.2.3.3.2.1 *auditcat*

The *auditcat* command reads audit records from standard input or from a file, and processes the records and sends them to standard output or to the system audit trail.

6.2.3.3.2.2 *auditselect*

The *auditselect* command can be used as both a pre-processing and post-processing tool. As a pre-processing tool, the *auditselect* command serves the same purpose as *auditcat*, but adds the ability to specify conditions that an audit record must meet. This allows a system to be configured to save audit records that relate to login in one file, and audit records that relate to file access in a separate file.

Auditselect utilizes an expression to apply against the current audit record. The expression consists of one or more terms joined by the logical operators && (and), || (or) and ! (not). Each term in the expression describes a field, a relational operator and a value.

The following is an example expression to select all the *FILE_Open* events:

```
event==FILE_Open
```

The event field identifies that *auditselect* should query based on the name of the event. The operator is equal and the name of the event is *FILE_Open*.

Table 10: Available Fields. The available fields are used to build expressions with *auditselect*.

Field	Definition
event	Name of the audit event
command	Name of the command that generated the audit event
result	Status of the audit event. The value of the result field must be one of the following: OK, FAIL, FAIL_PRIV, FAIL_AUTH, FAIL_ACCESS, FAIL_DAC. FAIL matches all other error codes.
login	ID of the login user of the process that generated the audit event.
real	ID of the real user of the process that generated the audit event.
pid	ID of the process that generated the audit event.
ppid	ID of the parent of the process that generated the audit event.
tid	ID of the kernel thread that generated the event.
time	Time of day the audit event was generated.
date	Date the audit event was generated.
host	Hostname of the machine that generated the record. The reserved name UNKNOWN can be used to match any machines that are not listed in the /etc/security/audit/hosts file.
priv	Privilege name
sl	Sensitivity label name
tl	Integrity label name
role	Role name

Auditselect allows to selectively extract individual audit records.

6.2.3.4 Audit Review (AU.4)

Three different commands exist for the review of audit records in the system: *auditpr, auditmerge,* and *auditselect*.

The *auditpr* command formats audit records to a display device or to a printer for review. The *auditpr* command also allows the administrator to select which of the fields to include in the output as well as the order to display them. The fields available for inclusion with the output of the *auditpr* command are:

- Audit event
- User's login name
- Event status
- Time the records was written
- Command name
- Real user name
- Process ID
- Parent process ID
- Kernel thread ID
- Host name that generated the audit record
- Role names or IDs of the audited process
- Effective privilege
- Effective sensitivity label (SL)
- Effective integrity label (TL)
- WPAR name

The default values are the audit event, the user's login name, the audit status, the kernel thread ID and the command name *auditselect* allows the administrator to build an expression that will be applied to the stored audit records. The details of the *auditselect* command are listed in section 6.2.3.3, Audit Record Processing.

The *auditmerge* command provides a method of combining multiple audit trail files into a single audit trail file. These multiple files can come from different hosts, providing a centralized audit analysis function. As the two files are processed, the record with the oldest time stamp that still remains is written into the audit trail. This process continues until there are no more audit records to process. The Security Guide directs the system administrator to transfer the audit files to be merged to the same host.

The commands *auditpr* and *auditmerge* allow an authorized administrator to read the audit records and convert them to human readable formats.

6.2.3.5 Audit File Protection (AU.5)

The audit trail files, configuration files, bin files, and the */audit* directory are protected on each system using normal file system permissions. Each audit file grants read access to the root user and the audit group, and write access to only the root user. The AUDIT authorization is needed for the audit programs that are used to read these files. The AIX Security Guide instructs the administrator that if the cached and permanent audit trails are kept somewhere other than in the */audit* directory, then the alternate directory must be protected from access by non-root users.

6.2.3.6 Audit Record Loss Prevention (AU.6)

Bin mode auditing is susceptible to the exhaustion of disk space available to the */audit* directory or to a system crash. In the case of a system crash, all data in physical memory is lost, including any audit records that had not yet been flushed to disk. The audit subsystem enforces a 32K byte limit on the size of an individual audit record, and only one audit record can be in transit between a thread and the kernel at any given time. When the system is no longer able to write audit records to the audit bins either the system will stop in "panic" mode or a counter will show the number of audit records lost. This counter is written in an audit record the next time the system is able to produce audit records again. If the TOE stops in case it is unable to write audit records or if the TOE just counts the number of audit record lost is a configuration parameter that can be set by the System Administrator.

The AIX Security Guide includes instructions to the administrator to backup all files, including audit data, on a regular basis to avoid the loss of data due to hard disk failures.

6.2.3.6.1 Audit Record Loss Prevention for Bin Mode Auditing

AIX allows an administrator to define a threshold value for the amount of free space in the file system holding the audit files. When the amount of free space in this file system is below this defined threshold value this fact will be reported to an administrator. This allows the administrator to take the appropriate actions to prevent the system to enter the panic mode due to the inability to write events to the audit trail.

AIX provides a panic mode for use with bin mode auditing. The panic mode option halts the host when the current audit bin stops accepting additional records, preventing the unnecessary loss of audit records. This only occurs with the exhaustion of disk space. If a host halts because it cannot collect audit records, the other hosts in the distributed system are not affected, unless the host is acting as the administrative master. The AIX Security Guide contains instructions for enabling panic mode, as panic mode is not enabled by default.

The result of halting the system because panic mode was invoked would be the loss of any audit data presently in the host’s memory that had not been written to disk. In addition, audit records could be lost for operations that were underway but had not yet completed generating audit records. This minimizes the damage caused by the lack of disk space, because only the audit records that are currently in memory are lost.

A recovery process for audit bins exists in the evaluated configuration. If either of the bin files is not empty when audit is started, the *auditbin* daemon executes the bin mode post-processing command to process the bins.

The amount of audit data that can be lost in bin mode is minimized by the use of the binsize and bytethreshold parameters in the */etc/security/audit/config* file. The binsize parameter sets the maximum size a bin may reach before the auditbin daemon switches to the other bin, and executes the bin mode post-processing command. The bytethreshold parameter sets the amount of data in bytes that is written to a bin before a synchronous update is performed. The AIX Security Guide states that the binsize and bytethreshold parameters should be set to 64K bytes each to minimize audit data loss. The amount of audit data that could be lost due to a failure in bin mode is the combination of these two files, or 128K bytes.

6.2.3.7 Audit System Privileges (AU.7)

The enforcement of the TOE’s auditing policies is supported, in addition to the general access control policies (DAC, MAC (LSPP mode only), MIC (LSPP mode only)), by the following privileges, allowing a process to

1. record/add audit records (PV_AU_ADD);
2. turn on/off auditing or change audit system configuration (PV_AU_ADMIN);
3. query the status of the audit system or the audit mask of a process (PV_AU_PROC);
4. read a file marked as an audit file (PV_AU_READ);
5. write or delete a file marked as an audit file, or mark a file as an audit file (PV_AU_WRITE);
6. obtain all privileges listed in 1 through 5 (PV_AU).

See TP.9 for a description of file security flags that influence the behavior of the audit subsystem.

6.2.4 Discretionary Access Control (DA)

This section outlines the general DAC policy in AIX as implemented for resources. A subset of these resources are file system objects where access is controlled by one of two policies (i.e., a file system object can only have one policy associated with it at a time):

- AIXC policy – the AIX classic access control policy including the Encrypted File System (EFS)
- NFSv4 policy – the Network File System version 4 (NFSv4) access control policy

The AIXC policy uses permission bits and, optionally, extended permissions and encryption. The extended permissions are in the form of an access control list (ACL) where each entry in the ACL can define the permissions of a specific user or group. The encryption is in the form AES encrypted files using the EFS file system (an extended version of the JFS2 file system). This is described in more detail in the following sections.

The NFSv4 policy uses fine grained permissions. The fine grained permissions are in the form of an ACL where each entry in the ACL can enable a number of fine grained permissions for a user, group, or for everyone. This is described in more detail in the following sections.

Permission bits are the standard UNIX DAC mechanism and are used on all AIX file system named objects. Individual bits are used to indicate permission for read, write, and execute access for the object's owner, the object's group, and all other users (i.e. world). The extended permission and fine grained permission mechanisms are supported only for file system objects and provide a finer level of granularity than do permission bits.

The policies for all resources are based on user identity (and in some cases on group membership associated with the user identity). To allow for enforcement of the DAC policy, all users must be identified and their identities authenticated.

Details of the specific DAC policy applied to each type of resource are covered in the section "Discretionary Access Control: File System Objects" and the section "Discretionary Access Control: IPC Objects".

The general policy enforced is that subjects (i.e., processes) are allowed only the accesses specified by the class-specific policies. Further, the ability to propagate access permissions is limited to those subjects who have that permission, as determined by the class-specific policies.

The privilege PV_DAC will also grant full access regardless of the setting of permission bits or ACLs. A subset of PV_DAC can be used for explicit overrides, for example PV_DAC_W to override DAC restrictions on any file.

Finally, in a root enabled mode system, a subject with an effective UID of 0 is exempt from all restrictions and can perform any action desired, including the execution of files for which at least one exec DAC bit is set.

DAC provides the mechanism that allows users to specify and control access to objects that they own. DAC attributes are assigned to objects at creation time and remain in effect until the object is destroyed or the object attributes are changed. DAC attributes exist for, and are particular to, each type of object on AIX.

AIXC Permission-bit and Extended Permission Policy

A subject whose effective UID matches the file owner ID can change the file attributes, the base permissions, and the extended permissions. Changes to the file group are restricted to the owner.

For new files, the group identifier must either be the current effective group identifier or one of the group identifiers in the concurrent group set. In addition when using a root enabled mode system, a subject whose effective UID is 0 can make any desired changes to the file attributes, the base permissions, the extended permissions, and owning user of the file.

Encryption

The Encrypted File System (EFS) provides for the ability to encrypt and decrypt files using AES encryption. A subject who doesn't know or have access to the key to decrypt the file cannot view the contents of the file, including administrative users who might be able to override the permission bits of a file.

NFSv4 Policy

A subject whose effective UID matches the file owner ID can change the file attributes, the base permissions, and the fine grained permissions. (If an object has an NFSv4 ACL, the base permissions (excluding the setuid, setgid,and save text bits) are ignored when making access decisions, but they are set to approximate the value of the ACL.) Additional rules regarding who can manage NFSv4 ACLs and object attributes are provided in section 6.2.4.3.1.2.2.

For new files, the group identifier must either be the current effective group identifier or one of the group identifiers in the concurrent group set. In a root enabled mode system, if the subject's effective UID is 0, the group identifier can be any chosen value.

The NFSv4 policy allows for file system objects to inherit ACL entries from the parent directory's ACL. Subdirectories can inherit different entries than other file system objects. The ability to propagate the ACL entries to subdirectories can be limited to just the subdirectories within the parent directory.

6.2.4.1 Permission Bits (DA.1)

AIX uses standard UNIX permission bits to provide one form of DAC for file system named objects. There are three sets of three bits that define access for three categories of users: the owning user, users in the owning group, and other users. The three bits in each set indicate the access permissions granted to each user category: one bit for read (r), one for write (w) and one for execute (x). Each subject's access to an object is defined by some combination of these bits:

- rwx symbolizing read/write/execute
- r-x symbolizing read/execute
- r-- symbolizing read

--- symbolizing null When a process attempts to reference an object protected only by permission bits, the access is determined as follows:

- Effective UID = Object's owning UID and the owning user permission bits allow the type of access requested. Access is granted with no further checks.
- Effective GID, or any supplementary groups of the process = Object's owning GID, and the owning group permission bits allow the type of access requested. Access is granted with no further checks.
- If the process is neither the owner nor a member of an appropriate group and the permission bits for world allow the type of access requested, then the subject is permitted access.
- In root enabled mode, if the process is the root identity, then the subject is permitted read and write access.
- In root enabled mode, if the process is the root identity, and the attempted access is an execution of the object, the access is granted only if at least one of the execution bits is set.
- If none of the conditions above are satisfied and the process does not possess the needed PV_DAC privilege or the appropriate subset of PV_DAC then the access attempt is denied.

As a special case that has been modeled as part of the DAC Policy in this Security Target, a read-only bit for file systems can be set upon mount time, yielding the denial of every write request for the file system.

6.2.4.2 Extended Permissions (DA.2)

6.2.4.2.1 AIXC Extended Permissions

The extended permissions consist of an essentially unlimited number of additional permissions and restrictions for specific users and groups. Each entry in the extended permissions list consists of three parts: an entry type, a set of permissions, and an identity list.

- The entry type is the value permit, deny, or specify (indicating that the entry indicates a set of permissions to be allowed as supplemental to the listed identity(ies), denied to the listed identity(ies), or that the permissions permitted and the complementary set denied to the listed identity(ies) respectively).
- The permission set is zero or more of the permissions read, write, and execute.
- The identity list is one or more values specifying users and/or groups. The entry is applied if the process' effective UID, effective GID, and supplemental groups match all values in the list. The term "match" means that for each value in the identity list, if the value is for a UID, that the specified UID is the same as the process' effective UID, and if the value is for a GID, that the specified GID is either the same as the process' effective GID or the specified GID is included in the process' list of supplemental GIDs.

There is no explicit ordering of entries within the extended permissions. To determine access rights, the kernel takes into account all entries that match the UID or GID of the process. For each entry, the permit and specify bits are added to a permissions list and the deny and bitwise negation of the specify are added to a restrictions list. The restrictions are bitwise removed from the permissions and the resulting list is used in the access determination.

The maximum size for the extended permissions is one memory page (4096 bytes). The entries are variable length. Each entry takes a minimum of 12 bytes (two for the length of the entry, two for the permission type and permissions allowed, two for the number of identity entries, two for the type of identity entry, and four for each UID/GID). As a result, there can be over 300 entries in an extended permissions list, which is in practice unlimited.

Collectively, the file attributes, base permissions, extended permissions, and extended attributes are known as the file AIXC Access Control List (ACL). AIXC ACLs have a textual representation (used with commands such as *aclget*) and binary representations (for storage in the file system).

When a process attempts to reference an object protected by an ACL, it does so through a system call (e.g., open, exec). If the object has been assigned an ACL, access is determined as according to the algorithm below:

> A subject must have search permission for every element of the pathname and the requested access for the object. A subject has a specific type access to an object if the type of access is within the union of all permission rights (grant entries) defined in the access control list of the object for the subject and is not within the logical union of all restrictions (deny entries) defined in the access control list of the object for the subject. If no entry in the extended permissions either allows or denies access, the access right defined in the permission bits apply. In any other case access is denied.

6.2.4.2.2 NFSv4 Fine Grained Permissions

An NFSv4 ACL consists of a list of entries with the following fields:

- Type Field – This field contains one of the following values:
    - ALLOW – Grants the subject, specified in the Who field, the permission(s) specified in the Mask field.
    - DENY – Denies the subject, specified in the Who field, the permission(s) specified in the Mask field.
- Mask Field – This field contains one or more of the following fine grained permission values:
    - READ_DATA / LIST_DIRECTORY – Read the data from a non-directory object or list the objects in a directory.
    - WRITE_DATA / ADD_FILE – Write data into a non-directory object or add a non-directory object to a directory.
    - APPEND_DATA / ADD_SUBDIRECTORY – Append data into a non-directory object or add a subdirectory to a directory.
    - READ_NAMED_ATTRS – Read the named attributes of an object. (There are no named attributes.)
    - WRITE_NAMED_ATTRS – Write the named attributes of an object. (There are no named attributes.)
    - EXECUTE – Execute a file or traverse/search a directory.
    - DELETE_CHILD – Delete a file or directory within a directory. (Applies to directories.)
    - READ_ATTRIBUTES – Read the basic (non-ACL) attributes of a file.
    - WRITE_ATTRIBUTES – Change the times associated with a file or directory.
    - DELETE – Delete a file or directory.
    - READ_ACL – Read the ACL.
    - WRITE_ACL – Write the ACL.
    - WRITE_OWNER – Change the owner and group.
    - SYNCHRONIZE – Synchronize access. (Exists for compatibility with other NFSv4 clients, but has no implemented function.)
- Flags Field – This field defines the inheritance capabilities of directory ACLs and indicates whether the Who field contains a group or not. The field contains zero or more of the following flags:
    - FILE_INHERIT – Specifies that, in this directory, newly created non-directory objects will inherit this entry.
    - DIRECTORY_INHERIT – Specifies that, in this directory, newly created subdirectories will inherit this entry.
    - NO_PROPAGATE_INHERIT – Specifies that, in this directory, newly created subdirectories will inherit this entry, but these subdirectories will not pass this entry to their newly created subdirectories.

    - INHERIT_ONLY – Specifies that this entry does not apply to this directory, only to the newly created objects that inherit this entry.
    - IDENTIFIER_GROUP – Specifies that the Who field represents a group; otherwise, the Who field represents a user or a special Who value.
- Who Field – This field contains one of the following values:
    - User – Specifies the user that this entry applies to.
    - Group – Specifies the group that this entry applies to.
    - Special – This attribute can be one of the following values:
        - OWNER@ - Specifies that this entry applies to the owner of the object
        - GROUP@ - Specifies that this entry applies to the owning group of the object.
        - EVERYONE@ - Specifies that this entry applies to all users of the system *including* the owner and group.

If the ACL is empty, access requires the PV_DAC_R privilege or, in the case of root enabled mode, an effective UID of 0.

The owner of an object implicitly has the following mask values regardless of what the ACL may or may not contain:

- READ_ACL
- WRITE_ACL
- READ_ATTRIBUTES
- WRITE_ATTRIBUTES

APPEND_DATA is implemented as WRITE_DATA. Effectively, there's no functional distinction between WRITE_DATA and APPEND_DATA. Both values must be set or unset in unison which is enforced by the TOE.

Object ownership can be modified through the use of WRITE_OWNER. Section 6.2.4.3.1.2.2 details how WRITE_OWNER works. When the owner is changed, the setuid bit is turned off. When the group is changed, the setgid bit is turned off.

The inheritance flags only have meaning in a directory's ACL and only apply to objects that are created in the directory after the inheritance flags have been set (i.e., existing objects are not affected by inheritance changes to the parent directory's ACL).

The entries in an NFSv4 ACL are order dependent. To determine if the requested access is allowed, each entry is processed in order. Only entries which have a Who field that matches the effective UID, if a user is specified in the entry, or effective GID, if a group is specified in the entry, of the subject are considered. Each entry is processed until all of the bits of the requester's access have been ALLOWED. Once an access type has been ALLOWED by an entry, it is no longer considered in the processing of later entries. If a DENY entry is encountered where the requester's access for that mask value is necessary and undetermined, the request is denied. If the evaluation reaches the end of the ACL, the request is denied.

The maximum supported ACL size is 64KB. Each entry in an ACL is of variable length and 64KB is the only limit on an entry.

6.2.4.3 Discretionary Access Control: File System Objects (DA.3)

The Discretionary Access Control (DAC) policy is described above. This section describes the details of DAC policies as they apply to file system objects.

6.2.4.3.1 Common File System Access Control

This section describes the common DAC policy applied to file system objects, including policies for object contents and attributes.

6.2.4.3.1.1 *DAC Contents Policy*

6.2.4.3.1.1.1 AIXC Permission-bit and Extended Permissions Contents Policy

The permission-bit and ACL DAC policy determines the effective access that a process may have to the contents of a file system object: some combination of read(r), write (w), and execute (x). In general, read access permits the object's contents to be read by a process, and write permits them to be written; execute is interpreted differently for different object types. Some object types (unnamed pipes, symbolic links) do not use the permission bits at all.

6.2.4.3.1.1.2 NFSv4 Contents Policy

The NFSv4 policy determines the effective access that a process may have to the contents of a file system object. How this policy works is described in DA.2. Some object types (unnamed pipes, symbolic links) do not use the NFSv4 policy at all. The permission bits (excluding the setuid, setgid, and save text bits), specifically the user/group/other bits, are ignored when making access control decisions if an NFSv4 ACL exists on the object.

6.2.4.3.1.2 *DAC Attributes Policy*

6.2.4.3.1.2.1 AIXC Permission-bit and Extended Permissions Contents Policy

In general, a process must be the object's owner, or have privilege, to change the objects attributes, and there are no DAC restrictions on viewing the attributes, so any process may view them. However, the following are exceptions to the rule:

- The permission bits and ACL (permission bits, extended permissions and attributes) of an object may be changed by an owner or by a subject having the PV_DAC privilege or the appropriate subset of PV_DAC or, in root enabled mode, by the root identity.
- The owning group ID of an object may be changed by an owner, but only to a group of which the process is currently a member, unless it has PV_DAC or the appropriate subset of PV_DAC or, in root enabled mode, by the root identity.
- The owning user ID of an object may only be changed by an administrator with PV_DAC or the appropriate subset of PV_DAC or, in root enabled mode, by the root identity.

6.2.4.3.1.2.2 NFSv4 Contents Policy

The NFSv4 policy provides control over who can read and write the attributes of an object. A subject with the PV_DAC privilege or, in root enabled mode, effective UID 0 can always override the NFSv4 policy. The object owner can allow others to read and write the attributes of an object using the READ_ATTRIBUTES, WRITE_ATTRIBUTES, READ_NAMED_ATTRS, and WRITE_NAME_ATTRS attributes of the ACL mask. The owner can control who can read and write the ACL using the READ_ACL and WRITE_ACL attributes of the ACL mask. The object owner always has READ_ATTRIBUTES, WRITE_ATTRIBUTES, READ_ACL, and WRITE_ACL access. The owner can also allow others to change the owner and group of the object using the WRITE_OWNER attribute. An object owner cannot change the owner or group of the object by default, but the owner can add a WRITE_OWNER entry to the ACL specifying themselves, or the object could inherit an ACL entry which specifies a WRITE_OWNER entry with a Who value of OWNER@.

There are some exceptions to the rules in root enabled mode.

- If the object is owned by UID 0, only UID 0 can change the owner, but the group can still be changed by a subject with WRITE_OWNER.
- If the object has a non-UID 0 owner, a non-UID 0 user with WRITE_OWNER can only change the owner to himself.
- The group can be changed to any group in the subject's concurrent group set with the exception that it can never be changed to GID 0 or GID 7 even if these two groups are in the concurrent group set of the subject.

6.2.4.3.1.3 *DAC Defaults*

6.2.4.3.1.3.1 AIXC Permission-bit and Extended Permissions Defaults

The default access control on newly created FSOs is determined by the permissions associated with the directory where the FSO was created, the effective user ID, group ID, and umask value of the process that created the FSO, and the specific permissions requested by the program creating the FSO.

- The owning user of a newly created FSO will be the effective UID of the creating process.

- If the setgid bit is set on the containing directory, then the owning group of a newly created FSO will be the owning group of the containing directory. If the setgid bit is not set on the containing directory, then the owning group of the newly created FSO will be the effective GID of the creating process.
- The initial access permissions on the FSO are those specified by the creating process bit- wise ANDed with the one's complement of the umask value. For example, if a program specified initial permissions of 0664 (read/write for owner, read/write for group, and read for world) but the umask value were set to 0027 (prevent write for group or world, prevent all permissions for world), then the initial file permissions would be set to 0640 (or 0644 bit-and 0750).
- There are initially no extended permissions associated with an FSO. Extended permissions can be set by applications or by users using AIX commands.

Base and extended access permissions can be changed by any process with an effective UID equal to the owning UID of the FSO, providing that the effective UID has at least the execute permission to the containing directory. Note that since a file may have multiple hard links, the process can use any of the containing directories (e.g., if there is any directory containing a link to the file, then that path could be used as a means to get to the file and change its permissions).

6.2.4.3.1.3.2 NFSv4 Defaults

If the parent directory does not have any NFSv4 inheritance entries applicable to the FSO being created, then the FSO will be created using the AIXC defaults mentioned above. Otherwise, the parent directory's inheritance entries will be copied into and become the ACL of the newly created FSO as per the rules of NFSv4 inheritance. NFSv4 inheritance is described in DA.2.

- The owning user of a newly created FSO will be the effective UID of the creating process.
- If the setgid bit is set on the containing directory, then the owning group of a newly created FSO will be the owning group of the containing directory. If the setgid bit is not set on the containing directory, then the owning group of the newly created FSO will be the effective GID of the creating process.

The permission bits are set on the object to approximate the values contained in the ACL.

6.2.4.3.1.4 *DAC Revocation on File System Objects*

With the exception of NFS objects, file system objects (FSOs) access checks are performed when the FSO is initially opened, and are not checked on each subsequent access. Changes to access controls (e.g., revocation) are effective with the next attempt to open the FSO.

For NFS objects, access is checked for each operation. A change to the access rights for an NFS FSO take effect as of the next NFS request.

In cases where the administrator determines that immediate revocation of access to an FSO is required, the administrator can reboot the computer, resulting in a close on the FSO and forcing an open of the FSO on system reboot. This method is described in the AIX Security Guide.

Applications that want to revoke tty access of other processes can use the revoke() and frevoke() system calls to revoke all current access of other processes, forcing them to reopen the file and to undergo the associated access checks again.

