Silicon Graphics, Inc.

Trusted IRIX/CMW Version 6.5.13

Security Target

Version 1.9

May 1, 2002

Prepared for:

Silicon Graphics, Inc.

1600 Amphitheatre Parkway

Mountain View, CA 94043

Prepared by:

Science Applications International Corporation

Common Criteria Testing Laboratory

7125 Columbia Gateway Drive, Suite 300

Columbia, MD 21046

Table of Contents

List of Tables

SGI Trusted IRIX/CMW 6.5.13 Security Target

1 SECURITY TARGET (ST) INTRODUCTION

1.1 INTRODUCTION

This section contains document management and overview information. The Security Target (ST) identification provides labeling and descriptive information for the ST and Target of Evaluation (TOE) to which it refers. The overview section summarizes the ST in narrative form. The CC conformance claims section states with which portions of the CC the TOE conforms. The terms section provides a list of acronyms and definitions.

1.2 IDENTIFICATION

ST Title: Silicon Graphics, Inc. (SGI) Trusted IRIX/CMW version 6.5.13 Security Target, Version 1.9

TOE Identification: Trusted IRIX/CMW version 6.5.13, with Patches 4354, 4451, 4452, 4373, and 4473 hosted on the Origin 200 workstations and Origin 3000 servers.

Common Criteria (CC) Identification: The Common Criteria for Information Technology Security Evaluation version 2.1, August 1999

Protection Profile (PP) Identification: Labeled Security Protection Profile (LSPP), version 1.b, October 8, 1999

Keywords: operating system, mandatory access control, discretionary access control, security target

1.3 ST OVERVIEW

This ST provides a basis for the evaluation of the Trusted IRIX/CMW TOE. It identifies the environment for which Trusted IRIX is intended. The following environment factors are described in this ST:

- Assumptions regarding the security environment and the intended usage of the TOE;
- Policies identified for the TOE and the data the TOE protects; and
- Security objectives that identify the responsibilities of the TOE and its environment in meeting the security needs.

Trusted IRIX/CMW, henceforth called Trusted IRIX, is an optional add-on package for SGI's IRIX operating system. It adds trusted system features to the base operating system. IRIX is a scalable, high performance implementation of the UNIX operating system. The security features provided by IRIX with the Trusted IRIX package installed include:

- Mandatory Access Control (MAC) subject and object labeling and MAC checks;
- Mandatory Integrity (MINT) subject and object labeling and MINT checks;
- Auditing of security relevant events;
- Support for shadow passwords in the Identification and Authentication utilities;
- MAC labeling of output and input;
- Trusted networking.

Trusted IRIX provides for a level of protection which is appropriate for IT environments that require protection against threats of inadvertent or casual attempts to breach the system security. Trusted IRIX is not intended to provide protection for hostile environments or against well-funded attacks. Trusted IRIX does not fully address the threats posed by malicious administrative or system development personnel. Trusted IRIX is suitable for use in both commercial and government environments.

The structure and content of this ST complies with the requirements specified in the CC Part 1, Annex C, and Part 3, Chapter 5.

1.4 COMMON CRITERIA CONFORMANCE CLAIMS

The TOE conforms to the CC Version 2.1, Parts 2 and 3 with augmentation. Evaluation Assurance Level (EAL) 3 has been augmented with the Security Policy Modeling (ADV_SPM.1) requirement. The TOE also conforms to the Labeled Security Protection Profile, version 1.b.

1.5 CONVENTIONS

The notation, formatting and conventions used in this Security Target are largely based upon those used in the Common Criteria, Version 2.1.

1.6 TERMS

This section describes terms that are used throughout the ST. When possible, terms are defined as they exist in the *Common Criteria for Information Technology Security Evaluation*.

- **Authorized administrator / Administrator** – A user in the administrator role is an authorized user who has been granted the authority to manage the TOE. These users are expected to use this authority only in the manner prescribed by the guidance given them. The term authorized administrator is taken from the CC and LSPP and is used in the ST in those sections that are derived from the LSPP or the CC directly. Otherwise, the term administrator is used. These terms are used interchangeably.
- **Authorized User** – a user who has been properly identified and authenticated. These users are considered to be legitimate users of the TOE.
- **Capabilities** – Fine-grained privileges to permit a user to perform an action otherwise restricted.
- **Discretionary Access Control Policy (DAC)** – A policy that allows authorized users and authorized administrators to control access to objects on the basis of individual user identity or membership in a group
- **Mandatory Access Control Policy (MAC)** – A policy that is a set of rules that determines access based upon the sensitivity (e.g., SECRET) or

category (e.g., PERSONNEL, MEDICAL) of the information being accessed and the access authority of the user attempting to access that information.

- **Non-kernel Objects** – Objects that are managed by trusted processes in user-mode.
- **Protection Profile (PP)** - An implementation-independent set of security requirements for a category of TOEs that meet specific consumer needs.
- **Security Level -** The combination of a hierarchical classification and a set of non-hierarchical categories that represents the sensitivity of information.
- **Security Target (ST)** - A set of security requirements and specifications to be used as the basis for evaluation of an identified TOE.
- **Sensitivity Labels** - Specific markings that identify the sensitivity of the information and the access rights of a user.
- **Target of Evaluation (TOE)** - An IT product of system and its associated administrator and user guidance documentation that is the subject of an evaluation.
- **TOE Security Functions (TSF)** - A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the TSP.
- **TOE Security Policy (TSP)** - A set of rules that regulate how assets are managed, protected, and distributed within a TOE.
- **TSF data** - Data created by and for the TOE that might affect the operation of the TOE.
- **TSF Scope of Control (TSC)** - The set of interactions that can occur with or within a TOE and are subject to the rules of the TSP.
- **User** – An individual who attempts to invoke a service offered by the TOE.

2 TARGET OF EVALUATION (TOE) DESCRIPTION

The Trusted IRIX system under evaluation is a system of Silicon Graphics Computer Systems, Inc. (SGI) Origin200 workstations and the Origin 3000 servers connected via an Ethernet. These UNIX-based multi-user, multi-tasking workstations provide high-performance, general-purpose computing in a reduced instruction set computer (RISC) workstation environment. The processor of the Trusted IRIX system workstation and server is the SGI MIPS R12000.

The SGI Origin 3000 Series is a family of modular computer server systems. The various internal components of the various SGI Origin 3000 servers and their functions are divided into separate units called "bricks" for system customization to meet various customer computing needs. These bricks are housed in short or tall rack enclosures.

The SGI Origin200 workstation is a multiprocessor system that consists of one or two chassis, which are called modules. The Origin200 GIGAchannel uses an additional chassis to provide four extra PCI slots and five XIO slots. Each Origin200 system ships from SGI in either a tower (free-standing) or rackmountable configuration.

The Trusted IRIX operating system is a security-enhanced version of the IRIX operating system. In addition to the IRIX identity-based discretionary access control (DAC) on system resources, Trusted IRIX controls access to system resources based on the sensitivity and integrity labels of each resource.

Trusted IRIX supports a set of access control policies; an identification and authentication capability to mediate and validate requests for entry into the system; an audit trail capability; and networking capability.

2.1 TOE PHYSICAL BOUNDARIES

The evaluated hardware is one or more Origin 200 workstations and Origin 3000 servers connected via an Ethernet with one or more SGI MIPS R12000 processors running Trusted IRIX. A set of devices may be attached and they are listed as follows:

- Dumb Terminal, (only on the Origin200, includes keyboard and monitor)
- Floppy Disk Drive,
- CD-ROM Drive
- SCSI Tape Drive,
- Fixed Disk Drives,
- Dumb Printer (only on the Origin200), and
- Network Adaptor.

The TOE does not include any physical network components between network adaptors of a connection. The ST assumes that any network connections, equipment, and cables are appropriately protected in the TOE security environment.

2.2 TOE LOGICAL BOUNDARIES

The diagram below depicts components of Trusted IRIX that comprise the TOE. The components are large portions of the Trusted IRIX operating system.

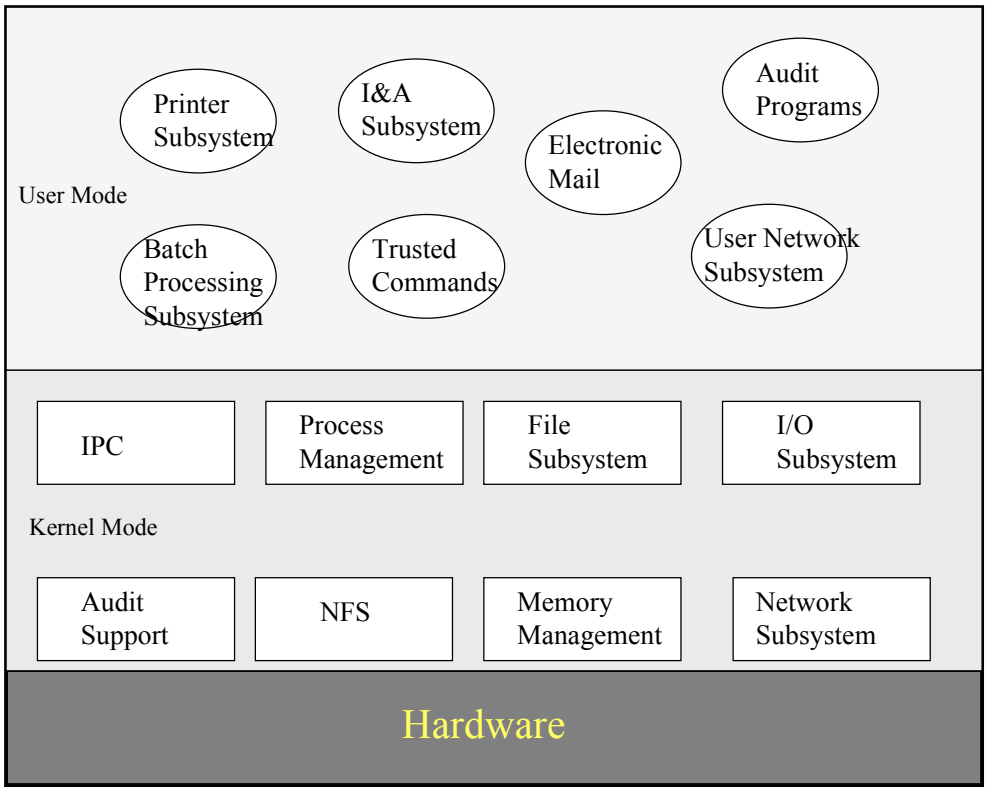

Figure 1 TOE Logical Architecture

Trusted IRIX is a security-enhanced version of IRIX. The major enhancements made to the base IRIX operating system (OS) are the additions of a label-based MAC policy based on the Bell and LaPadula [1] model and the Biba [2] model, an audit capability, and an extension of the IRIX DAC policy.

Trusted IRIX's MAC policy for sensitivity and integrity is implemented with labels. Each protected resource has associated with it a label that has two components: a sensitivity and an integrity component. These labels are used to determine access to a resource in accordance with the system's MAC policy. This policy defines a dominance relationship between the labels.

SGI uses the integrity component of the label as a TOE identification and isolation mechanism. All software components of the TOE are labeled with a system high integrity level. Therefore, this integrity level in the label identifies the resource as being within the TOE. Also, all system processes that run as part of the TOE execute with system high integrity. Administrators may have this integrity level as part of their label, but it is not within the range available to other users. Thus, users do not have the ability to write a program that can be executed by an administrator. The Trusted IRIX MAC policy does not allow any user to modify a TOE file and no administrator to invoke a program that is not in the TOE.

The Trusted IRIX kernel is responsible for providing memory management, process management, a consistent interface to system hardware, and mediating access to protected resources. It also provides a hierarchical view of files on mass storage devices, and network protocols for communication between machines.

A process is an instance of a program in execution. Trusted IRIX has three types of processes: user processes, daemon processes, and kernel processes. A user process is an active entity that operates on behalf of a user, runs in user mode, and requests services with system calls and unprivileged hardware instructions. Daemon processes are processes that perform system-wide functions such as line printer spooling. They are not associated with any users although, like user processes, they run in user mode and request services with system calls and unprivileged hardware instructions. A kernel process is an active entity that operates in kernel mode.

Trusted IRIX views stored data as unformatted streams of bytes and represents it as files organized in file systems. A file can be (1) a set of data stored on disk, (2) a device special file that represents an I/O device or a pseudo-device such as kernel memory or pseudo-terminals, or (3) other special mechanisms such as pipes and sockets. The files are organized into hierarchical file systems. Trusted IRIX supports multiple file system types in a manner transparent to the user. They include the IRIX eXtended File System (XFS) high-performance file system; the Network File System (NFS) for files mounted on remote disks that are exported read-only; and the Debug File System (DBG), a pseudo-file system in which a running process is represented as a file.

All the Trusted IRIX file systems are organized into a traditional UNIX-like hierarchy with files as nodes. Non-leaf files are directories, special files which contain references to other files. All files can be identified by their place in the hierarchy, i.e., their pathname. Associated with every file is a set of attributes. These attributes include the file owner, a set of users that are assigned access rights to the file as a group, and information that specifies the access rights of the owner, group, and all other system users. Each of these is a Trusted IRIX protected resource.

The Trusted IRIX kernel provides a variety of other mechanisms for interprocess communication. These include traditional UNIX mechanisms such as the pipe; System V mechanisms such as shared memory, semaphore sets, and message queues; and BSD TCP/IP sockets. Trusted IRIX protects the interprocess communications just listed.

Trusted IRIX supports a least privilege mechanism through the implementation of POSIX P1003.1eD17 capabilities. The SuperUser privileges have been broken out into a set of distinct capabilities, which can be granted and relinquished through a set of inheritance rules.

The Trusted IRIX system is a distributed system and supports a range of network protocols and services. Trusted IRIX uses a protocol-based label to ensure that data maintains its label and originator accountability as it traverses the system. Trusted IRIX places the label of the data and the sender's UID in each data packet transmitted between workstations of the Trusted IRIX system.

All workstations in the Trusted IRIX system hold an identification and authentication (I&A) database. A user information file, not visible to ordinary users, contains authentication and DAC related data. Other I&A related information is placed in files that are linked to the standard UNIX passwd file and group file; users are given read-only access to these files. Passwords are used to authenticate users of the Trusted IRIX system.

Audit records of security relevant events are generated on each workstation of the Trusted IRIX system. These records contain the initial login identifier of the user who initiated the audited event. Trusted IRIX commands allow the system administrator to selectively audit events. SGI provides tools to reduce the audit logs for analysis.

Trusted IRIX supports the UNIX setuid and setgid mechanisms that allow a process to run with the UID or GID of the owner or owning group of the invoked file. Some Trusted IRIX special user identifiers own Trusted IRIX programs and Trusted IRIX uses the setuid mechanism so that processes invoking these programs assume the special identity. Processes invoking these programs assume the special identity via the setuid mechanism.

3 TOE SECURITY ENVIRONMENT

This section 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. The material for the environmental description was taken from the LSPP.