6.2.4.3.2 DAC: Ordinary File

Ordinary files support the concept of execution. Execute access is required to execute the file as a program or script. When an executable file has the set-user-ID or set-group-ID flags set, and the file owner or file group is not the same as the process's current effective user-ID or group-ID the executing program changes the process's security attributes. Otherwise the attributes remain unchanged. AIX doesn't support set-UID or set-GID scripts with this mechanism.

Note that the privcmds table can override the set-user-ID and/or set-group-ID flags. See section 6.2.7.2 for more details.

6.2.4.3.3 DAC: Directory

The execute access for directories governs the ability to traverse the directory as part of a pathname. A process must have execute access in order to traverse the directory during pathname resolution.

Directories may not be written directly, but only by creating, renaming, and removing (unlinking) objects within them. These operations are considered writes for the purpose of the DAC policy.

6.2.4.3.4 DAC: UNIX Domain Socket Special File

UNIX domain socket files are treated as files in the AIX file system from the perspective of access control, with the exception that using the bind or connect system calls requires that the calling process must have both read and write access to the socket file.

UNIX domain sockets exist in the file system name space. The socket files are supported by both the AIXC and NFSv4 policies.

UNIX domain sockets consist of a socket special file (managed by the File System) and a corresponding socket structure (managed by IPC). The VFS controls access to the socket based upon the caller's rights to the socket special file.

6.2.4.3.5 DAC: Named Pipes

Named pipes are treated identically to any other file in the AIX file system from the perspective of access control. Therefore both AIXC and NFSv4 policies are supported by named pipes. For this reason named pipes are listed as file system objects (although they are used for interprocess communication). Note that named pipes follow the rules for IPC objects, if no ACLs are used (which probably is the normal case they are used).

6.2.4.3.6 DAC: Device Special File

The access control scheme described for FSOs is used for protection of character and block device special files. The DAC settings on most device special files are configured to allow read and write access by the root user, and read access by privileged groups. With the exception of terminal and pseudo-terminal devices and a few special cases (e.g., /dev/null and /dev/tty), devices are configured to be not accessible to normal users.

6.2.4.3.7 DAC: Special Cases for NFS File Systems

An NFS filesystem may contain any of the supported object types of the underlying filesystem. That includes device special files. Non-regular files or directories which occur on NFS filesystems are treated similar to objects defined on the local filesystem -- a device special file on an NFS mounted filesystem will reference the underlying device on the local system. It would not reference the device on the remote system.

DAC checks by the NFS server for file contents permit a subject with the same effective owning user ID as the file to have access to the contents regardless of the DAC attributes. This is used to support the standard UNIX semantics for access to open files, because such access is not re-validated when a file's DAC attributes change. This special case relies on the property that, ordinarily, only a file's owner changes its DAC while the file is open, and it is thus sufficient to handle the owner specially.

DAC changes do have immediate effect for users other than the owner, unlike local files: if an NFS-accessed file's DAC is changed to deny access, any subsequent read or write operation to an open file will fail if the operation would no longer be permitted by the new DAC attributes.

However, this can never grant additional access, because the client would have checked the access when the file was opened and not permitted more access than the DAC attributes allowed at open time.

The filesystem maintains a "handle" on the credentials which were used at the time an NFS file was opened. It is those credentials which are used to reference files via NFS, not the current process credentials which might be modified by setuid().

6.2.4.4 Discretionary Access Control: IPC Objects (DA.4)

In general, WPARs limit the scope of IPC objects to the scope of the WPAR. Thus, a WPAR can only see the IPC objects within itself.

6.2.4.4.1 DAC: Shared Memory

For shared memory segment objects (henceforth SMSs), access checks are performed when the SMS is initially attached, and are not checked on each subsequent access. Changes to access controls (e.g., revocation) are effective with the next attempt to attach to the SMS.

In cases where the administrator determines that immediate revocation of access to a SMS is required, the administrator can reboot the computer, thus destroying the SMS and all access to it.

This method is described in the AIX Security Guide. Since a SMS exists only within a single host in the distributed system, rebooting the particular host where the SMS is present is sufficient to revoke all access to that SMS.

If a process requests deletion of a SMS, it is not deleted until the last process that is attached to the SMS detaches itself (or equivalently, the last process attached to the SMS terminates).

However, once a SMS has been marked as deleted, additional processes cannot attach to the SMS and it cannot be undeleted.

The default access control on newly created SMSs is determined by the effective user ID and group ID of the process that created the SMS and the specific permissions requested by the process creating the SMS.

- The owning user and creating user of a newly created SMS will be the effective UID of the creating process.
- The owning group and creating group of a newly created SMS will be the effective GID of the creating process.
- The creating process must specify the initial access permissions on the SMS, or they are set to null and the object is inaccessible until the owner sets them.
- SMSs do not have extended permissions.
- SMSs do not support NFSv4 ACLs.

Access permissions can be changed by any process with an effective UID equal to the owning UID or creating UID of the SMS. In root enabled mode, access permissions can also be changed by any process with an effective UID of 0, also known as running with the root identity.

6.2.4.4.2 DAC: Message Queues

For message queues, access checks are performed for each access request (e.g., to send or receive a message in the queue). Changes to access controls (e.g., revocation) are effective upon the next request for access. That is, the change affects all future send and receive operations, except if a process has already made a request for the message queue and is waiting for its availability (e.g., a process is waiting to receive a message), in which case the access change is not effective for that process until the next request.

If a process requests deletion of a message queue, it is not deleted until the last process that is waiting for the message queue receives its message (or equivalently, the last process waiting for a message in the queue terminates). However, once a message queue has been marked as deleted, additional processes cannot perform messaging operations and it cannot be undeleted.

The default access control on newly created message queues is determined by the effective user ID and group ID of the process that created the message queue and the specific permissions requested by the process creating the message queue.

- The owning user and creating user of a newly created message queue will be the effective UID of the creating process.
- The owning group and creating group of a newly created message queue will be the effective GID of the creating process.
- The initial access permissions on the message queue must be specified by the creating process, or they are set to null and the object is inaccessible until the owner sets them.
- Message queues do not have extended permissions.
- Message queues do not support NFSv4 ACLs.

Access permissions can be changed by any process with an effective UID equal to the owning UID or creating UID of the message queue. Access permissions can also be changed by any process having the appropriate privilege or, in root enabled mode, with an effective UID of 0.

6.2.4.4.3 DAC: Semaphores

For System V (SysV) semaphores, access checks are performed for each access request (e.g., to lock or unlock the semaphore). Changes to access controls (e.g., revocation) are effective upon the next request for access. That is, the change affects all future SysV semaphore operations, except if a process has already made a request for the SysV semaphore and is waiting for its availability, in which case the access change is not effective for that process until the next request.

In cases where the administrator determines that immediate revocation of access to a SysV semaphore is required, the administrator can reboot the computer, thus destroying the semaphore and any processes waiting for it. This method is described in the AIX Security Guide. Since a SysV semaphore exists only within a single host in the distributed system, rebooting the particular host where the semaphore is present is sufficient to revoke all access to that semaphore.

If a process requests deletion of a SysV semaphore, it is not deleted until the last process that is waiting for the semaphore obtains its lock (or equivalently, the last process waiting for the semaphore terminates). However, once a SysV semaphore has been marked as deleted, additional processes cannot perform semaphore operations and it cannot be undeleted.

The default access control on newly created SysV semaphores is determined by the effective user ID and group ID of the process that created the semaphore and the specific permissions requested by the process creating the semaphore.

- The owning user and creating user of a newly created SysV semaphore will be the effective UID of the creating process.
- The owning group and creating group of a newly created SysV semaphore will be the effective GID of the creating process.
- The initial access permissions on the SysV semaphore must be specified by the creating process, or they are set to null and the object is inaccessible until the owner sets them.
- SysV semaphores do not have extended permissions.
- SysV semaphores do not support NFSv4 ACLs.

Access permissions can be changed by any process with an effective UID equal to the owning UID or creating UID of the semaphore, and can be overridden by privileges or, in root enabled mode, by any process with an effective UID of 0.

No security claims are made for non-SysV semaphores.

Note: In addition to the regular IPC semaphores, AIX also supports memory mapped semaphores that are accessible within the memory mapped address space of processes that can share the mapped address space. There is no explict access control on these semaphores (handled by the msem_init, msem_lock, msem_unlock, msem_remove, msleep and mwakeup subroutines). The mmap routines map files into shared memory and all access control is performed via the DAC protection mechanisms of the mapped files or memory.

6.2.4.5 Discretionary Access Control: VIOS (DA.5)

VIOS resides in a separate LPAR partition and provides basic discretionary access control between VIOS SCSI device drivers acting on behalf of LPAR partitions and SCSI-based logical volumes and physical volumes through mappings. An LPAR partition (via a VIOS SCSI device driver) may be mapped to 0 or more logical and physical volumes, but a volume can only be mapped to at most one LPAR partition. This mapping limits an LPAR partition to only the volumes assigned to it.

VIOS also controls the mapping of VIOS Ethernet adapter device drivers to VIOS Ethernet device drivers acting on behalf of groups of LPAR partitions sharing a virtual network. In the evaluated configuration, only a one-to-one mapping of an Ethernet adapter device driver to an Ethernet device driver acting on behalf of a group of LPAR partitions is allowed. The one-to-one mapping is configured by the administrator and enforced by the device drivers. Also, the Ethernet packets must not be tagged with a VLAN tag in the evaluated configuration. This mechanism can be used to limit which LPAR partitions see certain Ethernet packets.

VIOS is restricted to administrator access only. VIOS allows all administrative roles except the Service Representative roles to manage the access control mechanisms previously mentioned.

6.2.4.6 Discretionary Access Control: TCP Connections (DA.6)

This section applies to CAPP mode only.

TCP based services can be protected with ACLs as well. By specifying port, host/network,user combinations, ports can be restricted to specific hosts and/or users. For example specifying port 6000, machine colorado and user joe, only this user coming from machine colorado will be able to connect to the X server. The remote hosts use TCP AH headers to send the information about the user together with the connection request. AIX checks */etc/security/acl* for permitted clients.

With the DACinet feature of AIX the concept of privileged ports (ports that can only be opened by the superuser, typically all ports below 1024) is extended so that any port now can be a privileged port. A bitmap of privileged ports is defined to hold information on whether a port is privileged. A system administrator can modify this bitmap.

6.2.4.7 Discretionary Access Control: Encrypted Files (DA.7)

The JFS2 file system supports the Encrypted File System (EFS). EFS allows individual files to be selectively encrypted and it allows for the files in a directory to be encrypted by default. EFS can be enabled on an existing file system, but files on that system will not be automatically encrypted by the enablement.

For each encrypted file, a random symmetric file encryption key (called the file key) is generated to encrypt the file. The key size of the file key matches the algorithm used (e.g., a random 256-bit key is generated for AES 256-bit). The symmetric encryption mechanisms supported by the evaluated configuration are:

- AES 128-bit CBC (default) and ECB
- AES 192-bit CBC and ECB
- AES 256-bit CBC and ECB

The file key is than wrapped with the public key of the user creating the file and stored as part of the file’s meta data in the file system. For other users and groups to have access to the file’s content, the file key must be wrapped with their public keys too and stored in the file’s meta data.

When EFS is enabled, users and groups must have asymmetric (public/private) keys in the form of X.509v3 certificates. These keys are stored in individual PKCS #12 keystores for both users and groups. The user and group keystores are restricted to root read/write access only using file system DAC. The private keys are protected within the keystore using AES 256-bit encryption and an encryption key. For user keystores, the keystore encryption key is generated from a user supplied password (e.g. the user’s login password). For group keystores, the password is generated by AIX.

When the user’s keystore is opened using the correct password (typically at login time), the keys are loaded and stored in the kernel to improve the performance of EFS. The kernel protects the private keys from unauthorized access while the keys reside in the kernel. The keys are scrambled (i.e. not stored in the clear) while in the kernel.

The certificates can be generated using one of the following algorithms:

- RSA 1024-bit (default)
- RSA 2048-bit
- RSA 4096-bit

The CLiC module is used to implement the cryptographic functions of EFS.

6.2.5 Workload Partitions (WP)

6.2.5.1 WPAR Information Control (WP.1)

A System WPAR works like an independent AIX installation. The characteristics of a System WPAR are:

- Administration: Internally, each System WPAR has its own administrator(s) which can add and delete users, groups, modify RBAC settings, modify MAC and MIC labels (LSPP mode only), etc. of the System WPAR. These changes are contained within the System WPAR and, thus, are independent of the Global environment.
- Authentication: The authentication subsystem for a System WPAR can be different than the Global environment. Users log into the System WPAR using remote login commands like *rlogin* and *telnet* and by using the *clogin* command.
- Process space: Processes in a System WPAR can only see other processes within the same System WPAR. Thus, a System WPAR’s processes can only create IPC objects and send signals to processes within its System WPAR. Processes in the Global environment can signal processes in a System WPAR. The /dev/kmem device does not exist in a System WPAR.

- Network: Each System WPAR has its own IP address(es). This allows for network communications between WPARs. Raw socket access is not allowed without the WPAR being assigned the privilege to access raw sockets.
- File system space: System WPAR file systems can be isolated from other System WPARs. This is accomplished through the use of the *chroot()* system call and the *mount()* system call. The Global environment administrator controls the amount of file system isolation. When the System WPAR is started, the Global environment effectively *chroot*'s and/or *mount*'s the System WPAR to its file system space. Like any AIX system, file systems can be shared among WPARs.
- Kernel: Only one kernel image exists which performs the virtualization and, thus, is shared by all WPARs and the Global environment in an LPAR.
- Libraries & Kernel extensions: In general, only one instance of a library and kernel extension exists and is shared by all WPARs and the Global environment. It is possible for unique libraries to be loaded by a System WPAR and, under controlled conditions, for a kernel extension to be loaded by a System WPAR.
- Auditing: Each System WPAR has its own audit trail and has control over its own audit trail (e.g. enable/disable auditing within the System WPAR). The Global environment can independently audit the System WPAR (regardless of the state of the System WPAR audit subsystem) placing the audit records into the Global environment's audit trail.
- Authority: The actual power that a System WPAR can exert on a system is controlled by the Global environment administrator that creates and/or modifies the System WPAR.
- Other System WPARs: Each System WPAR looks like a separate AIX system to other WPARs.

Each WPAR has a unique ID called a CID (corral ID) which the kernel uses to support separation between the WPARs and to uniquely identify/track a WPAR in subsystems like the Global environment's auditing subsystem.

Privileges can be assigned to a WPAR allowing the WPAR creator to control the capabilities of a WPAR.

6.2.6 Role-Based Access (RA)

6.2.6.1 Role-Based Access Control (RA.1)

AIX provides a role-based access control (RBAC) mechanism. By default, RBAC contains the following administrative roles:

- Information System Security Officer (ISSO)
- System Administrator (SA)
- System Operator (SO)

It contains the following non-administrative role:

- Normal user (This is the default role when a user has no explicit role assigned to them or no explicit role active.)

RBAC provides the ability to define other site-based roles. The TOE associates authorization sets with a role and ensures that the authorizations required to assume a role are satisfied before allowing operations associated with the role to be performed. Through the use of authorizations, the scope of a role can be reduced or increased.

AIX supports up to 8 roles per user session. A user can switch between assigned roles by using the *swrole* (switch role) command. This command creates a new shell process and assigns the requested roles to the new shell. This allows a user to add or delete roles from their active set of roles within their login session. Users can only *swrole* to roles that have been assigned to them.

The roles and authorizations tables for RBAC are maintained in user space, compiled, and loaded into the kernel. When the tables are compiled, they are checked for consistency. The kernel loads the compiled tables during the boot process (if compiled tables do not exist, the boot process will attempt to create compiled tables). Through this methodology, the TOE preserves a secure state and can recover to a consistent, secure state.

System WPAR administrators can create and modify roles independently of the Global environment. The modifications affect only the modified System WPAR.

6.2.7 Privileges (PV)

A privilege is an attribute of a process that allows the process to bypass specific restrictions and limitations of the system (DAC, MAC, MIC, TCB). Privileges are used to override security constraints, to permit expanded use of certain system resources such as memory and disk space, and to adjust the performance and priority of the process. In addition, privileges can be used directly within a user-level program that is responsible for mediating or enforcing security.

The privileges for AIX allow for the administration of the system without the use of the all powerful root administrator ID by granting explicit privileges to users (via authorizations) or commands or WPARs.

Privilege sets can be attached to both subjects and (indirectly via privcmds to) objects, but object privilege sets are used only on executable files and only for modifying the process privilege sets when the file is executed and by WPARs for controlling the capabilities of a WPAR.

The privileges enforced by the TOE’s reference monitor are as follows:

- Audit privileges (identified in section 6.2.3.7)
- Authorization privileges (identified in FDP_ACF.1(3))
- DAC privileges (identified in FDP_ACF.1(1))
- MAC privileges (identified in FDP_IFF.2(1))
- MIC privileges (identified in FDP_IFF.2(3))
- Network/Driver/STREAMS privileges (identified in FDP_IFF.2(2))
- TCB privilege (identified in FDP_ACF.1(2))

6.2.7.1 Process Privilege Sets (PV.2)

There are three types of process privilege sets: *effective privilege* sets (EPS), *maximum privilege* sets (MPS), and *limiting privilege* sets (LPS).

The EPS is used to actually override system restrictions. A process can add or remove privileges from its own EPS subject to the limitations imposed by the MPS.

The MPS is the set of privileges over which a process has control. The MPS is always a superset of the process’s EPS. A process can always remove a privilege from its MPS. A process’s MPS can only be increased if the process has the appropriate privilege, and even then it is restricted by the LPS of the process. The MPS of a process can also be modified when the process runs an executable file, but this too is limited by the process’s LPS.

The LPS represents the maximum possible privilege set that the process can have. The LPS is always a superset of the MPS. Any process can remove a privilege from its LPS, but there is no override mechanism to add a privilege to the LPS. A process cannot acquire privileges from an executable file if the privileges are not in the process’s LPS.

Privileges are inherited by child processes just like the IDs associated with the process.

6.2.7.2 Privileged Commands (PV.3)

AIX provides a Privileged Commands (privcmds) mechanism that assigns privileges to commands (both binaries and shell scripts) based on the role(s) the user has currently active. The mechanism uses a kernel table that contains the list of privileged commands along with the authorizations, privileges, effective uid, and effective gid for each command. For all commands executed, the command is first located in the privcmds table prior to execution by the kernel. If the command is found in the table, the kernel deduces the user’s authorization from the user’s active set of roles. If the user has the proper authorizations to execute the command (as defined by the *accessauths* attribute in the table), the command will be executed as a privileged command where the kernel creates a new process, assigns the privileges from the table’s *innateprivs* attribute, conditionally assigns the privileges from the table’s *authprivs* attribute, assigns the effective uid/gid from the table's *euid* and *egid* attributes, and executes the command, overriding (ignoring) the DAC execution permissions and file system setuid/setgid bits existing on the command.

Each command defined in the table can include the following attributes:

- *accessauths* – Specifies the access authorizations where the user's current session must have at least one of the specified authorizations in order to execute it as a privileged command.
- *innateprivs* – Specifies the privileges to be assigned to the program if it's executed as a privileged command.
- *authprivs* – Specifies additional privileges to be assigned to the program based on the authorizations of the user's current session if it's executed as a privileged command.
- euid – Specifies the effective user ID to be assigned to the program if it's executed as a privileged command.
- egid – Specifies the effective group ID to be assigned to the program if it's executed as a privileged command.

Both the *accessauths* attribute and *authprivs* attribute can contain the following special values:

- ALLOW_OWNER – Allows the user that owns the command to execute the command as a privileged command.
- ALLOW_GROUP – Allows the group that owns the command to execute the command as a privileged command.
- ALLOW_ALL – Allows everyone to execute the command as a privileged command.

In the evaluated configuration, shell scripts defined in the privcmds table should not have the ALLOW_ALL value assigned to them in the *accessauths* attribute and/or the *authprivs* attribute.

The privcmds table is contained in the */etc/security/privcmds* file and protected from unauthorized access.

6.2.7.3 Device Privilege Sets (PV.4)

AIX supports the use of privileges for accessing a device. Each device can have two sets of privileges:

- readprivs – Privileges used to control reading from the device.
- writeprivs – Privileges used to control writing to the device.

Each privilege set supports up to 8 privileges. Only one privilege is needed from the privilege set to allow the type of access defined by the privilege set. When no privileges are assigned to the access type, then access is governed by DAC for the access type. By default, AIX does not define any privileges for any devices.

6.2.7.4 WPAR Privilege Set (PV.6)

The WPAR Privilege Set (WPS) defines the maximum privilege set that a System WPAR and Application WPAR can have. Each System WPAR and Application WPAR has its own WPS assigned to it by the Global WPAR administrator. System WPARs and Application WPARs cannot modify their WPS.

6.2.8 Authorizations (AZ)

Authorizations are the mechanism used to mark certain user accounts as being associated with special administrative roles, such as the information system security officer (ISSO), the system administrator (SA), or the system operator (SO). Roles are simply collections of authorizations. Authorizations can be used to enforce the two-man rule, such as adding new users to the system, where one user can add the account and another can set up the account security information.

6.2.8.1 User Authorizations (AZ.1)

Authorizations are used to limit access to privileged programs, such as the *mount* command and the *shutdown* command. This allows a distinction to be made between administrators and regular users and to divide administrative duties among different classes of administrators.

Authorizations are also used by trusted programs to provide different levels of functionality for different classes of users. For example, the *chmod* program will allow a user with ISSO role to change the permission bits of any file, but allow a regular user to change the permission bits only for his own files.

Authorizations are enforced even when superuser emulation mode (SEM) is enabled. Thus, some commands cannot be executed by user ID 0 even when SEM is enabled.

Authorizations may be used for the following:

1. to determine if a user (or a process running on the user's behalf) can run a program,
2. to determine what privileges a process can have or use while running a program, and
3. to determine what functionality is available to a user while running a specified program.

Cases 1 and 2 can be handled without any modification to the source or binary file. Case 3 requires that the source code of the program be modified to specifically check for authorization and to behave differently based on the authorization check.

There are two categories of authorizations:

- system-defined authorizations – authorizations defined by AIX and hardcoded into the system
- user-defined authorizations – authorizations modifiable by an authorized system administrator

The system-defined authorizations are hardcoded into the system and used by the Global environment and all WPARs, but the user-defined authorizations are located in the */etc/security/authorizations* file. The Global environment and each System WPAR can have its own independent set of user-defined authorizations. The system-defined authorizations and the user-defined authorizations are combined to create the complete set of authorizations for the specific environment.

Section 6.2.7.2 defines how authorizations play a role in the Privileged Commands mechanism.

An authorization consists of the following components:

- a unique name – dot separated notation string of up to 63 printable characters
- a unique numeric identifier - in the range from 1 to 10000 inclusive for system-defined authorizations and greater than 10000 for user-defined authorizations
- a default description message – textual description of the authorization
- message catalog information for a description message

Users have roles assigned to them and can control which assigned roles are active. Roles are a collection of authorizations. The kernel maintains the set of assigned user roles for a user, the set of active roles of the user, and the authorizations associated with all the defined roles in the environment (i.e. Global environment or WPAR) in tables within the kernel. Thus, the kernel can determine for each user the set of active authorizations. The kernel uses these user authorizations to determine (using the privcmds table) if the user can execute a privileged command and the set of privileges the kernel should assign to the process.

Additionally, processes can obtain the invoker's list of active authorizations from the kernel and make access decisions based on the invoker's active authorization set.

Because the authorization set associated with a process is a function of the process's RUID, the authorizations of a process may change dynamically, such as after a *setuid* system call.

6.2.9 Mandatory Access Control (MAC)

This section applies to LSPP mode only.

The TOE provides a MAC policy that enforces access to named objects listed in Table 6 based on sensitivity labels. A sensitivity label is an access control structure that enumerates the MAC access control properties for the object. Sensitivity labels exist for all subjects and named objects on the system. A sensitivity label contains a single classification and a set of zero or more categories. The TOE supports up to 32,001 hierarchical classifications and up to 1,024 categories.

The MAC policy enforced by the TOE can be described using the concept of sensitivity label dominance. A sensitivity label dominates another if the classification of the first equals or exceeds the classification of the second, and if every category in the second is included in the first. For a subject to read an object, the subject's sensitivity label must dominate the object's sensitivity label. The TOE uses a label encodings file to designate TOE labels and their dominance relationships that are used for MAC enforcement. For a subject to write an object, the object's sensitivity label must be equal to the subject's sensitivity label.

The TOE can be configured with an accreditation range. The accreditation range is defined by a System High and System Low sensitivity label. The System High label must dominate all data processed on the system. All data on the system must dominate the System Low sensitivity label.

Processes can use privileges to override MAC restrictions as defined in FDP_IFF.2(1).

The import and export of data is implemented through a trusted version of the *backup/restore* utilities. These utilities have been extended to handle labeling. The extensions are transparent to the user and, aside from the labeling extensions, the command functions in the standard fashion. A combination of privilege and authorization mechanisms protects the import/export system.

Import and Export of labeled data describes the system's ability to maintain security attributes when objects are imported and exported to and from the system using predefined security enforcing interfaces. Import and Export of unlabeled data describes the system's ability to disregard security attributes, and the restrictions that are in place to maintain the system's integrity. The TOE enforces mandatory access control when exporting labeled and unlabeled data. Mandatory access control decisions are based on the files sensitivity label and the configured device sensitivity label or label range. The subject exporting the data on behalf of a user must have appropriate privileges, or the user must have appropriate authorizations.

Labeled functionality is included on the TOE. Data can be sent to both hardware devices and to files. Labels are automatically included in backup headers. An unauthorized user cannot override the placing of labels with the data. Only an ISSO authorized administrator can export data without security labels. The TOE provides the tar command that can be used to export data, and does not preserve security information. The TOE requires that a manual change in the device state be performed before using a device to export data without security attributes. This action is auditable.

When writing to disks or tapes with *backup* sensitivity labels (SLs) are included with the data. A user must have the ISSO role as an active authorization to use backup/restore to import or export unlabeled data from tapes or disks. When writing unlabeled data, the data receives the SL of the writing process. When importing unlabeled data, the data is assigned the SL of the importing process. However, if unlabeled data were written with a high-level process, the reading process must have an equal or higher level to read the data.

The evaluated TOE generates printer output using the Postscript Version 2.0 standard and PCL Version 5. The ISSO has the ability to specify the printable label assigned to the sensitivity label of a print job that is sent through the SystemV printing subsystem of AIX. Printed output is labeled with banner pages and security headers and footers. The banner pages contain the human readable sensitivity label.

Printers and mounted file systems are the devices that are available to users for importing/exporting **labeled** data only. The TOE places labels on the data being imported or exported via these devices. If an administrator changes the device such that it does not handle labeled data, the system is outside the evaluated configuration.

For remote access via NFS, as well as the other single level file systems cdrfs, udfs, and procfs, a single level policy is implemented: On the client side of NFS, as well as for the other single level file systems, the labels of the mount point are applied to the entire file system (precisely, the max SL and min SL for each file system object located within the single level file system is equal to the max SL and min SL of the mount point).

The following devices are available to users for importing/exporting **unlabeled** data only:

- floppy drive
- raw hard disk drive
- CD-ROM writeable
- serial port
- other devices

The TOE does not place labels on data being imported or exported via these devices. If an administrator changes the device such that it handles labeled data, the system is outside the evaluated configuration. Although, administrators can make these devices unavailable, this action does not change the device state from unlabeled to labeled.

6.2.10 Networking (TN)

This section contains 3 TSFs: TN1, TN.2, and TN.3. TN.1 and TN.2 comprise the Trusted Network feature of Trusted AIX and apply to LSPP mode only. TN.3 is the AIX IP Filtering feature and applies to both CAPP mode and LSPP mode.

In LSPP mode, the following devices can be configured for both labeled and unlabeled data.

- network

6.2.10.1 Network and interface rules (TN.1)

This section applies to LSPP mode only and is a part of the Trusted Network feature.

Labeling of data is handled by TN, which reads settings from rules loaded into the kernel from in configuration files. On initial installation, the rules are set to import and export labeled data. The rules can be changed with the netrule command. Such changes are not saved on power down. The tninit command saves changes between sessions.

The configuration files for TN are /etc/security/rules.host and /etc/security/rules.int. These are in binary format. If /etc/security/rules.int does not exist, the system will create a rules file from the ASCII text file /etc/tn/scripts/iniRules.txt by running tninit during the boot sequence.

The “change in device state” from labeled to unlabeled import and export of data is determined by the TN rules. Therefore, using netrule or tninit to change the TN rules constitutes the “manual change in device state” required for the evaluation.

The netrule and tninit commands generate audit events when rules are changed. This action satisfies the Security Target requirement for auditing the “change in device state”.

Exported data, even if labeled, may be read by a non-LSPP mode machine. For example, if an administrator uses backup to archive data onto a tape, then the tape can be read by an unmodified machine using restore, and all labels will be ignored, granting access to all data on the tape. Administrators should be aware that this is not a supported option.

TN provides two sets of networking rules: network interface and host filtering. Both types of networking rules determine what processing occurs on a packet before its transmission or when it is received. These rules are an extension of the MAC policy because 1) they apply sensitivity labels to packets, and 2) they enforce MAC restrictions on packets according to those labels. The TOE enforces MAC between subjects and TN named objects.

Network interface rules enforce packet label processing based on the physical network interface of the host. Host rules enforce packet label processing based on the source and destination IP addresses (with network masking allowed) of the packet, the source and destination ports of the request, and the protocol being used. Both types of rules provide several criteria for determining which packets to drop and which to pass.

Information flow is permitted only if the following sequential ordering relationships between security attributes hold:

a. if the IP address of the packet is equal to the IP address specified in the rule;

b. if the IP address of the packet is within the network mask specified for the rule;

c. if the direction of the packet flow corresponds to the direction of the rule (IN/OUT);

d. if the protocol of the packet is equal to the protocol specified in the rule;

e. if the port is within the range specified in the rule;

f. if the network interface of the packet is equal to the network interface specified in the rule;

g. if the IPSO labels are within the range defined by the rule, and the rule set to allow IPSO labels;

h. if the packet’s SL is within the minimum and maximum SL specified for the rule.

6.2.10.2 Internet Protocol Security Option (IPSO) (TN.2)

This section applies to LSPP mode only and is a part of the Trusted Network feature.

The TOE supports the Revised Internet Protocol Security Option (RIPSO) specified in RFC 1108 and the Common Internet Protocol Security Option (CIPSO) as specified in FIPS PUB 188, which allows the transmission of labeled data over IP networks. It also supports the DoD Basic Security Option (BSO) and Extended Security Option (ESO) specified in RFC 1108. The TOE uses a standards compliant implementation for IPv4. Since the standards do not address IPv6, the TOE's implementation for IPv6 is specific to the TOE.

In order to preserve the labels for the transmitted data, IP datagrams are extended with IP options that provide for classification of the transmitted data. To translate system-specific sensitivity labels defined in /etc/security/enc/LabelEncodings into the RFC1108-specified RIPSO labels, a translation table has to be created in (for example) /etc/security/rfc1108 and its location to be supplied to *tninit*. CIPSO uses a security tag to indicate the rule used to construct the security data in the IPSO. TN supports the tag types 1, 2 and 5 as specified in FIPS PUB 188.

The *netrule* command (which requires authorizations in order to execute) is used to define TN rules that specify the usage of CIPSO or RIPSO for network connections (see above).

6.2.10.3 IP Filtering (TN.3)

This section applies to both CAPP mode and LSPP mode and is not part of the Trusted Network feature.

The TOE supports IP filtering of both IPv4 and IPv6 network packets. Authorized administrators can specify IP filtering rules used for IP filtering. The rules can either permit or deny packet flow. The rules can either always permit/deny the flow or they can permit/deny the flow for a fixed period of time. The rules can be based on IP version (4 or 6), protocols (UDP, ICMP, ICMPV6, TCP, ESP, AH), presumed source addresses, destination addresses, destination ports, and source routings. The rules can be applied to different interface types (e.g., tr0, en0, lo0, pp0). Initially, packet flows are unrestricted until rules are applied.

6.2.11 Mandatory Integrity Control (MIC)

This section applies to LSPP mode only.

Mandatory integrity control is a system-enforced means of restricting access to and modification of objects based on the integrity of the object and the clearance of the user. While MAC is concerned with the sensitivity of an object, MIC is concerned with its trustworthiness.

The TOE enforces MIC using a system of labels. All named objects have integrity labels (TLs) to identify the integrity level of the object. Processes also have TLs. Process TLs indicate what level of information integrity the processes are allowed to access. The higher the TL, the more trustworthy the object or process. A process must be at least as trustworthy as an object in order to modify it. This means a process must have a TL equal to or exceeding that of the object. Thus TLs can be used to make files accessible for read only. For creating new objects in a file system, a process must be at least as trustworthy as the directory the object is to be created in.

MIC for read access is not enforced in the evaluated configuration.

6.2.11.1 MIC Labels (MIC.1)

All system objects, such as files, processes, etc. have TLs. TLs are automatically placed on objects at the time of creation. All core dumps are considered objects on the system and are automatically labeled by the system.

Only processes with proper privileges and authorizations are able to change the TL of a file or process.

Unlike MAC, directories only have a single label for MIC. There is a special TL that can be put on a file or process, called NOTL. When NOTL is on an object or process, no MIC checks are performed on it. Only privileged users can set a TL to NOTL, or change a TL if it is currently NOTL.

6.2.12 Object Reuse (OR)

Object Reuse is the mechanism that protects against scavenging, or being able to read information that is left over from a previous subject's actions. The following general techniques are applied to meet this requirement in the AIX distributed system:

- explicit initialization of resources on initial allocation or creation,
- explicit clearing of resources on release or deallocation,
- management of storage for resources that grow dynamically, and

- administrator-initiated wiping of hard disk drives.

Explicit initialization is appropriate for most TSF-managed abstractions, where the resource is implemented by some TSF internal data structure whose contents are not visible outside the TSF: queues, datagrams, pipes, and devices. These resources are completely initialized when created, and have no information contents remaining.

Explicit clearing is used in AIX only for directory entries, because they are accessible in two ways: through TSF interfaces both for managing directories and for reading files. Because this exposes the internal structure of the resource, it must be explicitly cleared on release to prevent the internal state from remaining visible.

Storage management is used in conjunction with explicit initialization for object reuse on files, and processes. This technique keeps track of how storage is used, and whether it can safely be made available to a subject.

Hard disk wiping provides means to an administrator to overwrite information on hard disks with bit patterns in order to render previously stored information on the disks unrecoverable. Rather than being a system property, this is a function that can be invoked by the administrator.

The following sections describe in detail how object reuse is handled for the different types of objects and data areas.

6.2.12.1 Object Reuse: File System Objects (OR.1)

All file system objects (FSOs) available to general users are accessed by a common mechanism for allocating disk storage and a common mechanism for paging data to and from disk. This includes the Journaled File System (JFS2) and Network File System (which exists physically as a JFS2 volume on a server host). This includes both normal and large JFS2 file systems.

Object reuse is irrelevant for the CD-ROM File System (CDRFS) and the Universal Data Standard file system (UDFS) because they are read-only file systems, making it impossible for a user to read residual data left by a previous user. File systems on other media (tapes, diskettes) are irrelevant because of warnings in the AIX Security Guide not to mount file systems on these devices.

For this analysis, the term FSO refers not only to named file system objects (files, directories, device special files, named pipes, and UNIX domain sockets) but also to unnamed abstractions that use file system storage (symbolic links and unnamed pipes). All of these, except unnamed pipes, device special files and UNIX domain sockets, have a directory entry that contains the pathname and an inode that controls access rights and points to the disk blocks used by the FSO.

In general, file system objects are created with no contents, directories and symbolic links are exceptions, and their contents are fully specified at creation time.

6.2.12.1.1 Object Reuse: Files

Storage for files is allocated automatically in pages as a file grows. These pages are cleared before they become accessible, within the file. However, when a file is deleted the space holding the data from the file, both in memory and on disk, is not cleared. This data will persist until the space is assigned to another file, when it will be cleared. These internal fragments of deleted files are protected by the kernel to prevent accessing of deleted data.

If data is read before it is written, it will read only as zeroes. Reads terminate when the end-of-file (EOF) is detected. It is possible to seek past the EOF, but any reads will return zeros. File writes may cause the file to grow, thus overwriting any residual data and moving the EOF. If the file pointer is advanced past the EOF and then written, this leaves a hole in the file that will subsequently be read as zeroes.

6.2.12.1.2 Object Reuse: Directories and Directory Entries

In part, object reuse for directories is handled as for ordinary files: pages allocated are always cleared before being incorporated into the directory. When a directory is first created, it is explicitly initialized to have the entries “.” and “..”, but the remainder of the directory’s storage is cleared.

Individual directory entries are manipulated as distinct resources, such as when referencing file system objects, and as part of the directory, such as when reading the entire directory itself. When a directory entry is removed or renamed the space occupied by that directory entry is either combined with the previous entry as free space or else the inode number of the entry is set to zero when the entry occurs on a 512 byte boundary.

When a directory entry does not occur on a 512-byte boundary, the size of the preceding directory entry is incremented by the size of the directory entry which has been removed. The space in a directory entry in excess of that which is needed to store the necessary information may be allocated when a directory entry is to be created. The fields of the directory entry remain unchanged.

When a directory entry occurs on a 512-byte boundary, the inode number is set to zero to indicate that this entry is now available for re-use. All other fields of the directory entry remain unchanged.

The directory entry is no longer visible to interfaces which perform file name operations and may only be seen when the entire directory is examined and the process has read access to the directory.

6.2.12.1.3 Object Reuse: Symbolic Links

Symbolic links have their contents (the link pathname) fully specified at creation time, and the readlink operation returns only the string specified at creation time, not the entire contents of the block it occupies.

6.2.12.1.4 Object Reuse: Device Special Files

All device special files are initialized to a known state on first open and never grow. Device special files refer to actual hardware or else to virtualized objects. There are no file system blocks, unless the device references a file system (in which case the mechanism for object reuse of file system objects apply). Nor is there memory, unless the device is associated with memory (in which case the object reuse mechanisms for memory objects apply).

6.2.12.1.5 Object Reuse: Named Pipes

FIFOs are created empty. Buffers are allocated to contain data written to a pipe, but the read and write pointers are managed to ensure that only data that was written to the pipe can ever be read from it.

6.2.12.1.6 Object Reuse: Unnamed Pipes

Unnamed pipes are created empty. Buffers are allocated to contain data written to a pipe, but the read and write pointers are managed to ensure that only data that was written to the pipe can ever be read from it.

6.2.12.1.7 Object Reuse: Socket Special File (UNIX Domain)

UNIX domain sockets have no contents; they are fully initialized at creation time.

6.2.12.2 Object Reuse: IPC Objects (OR.2)

AIX shared memory, message queues, and semaphores are initialized to all zeroes at creation. These objects are of a finite size (shared memory segment is from one byte to 256 MBytes, semaphore is one bit), and so there is no way to grow the object beyond its initial size.

No processing is performed when the objects are accessed or when the objects are released back to the pool.

6.2.12.3 Object Reuse: Queuing System Objects (OR.3)

6.2.12.3.1 Object Reuse: Batch Queue Entries

cron and at jobs are defined in batch files, which are subject to the object reuse protections specified for files as described previously.

6.2.12.4 Object Reuse: Miscellaneous Objects (OR.4)

6.2.12.4.1 Object Reuse: Process

A new process’s context is completely initialized from the process’s parent when the fork system call is issued. All program visible aspects of the process context are fully initialized. All kernel data structures associated with the new process are copied from the parent process, then modified to describe the new process, and are fully initialized.

The AIX kernel zeroes each memory page before allocating it to a process. This pertains to memory in the program’s data segment and memory in shared memory segments. When a process requests more memory, the memory is explicitly cleared before the process can gain access to it.

When the kernel performs a context switch from one thread to another, it saves the previous thread’s General Purpose Registers (GPRs) and restores the new thread’s GPRs, completely overwriting any residual data left in the previous thread's registers. Floating Point Registers (FPRs) are saved only if a process has used them. The act of accessing an FPR causes the kernel to subsequently save and restore all the FPRs for the process, thus overwriting any residual data in those registers.

Processes are created with all attributes taken from the parent. The process inherits its memory (text and data segments), registers, and file descriptors from its parent. When a process execs a new program, the text segment is replaced entirely.

6.2.12.5 Object Reuse: Hard disk drives (OR.5)

For SCSI disks that do not participate as "pdisks" in RAID arrays, the diagnostic subsystem (cf. section 6.2.14.7) offers a "Hard File Erase Disk" functionality to administrators of the TOE. SCSI disks which are part of a pdisk must be detached from the pdisk for erasure.

This option can be used to overwrite (remove) all data stored in currently user-accessible blocks of the disk. The Erase Disk option writes one or more user-specifiable patterns to the disk.

The administrator can specify the number (0-3) of patterns to be written to the hard disk. The patterns are written serially; that is, the first pattern is written to all blocks. Then the next pattern is written to all blocks, overlaying the previous pattern. Also, a random pattern can be written.

Please note that there are two abstraction layers in the underlying environment involved that restrict the TOE to the deletion of user-accessible blocks on those hard disk drives only: Firmware of SCSI hard disk drives and firmware of SCSI disk controllers may remap "bad blocks" containing user or TSF data to healthy blocks on the physical hard disk drive and maintain a pool of unallocated blocks for this purpose. The TOE is not able to (and does not claim to) overwrite such blocks, since it is using the generic SCSI interfaces to access the hard disk drive. Since the hard disk drive stays within the TSC it is ensured that users of the TOE, accessing the drive via the TOE-provided interfaces, won't be able to recover any residual information on it.

6.2.13 Security Management (SM)

This section describes the functions for the management of security attributes that exist within AIX.

6.2.13.1 Roles (SM.1)

AIX provides a role-based access control (RBAC) mechanism. See section 6.2.6 for more details.

In LSPP mode, the administrator is not root, but one of the ISSO, SA and SO roles. Root itself will not be used as a user ID that a user can directly log in to or for administration.

In CAPP mode, the root user can be allowed to log in (root enabled mode) or root login can be disabled (root disabled mode). Note that **root disabled mode is not supported** in the evaluated configuration in CAPP mode.

Users that have the appropriate authorizations associated with their account may run administrative programs associated with those authorizations/roles.

6.2.13.1.1 Normal Users

In AIX, normal users cannot perform actions that require administrator privileges. They can only execute those setuid root programs and privileged programs they have access to (either via DAC or authorizations or the privcmds table). In the evaluated configuration this is restricted to those programs they need, like the *passwd* program that allows a user to change his/her own password.

6.2.13.1.2 VIOS Roles

All VIOS roles are authorized administrative roles. VIOS doesn't support the concept of normal users. VIOS defines the following roles:

- Prime Administrator (a.k.a. padmin) – This role can execute every command provided by the VIOS command-line interface including the user ID commands and security commands. This role is limited to a single user ID: *padmin*. This user ID is defined in the installation image and no other users can be a Prime Administrator.
- System Administrator– This role can execute every command provided by the VIOS command-line interface except for the security commands and user ID commands (exception: they can change their own passwords). System Administrator user accounts do not exist until the Prime Administrator creates one or more.

- Development Engineer (DE) – This role is used only by IBM personnel to debug problems and run diagnostics. Development Engineer user accounts do not exist until the Prime Administrator creates one or more.
- Service Representative (SR) – This role allows a service representative to run commands that are required to service the system (shutdown, restart, update system microcode, configure/unconfigure devices, certify, format, etc.). Service Representative user accounts do not exist until the Prime Administrator creates one or more.

6.2.13.2 Audit Configuration and Management (SM.2)

Audit control consists of the files used to maintain the configuration of the audit subsystem and a description of the AUDIT command and its associated parameters

Table 11: Audit Control Files. The audit control files maintain the configuration for the auditing subsystem.

Audit Control File	Description
/etc/security/audit/config	Defines whether bin mode auditing is enabled, the names of the files used to store audit data and the names of the available classes. Also defines the audit classes, i.e., for each audit class the audit events belonging to the class are defined
/etc/security/audit/events	Defines audit events to be used on the system. An event needs to be defined in this file to be formatted correctly.
/etc/security/audit/objects	Contains a list of the objects whose access will be audited
/etc/security/audit/bincmds	Contains the post-processing command or commands for bin mode auditing
/etc/security/user	Contains a record for each user which specifies which classes will apply to the user account

There are two different types of audit event selection: per-user and per-object. Per-user auditing allows the administrator to specify specific classes of audit events that will be recorded for that user. Each process stores a copy of the audit classes that apply to that user as part of the process table. An audit class is a subset of the total number of audit events available and is defined in the file */etc/security/audit/config*.

Per-object auditing allows the administrator to specify file system objects that will be audited. This is defined in the file */etc/security/audit/objects*. There, for individual objects, the audit event for the access modes that one wants to be audited is defined.

These objects can be audited based on accesses of a specified mode (read/write/execute) and record the result of the access attempt (success/failure).

The *audit* command is used to start and stop the auditing subsystem, to temporarily switch the auditing subsystem on or off, and to query the audit subsystem for the current audit parameters.

The *audit* command is started from the host’s rc initialization script, as stated in the Security Guide.

The on and off parameters of the *audit* command enable and disable audit, without modifying the current configuration that is stored in the kernel. The on parameter can have an additional parameter, panic, which causes the system to shut down if bin data collection is enabled and records cannot be written to one of the bin files. The bin mode panic option can also be specified in */etc/security/audit/config*.

When the *audit* command is issued with the shutdown parameter, the collection of audit records is halted, and all audit configuration information is flushed from the kernel’s tables. All audit records are flushed from the kernel’s buffers and processed. The collection of audit data is halted until the next audit start command is entered.

When the *audit* command is issued with the start parameter, the following events occur:

- the */etc/security/audit/config* file is read
- the */etc/security/audit/objects* files is read and the objects that will be audited based on access are written into kernel tables
- the audit class definitions are written into kernel tables from */etc/security/audit/config*
- the auditbin daemon is started, depending on the options in */etc/security/audit/config*

- auditing is enabled for users specified in the user's stanza of the */etc/security/audit/config* file
- auditing is turned on, with panic mode enabled or turned off, depending on what mode is specified in */etc/security/audit/config*

To access the audit functions, the appropriate PV_AU privileges are required. The AUDIT authorization is needed for reading audit records.

The events that are audited can be selected on a per user basis, per event basis and per object basis using the configuration files described above.

A description of the structure of those files and the syntax of the entries can be found in *AIX 6.1 Files Reference* document.

A System WPAR contains its own audit trial which functions independently from the Global environment's audit subsystem. A Global environment determines what it wants to audit of the System WPAR and those events are added to the Global environment's audit trial regardless of the state of the auditing subsystem in the System WPAR. Similarly, a System WPAR configures and controls its own audit trail. An Application WPAR does not have a separate auditing subsystem, so it is under the audit control of the Global environment.

6.2.13.3 Access Control Configuration and Management (SM.3)

Discretionary access control to objects is defined by the permission bits and by the Access Control Lists (for those objects that have access control lists associated with them) or by NFSv4 ACLs. Default access permission bits are defined in the system configuration files that define the value of the access control bits for objects being created without explicit definition of the permission bits. The system administrator can define and modify those default values.

Permissions can be changed by the object owner and the system administrator. When an object is created the creator is the object owner. Object ownership can be transferred except for TCP ports, where the owner always remains the system administrator. In the case of IPC objects, the creator will always have the same right as the owner, even when the ownership has been transferred.

LSPP Mode Only: For MAC, TN, and MIC, sensitivity and integrity labels are assigned to all objects and users, whereas users are assigned a specific range of levels (clearance) within which they can operate.

IP Filtering rules are managed by authorized administrators. Authorized administrators can create, modify, delete, activate, de-active, and query rules.

TCB files are identified by a specific file security flag that can be set by authorized administrators. Authorizations can be granted to users by authorized administrators.

NFSv4 ACLs provide a mechanism which allows the object owner to give others the ability to modify the entries within the ACL. Directory NFSv4 ACLs can include entries that are inherited by child objects.

EFS provides a mechanism (including the commands to enable/disable and manage it) to encrypt/decrypt files. Encryption prevents the disclosure of the data in the files except by those who have the proper key to decrypt the data.

File Integrity Verification (FIV) provides a mechanism for the system administrator to determine if system critical objects have the appropriate security attributes (user, group, permission bits, etc.) set on the objects.

System WPARs are created, managed, and deleted by Global environment administrators within the Global environment. The Global environment administrator can control the file systems available to a WPAR, the mode of the file systems available to a WPAR, the devices available to a WPAR, and the privileges available to the processes of a WPAR.

For VIOS, both the VIOS SCSI discretionary access control and the VIOS Ethernet discretionary access control are managed by the system administrators. VIOS provides an administrative interface for managing these functions. VIOS SCSI device drivers acting on behalf of the LPAR partitions are not allowed to access a logical or physical volume until the mapping is created in VIOS. A VIOS Ethernet device driver acting on behalf of a group of LPAR partitions sharing a virtual network cannot access a VOS Ethernet adapter device driver and vice versa until a mapping is created in VIOS.

6.2.13.4 Management of User, Group and Authentication Data (SM.4)

Each System WPAR can create and manage its own set of users, groups, and authentication data independently from the Global environment and from other System WPARs.

6.2.13.4.1 Creating New Users

An administrator (role SA) can create a new user and can assign a unique user ID to this user. The initial password and other security attributes have to be defined by the ISSO using various utilities (e.g. the *passwd* command). The new user will be disabled until the initial password is set.

Attributes that can be set for each user are among others (a complete list can be found in the description of the *chuser* command and the description of the content of the file */etc/security/user*):

- Lock attribute (i.e., temporarily locking a user account)
- Administrative status of the user
- List of audit classes for the user
- List of groups the user belongs to
- Home directory for this user
- Number of consecutive unsuccessful login attempts allowed before the user account is locked
- Password parameter including the maximum and minimum age of a password, minimum length, difference to the old password, etc.