3.1 THREATS

The LSPP has derived all security objectives from the statement of Organizational Security Policy found in the following section. Therefore, there is no statement of the explicit threats countered by the LSPP.

3.2 ORGANIZATIONAL SECURITY POLICIES

An Organizational Security Policy is a set of rules or procedures imposed by an organization upon its operations to protect its sensitive data. Although the organizational security policies described below are drawn from DoD Manual 5200.28-M (Techniques and procedures for Implementing, Deactivating and Evaluating Resource Sharing ADP Systems) they apply to many non-DoD environments.

P.AUTHORIZED_USERS

Only those users who have been authorized to access the information within the system may access the system.

P.NEED_TO_KNOW

The system must limit the access to, modification of, and destruction of the information in protected resources to those authorized users which have a “need to know” for that information.

P.ACCOUNTABILITY

The users of the system shall be held accountable for their actions within the system.

P.CLASSIFICATION

The system must limit the access to information based on sensitivity, as represented by a label, of the information contained in objects, and the formal clearance of users, as represented by subjects, to access that information. The access rules enforced prevent a subject from accessing information which is of higher sensitivity than it is operating at and prevent a subject from causing information from being downgraded to a lower sensitivity.

The method for classification of information is made based on criteria set forth by the organization. This is usually done on a basis of relative value to the organization and its interest to limit dissemination of that information. The determination of classification of information is outside the scope of the IT system; the IT system is only expected to enforce the classification rules, not determine classification.

The method for determining clearances is also outside the scope of the IT system. It is essentially based on the trust placed in individual users by the organization. To some extent is also dependent upon the individual's role within the organization.

3.3 ASSUMPTIONS

This section describes the security aspects of the environment in which the TOE will be, or is intended to be used. This includes information about the physical, personnel, and connectivity aspects of the environment.

The TOE is assured to provide effective security measures in a cooperative non-hostile environment only if it is installed, managed, and used correctly. The operational environment must be managed in accordance with assurance requirements documentation for delivery, operation, and user/administrator guidance. The following specific conditions are assumed to exist in an environment where the TOE is employed.

3.3.1 Physical Assumptions

LSPP-conformant TOEs are intended for application in user areas that have physical control and monitoring. It is assumed that the following physical conditions will exist:

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.

3.3.2 Personnel Assumptions

It is assumed that the following personnel conditions will exist:

A.MANAGE

There will be one or more competent individuals 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.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.

3.3.3 Procedural Assumptions

The ability of the LSPP to enforce the intent of the organizational security policy, especially with regard to the Mandatory Access Controls, is dependent upon the establishment of procedures. It is assumed that the following procedural controls exist. In addition to the LSPP assumptions, the A.LABELS assumption has been added.

A. CLEARANCE

Procedures exist for granting users authorization for access to specific security levels.

A. SENSITIVITY

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.

A.LABELS

Procedures exist for the administrator to ensure that all internal representations of security levels are consistent between all machines.

3.3.4 Connectivity Assumptions

The LSPP contains no explicit network or distributed system requirements. However, it is assumed that the following connectivity conditions exist:

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. LSPP-conformant TOEs are applicable to networked or distributed environments only if the entire network operates under the same constraints and resides within a single management domain. There are no security requirements, which address the need to trust external systems or the communications links to such systems.

A.MGMT

The TOE must operate under a single management domain with each TSF sharing the same identification and authentication database.

A.CONNECT

All connections to peripheral devices reside within the controlled access facilities. LSPP-conformant TOEs only address security concerns related to the manipulation of the TOE through its authorized access points. Internal communication paths to access points such as terminals are assumed to be adequately protected.

4 SECURITY OBJECTIVES

This section identifies the security objectives for the TOE and its environment. The security objectives identify the responsibilities of the TOE security functions (TSFs) and their environment in meeting the security needs.

4.1 INFORMATION TECHNOLOGY (IT) SECURITY OBJECTIVES

The following are the TOE security objectives:

O.AUTHORIZATION

The TSF must ensure that only authorized users gain access to the TOE and its resources.

O.DISCRETIONARY_ACCESS

The TSF must control access to resources based on identity of users. The TSF must allow authorized users to specify which resources may be accessed by which users.

O.MANDATORY_ACCESS

The TSF must control access to resources based upon the sensitivity and categories of the information being accessed and the clearance of the subject attempting to access that information.

O.AUDITING

The TSF must record the security relevant actions of users of the TOE. The TSF must present this information to authorized administrators.

O.RESIDUAL_INFORMATION

The TSF must ensure that any information contained in a protected resource is not released when the resource is recycled.

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.

O.ENFORCEMENT

The TSF must be designed and implemented in a manner which ensures that the organizational policies are enforced in the target environment.

4.2 SECURITY OBJECTIVES FOR THE ENVIRONMENT

The TOEs operating environment must satisfy the following objectives. These objectives do not levy any IT requirements but are satisfied by procedural or administrative measures.

O.INSTALL

Those responsible for the TOE must ensure that the TOE is delivered, installed, managed, and operated in a manner which maintains IT security objectives.

O.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.

O.CREDEN

Those responsible for the TOE must ensure that all access credentials, such as passwords or other authentication information, are protected by the users in a manner which maintains IT security objectives.

5 IT SECURITY REQUIREMENTS

This chapter defines the functional requirements for the TOE. Functional requirements components in this ST were drawn from the LSPP. The LSPP uses Part 2 of the CC. Some functional requirements are extensions to those found in the CC.

CC defined operations for assignment, selection, and refinement were used to tailor the requirements to the level of detail necessary to meet the stated security objectives. These operations are indicated through the use of underlined (assignments and selections) and italicized (refinements) text.

Security Requirement	Functional Components	
Class FAU: Security Audit	FAU_GEN.1	Audit data generation
	FAU_GEN.2	User Identity Association
	FAU_SAR.1	Audit review
	FAU_SAR.2	Restricted audit review
	FAU_SAR.3	Selectable audit review
	FAU_SEL.1	Selective audit
	FAU_STG.1	Guarantees of audit data availability
	FAU_STG.3	Action in case of possible audit data loss
	FAU_STG.4	Prevention of audit data loss
Class FDP: User Data Protection	FDP_ACC.1	Discretionary Access Control Policy
	FDP_ACF.1	Discretionary Access Control Functions
	FDP_ETC.1	Export of unlabeled user data
	FDP_ETC.2	Export of labeled user data
	FDP_IFC.1	Mandatory Access Control Policy
	FDP_IFF.2	Mandatory Access Control Functions
	FDP_ITC.1	Import of unlabeled user data
	FDP_ITC.2	Import of labeled user data
	FDP_RIP.2	Object Residual Information Protection
	Note 1	Subject Residual Information Protection
Class FIA: Identification and Authentication	FIA_ATD.1	User attribute definition
	FIA_SOS.1	Strength of authentication data
	FIA_UAU.1	Timing of authentication
	FIA_UAU.7	Protected Authentication Feedback
	FIA_UID.1	Timing of identification
	FIA_USB.1	User-subject binding
Class FMT: Security Management	FMT_MSA.1	Management of object security attributes
	FMT_MSA.3	Static attribute initialization
	FMT_MTD.1	Management of the audit trail
	FMT_MTD.1	Management of audited events
	FMT_MTD.1	Management of user attributes
	FMT_MTD.1	Management of authentication data
	FMT_REV.1	Revocation of user attributes
	FMT_REV.1	Revocation of object attributes
	FMT_SMR.1	Security roles

Class FPT: Protection of the TOE Security Functions	FPT_AMT.1	Abstract machine testing
	FPT_RVM.1	Non-bypassability of the TSP
	FPT_SEP.1	TSF domain separation
	FPT_STM.1	Reliable time stamps

Table 1 TOE Functional Requirements

5.1 SECURITY AUDIT (FAU)

5.1.1 FAU_GEN.1 Audit data generation

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the auditable events *listed in column "Event"* Table 2 Auditable Events. *This includes all auditable events for the basic level of audit, except FIA_UID.1's user identity during failures*.

Section	Component	Event	Details
5.1.1	FAU_GEN.1	Start-up and shutdown of audit functions	
5.1.2	FAU_GEN.2	None	
5.1.3	FAU_SAR.1	Reading of information from the audit records.	
5.1.4	FAU_SAR.2	Unsuccessful attempts to read information from the audit records.	
5.1.5	FAU_SAR.3	None	
5.1.6	FAU_SEL.1	All modifications to the audit configuration that occur while the audit collection functions are operating.	
5.1.7	FAU_STG.2	None	
5.1.8	FAU_STG.3.	Actions taken due to exceeding of a threshold	
5.1.9	FAU_STG.4	Actions taken due to the audit storage failure.	
5.2.1	FDP_ACC.1	None	
5.2.2	FDP_ACF.1	All requests to perform an operation on an object covered by the SFP.	The identity of the object.
5.2.3	FDP_ETC.1	All attempts to export information.	
5.2.4	FDP_ETC.2	All attempts to export information.	
5.2.4	FDP_ETC.2	Overriding of human-readable output marking. (Additional)	
5.2.5	FDP_IFC.2	None	
5.2.6	FDP_IFF.2	All decisions on requests for information flow.	
5.2.7	FDP_ITC.1	All attempts to import user data, including any security attributes.	
5.2.8	FDP_ITC.2	All attempts to import user data, including any security attributes.	
5.2.9	FDP_RIP.2	None	
5.2.10	Note 1	None	
5.3.1	FIA_ATD.1	None	
5.3.2	FIA_SOS.1	Rejection or acceptance by the TSF of any tested secret.	
5.3.3	FIA_UAU.1	All use of the authentication mechanism.	
5.3.4	FIA_UAU.7	None	
5.3.5	FIA_UID.1	All use of the user identification mechanism, including the identity provided during successful attempts.	The origin of the attempt (e.g. terminal identification.)

Section	Component	Event	Details
5.3.6	FIA_USB.1	Success and failure of binding user security attributes to a subject (e.g. success and failure to create a subject).	
5.4.1	FMT_MSA.1	All modifications of the values of security attributes.	
5.4.2	FMT_MSA.3	Modifications of the default setting of permissive or restrictive rules. All modifications of the initial value of security attributes.	
5.4.3	FMT_MTD.1	All modifications to the values of TSF data.	
5.4.4	FMT_MTD.1	All modifications to the values of TSF data.	The new value of the TSF data.
5.4.5	FMT_MTD.1	All modifications to the values of TSF data.	The new value of the TSF data.
5.4.6	FMT_MTD.1	All modifications to the values of TSF data.	
5.4.7	FMT_REV.1	All attempts to revoke security attributes.	
5.4.8	FMT_REV.1	All modifications to the values of TSF data.	
5.4.9	FMT_SMR.1	Modifications to the group of users that are part of a role.	
5.4.9	FMT_SMR.1	Every use of the rights of a role. (Additional / Detailed)	The role and the origin of the request.
5.5.1	FPT_AMT.1.	Execution of the tests of the underlying machine and the results of the test.	
5.5.2	FPT_RVM.1	None	
5.5.3	FPT_SEP.1	None	
5.5.4	FPT_STM.1	Changes to the time.	

Table 2 Auditable Events

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

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

a) *The sensitivity labels of subjects, objects, or information involved*; and

b) *The additional information specified in the Details column of Table 2 Auditable Events*.

5.1.2 FAU_GEN.2 User Identity Association

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

5.1.3 FAU_SAR.1 Audit review

FAU_SAR.1.1 The TSF shall provide *auditor* with the capability to read all audit information from the audit records.

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

5.1.4 FAU_SAR.2 Restricted audit review

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

5.1.5 FAU_SAR.3 Selectable audit review

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

a) User identity

b) Subject sensitivity label;

c) Object sensitivity label;

d) Date and time;

e) Event type;

f) Object Identifier; and

g) Success or failure of related event.

5.1.6 FAU_SEL.1 Selective audit

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

a) User identity;

b) Subject sensitivity label; and

c) Object sensitivity label.

5.1.7 FAU_STG.1 Guarantees of audit data availability

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

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

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

FAU_STG.3.1 The TSF shall generate an alarm to the *auditor* if the audit trail exceeds 90% capacity.

5.1.9 FAU_STG.4 Prevention of audit data loss

FAU_STG.4.1 The TSF shall *be able to* prevent auditable events, except those taken by the *auditor*, and overwrite old records if the audit trail is full.

5.2 USER DATA PROTECTION (FDP)

5.2.1 FDP_ACC.1 Discretionary Access Control Policy

FDP_ACC.1.1 The TSF shall enforce the Discretionary Access Control Policy on subjects: share groups acting on the behalf of users, objects: files, directories, named pipes, symbolic links, unnamed pipes, processes, System V IPC objects, at jobs, crontab files, and print queue entries, and all operations among subjects and objects covered by the DAC policy.

5.2.2 FDP_ACF.1 Discretionary Access Control Functions

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

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

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

i) Unix Permission bits[1];

ii) ACL;

iii) Object Ownership; and

iv) Object Creator.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed[2]:

1. If the object has an explicit ACL, access is granted if one of the following is true:
    a. An ACL entry explicitly grants access to a user,
    b. An entry explicitly grants access to a group of which the subject is a member and the access has not been denied by a previous entry in the ACL, or
    c. An ACL entry explicitly grants access to the world and the access has not been denied by a previous entry in the ACL.
2. If an object does not have an ACL and has permission bits, access is granted if one of the following is true:
    a. If the subject is the owner of the object and the object's owner Unix permission bits indicate that the operation required accesses are allowed.

[1] This refers to the standard user/group/world read, write, and execute UNIX permission bits.
[2] Note that NFS file systems are mounted read-only.

b. If the subject is a member of the object owning group and the object's group Unix permission bits indicate that the operation required accesses are allowed, or

c. If the object's world Unix permission bits indicate that the operation required accesses are allowed.

3. The subject's effective or real UID is the same as the object owner or creator and the operation is performed on a process, System V IPC object, at job, crontab file, or print queue entry

4. The subject's process group ID is the same as the object owner or creator and the operation is performed on a process.

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based in the following additional rules: If a subject has the appropriate capability[3], the TSF shall authorize access of the subject to any object, even if such access is disallowed by FDP_ACF.1.2. In some instances, if a subject has the UID of 0, access to the object may be granted, even if such access is disallowed by FDP_ACF.1.2.

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

5.2.3 FDP_ETC.1 Export of unlabeled user data

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

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

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

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.

5.2.4 FDP_ETC.2 Export of labeled user data

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

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

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

[3] Capabilities to override DAC are not applicable to NFS.

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

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; and

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

5.2.5 FDP_IFC.1 Mandatory Access Control Policy

FDP_IFC.1.1 The TSF shall enforce the Mandatory Access Control Policy on subject: share groups, and objects: files, directories, named pipes, symbolic links, unnamed pipes, System V IPC objects, BSD TCP/IP sockets, at jobs, crontab files, and print queue entries and all operations among subjects and objects covered by the MAC policy.

5.2.6 FDP_IFF.2 Mandatory Access Control Functions

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

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.

FDP_IFF.2.2 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 *and the mandatory integrity policy rules hold:*

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.

FDP_IFF.2.3 The TSF shall enforce the mandatory integrity policy. The mandatory integrity policy permits an information flow between a controlled subject and controlled information via a controlled operation if the following rules, based on the ordering relationships, are true *and the mandatory access control policy rules hold*:

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

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

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

FDP_IFF.2.4 The TSF shall provide an integrity label with each subject and object. An integrity label consists of the following:

- A hierarchical grade; and
- A set of non-hierarchical divisions.

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

a) If the sensitivity label of the subject is greater than or equal to the sensitivity level of the object and the integrity label of the subject is less than or equal to the integrity level of the object, then a read operation is allowed.

b) If the sensitivity label of subject A is greater than or equal to the sensitivity level of the subject B and the integrity label of the subject A is less than or equal to the integrity level of the subject B, then a read operation is allowed

c) If the sensitivity label of the subject is equal to the sensitivity level of the object and the integrity label of the subject is equal to the integrity level of the object, then a write operation is allowed.

d) If the sensitivity label of subject A is equal to the sensitivity level of subject B and the integrity label of subject A is equal to the integrity level of subject B, then a write operation is allowed.

e) If a subject has the appropriate capability, the TSF shall authorize access of the subject to any object, even if such access is disallowed by FDP_IFF.2.6[4].

FDP_IFF.2.6 The TSF shall explicitly deny an information flow based on the following rules: [none]

[4] Capabilities to override MAC are not applicable to NFS.

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

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*.

5.2.7 FDP_ITC.1 Import of unlabeled user data

FDP_ITC.1.1 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.1.2 The TSF shall ignore any security attributes associated with the *unlabeled* user data when imported from outside the TSC.

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

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.

5.2.8 FDP_ITC.2 Import of labeled user data

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

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

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

FDP_ITC.2.4 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.2.5 The TSF shall enforce the following rules when importing *labeled* user data controlled under the *MAC policy* from outside the TSC:

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) Sensitivity label, consisting of the following:

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

5.2.9 FDP_RIP.2 Object Residual Information Protection

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

5.2.10 Note 1 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.

5.3 IDENTIFICATION AND AUTHENTICATION (FIA)

5.3.1 FIA_ATD.1 User attribute definition

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

a) User identifier;

b) Group memberships;

c) Authentication data;

d) User clearances;

e) Security-relevant roles; and

f) Capabilities.

5.3.2 FIA_SOS.1 Strength of authentication data

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

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.

5.3.3 FIA_UAU.1 Timing of authentication

FIA_UAU.1.1 The TSF shall allow no TSF-mediated actions on behalf of the user to be performed before the user is authenticated.

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

5.3.4 FIA_UAU.7 Protected Authentication Feedback

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

5.3.5 FIA_UID.1 Timing of identification

FIA_UID.1.1 The TSF shall allow no TSF-mediated actions on behalf of the user to be performed before the user is identified.

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

5.3.6 FIA_USB.1 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:

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) The sensitivity label used to enforce the Mandatory Access Control Policy, which consists of the following:

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

e) The capabilities associated with a user identity.

Note 2.1 *The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of a user*:

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

b) The security attributes shall be a subset of those defined for the user.

Note 2.2 *The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of a user:*

a) The effective user identity associated with a subject can be changed to another user's identity via a command, provided that successful authentication as the new user identity has been achieved;

b) When executing a file which has the set UID permission bit set, the effective user identity associated with the subject shall be changed to that of the owner of the file;

c) When executing a file which has the set GID permission bit set, the effective group identity associated with the subject shall be changed to that of the group attribute of the file.

5.4 SECURITY MANAGEMENT (FMT)

5.4.1 FMT_MSA.1 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 the object owner or users with the appropriate capability.

FMT_MSA.1.2 The TSF shall enforce the Mandatory Access Control Policy to restrict the ability to modify the sensitivity label associated with an object to users with the appropriate capability.

5.4.2 FMT_MSA.3 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 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.3.2 The TSF shall allow the creator to specify alternative initial values to override the default values when an object or information is created.

5.4.3 FMT_MTD.1 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.

5.4.4 FMT_MTD.1 Management of audited events

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

5.4.5 FMT_MTD.1 Management of user attributes

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

5.4.6 FMT_MTD.1 Management of authentication data

FMT_MTD.1.1 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:

a) authorized administrators; and

b) users authorized to modify their own authentication data.

5.4.7 FMT_REV.1 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 authorized administrators.

FMT_REV.1.2 The TSF shall enforce the rules:

a) The immediate revocation of security-relevant authorizations.

5.4.8 FMT_REV.1 Revocation of object attributes

FMT_REV.1.1 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 or Mandatory Access Control policies.

FMT_REV.1.2 The TSF shall enforce the rules:

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

b) The rules of the Mandatory Access Control policy (5.2.6) are enforced on all future operations.

5.4.9 FMT_SMR.1 Security roles

FMT_SMR.1.1 The TSF shall maintain the roles:

a) authorized administrator;

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

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

d) users authorized to modify their own authentication data;

e) auditor; and

f) lp.

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

5.5 PROTECTION OF THE TOE SECURITY FUNCTIONS (FPT)

5.5.1 FPT_AMT.1 Abstract machine testing

FPT_AMT.1.1 The TSF shall run a suite of tests during initial start-up, or 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.

5.5.2 FPT_RVM.1 Non-bypassability of the TSP

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

5.5.3 FPT_SEP.1 TSF domain separation

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

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

5.5.4 FPT_STM.1 Reliable time stamps

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

5.6 STRENGTH OF FUNCTION REQUIREMENT

The minimum strength of function level for the security functional requirements is SOF-medium.

6 ASSURANCE REQUIREMENTS

This chapter defines the assurance requirements for the TOE. Assurance requirements are taken from the LSPP. The LSPP assurance requirements are EAL3 augmented with the ADV_SPM.1 requirement, and are derived from the CC, Part 3.

Assurance Requirements	Assurance Components
Class ACM: Configuration Management	ACM_CAP.3 Configuration Items ACM_SCP.1 Coverage
Class ADO: Delivery and Operation	ADO_DEL.1 Delivery Procedures ADO_IGS.1 Installation, Generation, And Start-Up Procedures
Class ADV: Development	ADV_FSP.1 Informal Functional Specification ADV_HLD.2 Descriptive High-Level Design ADV_RCR.1 Informal Correspondence Demonstration ADV_SPM.1 Security Policy Modeling
Class AGD: Guidance Documents	AGD_ADM.1 Administrator Guidance AGD_USR.1 User Guidance
Class ALC: Life Cycle Support	ALC_DVS.1 Identification of Security Measures
Class ATE: Tests	ATE_COV.2 Evidence Of Coverage ATE_DTP.1 Depth ATE_FUN.1 Functional Testing ATE_IND.2 Independent Testing
Class AVA: Vulnerability Assessment	AVA_MSU.1 Examination of Guidance AVA_SOF.1 Strength Of TOE Security Function Evaluation AVA_VLA.1 Developer Vulnerability Analysis

Table 3 Assurance Components

6.1 CONFIGURATION MANAGEMENT (ACM)

6.1.1 Configuration Items (ACM_CAP.3)

ACM_CAP.3.1D The developer shall provide a reference for the TOE.

ACM_CAP.3.2D The developer shall use a CM system.

ACM_CAP.3.3D The developer shall provide CM documentation.

ACM_CAP.3.1C The reference for the TOE shall be unique to each version of the TOE.

ACM_CAP.3.2C The TOE shall be labeled with its reference.

ACM_CAP.3.3C The CM documentation shall include a configuration list and CM plan.

ACM_CAP.3.4C The configuration list shall describe the configuration items that comprise the TOE.

ACM_CAP.3.5C The CM documentation shall describe the method used to uniquely identify the configuration items.

ACM_CAP.3.6C The CM system shall uniquely identify all configuration items.

ACM_CAP.3.7C The CM plan shall describe how the CM system is used.

ACM_CAP.3.8C The evidence shall demonstrate that the CM system is operating in accordance with the CM plan.

ACM_CAP.3.9C The CM documentation shall provide evidence that all configuration items have been and are being effectively maintained under the CM system.

ACM_CAP.3.10C The CM system shall provide measures such that only authorized changes are made to the configuration items.

ACM_CAP.3.1E The evaluator shall confirm that the information provided meets all the requirements for content and presentation of evidence.

6.1.2 Coverage (ACM_SCP.1)

ACM_SCP.1.1D The developer shall provide CM documentation.

ACM_SCP.1.1C The CM documentation shall show that the CM system, as a minimum, tracks the following: the TOE implementation representation, design documentation, test documentation, user documentation, administrator documentation, and CM documentation.

ACM_SCP.1.2C The CM documentation shall describe how configuration items are tracked by the CM system.

ACM_SCP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.2 DELIVERY AND OPERATION (ADO)

6.2.1 Delivery Procedures (ADO_DEL.1)

ADO_DEL.1.1D The developer shall document procedures for delivery of the TOE or parts of it to the user.

ADO_DEL.1.2D The developer shall use the delivery procedures.

ADO_DEL.1.1C The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to a user's site.

ADO_DEL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.2.2 Installation, Generation, and Start-up Procedures (ADO_IGS.1)

ADO_IGS.1.1D The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE.

ADO_IGS.1.1C The documentation shall describe the steps necessary for secure installation, generation, and start-up of the TOE.

ADO_IGS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_IGS.1.2E The evaluator shall determine that the installation, generation, and start-up procedures result in a secure configuration.

6.3 DEVELOPMENT (ADV)

6.3.1 Informal Functional Specification (ADV_FSP.1)

ADV_FSP.1.1D The developer shall provide a functional specification.

ADV_FSP.1.1C The functional specification shall describe the TSF and its external interfaces using an informal style.

ADV_FSP.1.2C The functional specification shall be internally consistent.

ADV_FSP.1.3C The functional specification shall describe the purpose and method of use of all external TSF interfaces, providing details of effects, exceptions and error messages, as appropriate.

ADV_FSP.1.4C The functional specification shall completely represent the TSF.

ADV_FSP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.1.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the TOE security functional requirements.

6.3.2 Descriptive High-Level Design (ADV_HLD.2)

ADV_HLD.2.1D The developer shall provide the high-level design of the TSF.

ADV_HLD.2.1C The presentation of the high-level design shall be informal.

ADV_HLD.2.2C The high-level design shall be internally consistent.

ADV_HLD.2.3C The high-level design shall describe the structure of the TSF in terms of subsystems.

ADV_HLD.2.4C The high-level design shall describe the security functionality provided by each subsystem of the TSF.

ADV_HLD.2.5C The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.

ADV_HLD.2.6C The high-level design shall identify all interfaces to the subsystems of the TSF.

ADV_HLD.2.7C The high-level design shall identify which of the interfaces to the subsystems of the TSF are externally visible.

ADV_HLD.2.8C The high-level design shall describe the purpose and method of use of all interfaces to the subsystems of the TSF, providing details of effects, exceptions and error messages, as appropriate.

ADV_HLD.2.9C The high-level design shall describe the separation of the TSF into TSP-enforcing and other subsystems.

ADV_HLD.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD.2.2E The evaluator shall determine that the high-level design is an accurate and complete instantiation of the TOE security functional requirements.

6.3.3 Informal Correspondence Demonstration (ADV_RCR.1)

ADV_RCR.1.1D The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided.

ADV_RCR.1.1C For each adjacent pair of provided TSF representations, the analysis shall demonstrate that all relevant security functionality of the more abstract TSF representation is correctly and completely refined in the less abstract TSF representation.

ADV_RCR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.3.4 Security Policy Modeling (ADV_SPM.1)

ADV_FSP.1.1D The developer shall provide a TSP model.

ADV_SPM.1.2D The developer shall demonstrate correspondence between the functional specification and the TSP model.

ADV_SPM.1.1C The TSP model shall be informal.

ADV_SPM.1.2C The TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.

ADV_SPM.1.3C The TSP model shall include a rationale that demonstrates that it is consistent and complete with respect to all of the TSP that can be modeled.

ADV_SPM.1.4C The demonstration of the correspondence between the TSP model and the functional specification shall show that all the security functions in the functional specification are consistent and complete with respect to the TSP model.

ADV_SPM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.4 GUIDANCE DOCUMENTS (AGD)

6.4.1 Administrator Guidance (AGD_ADM.1)

AGD_ADM.1.1D The developer shall provide administrator guidance addressed to system administrative personnel.

AGD_ADM.1.1C The administrator guidance shall describe the administrative functions and interfaces available to the administrator of the TOE.

AGD_ADM.1.2C The administrator guidance shall describe how to administer the TOE in a secure manner.

AGD_ADM.1.3C The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.

AGD_ADM.1.4C The administrator guidance shall describe all assumptions regarding user behaviour that are relevant to secure operation of the TOE.

AGD_ADM.1.5C The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate.

AGD_ADM.1.6C The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_ADM.1.7C The administrator guidance shall be consistent with all other documentation supplied for evaluation.

AGD_ADM.1.8C The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator.

AGD_ADM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.4.2 User Guidance (AGD_USR.1)

AGD_USR.1.1D The developer shall provide user guidance.

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

AGD_USR.1.2C The user guidance shall describe the use of user-accessible security functions provided by the TOE.

AGD_USR.1.3C The user guidance shall contain warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

AGD_USR.1.4C The user guidance shall clearly present all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment.

AGD_USR.1.5C The user guidance shall be consistent with all other documentation supplied for evaluation.

AGD_USR.1.6C The user guidance shall describe all security requirements for the IT environment that are relevant to the user.

AGD_USR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.5 LIFE CYCLE SUPPORT (ALC)

6.5.1 Identification of Security Measures (ALC_DVS.1)

ALC_DVS.1.1D The developer shall produce development security documentation.

ALC_DVS.1.1C The development security documentation shall describe all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.

ALC_DVS.1.2C The development security documentation shall provide evidence that these security measures are followed during the development and maintenance of the TOE.

ALC_DVS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_DVS.1.2E The evaluator shall confirm that the security measures are being applied.

6.6 TESTS (ATE)

6.6.1 Evidence of Coverage (ATE_COV.2)

ATE_COV.2.1D The developer shall provide an analysis of the test coverage.

ATE_COV.2.1C The analysis of the test coverage shall demonstrate the correspondence between the tests identified in the test documentation and the TSF as described in the functional specification.

ATR_COV.2.2C The analysis of the test coverage shall demonstrate that the correspondence between the TSF as described in the functional specification and the tests identified in the test documentation is complete.

ATE_COV.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.6.2 Depth (ATE_DPT.1)

ATE_DPT.1.1D The developer shall provide the analysis of the depth of testing.

ATE_DPT.1.1C The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TSF operates in accordance with its high-level design.

ATE_DPT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.6.3 Functional Testing (ATE_FUN.1)

ATE_FUN.1.1D The developer shall test the TSF and document the results.

ATE_FUN.1.2D The developer shall provide test documentation.

ATE_FUN.1.1C The test documentation shall consist of test plans, test procedure descriptions, expected test results and actual test results.

ATE_FUN.1.2C The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed.