Those attributes are stored in the following files:

- */etc/group*
- */etc/passwd*
- */etc/security/passwd*
- */etc/security/user*
- */etc/security/audit/config*

6.2.13.4.2 Modification of User Attributes

User attributes can be modified by the system administrator (ISSO). Modifications of user attributes require the modification of the administration database that contains the user attributes (mainly */etc/security/user*).

6.2.13.4.3 Management of Authentication Data

The system administrator (ISSO) has the capability to define rules and restrictions for passwords used to authenticate users. The parameters available are:

minage	Minimum number of weeks that must pass before a password can be changed.
maxage	Maximum number of weeks that can pass before a password must be changed.
maxexpired	Maximum number of weeks beyond maxage that a password can be changed before administrative action is required to change the password. (Root is exempt.)
minalpha	Minimum number of alphabetic characters the new password must contain.
minother	Minimum number of non-alphabetic characters the new password must contain. (Other characters are any ASCII printable characters that are non-alphabetic and are not national language code points).
minlen	Minimum number of characters the new password must contain.
maxrepeats	Maximum number of times a character can be used in the new password.
mindiff	Minimum number of characters in the new password that must be different from the characters in the old password.
histexpire	Number of weeks that a user is unable to reuse a password.
histsize	Number of previous passwords that cannot be reused.
dictionlist	List of dictionary files checked when a password is changed. Dictionary files contain passwords that are not allowable.

Users are also allowed to change their own password using the *passwd* command. The password restrictions defined by the system administrator apply.

VIOS supports the following passwords parameters only: maxage, maxexpired, minother, minlen, maxrepeats, histexpire, and histsize.

6.2.13.5 Time Management (SM.5)

AIX provides the standard UNIX functions to manage the system clock. The time can be set or modified by an administrator (ISSO). Modifications to the system time are audited (if configured) allowing a system administrator to extract the differences between the “old” and “new” value of the system clock. The value of the system clock cannot be manipulated by normal users.

6.2.14 TSF Protection (TP)

While in operation, the kernel software and data are protected by the hardware memory protection mechanisms described in the high level design and the hardware reference manuals of AIX. The memory and process management components of the kernel ensure a user process cannot access kernel storage or storage belonging to other processes.

Non-kernel TSF software and data are protected by DAC, MAC (LSPP mode only), MIC (LSPP mode only), and TCB controls and process isolation mechanisms. In general, files and directories containing internal TSF data (e.g., audit files, batch job queues) are also protected from reading by access control permissions.

The TSF and the hardware and firmware components are required to be physically protected from unauthorized access. The system kernel mediates all access to the hardware mechanisms themselves, other than program visible CPU instruction functions.

The boot image for each host in the distributed system is adequately protected. A description of the boot logical volume can be found in section 5.3.16, Initialization and Shutdown.

6.2.14.1 TSF Invocation Guarantees (TP.1)

All system protected resources are managed by the TSF. Because all TSF data structures are protected, these resources can be directly manipulated only by the TSF, through defined TSF interfaces. This satisfies the condition that the TSF must be “always invoked” to manipulate protected resources.

Resources managed by the kernel software can only be manipulated while running in kernel mode.

Processes run in user mode and can call functions of the kernel only as the result of an exception or interrupt. The hardware and the kernel software handling these events and ensure that the kernel is entered only at pre-determined locations, and within pre-determined parameters. All kernel managed resources are protected such that only the kernel software is able to manipulate them.

Trusted processes implement resources managed outside the kernel. The trusted processes and the data defining the resources are protected as described above depending on the type of interface. For directly invoked trusted processes the program invocation mechanism ensures that the trusted process always starts in a protected environment at a predetermined point. Other trusted process interfaces are started during system initialization and use well defined protocol or file system mechanisms to receive requests.

Some system calls or parameters of system calls are reserved for trusted processes. When called, the kernel checks that the calling process runs with the appropriate privileges.

The TOE contains a Stack Execution Disable (SED) facility. When enabled, SED ensures that code residing on the stack of selected processes cannot be executed by the processes. This facility is configured by an administrator with the *sedmgr* command. If a process is configured to deny execution of code on its stack and the process attempts to execute code on its stack, the system will generate an exception and terminate the process. This helps prevent buffer overflow attacks by not allowing attackers to execute arbitrary code on the stack of an executable.

The TOE implements the TSP through a reference monitor. The kernel reference monitor (KRM) is a single C-language function that is called any time the security policy may need to be enforced. The KRM accepts as arguments all of the security attributes of the subject and object associated with the security check, along with an indication of what check or checks and auditing need to be performed. The KRM itself consists of many code modules that are called to handle the appropriate check or checks that are required. The KRM is used to enforce the security policy in the following instances:

- Systems calls
- File system creation, deletion, and access events
- Interprocess communication, including signals
- Network packet checks

The KRM handles all of the following kernel security mediation events:

- Discretionary access control (DAC) checks
- Mandatory access control (MAC) checks
- Trusted Network (TN) rules
- Mandatory integrity control (MIC) checks
- Trusted computing base (TCB) checks
- System security flag (SSF) checks
- File security flag (FSF) checks
- Privilege (PV) checks
- Authorization (AZ) checks
- Auditing (AUD) for all of the above

The KRM accepts as arguments the following:

- subject security attributes
- action to be performed by the reference monitor
- security attributes and object type for up to 4 objects
- flag indicating if auditing should be done
- pointer to the system-wide security settings

Although the KRM can check MAC, MIC, DAC, TCB, FSF, SSF, PV, AZ, and AUD components, not all actions require all checks. When multiple checks are required, they are performed in the following order:

- MAC (with SSF, FSF, PV and AUD as needed)
- MIC (with SSF, FSF, PV and AUD as needed)
- FSF (with PV and AUD as needed)
- TCB (with SSF, PV and AUD as needed)
- DAC (with PV and AUD as needed)
- PV (with SSF, FSF, and AUD as needed)
- AZ (with SSF, FSF, PV, and AUD as needed)

System security flags (SSF), also known as kernel security flags, support the enforcement of the TSP.

CAPP Mode Only: The evaluated configuration mandates the following settings for SSF in order to ensure that the TSP be enforced accurately:

Table 12: System/kernel security flags for CAPP mode only.

SYSTEM SECURITY FLAG	**OPERATIONAL MODE**
Root (ROOT)	ENABLED

LSPP Mode Only: The evaluated configuration mandates the following settings for SSF in order to ensure that the TSP be enforced accurately:

Table 13: System/kernel security flags for LSPP mode only.

SYSTEM SECURITY FLAG	**OPERATIONAL**

	MODE
Sl_enforced (MAC)	ENABLED
trustedlib_enabled (TLIB)	ENABLED
tl_read_enforced (MIC)	DISABLED
tl_write_enforced (MIC)	ENABLED
tnet_enabled (TN)	ENABLED
Root (ROOT)	DISABLED

6.2.14.2 Kernel (TP.2)

The AIX software consists of a privileged kernel and a variety of non-kernel components (trusted processes). The kernel operates on behalf of all processes (subjects).

The kernel runs in the CPU's privileged mode and has access to all system memory. All kernel software, including kernel extensions and kernel processes, execute with kernel privileges but only defined subsystems within the kernel are part of the TSF. The kernel is entered by some event that causes a context switch such as a system call, I/O interrupt, or a program exception condition.

Upon entry the kernel determines the function to be performed, performs it, and, when finished, performs another context switch to return to user processing (eventually on behalf of a different subject).

The kernel is shared by all processes, and manages system wide shared resources. It presents the primary programming interface for AIX in the form of system calls.

Because the kernel is shared among all processes, any process running "in the kernel" (that is, running in privileged hardware state as the result of a context switch) is able to directly reference the data structures that implement shared resources.

The major components of the kernel are memory management, process management, the file system, the low-level I/O system, WPAR management, and the kernel extensions like implementing for example network protocols (IP, TCP, UDP, and NFS).

6.2.14.3 Kernel Extensions (TP.3)

Kernel extensions are dynamically loaded code modules that add function to the kernel. They include device drivers, virtual file systems (e.g., CDRFS, NFS), inter-process communication methods (e.g., named pipes), networking protocols, and other supporting services. Kernel extensions can be loaded only at system boot in the evaluated configuration.

Kernel extensions run with kernel privilege, similarly to kprocs. However, extensions differ from kprocs in that the kernel does not schedule them. Instead, kernel extensions are invoked from user processes by system calls, or internal calls within the kernel, or started to handle external events such as interrupts.

Kernel extensions run entirely within the kernel protection domain. An extension may export system calls in addition to those exported by the base AIX kernel. User-domain code can only access these extensions through the exported system calls, or indirectly via the system calls exported by the base kernel.

Device drivers are kernel extensions that manage specific peripheral devices used by the operating system. Device drivers shield the operating system from device-specific details and provide a common I/O model for user programs to access the associated devices. For example, a user process calls read to read data, write to write data, and ioctl to perform I/O control functions.

6.2.14.4 Trusted Processes (TP.4)

Trusted processes in AIX are processes running in user mode but with privileges. Some high-level TSF functions are performed by trusted processes particularly those providing distributed services.

A trusted process is distinguished from other user processes by the ability to affect the security policy. Some trusted processes implement security policies directly (e.g., identification and authentication) but many are trusted simply because they operate in an environment that confers the ability to access TSF data (e.g., programs run by administrators or during system initialization).

Trusted processes have all the kernel interfaces for which they have the appropriate privilege available for their use, but are limited to kernel-provided mechanisms for communication and data sharing, such as files for data storage and pipes, sockets and signals for communication.

The major functions implemented with trusted processes include user login, identification and authentication, batch processing, audit data management and reduction, some network operations, system initialization, and system administration.

The kernel will check for each system call that requires privileges if the process that issued the call has those privileges. If not, the kernel will refuse to perform the system call. The kernel will also, for each access to an object protected by any of DAC, MAC (LSPP mode only), and MIC (LSPP mode only) mechanisms, check if the process has the required access rights for the attempted type of access. Note that commands listed in the privcmds database can override the DAC rules if the user as the proper roles.

Any program executed with DAC override privileges has the ability to perform the actions of a trusted process. It is therefore important that a site operating AIX strictly controls those programs and prohibits that those programs are modified and prohibits that programs from untrusted sources are executed with root privileges.

Trusted processes are not part of the kernel and (except for those processes that perform system initialization and identification and authentication) not part of the TSF itself.

Trusted processes provide a contribution to security management and identification and authentication.

Note: Trusted processes may use system management commands or system calls as mentioned in the section on supporting functions that are not part of the TSF. But in any case the kernel will verify that the process has the right to perform the system call with the parameter specified by the caller and has the right to access all files with the intended access mode.

6.2.14.5 TSF Databases (TP.5)

Table 14 identifies the primary TSF databases used in AIX and their purpose. These are listed both as individual files (by pathname) or collections of files.

With the exception of databases listed with the User attribute (which indicates that a user can read, but not write, the file), all of these databases shall only be accessible to administrators. None of these databases shall be modifiable by a user other than a system administrator with the appropriate authorization.

Those databases are part of the file system and therefore the file system protection mechanisms of the TOE have to be used to protect those databases from unauthorized access. It is the task of the persons responsible for setting up and administrating the system to ensure that the access control features of the TOE are used throughout the lifetime of the system to protect those databases.

VIOS, which uses only file-based I&A, uses the same TSF database file mechanism as AIX, but it maintains its files separately from AIX.

For LDAP-based I&A, just the user and group information is stored in LDAP. Therefore, LDAP replaces the following files:

- /etc/group
- /etc/passwd
- /etc/security/lastlog
- /etc/security/passwd
- /etc/security/user
- /etc/security/user.roles

When the NFSv4 client and server map user/group string names to UIDs/GIDs, they use the local OS authentication mechanism to make the mapping; thus, they will use the file-based I&A mechanism.

Table 14: Administrative Databases. This table lists other administrative files used to configure the TSF.

Database	Purpose
/etc/ftpusers	Limits access to FTP.
/etc/group	Stores group names, supplemental GIDs, and group members for all system groups.
/etc/hosts	Contains hostnames and their address for hosts in the network. This file is used to resolve a hostname into an Internet address in the absence of a domain name server.
/etc/inetd.conf	Configures start of network daemons.
/etc/inittab	Controls the system startup by running the appropriate command scripts and SRC invocations
/etc/krb5/krb5.conf	Specifies the default NAS (Kerberos) client configuration data.
/etc/nfs/security_default	Specifies the default NFSv4 client authentication data.
/etc/passwd	Stores user names, UIDs, primary GID, home directories for all system users.
/etc/security/acl	Specification of TCP port, host (or subnet), and user/group at that host or subnet allowed access to the port.
/etc/security/audit/bincmds	Specifies the pipeline of commands to be performed by the auditbin daemon.
/etc/security/audit/config	Specifies who and what is going to be audited, where the bin audit data will reside, and how auditing will be performed.
/etc/security/audit/events	Defines all of the audit events that are recognized by the system and the form of their tail data.
/etc/security/audit/objects	Specifies file system objects whose access is to be audited along with for what access modes it will be done.
/etc/security/authorizations	Contains the list of valid user-defined authorizations.
/etc/security/enc/LabelEncodings	Label mappings
/etc/security/lastlog	Stores time/date of last successful and unsuccessful login attempts for each user. Stores the number of unsuccessful login attempts since the last successful one.
/etc/security/ldap/ldap.cfg	Defines configuration attributes (including the SSL attributes) enforced by the LDAP client when LDAP-based I&A is used.
/etc/security/login.cfg	Defines attributes enforced when logging in or changing passwords.
/etc/security/mkuser.default	Defaults for user account creation.
/etc/security/passwd	Defines user passwords in one-way encrypted form, plus additional characteristics including previous passwords, password quality parameters.
/etc/security/portlog	Records ports locked as a result of login failures
/etc/security/priv	Defines the privileged ports as part of the access control on TCP ports
/etc/security/privcmds	Contains the security attributes for privileged commands.
/etc/security/privdevs	Contains the security attributes for privileged devices.
/etc/security/privfiles	Contains the security attributes for privileged files.
/etc/security/roles	Contains the RBAC role definitions for user-defined roles.
/etc/security/rules.host and /etc/security/rules.int	Configuration files for network port protection and labels. (LSPP mode only)
/etc/security/services	Specification of service names to be used by DACINET in the style of /etc/services.
/etc/security/tsd/tsd.dat	Integrity checking database (Trusted Signature Database)
/etc/security/user	Defines supplementary data about users, including audit status, required password characteristics, access to su command.
/etc/security/user.roles	Contains the mapping of users to roles.
/etc/wpar/devexports	Contains the list of exported devices to System WPARs.
/usr/ldap/clientkey.kdb	Contains the LDAP client's Kerberos key data. (The actual pathname of this file is specified by the administrator.)
/var/efs/*	Contains the key storage for EFS.

Database	Purpose
/var/krb5/security/creds/krb5cc_<pid>	Specifies the default NAS (Kerberos) client ticket file.
ODM attributes: mls_config and mls_operation	Specification of the kernel security flags for configuration and operational mode.
ODM attributes: ipsec_filter	Specifies the IP Filter rules used by the system.

These tables are not functions, but they are part of the management of the TSF.

The following databases are converted into binary format and loaded into the kernel:

- /etc/security/authorizations → KAT (Kernel Authorization Table)
- /etc/security/roles → KRT (Kernel Role Table)
- /etc/security/privcmds → KCT (Kernel Privileged Command Table)
- /etc/security/devices → KDT (Kernel Privileged Device Table)

6.2.14.6 Internal TOE Protection Mechanisms (TP.6)

All kernel software has access to all of memory, and the ability to execute all instructions. In general, however, only memory containing kernel data structures is manipulated by kernel software. Parameters are copied to and from process storage (i.e., that accessible outside the kernel) by explicit internal mechanisms, and those interfaces only refer to storage belonging to the process that invoked the kernel (e.g., by a system call). Functions implemented in trusted processes are more strongly isolated than the kernel. Because there is no explicit sharing of data, as there is in the kernel address space, all communications and interactions between trusted processes take place explicitly through files and similar mechanisms.

This encourages an architecture in which specific TSF functions are implemented by well-defined groups of processes.

6.2.14.7 Diagnosis (TP.7)

AIX provides a diagnosis program that can be used to check the correct operation of the underlying hardware of the system. This program can be executed by administrators or by hardware maintenance personnel. Results of the diagnosis program are stored in the diagnostic error log file, which can be protected by the discretionary access control functions of AIX against access by normal users.

6.2.14.8 File Integrity Verification & Integrity Checks (TP.8)

AIX provides a File Integrity Verification (FIV) mechanism that allows an administrator to check the integrity of the trusted files on the system. The database, called the Trusted Signature Database (TSD), is created during the build process of AIX. For each object listed in TSD, TSD contains the object owner, group, permission bits, hash value, and other security related information. The system administrator can use the *trustchk* command to verify the integrity of the objects listed in TSD. An administrator can also add, delete, and modify entries in TSD. The database is located at:

```
/etc/security/tsd/tsd.dat
```

The *trustchk* command verifies the attributes of the trusted file system objects by comparing them to the stored values in TSD. If the attributes of the object do not match what is stored in TSD, the command reports the error.

The AIX build process uses SHA-256 when creating a hash value (message digest) for each object. It then signs the hash values using a private key from a signing certificate that's unique to that version of AIX. These signatures are stored in TSD with each object. The public key is provided with the AIX installation image so that the signatures can be verified on the installed system.

TSD supports SHA-1, SHA-256, and SHA-512 message digests (hash algorithms). The CLiC module is used to implement the cryptographic functions of FIV.

6.2.14.9 File Security Flags (TP.9)

This section applies to LSPP mode only.

File security flags (FSF) are used to mark files with various types of information which are then evaluated as part of the checks implemented in the reference monitor. The file security flags supported in the evaluated configuration are identified and explained in the following table:

Table 15: File Security Flags (FSF)

FSF	**Semantics of the flag being enabled/set**
FSF_APPEND	If this FSF is set, a file can only be appended to and not altered otherwise in operational mode. In configuration mode, this can be overridden by the PV_TCB privilege.
FSF_AUDIT	Marks a file as being part of the audit subsystem and allows reading only if a process has the PV_AU_READ privilege and writing only with the PV_AU_WRITE privilege. (See also AU.7)
FSF_MAC_EXMPT	A process with the PV_MAC_OVRRD privilege will ignore MAC restrictions when attempting to access the file system object.
FSF_PDIR	Identifies a partitioned directory.
FSF_PSDIR	Identifies a partitioned subdirectory.
FSF_PSSDIR	Identifies a partitioned sub-subdirectory.
FSF_TLIB	The object is marked as part of the Trusted Library. In the evaluated configuration, this flag can only be changed when the system is in configuration mode or when the system security flag trustedlib_enabled is disabled.
FSF_TLIB_PROC	For executables, a process marked as a TLIB process will only be able to link to shared libraries (*.so) that have the FSF_TLIB flag set. In the evaluated configuration, this flag can only be changed when the system is in configuration mode or when the system security flag trustedlib_enabled is disabled.

6.2.14.10 NFSv4 Inter-TSF Communication (TP.10)

NFSv4 provides an optional inter-TSF trusted channel between NFSv4 clients and servers. To establish this channel, the TOE user contacts the Kerberos Version 5 server (contained in the TOE environment) to obtain a Kerberos ticket granting ticket (TGT) which is then used by the NFSv4 client to establish a trusted channel with the NFSv4 server. The channel created between the NFSv4 client and server use the SHA-1 message digest encrypting the data with one of the following encryption algorithms:

- 256 bit AES encryption in CTS mode
- 128 bit AES encryption in CTS mode
- 168 bit TDEA encryption in CBC mode

This communications channel uses the CLiC module on both the client and server side, but Kerberos uses its own encryption library.

Note that the Kerberos encryption library has not been FIPS certified nor has it been analyzed or tested to conform to cryptographic standards during this evaluation. All cryptography has only been asserted as tested by the vendor.

6.3 Supporting functions not part of the TSF

6.3.1 System Management Tools

The evaluated configuration of AIX provides the “System Management Interface Tool” (SMIT) for administration. This tool provides a more convenient way for an administrator to perform the administration activities. The web based administration tool WebSM is not included in the evaluated configuration and shall therefore not be used for system administration when operating the evaluated configuration of AIX.

SMIT itself is just a front-end tool which provides the administrator with a nice interface. SMIT generates scripts that use the system management commands provided by AIX. The administrator can review the shell scripts before they are executed by hitting the function key F6 before he executes the script.

In addition the administrator can use the commands provided by AIX for system management activities. Those commands are also seen as part of the system management tools.

This function contributes to satisfy the security requirements associated with the management of security attributes.

Note: System management tools and commands do not enforce any part of the TOE security policy. They just provide the tools for the administrator to perform his administrative functions. The TSF still check that the caller is allowed to invoke the system calls used by those tools and checks that the caller has the required access rights to the objects (like configuration files) he is going to access. SMIT generates a script with commands that the administrator can check before it is executed. Therefore SMIT itself is not seen as part of the TSF. The commands themselves are also bound by the restrictions imposed by the system call interface and the access rights to the files in the administrative database.

6.3.2 User Processes

The AIX TSF primarily exists to support the activities of user processes. A user, or non-TSF, process has no special privileges or security attributes. The user process is isolated from interference by other user processes primarily through the CPU execution state and address protection mechanisms and the way they are used by the kernel, and also through the protections on TSF interfaces for process and file manipulation.

User processes are by definition untrusted and therefore do not contribute to any security function. The TSF ensure that user processes are encapsulated in such a way that they are separated from the TSF and from processes (trusted and untrusted) running with different attributes and will only be able to communicate with them using the defined TSF interfaces. User processes therefore do not contribute to any security function of the TOE.

6.4 Assurance Measures

The following table provides an overview, how the assurance measures of EAL4 are met by AIX.

Table 16: Mapping Assurance Requirements to Documentation

Assurance Component	Documentation describing how the requirements are met
ACM_AUT.1	IBM uses automated CM tools with discretionary access control to provide configuration management for all evaluation evidence.
ACM_CAP.4	See above.
ACM_SCP.2	IBM has a problem tracking procedure in place that covers all aspects of ACM_SCP.2. This description is included in the documentation of the configuration management and the software development procedures.
ADO_DEL.2	IBM uses commercially established techniques for secure delivery of the TOE.
ADO_IGS.1	Installation, generation and start-up procedures are described as part of the guidance provided with the TOE.
ADV_FSP.2	A functional specification is provided.
ADV_HLD.2	A high-level design is provided.
ADV_IMP.1	Source code samples are provided as requested for the evaluation.
ADV_LLD.1	Low-level design documentation is provided for all subsystems that implement TSF.
ADV_RCR.1	The correspondence information is provided.
ADV_SPM.1	A Security Policy Model is provided.
AGD_ADM.1	Administrator guidance is provided with the TOE.
AGD_USR.1	User guidance is provided with the TOE.
ALC_DVS.1	The development security procedures are documented.
ALC_FLR.3	IBM has a mature customer support interface in place, including first, second and third level support and sophisticated means to automatically inform subscribed customers about potential security flaws and workarounds

Assurance Component	Documentation describing how the requirements are met
	and fixes available to prevent their exploitation.
ALC_LCD.1	The life cycle definition is described in separate documents.
ALC_TAT.1	See above.
ATE_COV.2	Detailed test plans and a test coverage and depth analysis are produced to test the TOE.
ATE_DPT.1	See above.
ATE_FUN.1	Testing is performed on different platforms that are defined in this Security Target. Test results are documented.
ATE_IND.2	The evaluation facility performs and documents independent tests.
AVA_MSU.2	The misuse analysis is provided.
AVA_SOF.1	The Strength of Function Analysis is provided.
AVA_VLA.2	A vulnerability analysis is provided.

6.5 TOE Security Functions Requiring a Strength of Function

Please refer to section 8.4.3 for a discussion of SOF-rated mechanisms.

7 Protection Profile Claims

7.1 PP Reference

This Security Target claims conformance with the following protection profiles:

- "Controlled Access Protection Profile" [CAPP]. This Protection Profile is listed on the TPEP web site of NSA as a "Certified Protection Profile".
- "Labeled Security Protection Profile" [LSPP] when the TOE is operated in LSPP mode. This Protection Profile is listed on the TPEP web site of NSA as a "Certified Protection Profile".
- "Role-Based Access Control Protection Profile" [RBACPP]. This Protection Profile is *not* listed as a certified protection profile.

7.2 PP Tailoring

Please refer to Table 4 and Table 8 for an identification of SFRs that have been added in addition to the ones derived from the protection profiles,(PPs) and for an identification of the operations performed on the SFRs derived from the PPs.

Two SFRs (FIA_UAU.1 and FIA_UID.1) defined in [CAPP] and [LSPP] have been substituted by hierarchically superior ones (FIA_UAU.2 and FIA_UID.2). This does not affect the compliance to these PPs. Since those components don't imply additional dependencies, the dependency analysis performed on these PPs still applies.