ATE_FUN.1.3C The test procedure descriptions shall identify the tests to be performed and describe the scenarios

for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests.

ATE_FUN.1.4C The expected test results shall show the anticipated outputs from a successful execution of the tests.

ATE_FUN.1.5C The test results from the developer execution of the tests shall demonstrate that each tested security function behaved as specified.

ATE_FUN.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.6.4 Independent Testing (ATE_IND.2)

ATE_IND.2.1D The developer shall provide the TOE for testing.

ATE_IND.2.1C The TOE shall be suitable for testing.

ATE_IND.2.2C The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.

ATE_IND.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.2.2E The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified.

ATE_IND.2.3E The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

6.7 VULNERABILITY ASSESSMENT (AVA)

6.7.1 Examination of Guidance (AVA_MSU.1)

AVA_MSU.1.1D The developer shall provide guidance documentation.

AVA_MSU.1.1C The guidance documentation shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.

AVA_MSU.1.2C The guidance documentation shall be complete, clear, consistent and reasonable.

AVA_MSU.1.3C The guidance documentation shall list all assumptions about the intended environment.

AVA_MSU.1.4C The guidance documentation shall list all requirements for external security measures (including external procedural, physical and personnel controls).

AVA_MSU.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_MSU.1.2E The evaluator shall repeat all configuration and installation procedures to confirm that the TOE can be configured and used securely using only the supplied guidance documentation.

AVA_MSU.1.3E The evaluator shall determine that the use of the guidance documentation allows all insecure states to be detected.

6.7.2 Strength of TOE Security Function Evaluation (AVA_SOF.1)

AVA_SOF.1.1D The developer shall perform a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim.

AVA_SOF.1.1C For each mechanism with a strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the minimum strength level defined in the PP/ST..

AVA_SOF.1.2C For each mechanism with a specific strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the specific strength of function metric defined in the PP/ST.

AVA_SOF.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_SOF.1.2E The evaluator shall confirm that the strength claims are correct.

6.7.3 Developer Vulnerability Analysis (AVA_VLA.1)

AVA_VLA.1.1D The developer shall perform and document an analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP.

AVA_VLA.1.2D The developer shall document the disposition of obvious vulnerabilities.

AVA_VLA.1.1C The documentation shall show, for all identified vulnerabilities, that the vulnerability cannot be exploited in the intended environment for the TOE.

AVA_VLA.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VLA.1.2E The evaluator shall conduct penetration testing, building on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.

7 TOE SUMMARY SPECIFICATION

This section describes the SGI Trusted IRIX in terms of security functions and then explains how the requirements are satisfied by those functions.

7.1 SECURITY FUNCTIONS

7.1.1 User-subject binding

Share groups are used to associate users with subjects on the operating system. A subject is an entity within the TSC that causes operations to be performed. A share group is a set of processes that potentially all share the same address space. The degenerate case of a share group is a single process. A process is one thread of execution within a share group. Each process in a share group has its own stack. Every process in a share group potentially has access to the data (including the stack) of all the others. Other process attributes that may be shared include the address space, open file table, the current and root directories, the umask, the maximum file size, and the real and effective user identifiers (UIDs) and group identifiers (GIDs). The reason the entire share group must be a subject (vs. the individual processes within it) is that there is potentially no isolation between the members of the share group. They share the same address space.

There are a number of security attributes associated with a subject. The attributes are assigned to a subject at creation time. These attributes are:

- Real UID
- Saved UID
- Saved GID
- Process mandatory access control (MAC) label
- Process group ID and session ID
- Process group leader ID
- Effective UID
- Audit UID (SAT ID)
- Real GID
- Effective GID
- Group list
- Process umask
- Effective Capability Set
- Inherited Capability Set
- Permitted Capability Set

Only administrative users (i.e., administrator and auditor) have capabilities assigned by default. The process MAC label is stored in the system label list This label also contains a flag designating if the process is moldy or not. A moldy process may create and directly view multi-level directories.

An existing subject can create a new subject by using the fork or exec system calls. A fork creates a new process that is a copy of the creating process. The new process is a child of the creator and has a new unique process ID. When a process within a share group performs an exec, it becomes the first process in a new share group and therefore, a new subject. A sproc system call is used to create new process within an existing share

group, but does not create a new subject unless and until the new process performs an exec system call, which changes its security attributes. The operations that can create new subjects are:

- A user logging into workstation locally or remotely
- A process executing the fork or exec system calls as described above
- A batch job being activated

In general the subject attributes are taken from the creating subject. The exception to this is the executing of a setuid or setgid program or the execution of the su program. When executing a setuid or setgid program, the effective UID or GID, respectively, of the process will be changed. The new effective ID is taken from the file owner of the file being executed. The process may then use the setreuid system call or the setregid system call to interchange the real, effective, and saved UID or GID. In addition the process may use the setregid system call to set its real or effective GID to one of the groups in the process group structure. Successful execution of the su program allows the subject to taken on the targeted identity.

7.1.2 Audit

7.1.2.1 Audit Start/Stop

Trusted IRIX maintains an audit trail using the system audit trail subsystem (SAT). This comprises a set of programs, system calls, library functions, administrator commands, and databases. Only the auditor can start and stop the audit function.

7.1.2.2 Audit Generation

Audit records are generated in the kernel and user-mode portions of the TOE. When a user logs on, a copy of the real UID is placed in the SAT ID field of the process. When a subject makes a system call that causes a potentially auditable event, that system call first checks to see if the event has been selected for auditing. If the event has been selected, the system call makes an event-specific function call that generates a record for the auditable event. For example, the system call chdir will call the function sat_chdir, which will generate a chdir audit record. If the event has not been selected, the system call will ignore it and continue processing. When the record is complete, the record is placed onto the audit record queue, a kernel memory buffer that holds generated records.

The audit subsystem uses a trusted process timed (see Section 7.1.12 Time Daemon) to supply a timestamp for the audit records.

7.1.2.3 Audit Management

Audit records are moved from the audit record queue into an audit file. If the auditor has specified a regular file as the audit file, the audit subsystem copies the records into that file. If the auditor has specified a directory as the destination, the audit subsystem will create files in that directory and name them with the file creation date/time in the format satyymmddhhmm. It will then fill them with records.

The auditor can direct the audit file to a standard output device using a parameter to the command line. The auditor can also specify a rotation scheme among the audit files. One option is if the last in a series of audit files is full, Trusted IRIX will go back to the first audit file to start writing and overwrite any previously recorded data. The second option

is once all the audit files are filled, Trusted IRIX will shut down. When the audit trails fill, a message is written to the syslog to alert the auditor.

Audit files are owned by the auditor and set to allow read-write access for the owner, null access for group and world. They are all labeled dbadmin. This renders them inaccessible to users.

The auditor uses the sat_select tool to control audit event selection based several criteria such as user identity, subject sensitivity label and object sensitivity label. This tool allows the auditor to designate which audit events to record, to display the current list of selected events, and to restore the default set of auditable events. To exclude a particular event, the auditor must select all other events except that event. The list of attributes that the auditor must be able to exclude is object label, subject label, and user id. Since each of these is from a finite list, the auditor can include all other elements from the list and that effectively excludes the missing event.

7.1.2.4 Audit Event Selection

All Audit files have a common format: a file header and a set of audit records, each of which contains a record header and record body. The file header comprises a version number (indicating which audit record structure is being used), the start and stop of audit file generation time, timezone information, and the host machine name where the audit file is stored.

All audit records have a common header, which contains information common to all events; the record body varies according to the specific event being recorded. The information contained in the header is as follows:

- Event type - the type of audit event
- Event outcome - the success or failure of the event
- Event sequence number - the sequence number for this type of event
- Time of Event - the date and time the event took place
- System Call name - the system call that took place
- Process ID - the identifier for the subject process that generated the event
- Parent Process ID - the identifier of the subject's parent process
- Host ID - the identifier of the host that generated the audit event.
- Current Working Directory - the subject's current working directory
- Process Label - the current label of the subject process
- SAT ID - the unique audit ID for the user
- UID - the UID of the process that caused the event
- GID - the GID of the process that caused the event
- Group list entries - the list of user IDs for the above GID

For most but not all event records, there is also a record body, which usually contains information about the object affected by the event. The following tables document the list of auditable events and provide a brief description of each event:

Auditable Event	**Description**
sat_access_denied	Access to the file or some element of the path was denied due to enforcement of MAC or DAC permissions.
sat_access_failed	Access to a file was denied because the path specified does not exist.

sat_chdir	Current working directory was changed with *chdir*.
sat_chroot	Current root directory was changed with *chroot*.
sat_open	A file was opened with write permission.
sat_open_ro	A file was opened read-only.
sat_read_symlink	The contents of a symbolic link were read with *readlink*. Note that the file the link "points" to is not accessed in any way.
sat_file_crt_del	A file was added or removed from a directory.
sat_file_crt_del2	This is the same as sat_file_crt_del, but reports that two files (perhaps a link) were removed.
sat_file_write	The data in a file was modified by *truncate*.
sat_mount	A filesystem was mounted or unmounted.
sat_file_attr_read	The attributes of a file were read by *stat*.
sat_file_attr_write	The attributes of a file were written by *chmod*.
sat_exec	A new process has been introduced by *exec*.
sat_fchdir	The user changed from the current working directory to the directory "pointed" to by the given open descriptor.
sat_fd_read	Information was read from a file descriptor using *read*.
sat_fd_read2	The same event as sat_fd_read, but with multiple file descriptors.
sat_tty_setlabel	The user set the label of a port via *ioctl*.
sat_fd_write	The user finalized a change to a file descriptor.
sat_fd_attr_write	The user changed the attributes of the file "pointed" to by the given file descriptor using *fchmod*.
sat_pipe	The user created an unnamed pipe.
sat_dup	The user duplicated a file descriptor.
sat_close	The user closed a file descriptor.
sat_proc_read	The user read from a process's address space using *ptrace*.
sat_proc_write	The user finalized a changes to a process's address space using *ptrace*.
sat_proc_attr_read	The user read a process's attributes.
sat_proc_attr_write	The user finalized a change to a process's attributes.
sat_fork	The user duplicated the current process (thereby creating a new process).
sat_exit	The user ended the current process.
sat_proc_own_attr_write	Process attributes were changed.
sat_clock_set	The system clock was set.
sat_hostname_set	The hostname was set.
sat_domainname_set	The domain name was set.
sat_hostid_set	The host ID was set.
sat_check_priv	Action requiring superuser privilege was performed.
sat_control	The *sat_select* command was used.
sat_svipc_access	The user accessed a System V IPC data structure.
sat_svipc_create	The user created a System V IPC data structure.
sat_svipc_remove	The user removed a System V IPC data structure.
sat_svipc_change	The user set some attribute of a System V IPC data structure.
sat_bsdipc_create	The user created a socket.
sat_bsdipc_create_pair	The user created a socket pair.
sat_bsdipc_address	A network address was used explicitly via the *accept*, *bind*, or *connect* system calls.
sat_bsdipc_if_config	An interface structure's attributes were changed.

Table 4 Kernel Audit Events

sat_ae_identity	A login- or logout- related event occurred.
sat_ae_dbedit	A file was modified using the *dbedit* utility. (This utility is available only with the Trusted IRIX optional product.)
sat_ae_mount	An NFS filesystem was mounted.
sat_ae_audit	Audit started/stopped
sat_ae_lp	Print job activity
sat_ae_custom	An application-defined event occurred. Application developers can engineer their applications to generate this event.

Table 5 User-mode Audit Events

7.1.2.5 Audit Reduction

The auditor has several tools available for the reduction and viewing of audit data. The auditor can invoke the following programs to manage the audit data:

- sat_reduce applies filter to audit data. Data can be filtered on various attributes including date and time, event type, object identifier, and success or failure of related event.
- sat_summarize produces a statistical summary of the contents of an audit file, such as the total number of audit records in the file by event type, by user, or by various timing parameters.
- sat_interpret converts the contents of an audit file into human-readable form.
- Covici is used to track changes to user attributes. In order for covici to track changes to user attributes, it must be used to make all changes to the user attributes.
- The SYSLOG is used to record login attempts for rlogin and ftp sessions.
- An awk script is provided to assist in filtering audit events based on object label.

7.1.3 Discretionary Access Control

Trusted IRIX enforces a discretionary access control (DAC) policy on all subjects and objects. Discretionary Access Control (DAC) is the mechanism by which an access to data is controlled based solely on the identity of the user and the resource. The implementation of DAC is accomplished by association of attributes which are specific to the type of resource. This section first identifies the DAC attributes of subjects and each object. The DAC policies are specific to the type of object and are described following the attribute identification.

7.1.3.1 Subject DAC Attributes

As stated in Section 7.1.1 above, the subject is share group (which may consist of a single process) Following are the DAC-related attributes of a subject:

- Real UID
- Saved UID
- Saved GID
- Process group ID and session ID
- Process group leader ID

- Effective UID
- Real GID
- Effective GID
- Group list
- Process umask
- Effective Capability Set
- Inherited Capability Set
- Permitted Capability Set

7.1.3.2 Object DAC Attributes

This section identifies the DAC attributes for each of the object types.

7.1.3.2.1 File System Object Attributes

Following are the DAC-related attributes of a file system object:

- Owner
- Owning Group
- Protection Mode
- Access Control List (optional)
- Default Access Control List (optional, relevant only for directories)
- Permitted Capabilities (optional, relevant only for executables)
- Effective Capabilities (optional, relevant only for executables)
- Inherited Capabilities (optional, relevant only for executables)

7.1.3.2.2 Process Object Attributes

Following are the DAC-related attributes of a process object:

- Real User ID
- Effective User ID
- Saved User ID
- Process ID
- Process Group ID
- Process Group Structure
- Session Structure

7.1.3.2.3 System V IPC Object Attributes

The System V inter-process communication facilities provide for semaphore sets, message queues, and shared memory segments. Following are the DAC-related attributes of a System V IPC object:

- Owner's User ID
- Owning Group ID
- Creator's User ID
- Creator's Group ID
- Protection Mode

7.1.3.2.4 Non-Kernel Objects

Non-kernel objects are at jobs, crontab file sand print queue entries. Following is the DAC-related attribute of a non-kernel object:

- Object Owner

7.1.3.3 Permissions Checking

Permission checking relies on several object attributes. Two of those attributes are permission bits (i.e., protection mode) and Access Control Lists (ACLs). Other object attributes identified above are used to perform permission checking. This section describes permission bits and ACLs and their access checking algorithms. Other attributes are not described here since they are simple comparisons (e.g., does the process ID match the owner ID?). The comparisons are identified in the next section.

Permission bits divide permissions into three categories and users into three relative groups. The three categories of permissions are read, write, and execute. They are denoted as "r" for read, "w" for write, and "x" for execute. The three relative groups are the owner of the file, the owner's group, and every other user.

Trusted IRIX includes an additional DAC facility, Access Control Lists (ACLs) for files and directories. The ACL is an optional component of the DAC. ACLs are used to provide more fine-grained protection than the group permissions. In the abstract, an ACL consists of a set of pairs (name, permissions), where name is a user or group name, and permissions are the list of permitted or denied access types for name. Files may have a single ACL. Directories may have two distinct ACLs. The first is the access control ACL, which is used in DAC for access to the directory itself. The second is the default ACL, which is used to initialize files created in that directory.

Permissions checking on permission bits checks for user permission first, followed by group, followed by other users. Permission is granted by the first permission set matched.

Permission checking for ACLs is also performed in the order of user, group, and other. ACLs are order dependent. The first match that is encountered on the ACL is the access granted.

Before permission checking is performed, capability checks are made to determine if the user has a capability to bypass the standard DAC checks just described. The capabilities checked are:

- CAP_DAC_READ_SEARCH for read access,
- CAP_DAC_WRITE for write access, and
- CAP_DAC_EXECUTE for execute access.

Other capabilities are checked for specific access modes and are described in the next section. In some instances, root checks are performed in lieu of capability checks. This is consistent with the capability check since root is defined to have all capabilities. The root check simplifies the case where multiple capability combinations exist.

7.1.3.4 Object DAC Policies

7.1.3.4.1 File System Objects

There are two ways to reference a file system object in Trusted IRIX, by pathname or by descriptor. In either case, if the subject has the appropriate capability, access is always granted. Otherwise, separate policies are used depending on whether the subject references the object by pathname or by descriptor. Typically, a subject makes a system call such as open that references a pathname and returns a file descriptor to the same object; reading and writing to that object is then carried out using the descriptor. Access checks are made against an ACL if one exists, or permissions bits if no ACL is present.

In order to read data from an object with a pathname, a subject must have execute/search access to each directory in an object's pathname, and read access to the file. In order to write data into an object, a subject must have execute/search access to each directory in the pathname, and write access to the file.

A subject may read a pathname object's attributes if the subject has execute access to each directory in the path. If the subject is also the owner, that subject may modify the object's attributes. A subject may execute the object if the subject has execute/search access to all portions of the pathname and execute access to the file. In addition to these policies, Trusted IRIX maintains a special policy regarding directories. Within the object's attributes is a special bit called the "sticky bit". When this bit is set in a directory's attributes, no entry in that directory can be deleted except by its owner or the directory's owner. It has no significance for other types of files.

A separate set of policies applies to file descriptors. A process may read a file to which it has the file descriptor, and may write the file if the process requested write access when obtaining the descriptor. If the process' UID owns the file, the process may change the file's attributes. Note that unnamed pipes have no pathname; hence access control is applied using the file descriptor.

When a symbolic link is encountered, its DAC attributes are taken from the file whose pathname is stored in the data portion of the symbolic link. Access checking is performed using those attributes.

7.1.3.4.2 Process Objects

Trusted IRIX has separate DAC policies for the process object when it is the current process and when it is another process.

Current Process

A process can always read and writes its own data in user space. A process can also always read its own attributes. Most process attributes either can never be written (e.g., process id) or can always be written. However, there are five process attributes governed by specific write policies: (1) real and effective UID, (2) group ID, (3) session ID, (4) group list, and (5) process label. These policies are:

1. A process with the CAP_SETUID capability can change its real and effective user identifier (UID). Otherwise, it can change its real and effective UID only if the requested value is equal to its real or saved UID.
2. A process with the CAP_SETGID capability can change its real and effective group identifier (GID). Otherwise, it must have the same effective UID as the invoker or be a member of the same session as the calling process.
3. A process can change its session ID, provided that it is not the process group leader and that no other process has a process group ID that matches the calling process' process ID.
4. Only a process with the CAP_PROC_MGT capability can change the group list.

5. Only a process with the CAP_MAC_RELABEL_SUBJ capability can change its label.

Another Process

A process can read or write another process' data if the reading process is the target process' parent. A process can read the attributes of any process that it can name (via the process ID), even if they are in different share groups. There are two process attributes that may be changed: the process group ID and sending a signal mask.

A process can change the process group ID of another process if one of the following is true:

- the changing process is the changed process' parent
- the two processes have the same effective UID or are part of the same session

A process can write the signal mask of another process if one of the following is true:

- the sending process has the appropriate capability
- the real or effective UID of the sending process matches the real or saved UID of the receiving process
- the receiving process is a child of the sending process.
- the signal is SIGCONT and the sending process is an ancestor of the receiving process
- the signal is SIGCONT and the sending and receiving processes are members of the same session.

7.1.3.4.3 System V Objects

Another set of policies applies to the System V IPC objects. The access function will approve access if the process has the CAP_DAC_OVERRIDE or CAP_FOWNER capability. If not, the access function checks to see if the user is the owner or the creator or, failing that, has the same effective GID. When the proper type of user is determined, the function checks the access bits for that user type to see if the requested access mode is valid. Access is granted accordingly. Access control on System V IPC objects is performed by all system calls that access these types of objects.

7.1.3.4.4 Non-Kernel Objects

DAC on non-kernel objects is enforced by the Batch and LP Subsystems. All at jobs and crontab files have their owner set to the user who submitted the job, and access is allowed to that owner only.

The at command allows users to list or remove at jobs they own in the batch queue. Users may not access other user's at jobs. A user with appropriate capability may access any at jobs. Users may display or remove crontab files they have submitted. Users may not access other users' crontab files. A user with the appropriate capability may access any crontab files.

All print queue entries are owned by lp. A user may use lp commands to list any entry or delete any print queue entry that the user submitted. A user with the root access may list or delete any entry.

7.1.3.4.5 DAC Initialization and Revocation

If the parent directory of the file has a default ACL, the default ACL is used as the initial ACL on the newly created file. The file system DAC permission bits are set to the value of the subject's umask at creation. The umask contains the initial permission settings and may be set as restrictively as the subject wants. In addition to permissions for owner, owning group, and world, System V IPC objects include the UIDs for the object creator and the creator's group. The creator and creating group are taken from the effective UID and GID of the process that created the object. The permissions for the creator are the same as the permissions for the owner.

DAC permissions can be changed on an object to which a subject currently has a descriptor. A change in an object's DAC permissions will not impact subjects, which already have a descriptor to this object. Changes to an object's DAC permissions become effective upon future attempts to open that object.

7.1.4 Mandatory Access Control

This section describes Mandatory Access Control (MAC) labels and policy, as well as how those labels and policies apply to objects within the Trusted IRIX.

7.1.4.1 Labels

Trusted IRIX assigns MAC labels to all subjects and objects under its control. A MAC label in Trusted IRIX contains two major components, the mandatory sensitivity (MSEN) label, and the mandatory integrity (MINT) label. These labels form the basis of the Trusted IRIX MAC policy.

The MSEN label allows Trusted IRIX to enforce MAC sensitivity controls. These controls prevent unauthorized disclosure of sensitive data. An MSEN label has three components:

- A type specifies the nature of the sensitivity label
- A hierarchical level reflects classification or clearance
- A non-hierarchical set of categories allows compartmentalization within a given clearance

The level and categories are the traditional Bell-LaPadula model hierarchical and non-hierarchical components. The type serves two functions. First, it allows Trusted IRIX to distinguish between several system defined labels and a user label. Second, it allows for possible expansion of the Trusted IRIX sensitivity label or enhancements to MAC algorithms without invalidating previously existing labels.

The possible values for the type field of an MSEN label are:

- msenhigh - a label that dominates all other MSEN labels.
- msenlow - a label dominated by all other MSEN labels.
- msenadmin - a label never dominated by any MSEN label assigned to a user
- msenequal - a label equal to all system MSEN labels
- msentcsec - an MSEN label that can be assigned to a user

Labels of the type msenhigh, msenlow, and msenequal are used for labeling system objects and resources so that they operate correctly within the MAC policy and are appropriately protected from user activity. Labels of these types make no use of any further level or compartment information, since their MAC relationships are already defined.

Labels of the type msenadmin can be assigned to administrators so that their operations can be properly protected from user interference. Labels of this type make no use of further level or compartment information, since their MAC relationships are already defined.

Labels of the type msentcsec are the only labels assigned to users of a Trusted IRIX system. They contain a level with a value between 0 and 255 and category set with any combination of up to 250 categories selected from an available set of 65,536 categories.

The MINT label has a similar structure to the MSEN label. It has the following fields:

- A type specifies the nature of the integrity label
- A hierarchical grade reflects trustworthiness
- A non-hierarchical set of divisions reflects suitability for a given purpose

These labels support an integrity mechanism based on the Biba integrity model. This model controls access to an object based on the degree to which the subject needs to be able to trust the information in the object. Within this model, a subject of high integrity is denied read access to objects of low integrity to prevent introduction of untrusted influences into trusted activities. Similarly, a subject of low integrity is denied write access to objects of high integrity to prevent corruption of high integrity data. This integrity mechanism supplements the existing access controls used by the base IRIX system to prevent corruption of trusted components.

Every MINT label has one of the following types:

- minthigh - a label that dominates all other MINT labels on the system.
- mintlow - a label dominated by all other MINT labels on the system.
- mintequal - a label that is equal to all other MINT labels on the system.
- mintbiba - a MINT label that can be assigned to users.

MINT labels of the type mintlow, minthigh, and mintequal are used for labeling system objects. Their MINT relationships with all labels are determined by their type, so they contain no further grade or division information. MINT labels of the type mintbiba can be assigned to users. They can contain a grade with a value of 0 to 255 and up to 250 divisions taken from a total available set of 65,536 divisions. Grades and divisions allow for definition of different domains of integrity within the user community. As a practical matter, the mechanism is generally used only to protect system binaries and databases and user labels containing a non-zero grade or a non-empty set of divisions are rarely if ever assigned.

7.1.4.2 System Defined MAC Labels

Trusted IRIX maintains a set of reserved MAC labels that it assigns to administrators and system objects. This set of labels is as follows:

Label Name	MSEN Attributes			MINT Attributes		
	Type	Level	Categories	Type	Grade	Divisions
wildcard	msenequal	N/A	N/A	mintequal	N/A	N/A
dblow	msenlow	N/A	N/A	minthigh	N/A	N/A

dbadmin	msenadmin	N/A	N/A	minthigh	N/A	N/A
binary	msenlow	N/A	N/A	mintbiba	highestgrade	none
userlow	msentcsec	unclassified	none	mintbiba	highestgrade	none

Table 6 Trusted IRIX System Defined Labels

The Trusted IRIX installation procedure applies the wildcard label to certain objects that must be writable by all users. To qualify for the wildcard label, an object must retain no data after a write, or retain data that is only retrievable within the context of the process that originally wrote to the object. Examples of this are the device special files, /dev/null and /dev/tty.

The Trusted IRIX installation procedure applies the dblow label to system objects that do not qualify for or do not need the wildcard label for correct use.

The Trusted IRIX installation procedure applies the dbadmin label to private system objects, like the audit trail files or the shadow password file. Administrators also operate at the dbadmin label.

The Trusted IRIX installation procedure applies the binary label to untrusted but system supplied files. The userlow label defines the lowest label at which a user might log into a Trusted IRIX system.

7.1.4.3 MAC Policy

The mandatory access control policy enforced by Trusted IRIX is based on a combination of Bell and LaPadula's sensitivity policy and Biba's integrity policy. The sensitivity policy prevents a subject from reading information that is too sensitive for the subject's clearance. The MSEN mechanism allows read and execute access only to those processes whose sensitivity labels dominate the object (meaning that the user process has equal or greater sensitivity label than the file or program). Additionally, a process may only write to an object with the same sensitivity label.

The mandatory integrity policy prevents a subject from modifying information that has higher integrity. The MINT mechanism allows read and execute access only to those processes whose integrity labels are dominated by the object (meaning that the file or program has equal or greater integrity than the user process). Additionally, a process may only write to an object with the same integrity. This is to avoid reducing the integrity of a file by a user with lower integrity.

Mandatory Integrity is similar to MSEN in design and implementation, but addresses different issues and threats. While MSEN prevents a user from accessing information that is too sensitive or secret for the user's clearance, MINT prevents a user from accessing information or programs that are of unknown or lower quality or security. For example, a user running at the highest possible clearance who has access to the most secret and important system resources should not be allowed to run every program found on the system. Such a user should be permitted to execute only programs of known good integrity.

The mandatory access control and integrity policies are based on a dominance relationship between the subject and the object. For any two MAC labels A and B, the

label A dominates the label B if and only if the sensitivity component of A dominates the sensitivity component of B and the integrity component of A is dominated by the integrity component of B. The Trusted IRIX MAC policy is more restrictive than Bell and LaPadula's sensitivity policy and Biba's integrity policy. That is, the Trusted IRIX requires a write equal policy rather than a write up policy.

7.1.4.4 Object Policies and Access Checks

The following sections describe the access checks made for the objects.

7.1.4.4.1 File System Objects

There are two ways to reference a file system object in Trusted IRIX: by pathname or by file descriptor. A file descriptor can only be obtained after passing a MAC check. The MAC check is done only when a file system object is referenced by a pathname. In Trusted IRIX there is no separate MAC policy for directories - file system object MAC policies also apply to directories. Unless otherwise specified, the file system object MAC policies (and directory MAC policies) are as follow:

- To read an object or its attribute or to execute an object, the label of a subject must dominate the label of an object.
- To write to an object or to modify an object's attributes, the label of a subject must be equal to the label of an object.

Trusted IRIX does not allow a user to change a label for a file that is already opened. The attributes associated with a file system object of type symbolic link cannot be written and the data associated with a file system object of type symbolic link cannot be written after it is created. Once the MAC label of the file is obtained, a comparison is made between the file label and the process label.

Multilevel directories permit files with duplicate names to reside at different labels. This prevents sharing of a file resource by processes at different labels while still permitting each process to successfully access a file resource at the process' label. This is implemented in Trusted IRIX with a moldy attribute. The moldy attribute is part of the MAC label. The moldy attribute is only present on directories and processes.

When a directory has a moldy property in its label, accesses to the directory by processes with non-moldy properties in their labels are automatically redirected to a sub-directory of the directory based on the label of the process accessing the directory. If that sub-directory does not exist it is created with the same label as the process that is accessing the directory. Processes that have a moldy attribute can see the true file structure (within the limits of the process' MAC restrictions) and are not automatically redirected.

Directories are created without moldy attributes even if the creating process is moldy. Directories must specifically be made moldy. Untrusted processes may have a moldy attribute in their label. Untrusted processes may also add a moldy attribute to a directory. When a moldy directory is first accessed at a given label, a sub-directory is created at that label and with the DAC attributes of the moldy directory. All subsequent DAC changes to the directories are independent of each other. Only root may remove a moldy directory that has been accessed, because the subdirectory cannot be deleted by any other user.

7.1.4.4.2 Process Objects

The process IPC MAC policies are same as the file system objects MAC policies except for the attribute write policy. A process can read another process's data and attributes if and only if the label of the subject process dominates the label of the object process. The process IPC data write policy is that the label of the subject process must be equal to the label of the object process. Most process attributes cannot be written by another process (e.g., audit ID, user ID, process label, etc). For those attributes that can be written by another process (i.e., process group ID and a signal), a process can change the process group ID of another process if and only if the label of the subject process equals the object process and the subject process possesses the CAP_PROC_MGT capability.