Additionally, [RBACPP] requires FMT_SMR.2 whereas [CAPP] and [LSPP] require only FMT_SMR.1. Since FMT_SMR.2 is hierarchically superior to FMT_SMR.1, FMT_SMR.2 was substituted for FMT_SMR.1. This does not affect the compliance to [CAPP] and [LSPP]. Since those components don't imply additional dependencies, the dependency analysis performed on these PPs still applies.

Section 8.1 provides rationale for the augmentation of the security problem definition in chapter 3.

Sections 8.1 and 8.2 contain the additional assumptions, organizational security policies, threats, and objectives that this Security Target uses that are in addition to those defined in the protection profiles.

The assurance requirements of the combined Protection Profiles are those defined in the Evaluation Assurance Level EAL2 and EAL3 of the Common Criteria plus ADV_SPM.1 (required by [RBACPP]). This Security Target specifies an Evaluation Assurance Level EAL 4 augmented by ALC_FLR.3. Since the Evaluation Assurance Levels in the Common Criteria define a hierarchy and EAL4 also requires ADV_SPM.1, all assurance requirements of the Protection Profiles are included in this Security Target. ALC_FLR.3 which has been added to the assurance requirements defined in the [CAPP], [LSPP], and [RBACPP] has no dependency on any other security functional requirement or security assurance requirement and is therefore an augmentation that has no effect on the security functional requirements or security assurance requirements stated in the Protection Profiles.

8 Rationale

The rationale section provides additional information and demonstrates that the security objectives and the security functions defined in the previous chapter are consistent and sufficient to counter the threats defined in chapter 3.

The rationale is based on the rationale already provided by [CAPP], [LSPP], and [RBACPP]. In accordance with the "Guide for the production of protection profiles and security targets" [GUIDE], only those aspects are discussed that are additional to the rationale provided by [CAPP], [LSPP], and [RBACPP].

8.1 Rationale for the Augmentation of the Security Problem Definition

This section and its subsections include the augmentations of the security problem definition.

8.1.1 Rationale for Organizational Security Policies

The following table contains the list of organizational security policies (OSPs) and their origin if from a protection profile. OSPs without a protection profile entry are specific to this ST. Rationale for OSPs from protection profiles are located in the associated protection profile(s).

Note that [RBACPP] threat T.ENTRY has been remapped to OSP P.AUTHORIZED_USERS.

Table 17: Mapping of Policies to PPs

OSP	Protection Profile	Rationale
P.ACCESS	[RBACPP]	
P.ACCOUNTABILITY	[CAPP], [LSPP]	
P.AUTHORIZED_USERS	[CAPP], [LSPP], [RBACPP – T.ENTRY]	
P.CLASSIFICATION	[LSPP]	
P.COMPROT		The TOE implements trusted communications between NFSv4 clients and servers.
P.DIST_USERS		The TOE requires all user accounts in a distributed environment to be defined consistently across all instances of the TOE.
P.ERASE		The TOE implements a policy for HDD erasure.
P.INTEGRITY		The TOE enforces mandatory integrity control (MIC).
P.NEED_TO_KNOW	[CAPP], [LSPP]	
P.STATIC		The TOE must run in a static LPAR, not in a dynamic LPAR (DLPAR).
P.TCBINTEGRITY		The TOE implements a trusted computing base mechanism.

8.1.2 Rationale for Assumptions

The following table contains the list of assumptions and their origins if from a protection profile. Assumptions without a protection profile entry are specific to this Security Target. Rationale for assumptions from protection profiles are located in the associated protection profile(s).

Table 18: Mapping of Assumptions to PPs

Assumption	Protection Profile	Rationale

Physical Assumptions		
A.ASSET	[RBACPP]	
A.LOCATE	[CAPP], [LSPP], [RBACPP]	
A.PROTECT	[CAPP], [LSPP], [RBACPP]	
Personnel Assumptions		
A.ACCESS	[RBACPP]	
A.COOP	[CAPP], [LSPP]	
A.MANAGE	[CAPP], [LSPP], [RBACPP]	
A.NO_EVIL_ADM	[CAPP], [LSPP]	
A.OWNER	[RBACPP]	
A.UTRAIN		Added as an assumption because it was also mentioned in the ITSEC Security Target for AIX. Users are trained well enough to use the security functionality provided by the system appropriately. This addresses the aspect of access control, where a user is responsible to manage access control rights for file system and IPC objects he owns. Users that need to protect their assets from unauthorized access by other authorized users of the system need to understand the implications of managing the access rights to file system objects and IPC objects they own. User need also to be trained in the protection of their authentication data in the TOE environment.
A.UTRUST		Added as an assumption because it was also mentioned in the ITSEC Security Target for AIX. Users are trusted for some tasks or group of tasks within a secure IT environment by exercising complete control over their data. This also addresses the aspect of access control where user are trusted to use the access control mechanism provided by the TOE appropriately. Users should not blame the system for the loss of integrity and / or confidentiality of data they own when they don't use the access control mechanism provided by AIX appropriately.
Connectivity Assumptions		
A.CONNECT	[CAPP], [LSPP], [RBACPP]	
A.KERB_KEY		Added to reflect that a Kerberos server can be used to generate encryption keys used by the NFSv4 Inter-TSF trusted communications.
A.KERB_PROTECT		Added to reflect that a Kerberos server can be used to hold principle names and passwords. This database contains the TSF data used for NFSv4 Inter-TSF trusted communications. It's assumed that the LDAP server used for the Kerberos database and associated network connections do not modify data and, furthermore, provide data protection.
A.LDAP_PROTECT		Added to reflect that an LDAP server can be used to house a

		distributed administrative database. This database contains the TSF data used for identification and authentication. It's assumed that the LDAP server and associated network connections do not modify data and, furthermore, provide data protection.
A.NET_COMP		Added to reflect the fact that the TOE is used in a distributed environment where network components are involved in the communication. It is assumed that those network components do not modify data transmitted over the network. This assumption is necessary since some TSF rely on the correctness of data transmitted over the network to remote external systems.
A.PEER	[CAPP], [LSPP]	
A.RSA_KEY		Added to reflect that the TOE uses RSA encryption keys generated by the IT environment. It is assumed that these keys are generated in a secure manner.
Procedural Assumptions		
A.CLEARANCE	[LSPP]	
A.SENSITIVITY	[LSPP]	

8.1.3 Rationale for the Inclusion of Threats

The [CAPP] and [LSPP] protection profiles have derived all security objectives from the organizational security policies listed in the protection profiles. The [RBACPP] protection profile defines the following TOE threats:

- T.ACCESS
- T.ENTRY (This threat has been replaced by P.AUTHORIZED_USERS in this ST.)
- T.OPERATE
- T.ROLEDEV

The authors of this Security Target decided to include also threats the TOE is going to counter as well as threats that need to be countered in the environment. Since all the policies and assumptions defined in the protection profiles have been included in the Security Target, the inclusion of threats is not a violation of the conformance claim to the Security Target.

8.2 Security Objectives Rationale

The following tables provide a mapping of security objectives to the environment defined by the threats, policies and assumptions, illustrating that each security objective covers at least one threat, assumption or policy and that each threat, assumption or policy is covered by at least one security objective.

8.2.1 Security Objectives Coverage

The following tables contain the list of objectives and environment objectives and their origins if from a protection profile. Objectives and environment objectives without a protection profile entry are specific to this Security Target. In some cases for [RBACPP], the objectives from [RBACPP] were remapped to different objectives. The original mapping is supplied in the Protection Profile column.

Table 19: Mapping Objectives to threats, assumptions, and policies

Objective	Protection Profile	Threat / Policy
O.AUDITING	[CAPP], [LSPP], [RBACPP - O.AUDIT & O.ACCOUNT]	T.UAACTION, T.UAUSER, P.ACCOUNTABILITY

Objective	Protection Profile	Threat / Policy
O.AUTHORIZATION	[CAPP], [LSPP], [RBACPP - O.ENTRY]	T.UAUSER, P.AUTHORIZED_USERS
O.COMPROT		P.COMPROT
O.CONFINE		T.UAACTION
O.DISCRETIONARY_ACCESS	[CAPP], [LSPP], [RBACPP - O.KNOWN]	T.ACCESS, P.NEED_TO_KNOW
O.DUTY	[RBACPP]	T.ROLEDEV
O.ENFORCEMENT	[CAPP], [LSPP]	P.AUTHORIZED_USERS, P.NEED_TO_KNOW, P.ACCOUNTABILITY, P.CLASSIFICATION, P.INTEGRITY, P.DIST_USERS
O.ERASE		P.ERASE
O.HIERARCHICAL	[RBACPP]	T.ROLEDEV
O.IP_FILTER		T.ACCESS, T.IP_PASSTHRU
O.MANAGE	[CAPP], [LSPP], [RBACPP - O.ADMIN]	P.AUTHORIZED_USERS, P.NEED_TO_KNOW, P.ACCOUNTABILITY, P.CLASSIFICATION, P.INTEGRITY, T.UAUSER, T.UAACTION, T.OPERATE
O.MANDATORY_ACCESS	[LSPP]	P.CLASSIFICATION
O.MANDATORY_INTEGRITY		P.INTEGRITY
O.NETWORK_ACCESS		P.CLASSIFICATION
O.RESIDUAL_INFORMATION	[CAPP], [LSPP]	P.CLASSIFICATION, P.NEED_TO_KNOW, T.ACCESS
O.ROLE	[RBACPP]	T,ROLEDEV, P.ACCESS
O.STACK		T.UAUSER, T.ACCESS, P.AUTHORIZED_USERS
O.TCB_ACCESS		T.ACCESS, P.TCBINTEGRITY
O.VIOS		T.VIOS

Table 20: Mapping objectives for the environment to threats, assumptions and policies

Environment Objective	Protection Profile	Threat / Assumption / Policy
OE.ADMIN		A.MANAGE, A. NO_EVIL_ADM
OE.CREDEN	[CAPP], [LSPP]	A.COOP
OE.HW_SEP		TE.HW_SEP
OE.INFO_PROTECT	[RBACPP - O.CONNECT]	TE.COR_FILE, A.PROTECT, A.UTRAIN, A.UTRUST, A.ASSET, A.ACCESS, A.OWNER, A.CLEARANCE, A.SENSITIVITY
OE.INSTALL	[CAPP], [LSPP], [RBACPP - O.INSTALL]	TE.COR_FILE, A.MANAGE, A.NO_EVIL_ADM, A.PEER, A.NET_COMP, P.STATIC
OE.KERB_BIND		TE.KERB_BIND
OE.KERB_KEY		A.KERB_KEY
OE.KERB_PROTECT		A.KERB_PROTECT
OE.LDAP_PROTECT		A.LDAP_PROTECT

OE.LPAR		TE.LPAR
OE.MAINTENANCE		TE.HWMF
OE.PHYSICAL	[CAPP], [LSPP], [RBACPP - O.PHYSICAL]	A.LOCATE, A.PROTECT, A.CONNECT
OE.PROTECT		TE.COR_FILE, A.NET_COMP, A.CONNECT
OE.RECOVER		A.MANAGE, TE.HWMF, TE.COR_FILE
OE.RSA_KEY		A.RSA_KEY
OE.SERIAL_LOGIN		A.CONNECT
OE.SOFTWARE_IN		P.NEED_TO_KNOW

Table 21: Mapping threats to objectives

Threat	Protection Profile	Objective
T.ACCESS	[RBACPP]	O.DISCRETIONARY_ACCESS, O.RESIDUAL_INFORMATION, O.TCB_ACCESS, O.STACK, O.IP_FILTER
T.IP_PASSTHRU		O.IP_FILTER
T.OPERATE	[RBACPP]	O.MANAGE
T.ROLEDEV	[RBACPP]	O.DUTY, O.HIERARCHICAL, O.ROLE,
T.UAACTION		O.AUDITING, O.CONFINE, O.MANAGE
T.UAUSER		O.AUDITING, O.AUTHORIZATION, O.MANAGE, O.STACK
T.VIOS		O.VIOS
TE.COR_FILE		OE.PROTECT, OE.INSTALL, OE.INFO_PROTECT, OE.RECOVER
TE.HW_SEP		OE.HW_SEP
TE.HWMF		OE.MAINTENANCE, OE.RECOVER
TE.KERB_BIND		OE.KERB_BIND
TE.LPAR		OE.LPAR

Table 22: Mapping Assumptions to Objectives

Assumption	Objective
Physical Assumptions	
A.ASSET	OE.INFO_PROTECT
A.LOCATE	OE.PHYSICAL
A.PROTECT	OE.INFO_PROTECT, OE.PHYSICAL
Personnel Assumptions	
A.ACCESS	OE.INFO_PROTECT
A.COOP	OE.CREDEN
A.MANAGE	OE.ADMIN, OE.INSTALL, OE.RECOVER
A.NO_EVIL_ADM	OE.ADMIN, OE.INSTALL
A.OWNER	OE.INFO_PROTECT
A.UTRAIN	OE.INFO_PROTECT
A.UTRUST	OE.INFO_PROTECT
Connectivity Assumptions	

A.CONNECT	OE.SERIAL_LOGIN, OE.PROTECT, OE.PHYSICAL
A.KERB_KEY	OE.KERB_KEY
A.KERB_PROTECT	OE.INSTALL, OE.KERB_PROTECT
A.LDAP_PROTECT	OE.INSTALL, OE.LDAP_PROTECT
A.NET_COMP	OE.PROTECT, OE.INSTALL
A.PEER	OE.INSTALL
A.RSA_KEY	OE.RSA_KEY
Procedural Assumptions	
A.CLEARANCE	OE.INFO_PROTECT
A.SENSITIVITY	OE.INFO_PROTECT

Table 23: Mapping Policies to Objectives

Policy	Objective
P.ACCESS	O.ROLE
P.ACCOUNTABILITY	O.AUDITING, O.MANAGE, O.ENFORCEMENT
P.AUTHORIZED_USERS	O.AUTHORIZATION, O.MANAGE, O.ENFORCEMENT, O.STACK
P.CLASSIFICATION	O.MANDATORY_ACCESS, O.MANAGE, O.ENFORCEMENT, O.NETWORK_ACCESS, O.RESIDUAL_INFORMATION
P.COMPROT	O.COMPROT
P.DIST_USERS	O.ENFORCEMENT
P.ERASE	O.ERASE
P.INTEGRITY	O.ENFORCEMENT, O.MANDATORY_INTEGRITY
P.NEED_TO_KNOW	O.DISCRETIONARY_ACCESS, O.MANAGE, O.ENFORCEMENT, O.RESIDUAL_INFORMATION, OE.SOFTWARE_IN
P.STATIC	OE.INSTALL
P.TCBINTEGRITY	O.TCB_ACCESS

8.2.2 Security Objectives Sufficiency

8.2.2.1 Threats

T.ACCESS: The threat of an authorized user of the TOE accessing information resources without the permission from the user responsible for the resource is removed by O.DISCRETIONARY_ACCESS requiring access control for resources and the ability for authorized users to specify the access to their resources. This ensures that a user can access a resource only if the requested type of access has been granted by the user responsible for the management of access rights to the resource. In addition O.RESIDUAL_INFORMATION ensures that an authorized user can not gain access to the information contained in a resource after the resource has been released to the system for reuse. O.TCB_ACCESS contributes further by providing a trusted computing base that allows modifications to TCB files only in maintenance (single user) mode. O.STACK ensures that a user cannot hijack a trusted process through the use of a buffer overflow type attack to gain access to a resource without the resource owner's permission. O.IP_FILTER ensures that authorized administrators can control IP packet flow received by the system and destined for the TOE.

T.IP_PASSTHRU: The treat of a user (authorized or unauthorized) using the TOE's network interfaces as an unauthorized pass through to another system is removed by O.IP_FILTER ensuring that an authorized administrator can decide/control which packets are passed through the TOE's network interfaces to other systems.

T.OPERATE: The threat of IT asset compromise due to improper administration and operation of the TOE is removed by O.MANAGE providing functions and facilities necessary to support the administrative users responsible for the management of TOE security.

T.ROLEDEV: The threat of developing and assigning user roles in a way that undermines security is removed by O.DUTY which provides the 'separation of duties' capability, O.HIERARCHICAL which supports defining roles in terms of other roles, and O.ROLE which limits access to and operations on resources and objects to members of authorized roles that permit those operations.

T.UAACTION: The threat of undetected security policy violation is removed by O.AUDITING requiring the TOE to collect evidence of security relevant actions and make it accessible to authorized administrators. O.MANAGE provides a system administrator with the capability to manage the audit system such that it is capable to monitor all critical aspects of the security policy. The threat of untrusted applications damaging user data and/or the security of the system can be limited via O.CONFINE which allows an application to be executed in a confined/restricted environment (WPAR).

T.UAUSER: The threat of impersonation of an authorized user by an attacker is sufficiently diminished by O.AUTHORIZATION requiring proper authorization of users gaining access to the TOE and O.AUDITING which requires the collection of evidence of security relevant actions, which includes authorization attempts. O.MANAGE ensures that only authorized administrators (which are assumed to be trustworthy) have the ability to add new users or modify the attributes of users. O.STACK ensures that a user (authorized or unauthorized) of a trusted process cannot maliciously execute code placed on the process stack through a buffer overflow type attack. Together those objectives ensure that no unauthorized user can impersonate as an authorized user.

T.VIOS: The threat of a VIOS SCSI device driver acting on behalf of an LPAR partition attempting to access a logical volume that's not assigned to the partition is removed by O.VIOS which provides access control between VIOS SCSI device drivers and logical volumes. Similarly, the threat of a VIOS Ethernet device driver acting on behalf of a group of LPAR partitions attempting to access a VIOS Ethernet adapter device driver and vice versa is removed by O.VIOS which provides access control between the two entries.

TE.COR_FILE: The threat of undetected loss of integrity of security enforcing or relevant files of the TOE is diminished by OE.INSTALL requiring procedures for secure distribution, installation and configuration of systems thereby ensuring that the system has a secure initial state with the required protection of such files, OE.PROTECT requiring protection of transferred data in a networked environment and OE.INFO_PROTECT requiring procedures for the appropriate protection of those files when the system is up and running. OE.RECOVER ensures that the system is securely recovered, which includes the verification of the integrity of security enforcing or security relevant files as part of the recovery procedures.

TE.HW_SEP: The threat that the underlying hardware does not provide the functions required to implement an efficient self-protection of the TSF such that the TSF themselves and the TSF data can be efficiently protected from unauthorized access and modification by untrusted software is addressed by the objective OE.HW_SEP for the processor used to execute the TOE software. This is a basic fundamental requirement for secure operating systems where trusted and untrusted software are executed on the same processor using the same memory space and the same processor resources. For TSF self-protection a processor feature is required that controls access to processor resources and main memory such that the TSF can implement a self-protection function in the way that the TSF reserve processor resources and memory areas for themselves and prohibit that those resources can be used by non-TSF software.

TE.HWMF: The threat of losing data due to hardware malfunction is mitigated by OE.MAINTENANCE requiring the invocation of diagnostic tools during preventative maintenance periods. In addition OE.RECOVER requires the organizational procedures to be set up that are able to recover critical data and restart operation in a secure mode in the case such a hardware malfunction happens.

TE.KERB_BIND: The threat that a Kerberos password may be guessed by repeated bind attempts to Kerberos is addressed by OE.KERB_BIND requiring the password complexity rules for Kerberos accounts to be equivalent to or greater than that of the TOE's.

TE.LPAR: The threat that software running in another logical partition than the TOE itself, therefore having different hardware resource of the same machine assigned to it than the TOE, is addressed by the objective OE.LPAR, requiring from the IT environment (i.e. the underlying machine) to successfully restrict access to resources that are assigned to one logical partition to the operating system running in that partition, therefore preventing access from software in other partitions to the TOE's logical partition.

8.2.2.2 Physical Assumptions

A.ASSET: The assumption on the value of the stored assets meriting moderately intrusive attacks is covered by OE.INFO_PROTECT which requires that protection mechanisms are configured properly.

A.LOCATE: The assumption on physical protection of the processing resources of the TOE is covered by OE.PHYSICAL requiring physical protection.

A.PROTECT: The assumption on physical protection of all hard- and software as well as the network and peripheral cabling is covered by the objectives OE.INFO_PROTECT demanding the approval of network and peripheral cabling and OE.PHYSICAL requiring physical protection. Note: Physical protection of the network components and cabling is required by A.PROTECT which may seem to be redundant to A.CONNECT. But A.CONNECT also addresses protection against passive wiretapping, which may be done without having physical access to a hardware component.

8.2.2.3 Personnel Assumptions

A.ACCESS: The assumption that roles accurately reflect the user's job function, responsibilities, qualifications, an/or competencies within the enterprise is covered by OE.INFO_PROTECT which requires that administrators are trained to perform configuration tasks properly.

A.COOP: The assumption on authorized users to act in a cooperating manner is covered by the objective OE.CREDEN requiring the safe storage and non-disclosure of authentication credentials.

A.MANAGE: The assumption on competent administrators is covered by OE.ADMIN requiring competent and trustworthy administrators and OE.INSTALL requiring procedures for secure distribution, installation and configuration of systems as well as OE.RECOVER requiring the administrator to perform all the required actions to bring the TOE into a secure state after a system failure or discontinuity.

A. NO_EVIL_ADM: The assumption on administrators that are neither careless nor willfully negligent or hostile is covered by OE.ADMIN requiring competent and trustworthy administrators and OE.INSTALL requiring procedures for secure distribution, installation and configuration of systems.

A.OWNER: The limited right of users to create and manage new data object is covered by OE.INFO_PROTECT which requires that users are trained to perform these tasks properly and not to pass on information to somebody without the right to access the information.

A.UTRAIN: The assumption on trained users is covered by OE.INFO_PROTECT which requires that users are trained to protect the data belonging to them.

A.UTRUST: The assumption on user to be trusted to protect data is covered by OE.INFO_PROTECT which requires that users are trusted to use the protection mechanisms of the TOE adequately to protect their data.

8.2.2.4 Connectivity Assumptions

A.CONNECT: The assumption on controlled access to peripheral devices and protected internal communication paths is covered by OE.SERIAL_LOGIN for the protection of attached serial login devices, OE.PROTECT for the protection of data transferred between workstations and OE.PHYSICAL requiring physical protection.

A.KERB_KEY: The assumption on Kerberos generating the encryption keys for NFSv4 Inter-TSF trusted communication is implemented by OE.KERB_KEY.

A.KERB_PROTECT: The assumption on Kerberos properly protecting the principle database requires that Kerberos be properly installed on the host in the distributed system through OE.INSTALL and that Kerberos provide access control to the principle data in the database through OE.KERB_PROTECT.

A.LDAP_PROTECT: The assumption on LDAP properly protecting the administrative database requires that LDAP be properly installed on the host in the distributed system through OE.INSTALL and that LDAP provide access control to the administrative data in the database through OE.LDAP_PROTECT.

A.NET_COMP: The assumption on network components to not modify transmitted data is covered by the objective OE.PROTECT requiring procedures and/or mechanisms to ensure a safe data transfer between systems as well as OE.INSTALL requiring proper installation and configuration of all parts of the distributed system thus including also components that are not part of the TOE.

A.PEER: The assumption on the same management control and security policy constraints for systems with which the TOE communicates is covered by OE.INSTALL requiring procedures for secure distribution, installation and configuration of the distributed system.

A.RSA_KEY: The assumption for generating the RSA encryption keys for SSL communication is implemented by OE.RSA_KEY.

8.2.2.5 Procedural Assumptions

A.CLEARANCE: The assumption on the procedures for granting authorization for access to specific security levels is covered by OE.INFO_PROTECT which requires that DAC and MAC protections are set up correctly and that users are trained to perform these tasks properly.

A.SENSITIVITY: The assumption on the procedures for establishing the security level of all information imported to or exported from the system including the security level of peripheral devices is covered by OE.INFO_PROTECT which requires that DAC and MAC protections are set up correctly and that users are trained to perform these tasks properly.

8.2.2.6 Organizational Security Policies

P.ACCESS: The access rights to specific objects are determined by O.ROLE which provides role-based access control.

P.ACCOUNTABILITY: The policy demanding accounting for user actions is implemented by O.AUDITING requiring auditing for security relevant actions and supported by O.MANAGE for the management of this functionality and O.ENFORCEMENT ensuring the correct invocation.

P.AUTHORIZED_USERS: The policy demanding that users have to be authorized for access to the system is implemented by O.AUTHORIZATION and supported by O.MANAGE allowing the management of these functions and O.ENFORCEMENT and O.STACK ensuring the correct invocation of the functions.

P.CLASSIFICATION: The policy demanding access control based on sensitivity label is implemented by O.MANDATORY_ACCESS requiring labeling and associated access control. The O.RESIDUAL_INFORMATION objective ensures that information will not be given to users that do not have a cleared access, when resources are reused. The O.MANAGE supports this policy by requiring only authorized administrators manage the functions and O.ENFORCEMENT ensures that functions are invoked and operate correctly. The O.NETWORK_ACCESS security objective supports this policy when the TOE communications with other systems across a network.

P.COMPROT: The policy ensures that administrators can, at their own discretion, include trusted communications between NFSv4 client and servers and, when using LDAP-based authentication, between the TOE and the LDAP server as implemented by O.COMPROT.