7.1.4.4.3 System V IPC Objects

The System V IPC facilities provide for semaphore sets, message queues, and shared memory segments.

Untrusted processes are allowed to access a semaphore set, message queue, or shared memory segment only if the label of the process dominates the label of the object. When accessing the System V IPC objects, the MAC check is performed each time the objects are accessed.

7.1.4.4.4 Internet Domain IPC

When a TCP/IP socket is created it is assigned the MAC Label of the process that created it. When an untrusted process uses the TCP/IP socket, it is constrained to communication at the label of the process that created the socket. Attempts to connect to the TCP/IP socket by a process at a different label are disallowed by the kernel. To place a TCP/IP socket in trusted processing mode, a process must have the appropriate capability. In this mode the process can change the label of the TCP/IP socket if the process has one of the following (as appropriate):

- CAP_MAC_UPGRADE.
- CAP_MAC_DOWNGRADE.

A process can obtain the label of a TCP/IP socket only if the process is at the same label as the socket or the socket is in trusted processing mode.

While data may travel on a network at multiple MAC labels, all network connections and datagrams are single level objects. A network may be seen as a multilevel entity insofar as it carries multilevel data, but access to the raw network is strictly a privileged function of the system. Raw networks are never directly accessible to user domain processes. The labeling of data on the network travels with the data on the same physical media.

7.1.4.4.5 Non-Kernel Objects

There are three non-kernel objects in Trusted IRIX.

- crontab file - The crontab command enforces the MAC security policy on cron jobs. Jobs receive the label of the user process submitting the job. When cron runs this batch job, it checks if the label of the crontab file is within the user's current clearance range. If the label is not within the user's current clearance range, the job will not run. Users can display the contents of their crontab file if the user's current label equals the crontab file label. Users can also remove a crontab file if the user's current label is equal to the crontab file label.

- at job - The at command enforces the MAC security policy on at jobs. Jobs receive the label of the user process submitting the job. When cron runs this batch job, it checks if the job is within the user's current clearance range. If the label is not within the user's current clearance range, the job will not run. Users can list their own at jobs in the batch queue if the user's current label equals the at job label. Users can also remove an at job if the user's current label is equal to the at job label.
- print queue entry - The LP subsystem commands enforce the MAC security policy on print queue entries. Entries are submitted using the lp and pr commands and entries receive the label of the process issuing the command. A MAC check is done when the lpstat command is invoked to list the queue entries or the cancel command is invoked to delete queue entries. The lpstat command displays the information about the job currently printing and print entries whose label is dominated by the user's label. The cancel command deletes only those print queue entries whose label is the same as the user's label.

7.1.4.5 Importing and Exporting Data

Data can be imported and exported using the dumb terminal and tape devices. Data is labeled at the level the user is logged on for dumb terminals. The tape devices are owned by root, labeled dblow, and have DAC permissions granting owner-only read and write permissions. The tar command is the only TSF command provided to import and export data. It is also the only command provided to associate the label of a file with the exported file. The M option directs tar to maintain the security labels of all files placed on tape. An archive made using the M option will have a tar-created phantom file containing the saved file's MAC label directly preceding the saved file in the archive. The phantom files have an identifying string in their name that identifies them as phantom files.

If an untrusted user wants to import or export data, the user must coordinate with the administrator to get permission to the tape device and to have the label set properly on the device. It is the administrator's responsibility to change the label of the tape device file each time a user at a different label wishes to use the tape device. This manual changing of the label is an auditable event. Additionally, to ensure that the same device is not used to export data with and without security attributes, the Administrator Guide instructs the administrator to not allow access to the tape device to untrusted users. To export data with or without security attributes, users are instructed to make a request to the administrator. The administrator will then perform this action for them.

The label the user is logged on at controls the label of the data input and output via a dumb terminal. When a user logs on, the user's shell program receives a file descriptor with read/write access to the terminal. The user's programs inherit this file descriptor and use it to import and export unlabeled data. Other non-administrative users are denied read and write access to the terminal's special device file because the file is labeled dblow with read access restricted to the file's owner, root. Use of the MAC label dblow and user account root is restricted to administrators.

When data is exported in a human-readable form, Trusted IRIX will print up to 300 characters of label information on the header and trailer pages. In the event the label is longer than 300 characters, Trusted IRIX will delete the print job and nothing will be printed. Trusted IRIX will print up to 60 characters on each page of output. If the label is longer than 60 characters, then the keyword TRUNCATED will be added. The label on the output is the label of the user process requesting the print job.

7.1.5 Object Reuse

Trusted IRIX maintains a policy that no allocated storage shall contain information leftover from previous use. When a file system object is created, all of its fields are filled with attribute information for the new object, overwriting the old information

The control structures that underlay a process are initialized and assigned to the process at process creation time. System V IPC objects are created empty.

Data managed within the TCP/IP networking subsystem are kept in message buffers. When a message buffer is allocated for use, it is zeroed.

7.1.6 Login Process

7.1.6.1 Local Logins

The login process is a user-mode process that handles user logins to the system. When a login attempt originates on a hardwired terminal connection. This process prompts for an initial user name and then passes the name to the login command. The login command then prompts for a password. The login command will then continue to prompt for username/password pairs until a correct pair is entered or a configurable maximum number of attempts is exceeded. At this point, a configurable time delay occurs, and the login command exits.

7.1.6.2 Network Logins

When multiple Trusted IRIX workstations are connected via an Ethernet, a user may request another workstation to perform services on the user's behalf by invoking one of the following commands: rlogin, rsh, rcp, telnet, or ftp. When a user invokes the rlogin, rsh, rcp, telnet, or ftp command, a corresponding client process is invoked on the user's workstation. This client process attempts to communicate with a corresponding server process that will perform the service on the remote workstation.

In each case, this interaction results in the server process performing a service on a remote host on behalf of the user. The client processes rlogin, rsh, and rcp pass the identity (i.e., effective UID) of their user to the corresponding server process (rlogind or rshd), which "trusts" this identity and does not perform further authentication (unless the authentication fails and the user is prompted for a different identity). The server processes ftpd and telnetd, on the other hand, always identify and authenticate remote users.

7.1.6.3 User Attribute Storage

The /etc directory maintains the user attribute information. The identification and authentication subsystem maintains a database in /etc/passwd and /etc/shadow that provides basic authentication data such as user name, user ID, and password. This information is supplemented by the following related databases:

- group membership (/etc/group)
- clearance information (/etc/clearance)
- administrative capability assignments (/etc/capability)

A user's encrypted password is stored only in /etc/shadow. This file is protected by DAC permissions that allow reading and writing only by root and the dblow MAC label that prevents non-administrators from writing the file. The user is never allowed to see the plain text form of a password, or alter the password database directly. Only an

administrator can manage user attributes. When an administrator makes a change to a user attribute using the covici tool, the change takes affect on the next login. To ensure an immediate change, the administrator must require the user to logoff.

7.1.6.4 Password Management

Users are allowed to change their own passwords using the passwd command. Passwords must be constructed to meet the following requirements:

- Each password must have at least six characters. Only the first eight characters are significant.
- Each password must contain at least two alphabetic characters and at least one numeric or special character. In this case, ``alphabetic'' means upper and lower case letters.
- Each password must differ from the user's login name and any reverse or circular shift of that login name. For comparison purposes, an upper case letter and its corresponding lower case letter are equivalent.
- New passwords must differ from the old by at least three characters. For comparison purposes, an upper case letter and its corresponding lower case letter are equivalent.

Administrators can change any user's password using the same passwd command. User changes to passwords using the passwd are constrained by password aging. Password aging means after a password change, the user may not change the password again for a site configured minimum period of time. This prevents the user from changing away from and back to an expired password too quickly. Once a password has been validated by the passwd command, it is encrypted and written into the password database. The clear text is never written into any database.

7.1.7 Capabilities

The Capabilities mechanism provides access to specific administrative functions. Only administrators and auditors are assigned capabilities. Capabilities are stored within the Process Definition and are referenced whenever a capability is required to perform a requested function.

Capabilities are inherited and granted through a set of rules. Every process has three capability vectors. Only the *effective* vector is used in access control decisions. The *permitted* vector is the capabilities that a process may request. The *inherited* vector is only used in the calculation of capability sets during exec processing.

As stated above, each program file has three capability vectors. These vectors influence the final capability set of a process, which invokes the program. When a new program is loaded the capability vectors are set to:

- Inherited-process-new = Inherited-file & Inherited-process-old
- Permitted-process-new = Permitted-file | (Permitted-process -old & Inherited-process-new)
- Effective-process-new = Effective-file & Permitted-process-new

The new inherited vector is the intersection of the process inherited vector and the program inherited vector. The new permitted vector is the union of the program permitted vector and the intersection of the new inherited vector and the process permitted vector. The new effective vector is the intersection of the new permitted vector and the program effective vector.

Figure 2 Capability Relationships demonstrates the relationship among the capability sets:

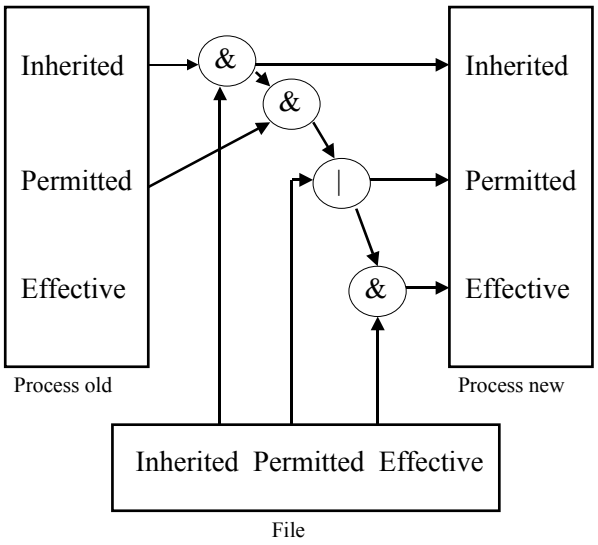

Figure 2 Capability Relationships

Note that the old effective vector does not influence any of the new vectors, and that the program inherited vector defines an upper bound on the capabilities available to the process through inheritance from the preceding process.

In the following identification of capabilities, for brevity the term *allows* used. The more complete description would be that the system call would successfully complete only if the capability is in the effective set of the process at the time of issuing the system call.

- CAP_ACCT_MGT - This capability allows the process to issue the acct system call.
- CAP_AUDIT_CONTROL - This capability shall override the restriction that a process cannot modify audit control parameters.
- CAP_AUDIT_WRITE - This capability shall override the restriction that a process cannot write data into the system audit trail.
- CAP_CHOWN - This capability allows the caller to change the owner of a file to an arbitrary UID/GID.
- CAP_CHROOT -This capability allows the process to issue the chroot call.
- CAP_DAC_EXECUTE -This capability shall override file mode execute access restrictions when accessing an object, and shall override the ACL execute access restrictions when accessing an object.
- CAP_DAC_READ_SEARCH - This capability shall override file mode read and search access restrictions when accessing an object, and shall override the ACL read and search access restrictions when accessing an object.

- CAP_DAC_WRITE - This capability shall override file mode write access restrictions when accessing an object, and shall override the ACL write access restrictions when accessing an object.
- CAP_DEVICE_MGT - This capability allows privileged operations on devices.
- CAP_FOWNER - This capability overrides the requirement that the user ID associated with a process be equal to the file owner ID, except in the cases where the CAP_FSETID capability is applicable. In general, this capability, when effective, will permit a process to perform all the functions that any file owner would have for their files.
- CAP_FSETID - This capability shall override the following restrictions: that the effective user ID of the calling process shall match the file owner when setting the set-user-ID (S_ISUID) and set-group-ID (S_ISGID) bits on that file; that the effective group ID or one of the supplementary group IDs of the calling process shall match the group ID of the file when setting the set-group-ID bit of that file; and that the set-user-ID and set-group-ID bits of the file mode shall be cleared upon successful return from chown.
- CAP_KILL - This capability shall override the restriction that the real or effective user ID of a process sending a signal must match the real or effective user ID of the receiving process.
- CAP_MAC_DOWNGRADE - This capability shall override the restriction that no process may downgrade the MAC label of a file.
- CAP_MAC_MLD - This capability prevents the automatic redirection of multilevel (moldy) directories.
- CAP_MAC_READ - This capability shall override mandatory read access restrictions when accessing objects.
- CAP_MAC_RELABEL_OPEN - This capability allows a process to alter the label of an open file.
- CAP_MAC_RELABEL_SUBJ - This capability shall override the restriction that a process may not modify its own MAC label.
- CAP_MAC_UPGRADE - This capability shall override the restriction that no process may upgrade the MAC label of a file.
- CAP_MAC_WRITE - This capability shall override mandatory write access restrictions when accessing objects.
- CAP_MEMORY_MGT - This capability overrides the restriction that a process may not manipulate the system memory management policies.
- CAP_MOUNT_MGT - This capability is required to mount and unmount filesystems.
- CAP_NETWORK_MGT - This capability is required to change the system network configuration. The functions enabled by this capability include:
  - downloading firmware to network device interfaces and starting them
  - setting the Media Access Control (MAC) address, e.g. the Ethernet address of an interface
  - retrieving device management information from network devices
  - setting, controlling and examining the FDDI SMT information
  - controlling the ARP mechanism
  - controlling the IP address(es), parameters, MAC labels, and flags of network interfaces
  - configuring the IP filter
  - controlling the TSIX interface
  - using the private interface for lockd
  - using the private interfaces for the NFS service daemons
- CAP_PRIV_PORT - This capability is required to use a privileged (less than 1024) TCP/IP port.

- CAP_PROC_MGT - This capability is required to override the restrictions on changing the attributes of other processes and to perform privileged process operations.
- CAP_QUOTA_MGT - This capability is required to modify disk quotas.
- CAP_SCHED_MGT - This capability is required to manipulate the system process scheduler.
- CAP_SETFCAP - Privilege to change the capability sets of a file.
- CAP_SETPCAP - Allows a process to change its capability sets.
- CAP_SETGID - This capability shall override the restriction in the setgid function that a process cannot change its real group ID or change its effective group ID to a value other than its real group ID. This capability also controls the setting of the process group and session group.
- CAP_SETPPRIV - This capability allows the process to set its effective set to include privileges not in its permitted set.
- CAP_SETUID - This capability shall override the restriction in the setuid() function that a process cannot change its real user ID or change its effective user ID to a value other than the current real user ID.
- CAP_SHUTDOWN - This capability is required to use the uadmin system call which can:
  - shut the system down
  - reboot the system
  - force remount of the root after automatic file system damage repair
  - notify all processes to terminate gracefully
  - power the system down (not supported on all systems)
- CAP_STREAMS_MGT - This capability is required to perform privileged STREAMS ioctls.
- CAP_SWAP_MGT - This capability is required to add or remove swap areas of the system.
- CAP_SYSINFO_MGT - This capability is required to manipulate the system identification information of the system.
- CAP_TIME_MGT - This capability is required to modify the system clock.

There is an instance where capabilities are not enforced as described in this section. Upon access to files via NFS, only permission bit checks are enforced and capabilities have no impact. The capabilities that are explicitly affected by this are: CAP_DAC_READ_SEARCH, CAP_DAC_WRITE, and CAP_DAC_EXECUTE, CAP_FOWNER.

7.1.8 Diagnostics

The hardware and firmware is tested as part of the TOE. Additionally, hardware tests are made available to the administrator to periodically validate the correct operation of the Trusted IRIX hardware and firmware. The tests are partitioned into two groups: Power ON (PON) diagnostics and the hardware instructions and memory tests. The PON tests are run at every system startup time and the other tests can be run by an administrator whenever the system is down.

7.1.9 Reference Monitor

The Trusted IRIX architecture is based on a kernel and processes. The kernel executes in the kernel mode of the processor, which is the most privileged mode. Untrusted processes run in the user mode of the processor. The mechanism for entering the kernel mode also transfers control to the kernel, which then arbitrates any requests for service and access to resources based on the security policy and the file descriptor submitted by

the user process. When untrusted processes access system resources in user-mode, they do so through well–defined server interfaces.

7.1.10 Domain Separation

7.1.10.1 Domains of Execution

The Trusted IRIX system executes instructions in two broad domains. Commands and applications execute in user-mode. In this mode, the memory management system and hardware restrictions on instruction execution isolate processes from each other, from operating system control data, and from direct hardware access. This establishes a strong separation of the instruction streams and data contained in one process from those found in another process.

A thread within a share-group may initiate an operating system request from user-mode through the system call trap mechanism. A system call trap is a software interrupt operation that causes a context switch from user-mode into kernel-mode. In kernel-mode, the thread can only execute the kernel defined code sequence that follows from a system call. This kernel code sequence, however, has unrestricted direct access to kernel control data and the hardware. The kernel-mode and user-mode distinctions are enforced by the hardware.

7.1.10.2 Process Trust Distinctions

A process in Trusted IRIX may have any of three levels of trust at any given time, depending on the code it executes. These three levels of trust define three trust domains for user-mode in a Trusted IRIX system:

- The user domain
- The administrative domain
- The system domain

The user domain contains processes without special authorization, operating on behalf of a single named user. Processes in the user domain may be simple or complex, but they cannot violate system policy restrictions, so they have no particular security function.

The administrative domain contains processes running with or without special authorization operating on behalf of administrators. Processes in the administrative domain always have the potential to assert administrative authority over policy restrictions. While these processes have a security function, that function is entirely under administrative control and, therefore, largely beyond comprehensive analysis. This limits analysis to correctness of function and potential administrative action.

The system domain contains processes executing with special authorization on behalf of potentially non-administrative users. This includes system daemons that provide services like authentication, batch processing, and so forth. It also includes applications that allow a user limited access to protected system resources, a password changing command, for example. Execution in the system domain implies policy enforcement by user-mode software. Policy enforcement requires well-defined and repeatable behavior, so analysis of system domain software includes a description of the controls placed by the software. Process isolation and memory protection features ensures that these processes have a protected execution environment

7.1.11 Roles

Trusted IRIX supports a restricted version of root, the traditional UNIX administrative user. However, root is constrained by the system MAC and DAC policies. Administrative privileges are granted through the use of capabilities. (see Section 7.1.7 Capabilities) The administrator has the ability to manage user accounts, the capability policy, default discretionary access policy, and the mandatory access control policy.

In addition to root, Trusted IRIX has defined the auditor and lp roles. The auditor owns all audit files; access permissions on the files give access only to the owner. The auditor may create, clear, and delete audit files. The auditor also selects which audit events to enable and can filter the resulting audit log. The lp role has the ability to manage printers.

7.1.12 Time Daemon

Trusted IRIX uses the time server daemon, timed, to provide a reliable time stamp. This time stamp is used in the audit trail to ensure accurate accounting of audit events.

7.2 ASSURANCE MEASURES

7.2.1 Configuration Management

The Configuration Management (CM) system applied by SGI ensures each product release is assigned a unique identifier. The CM system also identifies each hardware and software item that composes the TOE. The documentation and other programs managed by the CM system include: design documentation, test documentation, tests, user guide, administrator guide, and the configuration management plan. The CM plan is documented within the following document:

- SGI Configuration Management Plan

Assurance Requirements Satisfied: ACM_CAP.3 and ACM_SCP.1

7.2.2 Delivery and Operation

SGI has a set of Delivery and Operation documentation that describes the procedures for the delivery of the TOE. The documentation describes what is delivered with the TOE, instructions for installing and configuring the TOE, and warnings for the administrator to follow during installation. The Delivery and Operation documents are:

- IRIX/Trusted IRIX Delivery and Installation

Assurance Requirements Satisfied: ADO_DEL.1 and ADO_IGS.1

7.2.3 Development

SGI has a functional specification that describes the external interfaces of the TOE including the effects, exceptions, and error messages. There are also design documents that describe the security functions of the subsystems of the TOE. A correspondence exists that maps the high level design to the functional specification and the functional specification to the ST. An informal model describes the rules and characteristics of all

policies of the TSP and provides a mapping between the model and functional specification. The documents that meet the development assurance requirement:

- Subsystem specification for each TOE subsystem
- High Level Design Overview
- IRIX 6.5 Man Pages
- SGI Informal Security Policy Model

Assurance Requirements Satisfied: ADV_FSP.1, ADV_HLD.2, ADV_RCR.1, and ADV_SPM.1.

7.2.4 Guidance Documents

The Guidance Documents provided by SGI include both administrator and user manuals. The administrator manual describes the administrative functions and interfaces, provides guidance on how to administer the TOE securely, and contains warnings about functions and privileges that should be controlled. The user manual describes the functions and interfaces available to non-administrative users, describes the user-accessible security functions, and contains warnings to users about functions that should be controlled. The guidance documents are:

- Trusted IRIX/CMW Security Administration Guide, 007-3299-005, August, 2001
- IRIX Admin: Backup, Security, and Accounting, 007-2862-004
- Trusted IRIX/CMW Security Administration Guide Release Notes for Release 6.5.13 Common Criteria Evaluation
- Trusted IRIX/CMW Security Features User's Guide, 007-3300-003
- Trusted IRIX/CMW Security Features User's Guide Release Notes for Release 6.5.13 Common Criteria Evaluation, version 1.0

Assurance Requirements Satisfied: AGD_ADM.1 and AGD_USR.1

7.2.5 Life Cycle Support

The Life Cycle Support documentation describes how all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in it development environment are followed. The life cycle support document is

- IRIX/Trusted IRIX Life Cycle Procedures

Assurance Requirements Satisfied: ALC_DVS.1

7.2.6 Security Testing

SGI maintains a security test suite consisting of test plans, procedures, expected results, and actual results. SGI has performed and documented an analysis that the test suite adequately test the interfaces from both a coverage and depth perspective. A correspondence exists that maps the test suite to the functional specification. The test documents are:

- IRIX and Trusted IRIX/CMW CAPP and LSPP Evaluation Test Suite, Release 32 with addendums.

Assurance Requirements Satisfied: ATE_COV.2, ATE_DPT.1, ATE_FUN.1, and ATE_IND.2

7.2.7 Vulnerability Assessment

- SGI has provided guidance documents that identify all possible modes of operation of the TOE, their consequences, and implications for maintaining secure operation. The guidance documents list all assumptions about intended usage and the TOE's environment.

The Strength of Function analysis performed on the password mechanism is provided in the following SGI document:

- SGI Strength of Function Analysis

As part of its design and testing process, SGI performs a vulnerability assessment. This analysis includes a search for obvious ways in which a user can violate the TSP. For all identified vulnerabilities, SGI provides an explanation why the vulnerability cannot be exploited in the intended environment for the TOE. The following documents the vulnerability analysis:

- SGI Vulnerability Analysis

Assurance Requirements Satisfied: AVA_MSU.1, AVA_SOF.1, and AVA_VLA.1

8 PP CLAIMS

This section provides the PP conformance claims.

8.1 PP IDENTIFICATION

The TOE conforms to the Labeled Security Protection Profile, Version 1.b, October 8, 1999.

8.2 PP TAILORING

The following requirements from the Labeled Security Protection Profile were tailored in this Security Target:

- FAU_SAR.3 Selectable Audit Review
- FAU_SEL.1 Selective Audit
- FAU_STG.3 Action In Case Of Possible Audit Data Loss
- FAU_STG.4 Prevention of Audit Data Loss
- FDP_ACC.1 Discretionary Access Control
- FDP_ACF.1 Discretionary Access Control Functions
- FDP_ETC.1 Export of Unlabeled User Data
- FDP_ETC.2 Export of Labeled User Data
- FDP_IFC.1 Mandatory Access Control
- FDP_IFF.2 Mandatory Access Control Functions
- FDP_ITC.1 Import of Unlabeled User Data
- FDP_ITC.2 Import of Labeled User Data
- FIA_ATD.1 Use Attribute Definition
- FIA_UAU.1 Timing of Authentication
- FIA_UID.1 Timing of Identification
- FIA_USB.1 User-Subject Binding
- FMT_MSA.1 Management of Object Security Attributes
- FMT_MSA.3 Static Attribute Initialization
- FMT_REV.1 Revocation of User Attributes
- FMT_REV.1 Revocation of Object Attributes
- FMT_SMR.1 Security Management Roles
- FPT_AMT.1 Abstract Machine Testing

8.3 PP ADDITIONS

The following assumptions were added in this Security Target: A.LABELS and A.MGMT.

9 RATIONALE

This section provides the rationale for the selection of the IT security objectives, requirements, and security functions. It also provides rationale explaining all PP claims.

9.1 RATIONALE FOR IT SECURITY OBJECTIVES

The LSPP provides rationale for the security objectives demonstrating that security objectives are suitable to cover the intended environment. The rationale in the LSPP is valid for this ST. There are two additional assumptions in this ST. The following paragraph provides the rationale for the changes.

A.LABELS Procedures exist for the administrator to ensure that all internal representations of security levels are consistent between all machines.

This added assumption is a refinement of the PP management assumptions and do not cause the environment to be more benign.

A.MGMT The TOE must operate under a single management domain with each TSF sharing the same identification and authentication database

This added assumption is a refinement of the PP management assumptions and do not cause the environment to be more benign.

9.2 RATIONALE FOR SECURITY FUNCTIONAL REQUIREMENTS

The LSPP provides rationale for the security functional requirements demonstrating that security functional requirements are suitable to address the security objectives. The rationale in the LSPP is valid for this ST as no new security functional requirements or security objectives were added.

9.3 RATIONALE FOR SECURITY ASSURANCE REQUIREMENTS

The LSPP provides rationale for the security assurance requirements demonstrating that security assurance requirements are suitable for the intended environment. The rationale in the LSPP is valid for this ST as no new security assurance requirements or security objectives were added.

9.4 RATIONALE FOR TOE SUMMARY SPECIFICATION

This section shows that the TOE security functions and assurances are suitable to meet the TOE security requirements. This section summarizes the TOE Summary Specification; for more details about any requirement, see the corresponding section within Section 7. For the purposes of clarity, the four FMT_MTD.1 requirements have been labeled a, b, c, and d respectively. This is to distinguish them for the correspondence. Similarly, the FMT_REV.1 requirements have been labeled a and b.

The only security mechanism that is realized by a probabilistic or permutational implementation is the password mechanism identified in FIA_SOS.1. By requiring passwords consist of six characters with at least two alphabetic characters and at least one numeric or special character the claim exceeds the minimum strength of function requirement.

Functional Components	**Security Functions**											
	User Subj. Bind	Audit	DAC	MAC	OR	Login	Caps	Diagnostic	Ref Mon	Domain Sep.	Roles	Time
FAU_GEN.1		X										
FAU_GEN.2		X										
FAU_SAR.1		X										
FAU_SAR.2		X	X	X								
FAU_SAR.3		X										
FAU_SEL.1		X										
FAU_STG.1		X	X	X								
FAU_STG.3		X										
FAU_STG.4		X										
FDP_ACC.1			X									
FDP_ACF.1			X									
FDP_ETC.1				X								
FDP_ETC.2				X								
FDP_IFC.1				X								
FDP_IFF.2				X								
FDP_ITC.1				X								
FDP_ITC.2				X								
FDP_RIP.2					X							
Note 1					X							
FIA_ATD.1						X						
FIA_SOS.1						X						
FIA_UAU.1						X						
FIA_UAU.7						X						
FIA_UID.1						X						
FIA_USB.1	X											
FMT_MSA.1			X	X			X					
FMT_MSA.3			X	X			X					
FMT_MTD.1a			X	X								
FMT_MTD.1b			X	X								
FMT_MTD.1c			X	X								
FMT_MTD.1d			X	X		X						
FMT_REV.1a			X	X								
FMT_REV.1b			X	X								
FMT_SMR.1							X				X	
FPT_AMT.1								X				
FPT_RVM.1									X			
FPT_SEP.1										X		
FPT_STM.1												X

FAU_GEN.1 Trusted IRIX audit security function generates all the audit events listed in Table 2 Auditable Events on page 15. Within each audit record is the date, time, outcome of the event, and subject identity, as well as host name, subject sensitivity label, object

sensitivity level, process ID, and event type. For object access events, the identity of the object is in the audit record. For administrator events, the administrator action is audited as well as any security-relevant attribute changes.

FAU_GEN.2 The Trusted IRIX audit security function ensures each audit record has a user ID and process ID to associate each auditable event with the identity of the user that caused the event.

FAU_SAR.1 The Trusted IRIX audit security function provides the sat_interpret tool that converts the audit trail into human readable format so that auditors may read the entire contents of the audit trail.

FAU_SAR.2 The Trusted IRIX audit security function generates audit files. The DAC security function ensures that by default, audit files are owned by the auditor and set to allow read-write access for the owner, null access for group and world. The MAC security function ensures that by default audit files are labeled dbadmin so that only an auditor may access them.

FAU_SAR.3 Using the sat_reduce program provided in the audit security function, Trusted IRIX permits the auditor to perform searches of the audit data based on user identity, date, time, event type, object identity, and success/failure of the event. See Section 7.1.2.5 for a more detailed explanation of the audit reduction capabilities.

FAU_SEL.1 Using the sat_select program provided in the audit security function, Trusted IRIX permits the auditor to include or exclude auditable events from the set of audited events based on user identity, subject sensitivity label, and object sensitivity level.