P.DIST_USERS: The policy ensures that user accounts are defined consistently across all instances of the TOE in a distributed environment. This allows all the TOEs to consistently enforce access control across all TOE instances as supported by O.ENFORCEMENT especially in the case where NFS is involved.

P.ERASE: The policy asks for the provision of a hard disk erase function, which is implemented by O.ERASE requiring the TOE to offer overwriting of hard disk drives with bit patterns that prevent the recovery of the original information stored on the disks.

P.INTEGRITY: The policy ensures that the integrity level of data is maintained. This is implemented by O.MANDATORY_INTEGRITY using MIC and is enforced by O.ENFORCEMENT which ensures that the functions are invoked and operate correctly.

P.NEED_TO_KNOW: The policy to restrict access to and modification of information to authorized users which have a “need to know” for that information is implemented by O.DISCRETIONARY_ACCESS demanding an appropriate access control. It is supported by O.RESIDUAL_INFORMATION ensuring that resources do not release such information during reuse and by OE.SOFTWARE_IN preventing users other than authorized administrators from installing new software that might affect the access control functionality. O.MANAGE allows administrators to manage this functions, O. ENFORCEMENT ensures that the functions are invoked and operate correctly.

P.STATIC: The policy ensures, by demanding appropriate organizational measures, that no additional object reuse issues are imposed with the support of logical partitioning by the TOE: while the TOE provides functionality to support the dynamic allocation and release of hardware resources during operation, such dynamic partitioning (DLPAR) by the use of a Hardware Management Console for the underlying hardware must not be performed by an administrator while the TOE is running. This is implemented by OE.INSTALL, demanding a secure installation and configuration of TOE systems, which applies also to the underlying hardware.

P.TCBINTEGRITY: The integrity of the information belonging to the Trusted Computing Base is protected and verified by the objective O.TCB_ACCESS, which mandates the protections of files belonging to the TCB.

8.3 Security Requirements Rationale

SFRs for the TOE have been selected in addition to [CAPP] and [LSPP] in this ST to address additional functionality provided by the TOE without conflicting with [CAPP] and [LSPP]'s TSP.

No security functions for the non-IT environment have been added, since the procedures that need to be implemented can (and probably will) be different for each site running the evaluated version of AIX. Therefore no specific security functional requirements and security functions for the non-IT environment have been defined in this Security Target. Individual sites running AIX should validate that the procedures and physical security measures they have put in place are sufficient to cover the security objectives defined for the environment of the TOE in this Security Target.

Security requirements for the IT environment have been added to define the support required by the TOE from the underlying processor.

As with every operating system that also runs untrusted software, some kind of separation mechanism must exists that prohibits the untrusted software from tampering with trusted software and TSF data. In the case of this TOE the processor must supply a separation mechanism such that memory areas as well as hardware privileges required to directly access devices or memory management functions are protected from direct access by untrusted software. This is defined with an access control policy called "Memory Access Control Policy" that the underlying processor must support. This policy is expressed using FDP_ACC.1 and FDP_ACF.1 as well as FDP_MSA.3 from part 2 of the Common Criteria. Section 8.3.3 provides more detailed rationale for the selection of the security functional requirements for the IT environment.

8.3.1 Security Requirements Coverage

Section 7.2.2 of both [CAPP] and [LSPP] provides a table mapping security objectives for the TOE to the security functional requirements. In addition, this section of the Protection Profiles also provides a discussion with the detailed evidence of the coverage for each security objective. In accordance with [GUIDE] this rationale is seen as sufficient for discussion of the coverage of security requirements for those objectives and security requirements taken directly from the Protection Profile.

In addition to the rationale provided in the Protection Profiles, the following table shows that each security functional requirement, including the SFRs that have been selected in addition to those from LSPP, addresses at least one objective.

Table 24: Mapping Security Functional Requirements to Objectives

SFR	Objectives
FAU_GEN.1	O.AUDITING
FAU_GEN.2	O.AUDITING
FAU_SAR.1	O.AUDITING O.MANAGE
FAU_SAR.2	O.AUDITING
FAU_SAR.3	O.AUDITING O.MANAGE
FAU_SEL.1	O.AUDITING O.MANAGE
FAU_STG.1	O.AUDITING
FAU_STG.3	O.AUDITING O.MANAGE
FAU_STG.4	O.AUDITING O.MANAGE
FCS_CKM.1(1) SSL	O.COMPROT
FCS_CKM.1(2)	O.DISCRETIONARY_ACCESS
FCS_CKM.1(3)	O.DISCRETIONARY_ACCESS

SFR	Objectives
FCS_CKM.2(1) SSL	O.COMPROT
FCS_CKM.2(2) Kerberos	O.COMPROT
FCS_COP.1(1) SSL	O.COMPROT
FCS_COP.1(2) SSL	O.COMPROT
FCS_COP.1(3) Kerberos	O.COMPROT
FCS_COP.1(4)	O.COMPROT
FCS_COP.1(5)	O.DISCRETIONARY_ACCESS
FCS_COP.1(6)	O.DISCRETIONARY_ACCESS O.TCB_ACCESS
FCS_COP.1(7)	O.DISCRETIONARY_ACCESS O.TCB_ACCESS
FCS_COP.1(8)	O.DISCRETIONARY_ACCESS
FDP_ACC.1(1)	O.DISCRETIONARY_ACCESS
FDP_ACC.1(2)	O.TCB_ACCESS
FDP_ACC.1(3)	O.AUTHORIZATION
FDP_ACC.1(4)	O.ROLE
FDP_ACC.1(5)	O.VIOS
FDP_ACF.1(1)	O.DISCRETIONARY_ACCESS
FDP_ACF.1(2)	O.TCB_ACCESS
FDP_ACF.1(3)	O.AUTHORIZATION
FDP_ACF.1(4)	O.ROLE
FDP_ACF.1(5)	O.VIOS
FDP_ETC.1	O.MANDATORY_ACCESS
FDP_ETC.2	O.MANDATORY_ACCESS
FDP_IFC.1(1)	O.MANDATORY_ACCESS
FDP_IFC.1(2)	O.MANDATORY_INTEGRITY
FDP_IFC.1(3)	O.NETWORK_ACCESS
FDP_IFC.1(4) WPAR	O.CONFINE
FDP_IFC.1(5)	O.IP_FILTER
FDP_IFF.1(1) WPAR	O.CONFINE
FDP_IFF.1(2)	O.IP_FILTER
FDP_IFF.2(1)	O.MANDATORY_ACCESS
FDP_IFF.2(2)	O.NETWORK_ACCESS
FDP_IFF.2(3)	O.MANDATORY_INTEGRITY
FDP_ITC.1	O.MANDATORY_ACCESS
FDP_ITC.2	O.MANDATORY_ACCESS
FDP_RIP.2	O.RESIDUAL_INFORMATION
Note 1	O.RESIDUAL_INFORMATION
FDP_RIP.3-AIX	O.ERASE
FIA_ATD.1(1)	O.AUTHORIZATION O.DISCRETIONARY_ACCESS

SFR	Objectives
	O.MANDATORY_ACCESS O.MANDATORY_INTEGRITY O.AUDITING O.TCB_ACCESS O.NETWORK_ACCESS O.ROLE
FIA_ATD.1(2)	O.AUTHORIZATION
FIA_SOS.1	O.AUTHORIZATION
FIA_UAU.2	O.AUTHORIZATION
FIA_UAU.7	O.AUTHORIZATION
FIA_UID.2	O.AUTHORIZATION
FIA_USB.1(1)	O.DISCRETIONARY_ACCESS O.AUDITING O.AUTHORIZATION O.MANDATORY_ACCESS O.MANDATORY_INTEGRITY O.TCB_ACCESS O.NETWORK_ACCESS O.ROLE
FIA_USB.1(2)	O.AUTHORIZATION
FMT_MOF.1(1)	O.MANAGE
FMT_MOF.1(2)	O.MANAGE
FMT_MSA.1(1)	O.DISCRETIONARY_ACCESS O.MANDATORY_ACCESS
FMT_MSA.1(2)	O.NETWORK_ACCESS
FMT_MSA.1(3)	O.TCB_ACCESS
FMT_MSA.1(4)	O.MANDATORY_INTEGRITY
FMT_MSA.1(5)	O.AUTHORIZATION
FMT_MSA.1(6) RBAC	O.ROLE O.MANAGE
FMT_MSA.1(7) RBAC	O.ROLE O.MANAGE
FMT_MSA.1(8) RBAC	O.ROLE O.MANAGE
FMT_MSA.1(9) RBAC	O.ROLE O.MANAGE
FMT_MSA.1(10)	O.VIOS
FMT_MSA.1(11) WPAR	O.CONFINE
FMT_MSA.2 RBAC	O.AUTHORIZATION O.COMPROT
FMT_MSA.3(1)	O.DISCRETIONARY_ACCESS O.MANDATORY_ACCESS
FMT_MSA.3(2)	O.NETWORK_ACCESS
FMT_MSA.3(3)	O.TCB_ACCESS
FMT_MSA.3(4)	O.MANDATORY_INTEGRITY
FMT_MSA.3(5)	O.AUTHORIZATION
FMT_MSA.3(6) RBAC	O.ROLE O.MANAGE
FMT_MSA.3(7)	O.VIOS
FMT_MSA.3(8)	O.CONFINE

SFR	Objectives
WPAR	
FMT_MSA.3(9)	O.IP_FILTER
FMT_MTD.1(1)	O.AUDITING O.MANAGE
FMT_MTD.1(2)	O.AUDITING O.MANAGE
FMT_MTD.1(3)	O.AUDITING O.MANAGE
FMT_MTD.1(4)	O.MANAGE
FMT_MTD.1(5)	O.AUTHORIZATION O.MANAGE
FMT_MTD.1(6)	O.ENFORCEMENT O.MANAGE
FMT_MTD.1(7) RBAC	O.HIERARCHICAL O.MANAGE
FMT_MTD.1(8)	O.AUTHORIZATION O.MANAGE
FMT_MTD.1(9)	O.CONFINE O.MANAGE
FMT_MTD.1(10)	O.CONFINE
FMT_MTD.1(11)	O.IP_FILTER O.MANAGE
FMT_MTD.3	O.AUTHORIZATION
FMT_REV.1(1)	O.MANAGE
FMT_REV.1(2)	O.DISCRETIONARY_ACCESS O.MANDATORY_ACCESS O.ROLE O.MANAGE
FMT_REV.1(3)	O.MANAGE
FMT_SMF.1	O.MANAGE
FMT_SMR.1	O.MANAGE O.VIOS
FMT_SMR.2	O.MANAGE O.AUDITING O.DUTY O.ROLE
FPT_AMT.1	O.ENFORCEMENT
FPT_FLS.1	O.ENFORCEMENT
FPT_RCV.1	O.MANAGE
FPT_RCV.4	O.MANAGE
FPT_RVM.1	O.ENFORCEMENT
FPT_RVM.2-AIX	O.STACK
FPT_SEP.1	O.ENFORCEMENT
FPT_STM.1	O.AUDITING
FPT_TDC.1	O.NETWORK_ACCESS
FPT_TST.1	O.ENFORCEMENT O.TCB_ACCESS
FTA_LSA.1	O.ROLE
FTA_TSE.1	O.ROLE
FTP_ITC.1	O.COMPROT

Table 25: Mapping Security Functional Requirements for the IT Environment to Objectives

SFR	Objective
FCS_CKM.1(a)	OE.RSA_KEY
FCS_CKM.1(b)	OE.KERB_KEY
FDP_ACC.1(a)	OE.HW_SEP
FDP_ACC.1(b)	OE.LPAR
FDP_ACF.1(a)	OE.HW_SEP
FDP_ACF.1(b)	OE.LPAR
FIA_SOS.1(a)	OE.KERB_BIND
FIA_UID.2(a)	OE.LPAR
FMT_MSA.3(a)	OE.HW_SEP

8.3.2 Security Requirements Sufficiency for the TOE

This section discusses how the single objectives defined for the TOE are met by the SFRs selected in this ST. Whenever such rationale has already been sufficiently shown in [CAPP] and/or [LSPP] for an objective, only the additional aspects introduced by augmentations to [CAPP] and [LSPP] in this ST are discussed.

O.AUDITING is implemented by the SFRs discussed in [CAPP] and [LSPP]. In addition, it is supported by the security attributes required in FIA_ATD.1(1) and the management actions for auditing specified in FMT_MTD.1(3).

O.AUTHORIZATION is implemented by the SFRs discussed in [CAPP] and [LSPP]. An additional authorization mechanism enforced by the TOE is modeled as a discretionary access control policy in FDP_ACF.1(3) and called out in FDP_ACC.1(3), with supporting management functions in FMT_MSA.1(5) and FMT_MSA.3(5) and the security attributes required in FIA_ATD.1(1). For RBAC, only secure values for security attributes are used FMT_MSA.2 and accepted FMT_MTD.3. For VIOS, the security attributes are defined by FIA_ATD.1(2) and the user-subject binding by FIA_USB.1(2).

O.COMPROT is implemented by the cryptographic SFRs used for communications (FCS_CKM.1(1), FCS_CKM.2(1), FCS_CKM.2(2), FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), and FCS_COP.1(4)), the trusted channel SFR FTP_ITC.1, and secure security attribute SFR FMT_MSA.2.

O.CONFINE is addressed by the WPAR related SFRs for information flow control FDP_IFC.1(4) and FDP_IFF.1(1) and include the WPAR security attributes in FMT_MSA.1(11), their default value initialization in FMT_MSA.3(8), the privileges in FMT_MTD.1(9), and the corral ID in FMT_MTD.1(10).

O.DISCRETIONARY_ACCESS is implemented by the SFRs discussed in [CAPP] and [LSPP]. In addition, the Encrypted File System performs access control based on knowledge of the encryption key through FCS_CKM.1(2, 3) and FCS_COP.1(5, 6, 7, 8).

O.DUTY addresses the fact that the TOE must provide the capability of enforcing ‘separation of duties’. The enforcement of role separation by FMT_SMR.2 supports this objective.

O.ENFORCEMENT is implemented by the SFRs discussed in [CAPP] and [LSPP] and supported by the additional privilege enforcement mechanism implemented in the TOE, which are represented by the requirements in FPT_RVM.1 for a reference mechanism and additional management functionality in FMT_MTD.1(6). Verification of the TCB access control (see O.TCB_ACCESS) is supported by functionality to test the TCB’s integrity in FPT_TST.1. The TSF must support entering a fail secure mode on critical errors as defined by FPT_FLS.1.

O.ERASE is addressed by providing a mechanism to overwrite residual information on hard disk drives upon request of administrators in FDP_RIP.3-AIX.

O.HIERARCHICAL addresses the fact that the TOE must provide the capability of defining hierarchical roles as required by FMT_MTD.1(7).

O.IP_FILTER is implemented by the flow control SFRs FDP_IFC.1(5) and FDP_IFF.1(2) which provide the capability for the TOE to filter on several different IP packet attributes and includes the IP filter security attributes in

FMT_MSA.1(12) along with their default value initialization in FMT_MSA.3(9). The management actions for IP filtering are specified in FMT_MTD.1(11).

O.MANAGE is implemented by the SFRs discussed in [CAPP] and [LSPP] and by additional management functionality for the security functions that have been introduced in addition to the [CAPP] and [LSPP] requirements in FMT_MTD.1(3) and FMT_MTD.1(6). Security function management is called out in FMT_SMF.1. Authorized administrators can enable/disable auditing and specify events via FMT_MOF.1(1) and manage user authentication data via FMT_MOF.1(2). Users, especially administrators, can manage the roles of a user via FMT_MSA.1(6, 7, 8, 9) and FMT_REV.1(2). Administrators can define default values for security attributes as per FMT_MSA.3(6). TSF data can be managed via roles as per FMT_MTD.1(7). Administrators can also manage and recover from cases where there’s an RBAC database failure as per FPT_RCV.1 and FPT_RCV.4. WPAR security attributes can be managed by administrators via FMT_MTD.1(9). For VIOS, the roles are defined by FMT_SMR.1 with user attribute revocation defined by FMT_REV.1(3). For IP filtering, the IP filter rules are managed by administrators via FMT_MTD.1(11).

O.MANDATORY_ACCESS is implemented by the SFRs discussed in [LSPP], supported by security attributes required in FIA_ATD.1(1) and rules for revocation of security attributes related to MAC in FMT_REV.1(2).

O.MANDATORY_INTEGRITY is implemented by SFRs defining the mandatory integrity control policy in FDP_IFC.1(2) and FDP_IFF.2(3), user-subject binding in FIA_USB.1(1) and security attributes spelled out in FIA_ATD.1(1), as well as management functionality in FMT_MSA.1(4) and FMT_MSA.3(4).

O.NETWORK_ACCESS is implemented by the TOE’s Trusted Network policy, which has been defined in FDP_IFC.1(3) and FDP_IFF.2(2), with security attributes in FIA_ATD.1(1) and user-subject binding in FIA_USB.1(1) and management functionality in FMT_MSA.1(2) and FMT_MSA.3(2). A requirement for consistent interpretation of security labels in networked sessions is spelled out in FPT_TDC.1.

O.RESIDUAL_INFORMATION is implemented by the SFRs discussed in [CAPP] and [LSPP].

O.ROLE - The TOE must prevent users from gaining access to and performing operations on its resources/objects unless they have been granted access by the resource/object owner or they have been assigned to a role (by an authorized administrator) which permits those operations FMT_SMR.2. Role-based access control must have a defined scope of control FDP_ACC.1(4). The rules of the RBAC policy must be defined (FDP_ACF.1(4)). The security attributes of objects used to enforce the RBAC policy must be defined. The security attributes of subjects used to enforce the RBAC policy must be defined (FIA_ATD.1(1), FIA_USB.1(1)). Authorized users must be able to control who has access to objects (FMT_MSA.1(6, 7, 8, 9)) and be able to revoke that access (FMT_REV.1(2)). Protection of named objects must be continuous, starting from object creation (FMT_MSA.3(6)).

O.STACK is implemented by the TOE’s Stack Execution Disable (SED) feature which allows an authorized administrator to disable the execution of code residing on the stack of selected processes in FPT_RVM.2-AIX.

O.TCB_ACCESS is implemented by the TOE’s trusted computing base, which is modeled in FDP_ACC.1(2) and FDP_ACF.1(2), with security attributes in FIA_ATD.1(1) and user-subject binding in FIA_USB.1(1) and management functionality in FMT_MSA.1(3) and FMT_MSA.3(3) and a mechanism for testing the TCB integrity in FPT_TST.1. The TCB uses encryption and message digests to provide integrity checks via FCS_COP.1(6, 7).

O.VIOS is implemented by the TOE and modeled in FDP_ACC.1(5) and FDP_ACF.1(5) with management functionality in FMT_MSA.1(10) and FMT_MSA.3(7). The roles are defined by FMT_SMR.1.

8.3.3 Rationale for Security Requirements for the IT environment

In addition to the requirements of [CAPP], [LSPP], and [RBACPP], this Security Target has added security requirements for the IT environment. Those requirements define the need for an access control policy implemented in the underlying processor that allows to reserve the access and manipulation of critical processor and memory resources to specially software (instructions) operating with a defined privilege attribute (usually called “supervisor” or “system” mode). The TSF have to ensure that no untrusted software will ever execute with this privilege. Based on this the TSF can then control the access to memory objects and other processor resources and implement the high level access control functions as well as the TSF self protection.

To do this the underlying processor has to provide a basic access control mechanism where access to processor resources (like registers) and memory areas is controlled based on a processor attribute where the implementation of the TSF ensure that untrusted software never executes with this attribute. This is expressed with FDP_ACC.1(a) and FDP_ACF.1(a). Since the processor may allow read access to specific registers for software running without “supervisor” privilege, FDP_ACF.1.3 of FDP_ACF.1(a) is used to define this.

The requirements don't define the exact rules because those may differ slightly for different processor types without getting into the problem of interoperability problems. For example a new processor may implement additional instructions and additional register but still be fully downwards compatible. Since software developed for the older versions of the processor will not use the additional instructions and will not touch the additional register, the claims for the software still hold although the objects controlled by the new processor differ from those controlled by the old processor. Of course, if anybody wants to evaluate a PowerPC based processor those rules have to be defined precisely for the specific processor type that is the target of the hardware evaluation.

The "static attribute initialization" (FMT_MSA.3(a)) is here defined as the value of the processor attribute ("user" or "supervisor") at the start-up of the processor (after reset or power-up). This has to be "permissive" since the register and memory areas need to be initialized. It is therefore necessary that the software that perform those initialization activities is part of the TSF.

The security requirements for the IT environment address the security objective OE.HW_SEP since the Memory Access Control Policy allows the TOE to protect the TSF and the TSF data from unauthorized access by untrusted software. The TOE has to use the Memory Access Control Policy to allow memory access by untrusted software just to those memory areas that belong to the untrusted software itself. Access to special hardware register will be managed by the TSF such that this access will always be reserved to trusted software. This shows that the security requirements for the IT environment are sufficient to protect the TSF and TSF data from unauthorized access and modification when used correctly by the TOE.

In addition, the security requirements FDP_ACC.1(b) and FDP_ACF.1(b) have been chosen to express the need for an access control policy implemented in the underlying hardware to regulate access to parts of the hardware that are assigned to different logical partitions (LPARs). If an LPAR enabled underlying machine allows to run several operating systems in different logical partitions, with dedicated hardware resources assigned to those partitions, means are required to prevent the operating system running in one partition from accessing the resources assigned to another operating system running on the same machine. The hypervisor must also be able to identify each LPAR partition in order to allow communication channels to be setup between partitions (FIA_UID.2(a)) as well as to know the resources allocated to each partition.

Since the underlying hardware for the TOE provides LPAR support, access to the TSF and TSF data from other logical partitions than the one that belongs to the TOE must be prevented. Such protection has to be provided by the IT environment, which is expressed in the security requirements meeting the objective OE.LPAR.

The security requirement of FIA_SOS.1(a) has been added to express the need to protect the Kerberos accounts from password guessing attempts when binding to the Kerberos server, which is part of the TOE environment. The strength must be at least equivalent to that of the TOE.

The security requirement for FCS_CKM.1(a) has been added to express the need for the environment to generate RSA encryption keys for the SSL communications.

The security requirement for FCS_CKM.1(b) has been added to express the need for the Kerberos server to generate encryption keys for the NFSv4 Inter-TSF trusted communications.

8.3.4 Justification of Explicitly Expressed Security Requirements

The explicit requirement Note 1 is adopted from the [LSPP], the substantiation of the LSPP applies: The CC's FDP_RIP components only specify resources being allocated to objects and does not address resources used directly by subjects, such as memory or registers. This explicit requirement was added to ensure coverage of these resources. The words are identical to FDP_RIP.2 except "subject" replaces "object".

The name "Note 1" has been adopted from the Protection Profile for an easy mapping to the requirements defined there. It is known to the authors of this Security Target that this name is not compliant with the recommendations for the naming of components additional to the ones defined in part two of the CC.

The explicit requirement FDP_RIP.3-AIX has been introduced as a response to the objective O.ERASE. While it might have been possible to rush the implementation of this objective by using the existing components from the FDP_RIP family, the ST author felt that it was necessary to distinguish between the object reuse properties that are inherent in the management of shared resources of a TOE, as defined in FDP_RIP.1 and .2, and the administrator-invoke-able functionality to make residual information unavailable "on demand" before e.g. removing a resource from a system.

The explicit requirement FPT_RVM.2-AIX has been introduced in response to the objective O.STACK. This SFR describes an extended capability of AIX that improves the reference mediation of AIX beyond that described by

FPT_RVM.1. Specifically, it helps prevent the misuse of the TOE (specifically trusted processes) against its own programmatic shortcomings and possible compromise.

8.3.5 Security Requirements Dependency Analysis

This dependency analysis takes into account only the SFRs that have been selected in addition to those derived from [CAPP] and [LSPP]. With respect to dependencies in [CAPP] and [LSPP], the ST authors rely on the evaluation of the registered PP, noting that since then a dependency of components in [CAPP] and [LSPP] on FMT_SMF.1 has been introduced and is fulfilled by this ST, and that FIA_UAU.1 and FIA_UID.1 as defined in [CAPP] and [LSPP] have been replaced by the hierarchically higher components FIA_UAU.2 and FIA_UID.2. FIA_UAU.1 and FIA_UAU.2 have the same dependency on FIA_UID.1, which is resolved by the inclusion of FIA_UID.2, which is hierarchical to FIA_UID.1. FIA_UID.1 and FIA_UID.2 both have no dependency. Additionally, [RBACPP] uses FMT_SMF.2 which is hierarchically higher than FMT_SMF.1, so FMT_SMF.1 of [CAPP] and [LSPP] has been replaced with FMT_SMF.2.

Since no other additional security functional requirements to those defined in [CAPP] and [LSPP] have been defined for the TOE, the dependency analysis for the security functional requirements in section 7.3 of the Protection Profile applies for all security requirements taken from the LSPP.