FAU_STG.1 The Trusted IRIX audit security function generates audit files. The MAC and DAC security functions ensure audit files are protected from authorized deletion by the restricting the sensitivity level of the audit files and by the permissions set on the files. The default MAC label for audit files is dbadmin and only administrative users may access data at the dbadmin level. The default DAC settings for audit files are owned by the auditor and set to allow read-only access for the owner, null access for group and world. In the case where a system crash occurs before all audit data is written to disk, SGI has made engineering estimates of the number of audit records lost. satd writes data to the audit file in 8 KB blocks. The average size of an audit record is 100 bytes, giving approximately 80 bytes per write. SGI estimates that about 120 additional records will be either in the audit queue or being built throughout the system. In the event of a system crash, the 8 KB block and the other audit records will be lost. Thus, a total of about 200 audit records will be lost in the average case. The most extreme case of audit loss occurs when Trusted IRIX audits every intra-TSF packet delivery. SGI has found that the system will run out of memory to store audit records and crash within a few minutes. This case will result in the loss of approximately 32,000 audit records.

FAU_STG.3 If the audit trail has reached 90% capacity, the audit security function writes a message to the system log for indicating more audit storage is necessary.

FAU_STG.4 The audit security function provides the auditor the option of setting a flag indicating the system should shutdown if the audit trail is full. This action will prohibit all auditable events from occurring until the administrator can make additional space available for audit data. The auditor also has the option of allowing the audit trail to overwrite the oldest audit records in the audit trail if the audit log fills.

FDP_ACC.1 The DAC security function supports a DAC policy between share groups acing on behalf of users and the following objects: files, directories, named pipes, symbolic links, processes, unnamed pipes, System V IPC objects, at jobs, crontab files, and print queue entries. The DAC security functions ensure the DAC policy is enforced on all operations among subjects and objects.

FDP_ACF.1 The TSF enforces access to user objects based on user identity and group membership(s) associated with a subject, and permission bits, ACLs, object ownership, and creator associated with an object. The rules governing the access are described above in Section 7.1.4.4.

FDP_ETC.1 Unlabeled data is data that does not have security attributes associated with it after it is exported. Device special files used to communicate with the tape device are single level devices whose MAC label may be changed only by an administrator. These tape devices are initially owned by root and are not accessible to untrusted processes. To ensure that the same device is not used to export data with and without security attributes, the Administrator Guide instructs the administrator to not allow access to the tape device to untrusted users. To export data with or without security attributes, users are instructed to make a request to the administrator. The administrator will then perform this action for them. Changing the label of a device special file is an auditable event. When exporting unlabeled data to a tape device, no security attributes are associated with the data and auditing is maintained as described in Section 7.1.4.5. When exporting data via a dumb terminal, the data is exported single-level to a specific user. See Section 7.1.4.5 for a more detailed explanation.

FDP_ETC.2 Labeled data is data that has security attributes associated with it after it is exported. The MAC security function ensures that when labeled data is exported, it is controlled under the MAC policy. Tape devices are not accessible to untrusted. Access to a tape device is via device special files. When labeled data is exported to a single level device via the tar command using the M option, the label of the data and all its security attributes are preserved. Changing the label of a device special file is an auditable event and must be performed manually by the administrator. See Section 7.1.4.5 for a discussion of how labels are completely and unambiguously associated with the corresponding data.

To ensure that the same device is not used to export data with and without security attributes, the Administrator Guide instructs the administrator to not allow access to the tape device to untrusted users. To export data with or without security attributes, users are instructed to make a request to the administrator. The administrator will then perform this action for them. Changing the label of a device special file is an auditable event.

When data is exported in a human-readable form, Trusted IRIX will print up to 300 characters of label information on the header and trailer pages. In the event the label is longer than 300 characters, Trusted IRIX will delete the print job and nothing will be printed.

Trusted IRIX will print up to 60 characters on each page of output. If the label is longer than 60 characters, then the keyword TRUNCATED will be added. The label on the output is the label of the user process requesting the print job.

FDP_IFC.1 The MAC security function ensures that the MAC policy is applied on all operations between processes and the following objects: files, directories, named pipes, symbolic links, processes, unnamed pipes, System V IPC objects, BSD TCP/IP sockets, at jobs, crontab files, and print queue entries.

FDP_IFF.2 The MAC security function enforces the MAC policy between subjects and objects. Trusted IRIX assigns MAC labels to all subjects and objects under its control. A MAC label in Trusted IRIX contains two major components, the mandatory sensitivity (MSEN) label, and the mandatory integrity (MINT) label. These labels form the basis of the Trusted IRIX MAC policy.

The MSEN label allows Trusted IRIX to enforce MAC sensitivity controls. These controls prevent unauthorized disclosure of sensitive data. An MSEN label has three components:

- A type specifies the nature of the sensitivity label
- A hierarchical level reflects classification or clearance
- A non-hierarchical set of categories allows compartmentalization within a given clearance

The level and categories are the traditional Bell-LaPadula model hierarchical and non-hierarchical components.

There are 256 possible levels and 250 possible site-definable categories for a MSEN label.

The MINT label has a similar structure to the MSEN label. It has the following fields:

- A type specifies the nature of the integrity label
- A hierarchical grade reflects trustworthiness
- A non-hierarchical set of divisions reflects suitability for a given purpose

There are 256 possible levels and 250 possible site-definable divisions for a MINT label. These labels support an integrity mechanism based on the Biba integrity model. This model controls access to an object based on the degree to which the subject needs to be able to trust the information in the object. Within this model, a subject of high integrity is

denied read access to objects of low integrity to prevent introduction of untrusted influences into trusted activities. Similarly, a subject of low integrity is denied write access to objects of high integrity to prevent corruption of high integrity data. This integrity mechanism supplements the existing access controls used by the base IRIX system to prevent corruption of trusted components.

The mandatory access control policy enforced by Trusted IRIX is based on a combination of Bell and LaPadula's sensitivity policy and Biba's integrity policy. While the sensitivity policy prevents a subject from reading information that is too sensitive for the subject's clearance, the mandatory integrity policy prevents a subject from modifying information that has higher integrity. These policies are based on a dominance relationship between the subject and the object. For any two MAC labels A and B, the label A dominates the label B if and only if the sensitivity component of A dominates the sensitivity component of B and the integrity component of A is dominated by the integrity component of B. The Trusted IRIX MAC policy is more restrictive than Bell and LaPadula's sensitivity policy and Biba's integrity policy. That is, the Trusted IRIX requires a write equal policy rather than a write up policy.

For those system label components that have its own internal hierarchies the term dominate is defined as below:

- The label component of a subject dominates the label component of an object if the subject's hierarchical classification is greater than or equal to the object's hierarchical classification and the subject's non-hierarchical set is a superset of the object's set.
- The label component of a subject is equal to the label component of an object if the subject's hierarchical classification is equal to the object's classification and the subject's non-hierarchical set is equal to the object's set.
- The label component of a subject is dominated by the label component of an object if the subject's hierarchical classification is less than or equal to the object's classification and the subject's non- hierarchical set is a subset of the object's set.

FDP_ITC.1 Unlabeled data is data that does not have security attributes associated with it prior to it being imported. The MAC security function addresses importing unlabeled user data. The tape device and dumb terminals are the only means to import unlabeled data. All unlabeled data imported on the tape device, receives the label of the tape device and all data imported via dumb terminals receive the label of the device special file. The administrator controls who can use the tape device to import data. Import of data with security attributes (labeled data) is restricted to administrators only and requires that the administrator manually change the label of the tape device. Importing labeled user data requires a capability; therefore, an untrusted user cannot import labeled data. All MAC checks and the assignment of a label to a tape device are auditable.

FDP_ITC.2 Labeled data is data that has security attributes associated with it prior to it being imported. The MAC security function controls the import of labeled user data. All labeled user data is imported only by the administrator using the tape device and maintains its security attributes. All MAC checks and the assignment of a label to a tape device are auditable. See Section 7.1.4.5 for a discussion of how labels are completely and unambiguously associated with the corresponding data

FDP_RIP.2 The object reuse security function ensures that before any resource is made available to a subject, it is cleared upon allocation. File system objects are created empty. System V IPC objects and TCP/IP sockets are also cleared upon allocation. All non-kernel objects are cleared upon allocation.

Note 1 The object reuse security function is responsible for making sure that when a subject is created, it contains no residual data. To ensure this, processes take their attributes from their parent processes and all memory associated with the new subject is cleared.

FIA_ATD.1 Within the /etc file system are files that maintain the user databases. Each user has an associated user identifier, groups memberships, password, roles, and capabilities.

FIA_SOS.1 The login security function provides a password mechanism to perform authentication. For each attempt to use the password mechanism, the probability that a random attempt will succeed is less than one in 1,000,000. For multiple attempts to use the password mechanism during a one minute period, the probability that a random attempt during that minute will succeed is less than one in 100,000.

FIA_UAU.1 The login security function does not permit any TFS-mediated actions before a user is authenticated. Users must login using either the local login process or a remote login program before any TSF services are available.

FIA_UAU.7 The login security function ensures that when a user enter a password to perform authentication, only obscured feedback is provided to the user. This is true for local logins as well as network logins.

FIA_UID.1 The login security function does not permit any TFS-mediated actions before a user is identified. Users must login using either the local login process or a remote login program before any TSF services are available.

FIA_USB.1 The user-subject binding security function associates security attributes with users. Each user has the following security attributes associated with subjects acting on behalf of that user:

- Real UID
- Saved UID
- Saved GID
- Process group ID and session ID
- Process group leader ID
- Pointer to the process mandatory access control (MAC) label
- Effective UID
- Audit UID (SAT ID)

- Real GID
- Effective GID
- Group list
- Process umask
- Effective Capability Set
- Inherited Capability Set
- Permitted Capability Set

Attributes are bound to a subject at creation time. The operations that can create new subjects are:

- A user logging into workstation locally or remotely
- A process executing the fork or exec system calls as described above
- A batch job being activated

In general the subject attributes are taken from the creating subject. The exception to this is the executing of a setuid or setgid program or the use of the su program.

FMT_MSA.1 The DAC and capabilities security functions address this requirement by only allowing owners and administrators to change the DAC access rights associated with an object. The MAC and capabilities security functions ensure that only users with the appropriate capability can change the MAC label of an object.

FMT_MSA.3 The DAC security function provides for a restrictive default access to all objects. By default objects receive restrictive permissions defined in the creator's umask or default ACL. The MAC security function ensures that by default all user created objects are assigned the label of the creating processes. For both the MAC and DAC attributes, users with the appropriate capability can override the default values.

FMT_MTD.1a The DAC and MAC security functions ensure that by default, audit files are owned by the auditor, set to allow read-only access for the owner, null access for group and world, and labeled dbadmin so only the auditor may access them.

FMT_MTD.1b The MAC and DAC security functions ensure that only authorized auditors can run the saton program to modify the set of audit events and sat_select to filter the audit selection.

FMT_MTD.1c The DAC and MAC security functions ensure that only authorized administrators can initialize and modify user security attributes other than authentication data. Labels and permissions set on the /etc file system only permit administrators to change the security attribute files.

FMT_MTD.1d The MAC and DAC security function ensures that only authorized administrators may initialize authentication data. This initialization occurs when a user is added and only an administrator may update the /etc databases to add a user. Users may change their own

authentication data or an administrator may change it. This is controlled by the login security function.

FMT_REV.1a The revocation of security attributes is enforced by the MAC and DAC security function. Only administrators have access to revoke security attributes associated with a user. For the revocation to be immediate, the administrator must shutdown the TSF.

FMT_REV.1b The DAC security function ensures that owners and administrators may revoke access to an object. The revocation becomes effective the next time an access check is made against the object. The MAC security ensures that only administrators may change the label of an object and all future attempts to access the objects are checked against the new label.

FMT_SMR.1 The roles security function provides the security management roles on Trusted IRIX. Trusted IRIX supports the administrative role of root, the traditional UNIX administrative user. The root user is constrained by the system MAC and DAC policies. Administrative privileges are granted through the use of capabilities. Trusted IRIX also supports the role of auditor which can: create, clear, and delete the audit trail; modify audit events, and; read the audit trail. All users are permitted to change their own passwords.

FPT_AMT.1 The hardware and firmware is tested as part of the TOE. Additionally, hardware tests are made available to the administrator to periodically validate the correct operation of the Trusted IRIX hardware and firmware. The tests are partitioned into two groups: Power ON (PON) diagnostics and the hardware instructions and memory tests. The PON tests are run at every system startup time and the other tests can be run by an administrator whenever the system is down.

FPT_RVM.1 The reference monitor security function ensures that the TSF is always invoked before any functions are allowed to proceed. The Trusted IRIX architecture is based on a kernel and processes. The kernel executes in the kernel mode of the processor, which is the most privileged mode. When untrusted processes request services of the kernel, control is transferred to the kernel, which then arbitrates any requests for service and access to resources based on the security policy and the file descriptor submitted by the user process. When untrusted processes access system resources in user-mode, they do so through well–defined server interfaces

FTP_SEP.1 The domain separation security function enforces this requirement. The Trusted IRIX architecture is based on a kernel and process model. The kernel executes in the kernel mode of the processor, which is the most privileged mode. Untrusted processes run in the user mode of the processor. The memory separation in the kernel ensures that processes can only access their own address space or address spaces explicitly shared; this protects trusted processes from untrusted processes. Untrusted processes are managed by the kernel and have separate address spaces and process contexts.

FTP_STM.1 The time security function provides a reliable time stamp for the TSF.

9.5 RATIONALE FOR PP CLAIMS

The ST added two additional assumptions to the LSPP. Both the additional assumptions already refinements of the PP assumptions about management and do not cause the environment to be more benign. No objectives or security requirements have been added in this ST.

The following audit security functional requirements were refined with the auditor role instead of the LSPP authorized administrator role:

- FAU_SAR.1
- FAU_STG.3
- FAU_STG.4
- FMT_MTD.1

The auditor role is a subset of the authorized administrator role. It can only perform the audit-related tasks, whereas the authorized administrator can perform a variety of administrative tasks. Given this definition of auditor, the use of auditor role is consistent with LSPP.

In the FDP_IFF.2.2 security functional requirement, the MAC policy for write operations was refined to only allow write access when the sensitivity labels are equal. This refined requirement is a subset of the original requirement and is still consistent with the O.MANDATORY_ACCESS objective and the P.CLASSIFICATION Organization Security Policy.

References

[1] Bell, D. Elliot and Leonard J. LaPadula*, Secure Computer Systems: Unified Exposition and Multics Interpretation,* ESD-TR-75-306, Electronic Systems Division, Air Force Systems Command,Hanscom AFB, Bedford, Massachusetts, March 1976.

[2] Biba, K. J.*, Integrity Considerations for Secure Computer Systems,* ESD-TR-76-372, AD/A- 039-324, Electronic Systems Division, Air Force Systems Command, Hanscom AFB, Bedford, Massachusetts, April 1977.

[3] *Common Criteria for Information Technology Security Evaluation*, CCIMB-99-031, Version 2.1, August 1999.

[4] *Labeled Security Protection Profile,* Information Systems Security, NSA, Version 1.b, October 8, 1999

Acronyms

CC	Common Criteria
CM	Configuration Management
DoD	Department of Defense
EAL	Evaluation Assurance Level
GID	Group Identifier
IT	Information Technology
MAC	Mandatory Access Control
MSEN	Mandatory Sensitivity
MINT	Mandatory Integrity
PP	Protection Profile
ST	Security Target
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Functions
UID	User Identifier