Table 26: Dependency analysis for the TOE SFRs

SFR	Origin	Dependencies	Resolved?
FCS_CKM.1(1) (SSL session keys GSKit)	CC Part 2	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4 FMT_MSA.2	no FCS_COP.1(1) FMT_MSA.2
FCS_CKM.1(2) (symmetric keys EFS)	CC Part 2	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4 FMT_MSA.2	no FCS_COP.1(5) FMT_MSA.2
FCS_CKM.1(3) (RSA keys EFS)	CC Part 2	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4 FMT_MSA.2	no FCS_COP.1(6) FMT_MSA.2
FCS_CKM.2(1) (SSL RSA of session keys)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FCS_CKM.1(1) FMT_MSA.2
FCS_CKM.2(2) (Kererbos)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FCS_CKM.1(b)
FCS_COP.1(1) (SSL symmetric)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FCS_CKM.1(1) FMT_MSA.2
FCS_COP.1(2) (SSL RSA)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FCS_CKM.1(a)
FCS_COP.1(3) (NFSv4 AES, TDEA, SHA1)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FCS_CKM.1(b)
FCS_COP.1(4) (Kerberos AES, TDEA)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or	no

SFR	Origin	Dependencies	Resolved?
		FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	FCS_CKM.1(b)
FCS_COP.1(5) (Symmetric EFS)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FCS_CKM.1(2) FMT_MSA.2
FCS_COP.1(6) (Asymmetric EFS, FIV)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FCS_CKM.1(3) FMT_MSA.2
FCS_COP.1(7) (Digests EFS, FIV)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FMT_MSA.2
FCS_COP.1(8) (RNG EFS)	CC Part 2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	no FMT_MSA.2
FDP_ACC.1(2)	CC Part 2	FDP_ACF.1	FDP_ACF.1(2)
FDP_ACC.1(3)	CC Part 2	FDP_ACF.1	FDP_ACF.1(3)
FDP_ACC.1(4)	RBACPP	FDP_ACF.1	FDP_ACF.1(4)
FDP_ACC.1(5)	CC Part 2	FDP_ACF.1	FDP_ACF.1(5)

FDP_ACF.1(2)	CC Part 2	FDP_ACC.1
		FMT_MSA.3
			FDP_ACC.1(2)
			FMT_MSA.3(3)

FDP_ACF.1(3)	CC Part 2	FDP_ACC.1
		FMT_MSA.3
			FDP_ACC.1(3)
			FMT_MSA.3(5)

FDP_ACF.1(4)	RBACPP	FDP_ACC.1
		FMT_MSA.3
			FDP_ACC.1(4)
			FMT_MSA.3(6)

FDP_ACF.1(5)	CC Part 2	FDP_ACC.1
		FMT_MSA.3
			FDP_ACC.1(5)
			FMT_MSA.3(7)

FDP_IFC.1(2)	CC Part 2	FDP_IFF.1	FDP_IFF.2(3)
FDP_IFC.1(3)	CC Part 2	FDP_IFF.1	FDP_IFF.2(2)
FDP_IFC.1(4) (WPAR)	CC Part 2	FDP_IFF.1	FDP_IFF.1(1)
FDP_IFC.1(5) (IP Filter)	CC Part 2	FDP_IFF.1	FDP_IFF.1(2)

FDP_IFF.1(1)
(WPAR)
	CC Part 2	FDP_IFC.1
		FMT_MSA.3
			FDP_IFC.1(4)
			FMT_MSA.3(8)

FDP_IFF.1(2)
(IP Filter)
	CC Part 2	FDP_IFC.1
		FMT_MSA.3
			FDP_IFC.1(5)
			FMT_MSA.3(9)

FDP_IFF.2(2)	CC Part 2	FDP_IFC.1
		FMT_MSA.3
			FDP_IFC.1(3)
			FMT_MSA.3(2)

FDP_IFF.2(3)	CC Part 2	FDP_IFC.1
		FMT_MSA.3
			FDP_IFC.1(2)
			FMT_MSA.3(4)

FDP_RIP.3-AIX	ECD (section 5.1)	None	yes
FIA_ATD.1(2)	CC Part 2	None	yes
FIA_USB.1(2)	CC Part 2	FIA_ATD.1	FIA_ATD.1(2)
FMT_MOF.1(1)	MLOSPP	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MOF.1(2)	MLOSPP	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(2)	CC Part 2	[FDP_ACC.1 or	FDP_IFC.1(3)

SFR	Origin	Dependencies	Resolved?
		FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(3)	CC Part 2	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACF.1(2) FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(4)	CC Part 2	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_IFC.1(1) FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(5)	CC Part 2	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACF.1(3) FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(6)	RBACPP	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACF.1(4) FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(7)	RBACPP	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACF.1(4) FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(8)	RBACPP	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACF.1(4) FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(9)	RBACPP	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACF.1(4) FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(10)	CC Part 2	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_ACF.1(5) FMT_SMF.1 FMT_SMR.1
FMT_MSA.1(11) (WPAR)	CC Part 2	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_IFC.1(4) FMT_SMF.1 FMT_SMR.2
FMT_MSA.1(12) (IP Filter)	CC Part 2	[FDP_ACC.1 or FDP_IFC.1] FMT_SMF.1 FMT_SMR.1	FDP_IFC.1(5) FMT_SMF.1 FMT_SMR.2

FMT_MSA.2	CC Part 2	ADV_SPM.1
		[FDP_ACC.1 or
		FDP_IFC.1]
		FMT_MSA.1
		FMT_SMR.1
			ADV_SPM.1
			FDP_ACC.1(1)
			FMT_MSA.1(1)
			FMT_SMR.2

FMT_MSA.3(2)	CC Part 2	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1(2) FMT_SMR.2
FMT_MSA.3(3)	CC Part 2	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1(3) FMT_SMR.2
FMT_MSA.3(4)	CC Part 2	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1(4) FMT_SMR.2
FMT_MSA.3(5)	CC Part 2	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1(5) FMT_SMR.2
FMT_MSA.3(6)	RBACPP	FMT_MSA.1	FMT_MSA.1(9)

SFR	Origin	Dependencies	Resolved?
		FMT_SMR.1	FMT_SMR.2
FMT_MSA.3(7)	CC Part 2	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1(10) FMT_SMR.1
FMT_MSA.3(8) (WPAR)	CC Part 2	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1(11) FMT_SMR.2
FMT_MSA.3(9) (IP Filter)	CC Part 2	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1(12) FMT_SMR.2
FMT_MTD.1(3)	CC Part 2	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MTD.1(6)	CC Part 2	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MTD.1(7)	RBACPP	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MTD.1(8)	CC Part 2	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.1
FMT_MTD.1(9)	CC Part 2	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MTD.1(10)	CC Part 2	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MTD.1(11)	CC Part 2	FMT_SMF.1 FMT_SMR.1	FMT_SMF.1 FMT_SMR.2
FMT_MTD.3	RBACPP	FMT_MTD.1	FMT_MTD.1(7)
FMT_REV.1(3)	CC part 2	FMT_SMR.1	FMT_SMR.1
FMT_SMF.1	CC Part 2	None	yes
FMT_SMR.1	CC Part 2	FIA_UID.1	FIA_UID.2
FPT_FLS.1	RBACPP	ADV_SPM.1	ADV_SPM.1
FPT_RCV.1	RBACPP	AGD_ADM.1 ADV_SPM.1	AGD_ADM.1 ADV_SPM.1
FPT_RCV.4	RBACPP	ADV_SPM.1	ADV_SPM.1
FPT_RVM.2-AIX	CC Part 2	None	yes
FPT_TDC.1	CC Part 2	None	yes
FPT_TST.1	RBACPP	FPT_AMT.1	FPT_AMT.1
FTA_LSA.1	RBACPP	None	yes
FTA_TSE.1	RBACPP	None	yes
FTP_ITC.1	CC Part 2	None	yes

Table 27: Dependency analysis for IT environment SFRs

SFR	Dependencies	Resolved?
FCS_CKM.1(a) (SSL)	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4 FMT_MSA.2	no FCS_COP.1(2)
FCS_CKM.1(b) (Kerberos)	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4 FMT_MSA.2	no FCS_COP.1(3, 4)
FDP_ACC.1(a)	FDP_ACF.1	FDP_ACF.1(a)
FDP_ACC.1(b)	FDP_ACF.1	FDP_ACF.1(b)

FDP_ACF.1(a)	FDP_ACC.1
	FMT_MSA.3
		FDP_ACC.1(a)
		FMT_MSA.3(a)

FDP_ACF.1(b)	FDP_ACC.1 FMT_MSA.3	no FDP_ACC.1(b)
FIA_SOS.1(a)	None	yes
FIA_UID.2(a)	None	yes
FMT_MSA.3(a)	FMT_MSA.1	no

SFR	Dependencies	Resolved?
	FMT_SMR.1	

Unresolved dependencies are discussed in section 8.3.6.

The RSA keys in the SSL certificates used by the TOE for encryption and decryption in FCS_COP.1(2) are generated by the IT environment via FCS_CKM.1(a).

The Kerberos encryption/decryption keys used by the TOE in FCS_COP.1(3) and FCS_COP.1(4) and distributed by FCS_CKM.2(2) are generated by the IT environment via FCS_CKM.1(b).

The dependency analysis for the pre-defined EAL4 has already been performed by the creators of the Common Criteria and is not repeated in this section. ALC_FLR.3, which has been added as a security assurance requirement has no dependency on any security functional or security assurance requirement. Therefore the dependency analysis for EAL4 of the Common Criteria applies.

Since EAL4 is hierarchical to EAL2 and EAL3, plus EAL4 includes ADV_SPM.1 required by [RBACPP], all dependencies of security functional requirements on assurance requirements listed in the Protection Profiles are also resolved.

8.3.6 Justification of Unresolved Dependencies

8.3.6.1 TOE Unresolved Dependencies

As demonstrated in the dependency analysis above, all dependencies between security requirements for the TOE that come from [CAPP] and [LSPP] are resolved since all the dependencies within [CAPP] and [LSPP] are resolved, but dependencies of additional SFRs are not all resolved. EAL 4 as a pre-defined evaluation assurance level contains no unresolved references to other assurance components and no additional references to functional components and ALC_FLR.3 has no dependencies.

Neither the TOE nor the IT environment explicitly implements a key destruction function for any of the TOE cryptographic security functions. Thus, all TOE security functional requirements defined using FCS_CKM.1, FCS_CKM.2, and FCS_COP.1 have unresolved dependencies on the key destruction security functional requirement FCS_CKM.4.

The message digests and random number generators (RNGs) of the TOE do not require key generation, therefore, FCS_COP.1(7) and FCS_COP.1(8) have unresolved dependencies on the key generation security functional requirement FCS_CKM.1.

The Kerberos keys and SSL RSA keys are created by the IT environment. Because of this, FCS_CKM.2(2), FCS_COP.1(2, 3, 4) have an unresolved dependency on FMT_MSA.2 which is left to the author of the ST for Kerberos and SSL.

Key destruction is performed implicitly for the SSL symmetric session keys used by the Object Reuse function, which ensures that memory used to temporarily store the symmetric session key is cleared before it is assigned to another subject or object. This applies for both main memory as well as disk space (the session keys might be written to disk space as part of the paging function of the TOE. They are not stored in ordinary files).

With respect to the long-term public-private key pairs, the key destruction is performed by deleting the file containing the key. The Object Reuse function of the TOE ensures that the disk space previously allocated to the file storing those keys is cleared before it is assigned to another subject or object.

The TOE does not import non-RSA keys for SSL but generates all non-RSA keys itself as expressed in the security functional requirement FCS_CKM.1(1).

8.3.6.2 IT Environment Unresolved Dependencies

The dependencies of FMT_MSA.3(a) on FMT_MSA.1 and FMT_SMR.1 for the security requirements for the IT environment are not resolved, because the processor does not allow to “manage” the use of the processor attribute and there is no role model involved. The processor switches between “user” and “supervisor” mode under well defined conditions where the TSF defined in this Security Target are required to “manage” those conditions. Roles (especially human roles) are not involved here.

The dependency on FMT_MSA.3 from FDP_ACF.1(b) for the IT environment has not been resolved in this ST, since imposing a decision whether the management of LPAR resources during TOE operation should be manageable would be an unnecessary restriction for the ST author of the underlying system.

Neither the TOE nor the IT environment explicitly implements a key destruction function for any of the TOE cryptographic security functions. Thus, all IT environment security functional requirements defined using FCS_CKM.1 have unresolved dependencies on the key destruction security functional requirement FCS_CKM.4.

The Kerberos keys and SSL RSA keys are created by the IT environment. Because of this, FCS_CKM.1(a) and FCS_CKM.1(b) have unresolved dependencies on FMT_MSA.2 which is left to the author of the ST for Kerberos and SSL.

8.3.7 Strength of Function

This Security Target claims, in compliance with [CAPP] and [LSPP], an SOF rating SOF-medium. This claim applies for FIA_SOS.1, whereby [CAPP] and [LSPP] state that a 'one off' probability of guessing the password in 1,000,000 is given. The SFR is in turn consistent with the security objectives.

[RBACPP] defines an SOF rating of at least SOF-basic, thus, this Security Target is also in compliance with [RBACPP].

8.3.8 Evaluation Assurance Level

The EAL defined in [CAPP] and [LSPP] is EAL3, as both address a generalized environment with a moderate level of risk to the assets. [RBACPP] defines an EAL2 level of assurance plus ADV_SPM.1. This Security Target claims EAL4 augmented with ALC_FLR.3, meeting these assumptions on the environment as well by providing a higher evaluation assurance level.

8.4 TOE Summary Specification Rationale

8.4.1 Security Functions Justification

The following table demonstrates the relationship between the TSF and the individual TSF aspects described in the TSS and security functional requirements for the TOE from chapter 5.

Table 28: Mapping of TSF and TSF aspects to SFRs

Security Function or Security Function Aspects	Corresponding SFRs
IA – Identification and Authentication	
IA.1	FCS_CKM.1(1), FCS_CKM.2(1), FCS_COP.1(1), FCS_COP.1(2), FIA_ATD.1(1), FIA_ATD.1(2), FIA_SOS.1, FMT_MSA.2, FMT_MTD.1(4), FMT_SMF.1, FTP_ITC.1
IA.2	FAU_GEN.2, FIA_UAU.2, FIA_UID.2
IA.3	FAU_GEN.2, FIA_UAU.2, FIA_UID.2, FIA_UAU.7
IA.4	FAU_GEN.2, FIA_USB.1(1), FIA_USB.1(2)
IA.5	FIA_USB.1(1), FIA_USB.1(2)
IA.6	FIA_USB.1(1), FIA_USB.1(2)
AU – Auditing	
AU.1	FAU_GEN.1, FAU_GEN.2
AU.2	FAU_GEN.1, FAU_GEN.2
AU.3	FAU_SAR.1, FAU_SAR.3
AU.4	FAU_SAR.1, FAU_SAR.3
AU.5	FAU_SAR.2, FAU_STG.1, FMT_MTD.1(1)
AU.6	FAU_STG.3, FAU_STG.4
AU.7	FAU_SAR.2
DA – Discretionary Access Control	

Security Function or Security Function Aspects	Corresponding SFRs
DA.1 (Perms)	FDP_ACC.1(1), FDP_ACF.1(1)
DA.2 (Extended Perms)	FDP_ACC.1(1), FDP_ACF.1(1)
DA.3 (FSO)	FDP_ACC.1(1), FDP_ACF.1(1), FMT_MSA.1(1), FMT_MSA.1(9), FMT_MSA.3(1), FMT_SMF.1
DA.4 (IPC)	FDP_ACC.1(1), FDP_ACF.1(1), FMT_MSA.1(1), FMT_MSA.1(9), FMT_MSA.3(1), FMT_SMF.1
DA.5 (VIOS)	FDP_ACC.1(5), FDP_ACF.1(5), FMT_MSA.1(10), FMT_MSA.3(7), FMT_MTD.1(8), FMT_SMF.1
DA.6 (TCP)	FDP_ACC.1(1), FDP_ACF.1(1), FMT_MSA.1(1), FMT_MSA.1(9), FMT_MSA.3(1), FMT_SMF.1
DA.7 (EFS)	FDP_ACC.1(1), FDP_ACF.1(1), FCS_CKM.1(2), FCS_CKM.1(3), FCS_COP.1(5), FCS_COP.1(6), FCS_COP.1(7), FCS_COP.1(8), FMT_MSA.2
RBAC	
RA.1	FDP_ACC.1(4), FDP_ACF.1(4), FMT_MSA.1(6), FMT_MSA.1(7), FMT_MSA.1(8), FMT_MSA.1(9), FMT_MSA.3(6), FMT_SMR.2, FPT_FLS.1, FPT_RCV.1, FPT_RCV.4, FTA_LSA.1, FTA_TSE.1
WPAR	
WP.1	FDP_IFC.1(4), FDP_IFF.1(1), FMT_MSA.1(11), FMT_MSA.3(8), FMT_MTD.1(9), FMT_MTD.1(10)
PV – Privileges	
PV.2 (Process Privilege Sets)	FAU_SAR.2, FDP_ACC.1(1), FDP_ACC.1(2), FDP_ACC.1(3), FDP_ACF.1(1), FDP_ACF.1(2), FDP_ACF.1(3), FDP_IFC.1(1), FDP_IFC.1(2), FDP_IFC.1(3), FDP_IFF.2(1), FDP_IFF.2(2), FDP_IFF.2(3)
PV.3 (Privileged Commands)	FAU_SAR.2, FDP_ACC.1(1), FDP_ACC.1(2), FDP_ACC.1(3), FDP_ACF.1(1), FDP_ACF.1(2), FDP_ACF.1(3), FDP_IFC.1(1), FDP_IFC.1(2), FDP_IFC.1(3), FDP_IFF.2(1), FDP_IFF.2(2), FDP_IFF.2(3)
PV.4 (Device Privilege Sets)	FDP_ACC.1(1), FDP_ACC.1(2), FDP_ACC.1(3), FDP_ACF.1(1), FDP_ACF.1(2), FDP_ACF.1(3), FDP_IFC.1(1), FDP_IFC.1(2), FDP_IFC.1(3), FDP_IFF.2(1), FDP_IFF.2(2), FDP_IFF.2(3)
PV.6 (WPAR Privilege Set)	FDP_ACC.1(1), FDP_ACC.1(2), FDP_ACC.1(3), FDP_ACF.1(1), FDP_ACF.1(2), FDP_ACF.1(3), FDP_IFC.1(1), FDP_IFC.1(2), FDP_IFC.1(3), FDP_IFC.1(4), FDP_IFF.1(1), FDP_IFF.2(1), FDP_IFF.2(2), FDP_IFF.2(3), FMT_MSA.3(8)
AZ – Authorizations	
AZ.1	FDP_ACC.1(3), FDP_ACF.1(3)
MAC – Mandatory Access Control	FDP_ETC.1, FDP_ETC.2, FDP_IFC.1(1), FDP_IFF.2(2), FDP_ITC.1, FDP_ITC.2
TN – Networking	
TN.1 (Trusted Network)	FDP_ETC.1, FDP_ETC.2, FDP_IFC.1(3), FDP_IFF.2(3), FDP_ITC.1, FDP_ITC.2
TN.2 (Trusted Network)	FDP_ETC.2, FDP_ITC.1, FPT_TDC.1
TN.3 (IP Filter)	FDP_IFC.1(5), FDP_IFF.1(2), FMT_MSA.1(12), FMT_MSA.3(9)
MIC – Mandatory Integrity Control	
MIC.1	FDP_IFC.1(2), FDP_IFF.2(2)

Security Function or Security Function Aspects	Corresponding SFRs
OR – Object Reuse	
OR.1	FDP_RIP.2, Note 1
OR.2	FDP_RIP.2, Note 1
OR.3	FDP_RIP.2, Note 1
OR.4	FDP_RIP.2, Note 1
OR.5	FDP_RIP.3-AIX
SM – Security Management	
SM.1	FDP_IFC.1(4), FDP_IFF.1(1), FMT_SMR.1, FMT_SMR.2
SM.2	FAU_GEN.1, FAU_SEL.1, FDP_IFC.1(4), FDP_IFF.1(1), FMT_MOF.1(1), FMT_MTD.1(1), FMT_MTD.1(2), FMT_SMF.1
SM.3	FDP_IFC.1(5), FDP_IFF.1(2), FMT_MSA.1(1)-(6), FMT_MSA.1(12), FMT_MSA.3(1)-(7), FMT_MSA.3(9), FMT_MTD.1(8), FMT_MTD.1(11), FMT_SMF.1, FMT_REV.1(2)
SM.4	FDP_IFC.1(4), FDP_IFF.1(1), FIA_ATD.1(1), FIA_ATD.1(2), FIA_SOS.1, FMT_MOF.1(2), FMT_MSA.1(7), FMT_MSA.1(8), FMT_MTD.1(4), FMT_MTD.1(5), FMT_MTD.3, FMT_SMF.1, FMT_REV.1(1), FMT_REV.1(3)
SM.5	FPT_STM.1
TP – TSF Protection	
TP.1	FPT_RVM.1, FPT_RVM.2-AIX
TP.2	FDP_IFC.1(4), FDP_IFF.1(1), FPT_SEP.1
TP.3	FPT_SEP.1
TP.4	FIA_UAU.2, FIA_UAU.7, FIA_UID.2, FPT_SEP.1
TP.5	FAU_SEL.1, FMT_MSA.2, FMT_MSA.3(1)-(6), FMT_MTD.1(1)-(5), FMT_SMF.1, FTP_ITC.1
TP.6	FPT_SEP.1
TP.7	FAU_GEN,1. FPT_AMT.1
TP.8	FCS_COP1.(7), FMT_MSA.2, FPT_TST.1
TP.9	FAU_SAR.2, FAU_SEL.1, FDP_ACC.1(2), FDP_ACF.1(2), FDP_IFC.1(1), FDP_IFF.2(1)
TP.10 (NFS)	FCS_CKM.2(2), FCS_COP.1(3), FCS_COP.1(4), FMT_MSA.2, FTP_ITC.1

The following table shows that the IT security functions, as specified in the TOE summary specification, meet all security functional requirements for the TOE and work together to satisfy the TOE security functional requirements.

Table 29: Mapping Security Functional Requirements to Security Functions

SFR	Security Functions (TOE summary specification)
FAU_GEN.1	The requirement for the information to be recorded with an audit event are satisfied by the generation of audit data is fulfilled by the security functions AU.1 specifying the audit record format and SM.2 describing the audit control files which are used to define the events that can be audited. AU.2 describes the process used within the TOE to generate an audit record. The results of diagnostic tests are stored in a separate error log file as described in TP.7.
FAU_GEN.2	The association of auditable events with its causing identity is done in AU.1 specifying the audit record format, including the user and login ID of the creator of the auditable event, and AU.2 generating the audit record. IA.2 and IA.3 describe the authentication process which ensures that the ID of the a user is authenticated. IA.4 describes that although a user may change his / her effective user ID, the login user ID (which is recorded in the audit record) can not be changed. This ensures that the ID the user has used when he has authenticated to

SFR	Security Functions (TOE summary specification)
	the system is recorded with every audit event that this user has caused.
FAU_SAR.1	The system administrator can use the commands described in AU.3 and AU.4 to select, read, process and print audit records.
FAU_SAR.2	Read access to the audit records is granted only to explicitly privileged users as enforced by the audit file protection of AU.5. In addition, privileges are used to enforce restrictions with respect to audit management as in AU.7 and in PV.2 and PV.3. File security flags are used to identify audit-related files and restrict access to them (TP.9).
FAU_SAR.3	The requirement to allow searches of the audit data based on specified attributes is met by the audit review function described in AU.3 and AU.4. auditselect can be used as a tool to select audit records using an expression based on the audit record fields listed in table 6.2. This table contains all the selection criteria listed in FAU_SAR.3.
FAU_SEL.1	The in- or exclusion of auditable events from the set of audited events is provided by SM.2 (audit configuration and management). SM.2 describes how a system administrator can select the events to be audited based on event type, file name and user identity and defines the system configuration files used in this function System configuration files in general are also described in TP.5. A file security flag can be used to override the decision not to audit access to individual files (TP.9).
FAU_STG.1	Audit data is protected by AU.5. This prevents audit records to be deleted or modified by other users than the system administrator.
FAU_STG.3	AU.6 describes the process AIX takes when the audit trail exceeds a defined threshold.
FAU_STG.4	Prevention of audit data loss is described in AU.6. The system can be configured to go into “panic” mode and stop the host when the audit trail is full.
FCS_CKM.1(1)	IA.1 generates GSKit SSL session keys when communicating to LDAP.
FCS_CKM.1(2)	DA.7 uses CLiC to generate random numbers and uses the numbers as keys for encrypting files.
FCS_CKM.1(3)	DA.7 uses CLiC to generate RSA key pairs for encrypting the random number keys in FCS_CKM.1(2).
FCS_CKM.2(1)	IA.1 uses GSKit SSL which distributes SSL session keys when communicating to LDAP.
FCS_CKM.2(2)	TP.10 distributes and uses Kerberos keys using Kerberos encryption libraries.
FCS_COP.1(1)	IA.1 uses GSKit SSL which performs symmetric encryption and decryption when communicating to LDAP.
FCS_COP.1(2)	IA.1 uses GSKit SSL which generates and verifies digital signatures when communicating to LDAP.
FCS_COP.1(3)	TP.10 uses CLiC to perform symmetric encryption and decryption as well as digest generation and verification of the NFSv4 client/server communications.
FCS_COP.1(4)	TP.10 uses the Kerberos encryption library for symmetric encryption/decryption when communicating with Kerberos.
FCS_COP.1(5)	DA.7 uses CLiC to perform symmetric encryption and decryption on files.
FCS_COP.1(6)	DA.7 uses CLiC to perform asymmetric encryption and decryption for signing and data encryption.
FCS_COP.1(7)	DA.7 and TP.8 use CLiC for digest generation and verification.
FCS_COP.1(8)	DA.7 use CLiC for cryptographic random number generation.
FDP_ACC.1(1)	The discretionary access control policy is based on DA.1 and DA.2 defining permission bits for the subjects and objects as in DA.3 for file system objects, DA.4 for IPC objects, and DA.6 for TCP connections. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_ACC.1(2)	The trusted computing base model is implemented in TP.9. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3, and by file security flags described in TP.9.
FDP_ACC.1(3)	The implementation of the authorizations mechanism is described in AZ.1. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_ACC.1(4)	The role-based access control policy is based on RA.1 defining the subjects and objects covered by this policy.

SFR	Security Functions (TOE summary specification)
FDP_ACC.1(5)	The VIOS access control policy is based on DA.5 defining the mapping of LPAR partitions to logical/physical volumes and defining the mapping of Ethernet packets to groups of LPAR partitions sharing a virtual network.
FDP_ACF.1(1)	The discretionary access control is realized as described above by DA.1, DA.2, DA.3, DA.4, and DA.6. There the individual mechanisms for access control depending on the object type are described in detail. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_ACF.1(2)	The trusted computing base model is implemented in TP.9. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_ACF.1(3)	The implementation of the authorizations mechanism is described in AZ.1. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_ACF.1(4)	The role-based access control is realized as described above by RA.1. There the individual mechanisms for access control depending on the object type are described in detail.
FDP_ACF.1(5)	The VIOS access control policy is based on DA.5 defining the mapping of LPAR partitions to logical/physical volumes and defining the mapping of Ethernet packets to groups of LPAR partitions sharing a virtual network.
FDP_ETC.1	Export of unlabeled data is implemented in MAC (and TN.1 for network objects).
FDP_ETC.2	Export of labeled data is implemented in MAC (and TN.1, TN.2 for network objects).
FDP_IFC.1(1)	The implementation of the mandatory access control policy is described in MAC. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3, and by a file security flag described in TP.9.
FDP_IFC.1(2)	The mandatory integrity control policy is implemented as described in MIC.1. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_IFC.1(3)	TN.1 describes how the TOE implements the Trusted Network policy. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_IFC.1(4)	WP.1 describes how the TOE implements the Workload Partition Control Policy. This is supported by PV.6, SM.1, SM.2, SM.4, and TP.2.
FDP_IFC.1(5)	TN.3 describes how the TOE implements the IP Filter Control Policy. This is supported by SM.3.
FDP_IFF.1(1)	WP.1 describes how the TOE implements the Workload Partition Control Policy. This is supported by PV.6, SM.1, SM.2, SM.4, and TP.2.
FDP_IFF.1(2)	TN.3 describes how the TOE implements the IP Filter Control Policy. This is supported by SM.3.
FDP_IFF.2(1)	The implementation of the mandatory access control policy is described in MAC. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3, and by a file security flag detailed in TP.9.
FDP_IFF.2(2)	TN.1 describes how the TOE implements the Trusted Network policy. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_IFF.2(3)	The mandatory integrity control policy is implemented as described in MIC.1. This is supported by additional restrictions enforced with the privilege mechanism, which is detailed in PV.2 and PV.3.
FDP_ITC.1	Import of unlabeled data is implemented in MAC (and TN.1 for network objects).
FDP_ITC.2	Import of labeled data is implemented in MAC (and TN.1, TN.2 for network objects).
FDP_RIP.2	Object residual information protection is realized by security functions for object reuse on file system objects (OR.1), IPC objects (OR.2), queuing system objects (OR.3) and miscellaneous objects (OR.4).
Note 1	The residual information protection as realized by OR.1, OR.2, OR.3 and OR.4 (see above) applies as well to subjects.
FDP_RIP.3-AIX	Residual information protection for hard disk drives is implemented as described in OR.5.
FIA_ATD.1(1)	AIX security attributes belonging to individual users are realized by the user I&A data management of IA.1. Management of user attributes is described in SM.4.

SFR	Security Functions (TOE summary specification)
FIA_ATD.1(2)	VIOS security attributes belonging to individual users are realized by the user I&A data management of IA.1. Management of user attributes is described in SM.4.
FIA_SOS.1	The passwd function of IA.1 is able to enforce the verification of secrets as required. System management commands can be used to define parameters that can be used to (hopefully) enhance the strength of the passwords chosen by the user. Password management including the possible parameter to enhance the strength of passwords are explained in SM.4.
FIA_UAU.2	Authentication of each user before any action is realized by IA.2 (common authentication mechanism) and IA.3 (interactive login and related mechanisms). Authentication is initiated by a trusted process. Trusted processes are described in TP.4.
FIA_UAU.7	The login mechanisms of IA.3 provide only obscured feedback during authentication. Authentication feedback is managed by a trusted process. Trusted processes are described in TP.4.
FIA_UID.2	Identification of each user before any action is realized together with authentication as in IA.2 and IA.3 (see above). Identification is initiated by a trusted process. Trusted processes are described in TP.4.
FIA_USB.1(1)	For AIX, the required binding between subjects and users is implemented by the su functionality of IA.4 and login processing of IA.5. Function IA.6 describes the logoff process which releases the binding between subjects and users.
FIA_USB.1(2)	For VIOS, the required binding between subjects and users is implemented by the su functionality of IA.4 and login processing of IA.5. Function IA.6 describes the logoff process which releases the binding between subjects and users.
FMT_MOF.1(1)	The enabling/disabling and selection of audit events is described in SM.2.
FMT_MOF.1(2)	The management of user authentication data is described in SM.4.
FMT_MSA.1(1)	The management of object security attributes is implemented by the access control configuration and management function SM.3, the objects are described in DA.3 (file system objects), DA.4 (IPC objects), and DA.6 (TCP connections). In addition, security attributes of relevance for MAC are addressed here.
FMT_MSA.1(2)	The management of security attributes for TN.1, TN.2 and MAC is implemented in SM.3.
FMT_MSA.1(3)	The management of security attributes for TP.8 and TP.9 is implemented in SM.3.
FMT_MSA.1(4)	The management of security attributes for MIC is implemented in SM.3.
FMT_MSA.1(5)	The management of security attributes for AZ is implemented in SM.3.
FMT_MSA.1(6)	RA.1 provides the ability to manage user role authorizations.
FMT_MSA.1(7)	RA.1 defines the default role and SM.4 defines the system management abilities.
FMT_MSA.1(8)	RA.1 defines how a session owner can modify their active role(s) and SM.4 defines the system management of user roles.
FMT_MSA.1(9)	RA.1 provide control over the role definitions and DA3, DA.4, and DA.6 describe the abilities of object owners for the object types that they own.
FMT_MSA.1(10)	The management of VIOS access control policy is implemented by SM.3. The objects are described in DA.5.
FMT_MSA.1(11)	WP.1 defines the ability to modify the security attributes of a WPAR.
FMT_MSA.1(12)	TN.3 defines the ability to modify the security attributes of an IP Filter.
FMT_MSA.2	This SFR was added to support the SFRs for secure communications and file system encryption (FCS_CKM.1, FCS_CKM.2, and FCS_COP.1). These cryptographic SFRs are for IA.1, DA.7, TP.8, and TP.10.
FMT_MSA.3(1)	Restrictive default values for security attributes are defined for the objects when they are created. Default values can be defined by the system administrator for all object types and by the user for file system objects created under his control. (see above, e.g.,SM.3, DA.3, DA.4, DA.6). Some default values are defined in TSF databases as defined in TP.5.
FMT_MSA.3(2)	The management of default values for TN.1, TN.2 and MAC security attributes is implemented in SM.3, while some default values are defined in TSF databases as in TP.5.
FMT_MSA.3(3)	The management of default values for TP.8 and TP.9 security attributes is implemented in SM.3, while some default values are defined in TSF databases as in TP.5.
FMT_MSA.3(4)	The management of default values for MIC security attributes is implemented in SM.3, while some default values are defined in TSF databases as in TP.5.

SFR	Security Functions (TOE summary specification)
FMT_MSA.3(5)	The management of default values for AZ security attributes is implemented in SM.3, while some default values are defined in TSF databases as in TP.5.
FMT_MSA.3(6)	RA.1 defines the roles that can specify alternative initial values for security attributes.
FMT_MSA.3(7)	A VIOS SCSI device driver acting on behalf of an LPAR partition cannot access a logical volume or physical volume until the mapping is created in VIOS as mentioned in SM.3 with the access control described in DA.5. Similarly, a VIOS Ethernet adapter device driver cannot access a VIOS Ethernet device driver acting on behalf of a group of LPAR partitions sharing a virtual network and vice versa until the mapping is created in VIOS as mentioned in SM.3 with the access control described in DA.5.
FMT_MSA.3(8)	WP.1 uses restrictive values for WPAR security attributes modifiable by the Global environment administrator. These values include the privileges of PV.6.
FMT_MSA.3(9)	TN.3 uses permissive values for IP Filter security attributes modifiable by an authorized administrator.
FMT_MTD.1(1)Audi t Trail	The audit trail (and the restricted access to it) is realized by the audit file protection AU.5 and the audit configuration and management SM.2. TSF databases as defined in TP.5 contain configuration parameter of the audit trail.
FMT_MTD.1(2) Audit Events	Only authorized administrators are allowed to modify or observe the set of audited events, which is implemented by the audit configuration and management SM.2. Audit attributes are stored in TSF databases described in TP.5.
FMT_MTD.1(3)	The audit threshold is managed in SM.2, using TSF databases to store the configuration as identified in TP.5.
FMT_MTD.1(4) User Attributes	User security attributes are protected as required by the user identification and authentication data management IA.1 and during the creation of new users in SM.4. User attributes are stored in TSF databases described in TP.5.
FMT_MTD.1(5) Authen. Data	Initialization of authentication data is restricted to administrators during the creation of new users in SM.4. Authentication data (in encrypted form) and attributes are stored in TSF databases described in TP.5. Users are allowed to change their own authentication data within the limits defined by the system administrator. This is described in SM.4
FMT_MTD.1(6)	Privileges are managed by authorized administrators as described as described in PV.2, PV.3, PV.4, and PV.6.
FMT_MTD.1(7)	SM.1 provides the tools which use the RBAC roles to determine the ability to manage RBAC and system attributes.
FMT_MTD.1(8)	VIOS contains the mappings of logical volumes and physical volumes to LPAR partitions. It also contains the mappings of Ethernet packets to groups of LPAR partitions sharing a virtual network. The mappings are performed as described by DA.5. Authorized administrators are allowed to modify these mappings as described in SM.3.
FMT_MTD.1(9)	WP.1 restricts the ability to modify privileges of WPARS to Global environment administrators.
FMT_MTD.1(10)	WP.1 uses a corral ID (CID) to distinguish between the WPARs on the system and the Global environment. The CID is set by the kernel and immutable during the execution of the WPAR and Global environment.
FMT_MTD.1(11)	SM.3 provides the management tools for managing the IP filter rules (described in TN.3).
FMT_MTD.3	Secure TSF Data Secure values for TSF data are enforced by the user management functions of SM.4.
FMT_REV.1(1) User Attributes	The revocation of user security attributes as required in FMT_REV.1(1) is realized by the user management functions of SM.4.
FMT_REV.1(2) Obj. Attributes	Revocation of object security attributes is realized by the access control configuration and management function SM.3.
FMT_REV.1(3) VIOS User Attributes	The revocation of user security attributes as required in FMT_REV.1(3) is realized by the user management functions of SM.4.
FMT_SMF.1	Management of security functions is addressed in the following security functions: Object security attributes management: DA.3 (File system objects), DA.4 (IPC objects), DA.6 (TCP connections). In addition SM.3 defines some management functions, and the implementations of the respective access control policies in AZ, MAC, TP.9, MIC.1+.2, TN.1, TN.3, PV.2 and PV.3. User attribute management: SM.4

SFR	Security Functions (TOE summary specification)
	Authentication management: SM.4 and IA.1 Audit trail management: SM.2 Audit event management: SM.2 In addition most of the management functions use the TSF databases (TP.5) to store management configurations.
FMT_SMR.1	The required roles are maintained within the security management of the roles in function SM.1.
FMT_SMR.2	The required roles are maintained within the security management of the roles in function RA.1 and system administration SM.1.
FPT_AMT.1	Diagnostic functions are provided by TP.7 to allow abstract machine testing.
FPT_FLS.1	RA.1 provides for the preservation of a secure state in case of RBAC database failures.
FPT_RCV.1	RA.1 provides for a mode of operation in the case of RBAC failure and service discontinuity.
FPT_RCV.4	RA.1 provides for the functional recovery of the security functions when privilege data and role membership information is not available.
FPT_RVM.1	The TSF invocation guarantee functionality TP.1 ensure that TSP enforcement functions are always invoked before functions in the TSC are allowed to proceed.
FPT_RVM.2-AIX	The ability to prevent the execution of code on the stack of a process is enforced by the memory management mechanism of the kernel mentioned in TP.1.
FPT_SEP.1	The required domain separation for the TSF is realized by the kernel functionality TP.2 itself, the kernel extensions TP.3, trusted processes TP.4, the discretionary access control mechanism DA.3 and internal TOE protection mechanisms TP.6.
FPT_STM.1	Reliable time stamps are provided by the protected system clock of the time management function SM.5.
FPT_TDC.1	Labeling for network connections implemented in TN.2 provides for TSF data consistency.
FPT_TST.1	TP.8 provides a mechanism to check the integrity of TSF-relevant files by comparing various critical attributes with a reference database (which is also identified in TP.5). Similarly, TP.8 provides a command to check the status of the TCB.
FTA_LSA.1	RA.1 provides commands for the user to manage their active role set and limits the active role set to the set of authorized roles for the user.
FTA_TSE.1	RA.1 specifies that all users have an implicit/default role of Normal User even when no explicit role is assigned to the account.
FTP_ITC.1	IA.1 defines the communications between the TOE and LDAP. TP.5 defines the configuration files for LDAP and NFSv4 communications. TP.10 defines the inter-TSF communications between the NFSv4 client and server.

The above table shows how the security functions satisfy the security functional requirements for the TOE. The following rationale demonstrates how the SFRs support each other and how the security functions work together to support the SFRs.

As outlined in the TSS, the TOE implements a variety of access control mechanisms that provide for both mandatory and discretionary means to restrict access to information assets to authorized users. These access control mechanisms work together to implement a number of objectives:

- The Discretionary Access Control is specified in FDP_ACC.1(1) and FDP_ACF.1(1) and implemented as described for the security function DA.
- The Mandatory Access Control policy is specified in FDP_IFC.1(1) and FDP_IFF.2(1) and implemented in MAC. Special rules apply for the import and export of labeled (FDP_ITC.2, FDP_ETC.2) and unlabeled (FTP_ITC.1, FDP_ETC.1) data.
- A separate policy has been specified in FDP_IFC.1(3) and FDP_IFF.2(2) for mandatory access control in network transactions, which is implemented as part of the security function MAC and provides for the consistency of labels between the TOE and remote systems (FPT_TDC.1).
- A Mandatory Integrity Control policy is modeled in FDP_IFC.1(2) and FDP_IFF.2(3) and implemented as in MIC.

- An Role-Based Access Control (RBAC) policy is modeled in FDP_ACC.1(4) and FDP_ACF.1(4) and implemented in RA. [RBACPP] includes the additional requirements of FMT_MTD.3, FPT_FLS.1, FPT_RCV.1, FPT_RCV.4, FPT_TST.1, FTA_LSA.1, and FTA_TSE.1.
- A Workload Partition control policy is modeled in FDP_IFC.1(4) and FDP_IFF.1(1) and implemented in WP.
- An IP Filter control policy is modeled in FDP_IFC.1(5) and FDP_IFF.1(2) and implemented in TN.3.
- As additional integrity protection for elementary system resources, the TOE implements a trusted computing base as described in TP.8 and TP.9, which is specified in FDP_ACC.1(2) and FDP_ACF.1(2) and restricts modifications to certain TSF-relevant files to the maintenance mode of the system. Integrity checks for the TCB files are offered as modeled in FPT_TST.1.

The TOE offers a mechanism to override those access control mechanisms. Rather than disabling all TSF enforcement for a certain user ID, like on basic UNIX systems, (which can be simulated on the TOE as well) the TOE implements the concept of privileges and authorizations. As described in the TSS, privileges can be granted to processes that allow overriding limited aspects of access control enforcement – those privileges have been documented for the single access control security functions they relate to and in summary for the security function PV. In addition, trusted processes on the TOE enforce additional authorization checks as modeled in FDP_ACC.1(3) and FDP_ACF.1(3) and implemented in AZ – this allows to ensure that only users with certain authorizations can execute privileged programs. Roles as identified in FMT_SMR.2 are consequently implemented by means of authorizations assigned to users. For VIOS, roles identified in FMT_SMR.1 are consequently implemented by means of user IDs, group IDs, and the role mechanism.

For AIX to enforce those mechanisms, the TOE needs to identify and authenticate users. This is implemented in IA as modeled by FIA_UID.2, FIA_UAU.2 and FIA_UAU.7, with strength requirements for authentication passwords in FIA_SOS.1. Security attributes are maintained for users as in FIA_ATD.1(1) and associated with processes acting on behalf of those users (FIA_USB.1(1)). Revocation of user attributes as part of the management functionality is modeled in FMT_REV.1(1) and (2).

For VIOS to enforce its mechanisms, it needs to identify and authenticate users. This is implemented in I&A as modeled by FIA_UID.2, FIA-UAU.2, and FIA_UAU.7, with strength requirements for authentication passwords in FIA_SOS.1. Security attributes are maintained for users as in FIA_ATD.1(2) and associated with processes acting on behalf of those users (FIA_USB.1(2)). Revocation of user attributes as part of the management functionality is modeled in FMT_REV.1(3).

Security function management is offered for the functionality described above by the security function SM, as modeled in FMT_MOF.1(2), FMT_MSA.1(1) through FMT_MSA.1(12), FMT_MSA.3(1) through FMT_MSA.3(9), FMT_MTD.1(4) thru FMT_MTD.1(10), and called out in FMT_SMF.1.

Auditing for security-related events is modeled in FAU_GEN.1 and FAU_GEN.2 and implemented in AU, with management and protection functionality for audit trails in FAU_SAR.1, FAU_SAR.2, FAU_SAR.3 and FAU_STG.1, FAU_STG.3 and FAU_STG.4 – protection of audit trails is implemented based on the access control and privilege mechanisms described above. Additional management functionality for auditing is provided, specifically in FMT_MTD.1(1), (2) and (3), FAU_SEL.1, FMT_MOF.1(1), and implemented as described in SM. A reliable time source for audit record generation is provided as in FPT_STM.1.

Security functional requirements FCS_CKM.1(1), FCS_CKM.2(1), FCS_COP.1(1), and FCS_COP.1(2) have been added to address the SSL communications between the TOE and the LDAP authentication database server. Security management of this communications consists of the use of secure attributes (FMT_MSA.2).

Security functional requirements FCS_CKM.2(2), and FCS_COP.1(4) have been added to address the Kerberos communication protocol and Kerberos tickets which differs from SSL. Security management of this communications consists of secure attributes (FMT_MSA.2).

Security functional requirement FCS_COP.1(3) has been added to address the NFSv4 client to server communications. This communication uses the keys generated and distributed by Kerberos as the encryption keys (FCS_CKM.2(2)). Inter-TSF communications between NFSv4 client and server is addressed by FTP_ITC.1.

Security functional requirements FCS_CKM.1(2, 3) and FCS_COP.1(5, 6, 7, 8) have been added to address the Encrypted File System. Security management of this communications consists of the use of secure attributes (FMT_MSA.2). Additionally, FCS_COP.1(6) and FCS_COP.1(7) address the File Integrity Verification (FIV).

The VIOS access control is defined by the VIOS access control policy. Where the discretionary access control policy is for TOE users and objects, the VIOS access control policy is for VIOS specific objects; thus, FDP_ACC.1(5) and FDP_ACF.1(5) were added to make this distinction. Furthermore, the VIOS subsystem is not part of the LSPP, thus, separate SFRs were added to make this distinction clearer. The security management SFRs FMT_MSA.1(10) and FMT_MSA.3(7) were added to handle the dependencies for managing the VIOS access control policy. Since the roles are different from LSPP, the FMT_SMR.1 security functional requirement was added to satisfy the role requirements. FMT_SMF.1 was modified to include VIOS security management functions and FMT_MTD.1(8) was added to describe the management functions.

Object reuse is prevented for shared resources as specified in FDP_RIP.2, Note1, and FDP_RIP.3-AIX and implemented in OR.

Protection of the TSF is implemented by means of abstract machine testing and domain separation in TP, as modeled in FPT_AMT.1 and FPT_SEP.1, and by implementation of a central reference monitor meeting the requirements of FPT_RVM.1. RVM.2-AIX provides an additional protection against the possible malicious misuse of a common programming problem which can allow the security functions to be bypassed.

8.4.2 Assurance Measures Justification

The TOE summary specification in section 6.4 includes a justification that each TOE security assurance requirement is met by appropriate assurance measures.

8.4.3 Strength of function

The password mechanism used for authentication is the only mechanism in the TSF that is implemented by a permutational or probabilistic mechanism. For the password based authentication mechanism of the security function IA.1, a minimum strength of SOF-medium is claimed. This is done in accordance with the SOF claim for the related security functional requirement FIA_SOS.1. There is no other mechanism in this Security Target where a strength of function claim is required.

This claim is consistent with the security objective O.AUTHORIZATION and the statement in section 3.2 which says that the TOE should “protect against threats of inadvertent or casual attempts to breach the system security”. A highly skilled and well funded attacker is explicitly excluded from the threat scenario described in section 3.2. Therefore, a strength of SOF-medium is consistent with the description of the TOE environment.

The hard disk erasure functionality in OR.5, which is not per se probabilistic or permutational, can also be considered subject to a strength of function rating to the extent that it overwrites existing information on drives with a certain level of “noise” that makes it hard to retrieve the original information stored on the disk. However, since the hard disk drive does not leave the TSC and is only accessible via the TOE-provided interfaces, a SOF claim is not applicable for this mechanism.

9 Abbreviations

ACE	Access Control Entry
ACL	Access Control List
AIX	Advanced Interactive Executive
ANSI	American National Standards Institute
API	Application Programming Interface
CBC	Cipher-block chaining
CC	Common Criteria
CC	Common Criteria for Information Technology Security Evaluation
CDE	Common Desktop Environment
CDE	Computer Desktop Environment
CID	Corral ID
CIPSO	Common IP Security Option
CLiC	IBM CryptoLite for C
CM	Configuration Management
CTS	Ciphertext stealing
DAC	Discretionary Access Control
DAC	Discretionary Access Control
EAL	Evaluation Assurance Level
ECB	Electronic codebook
EGID	Effective Group ID
EOF	End of File
EUID	Effective User ID
FIFO	First In First Out
FIPS	Federal Information Processing Standard
FPR	Floating Point Register
FSO	File System Object
FSP	Functional Specification
FTP	File Transfer Protocol
GA	General Availability
GID	Group ID
GPR	General Purpose Register
HLD	High Level Design
HTML	Hypertext Markup Language
I&A	Identification and Authentication
ID	Identification
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IP	Internet Protocol
IPC	Internet Protocol Control56
ISO	International Standards Organization
ISSO	Information System Security Officer
JFS	Journalled File System
KAT	Kernel Authorization Table

LFS	Logical File System
LPP	Licensed Product Package
LSPP	Labeled Security Protection Profile
MAC	Mandatory Access Control
NFS	Network File System
NIM	Network Install Manager
OID	Object Identification
OR	Observation Report
PDF	Portable Data Format
PP	Protection Profile
PTF	Program Temporary Fix
RIPSO	Revised IP Security Option
RPC	Remote Procedure Call
RPC	Remote Procedure Call
RSH	Remote Shell
SA	System Administrator
SAR	Security Assurance Requirement
SED	Stack Execution Disable
SEM	Superuser Emulation Mode
SFP	Security Function Policy
SFR	Security Functional Requirement
SL	Sensitivity Label
SMIT	System Management Interface Tool
SO	System Operator
ST	Security Target
ST	Security Target
TCB	Trusted Computing Base
TCP	Transmission Control Protocol
TN	Trusted Network
TOE	Target of Evaluation
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Functions
TSF	TOE Security Function
TSP	TOE Security Policy
UDP	User Datagram Protocol
UID	User ID
VFS	Virtual File System
VIOS	Virtual I/O Server
VMM	Virtual Memory Manager