Apple Mac OS X 10.6
Security Target

Version: 1.0

Last Update: December 16, 2009

atsec is a trademark of atsec information security GmbH

Apple and the Apple logo are trademarks or registered trademarks of Apple Corporation in the United States, other countries, or both.

Intel, Xeon, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

This document is provided AS IS with no express or implied warranties. Use the information in this document at your own risk.

This document may be reproduced or distributed in any form without prior permission provided the copyright notice is retained on all copies. Modified versions of this document may be freely distributed provided that they are clearly identified as such, and this copyright is included intact.

Copyright © 2008 by atsec information security corporation, and Apple or its wholly owned subsidiaries.

Table of Content

© Apple, atsec 2009

Document History

Version	Date	Changes	Author
0.1	2008-03-15	Initial Version	Stephan Müller, atsec
0.2	2008-03-17	Replacement of FMT_AMT.1 with FPT_TEE.1	Stephan Müller, atsec
0.3	2008-03-20	Considering the comments from Apple and the evaluator	Stephan Müller, atsec
0.4	2008-03-28	Update of wording Inclusion of notify(3) mechanism into SFRs	Stephan Müller, atsec
0.5	2008-03-31	Update based on QA Update based on evaluator comments	Stephan Müller, atsec
0.6	2008-04-11	Added blowfish for SSH Update based on Apple comments Update based on evaluator comments Clarification of including Mac OS X server	Stephan Müller, atsec
0.7	2008-11-06	Update description of system clock management, update of FAU_SAR.3, mention that only English is the TOE language	Stephan Müller, atsec
0.8	2009-08-04	Update FAU_GEN.1 table	Stephan Müller, atsec
0.9	2009-08-31	Addition of diffie-hellman-group-exchange-sha1; update of kernel IPC permission validation rules	Stephan Müller, atsec
0.10	2009-09-02	Removal of public key authentication	Stephan Müller, atsec
1.0	2009-12-16	Publish document	Stephan Müller, atsec

References

[CAPP]	Controlled Access Protection Profile, Issue 1.d, 8 October 1999
[CC]	Common Criteria for Information Technology Security Evaluation, CCIMB-2007-09-001 to CCIMB-2007-09-003, Version 3.1 Revision 3, July 2009, Part 1 to 3
[CEM]	Common Methodology for Information Technology Security Evaluation, CCIMB-2007-09-004, Evaluation Methodology, Version 3.1 Revision 3, July 2009
[GUIDE]	ISO/IEC PDTR 15446 Title: Information technology – Security techniques – Guide for the production of protection profiles and security targets, ISO/IEC JTC 1/SC 27 N 2449, 2000-01-04
[MACSEC]	Security Overview, 2008-02-08, http://developer.apple.com/documentation/Security/Conceptual/Security_Overview/Security_Overview.pdf
[RFC4419]	RFC 4419: Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol, http://www.ietf.org/rfc/rfc4419.txt
[SSHAUTH]	RFC 4252: The Secure Shell (SSH) Authentication Protocol, http://www.ietf.org/rfc/rfc4252.txt
[SSHTRANS]	RFC 4253: The Secure Shell (SSH) Transport Layer Protocol, http://www.ietf.org/rfc/rfc4253.txt

© Apple, atsec 2009

1 ST Introduction

This Security Target documents the security characteristics of the Apple Mac OS X version 10.6 operating system ("Mac OS X" and "Mac OS X Server"). Unless otherwise noted, the terms "Mac OS X", "Mac OS X Server" and "the TOE" are used as synonyms in this document.

The TOE, therefore, includes both, Mac OS X as well as Mac OS X Server.

The TOE does not include the hardware or firmware used to run the software components.

1.1 ST Structure

The structure of this document is as defined by [CC] Part 1 Annex A:

- Section 1 is the TOE Overview Description.
- Section 2 provides the conformance claims.
- Section 3 provides the Security Problem Definition
- Section 4 provides the security objectives.
- Section 5 provides the extended components definition.
- Section 6 provides the security requirements.
- Section 7 provides the TOE summary specifications.

1.2 Terminology

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

Administrative User	A user in one of the defined administrative roles of a Mac OS X system. The TOE defines one single type of administrative role, the root user.
Authentication data	The password for each user of the product. Authentication mechanisms using other authentication data are not supported in the evaluated configuration.
Data	Arbitrary bit sequences in computer memory or on storage media.
Information	Any data held within a Mac OS X instance, including data in transit between systems.
Named Object	In Mac OS X, those objects that are subject to access control. This includes all objects except memory objects.
Object	For Mac OS X, objects are defined by the different iterations of FDP_ACC.1.
Product	Software components that comprise the Mac OS X operating system.
Target of Evaluation (TOE)	The Mac OS X operating system, running and tested on the hardware and firmware in the evaluated configuration specified in this Security Target.
User	Any individual who has a unique user identifier and who interacts with the Mac OS X operating system. Unauthorized users do not possess a valid user identifier.
User Security Attributes	As defined by functional requirement FIA_ATD.1, every user is associated with a number of security attributes which allow the TOE to enforce its security functions on this user.

1.3 ST Reference and TOE Reference

Title: Apple Mac OS X version 10.6

Version: 1.0

Authors: Stephan Müller

Publication Date: December 16, 2009

Keywords: Apple, Mac OS X, general-purpose operating system

1.4 TOE Overview

Mac OS X is a general-purpose operating system providing multi-tasking and multi-user capabilities. After successful login, the user has access to all general-purpose computing resources allotted to the user by the configured policies. Using this environment, the user is able to execute arbitrary applications, creating and accessing files as well as IPC mechanisms. Adequate separation mechanisms are provided to separate different users and to protect their data.

Privileged administrative commands are restricted to administrators for managing the TOE.

The evaluation applies to the operation of the TOE in a non-hostile environment.

1.4.1 TOE Type

Mac OS X is a general-purpose operating system with a graphical user interface.

1.4.2 Intended Method of Use

The evaluation is intended for the operation of the TOE in a networked environment with other instances of the TOE, as well as other well-behaved network systems and operating systems administered by the same management domain. All systems connected to the network must be configured according to a defined common security policy.

The evaluation assumes that responsibility for the user data protected by the TOE can be delegated to the respective owning users. Nevertheless, all user data is access controlled by the TOE and stored in named objects for which the TOE is able to associate access rights.

1.4.3 Major Security Features

The main security functions of the TOE are:

- Audit
- User Data Protection
- Identification and Authentication
- Residual Data Protection
- Secure Communication
- Security Management
- TOE Self Protection

The architectural properties of the TOE of domain separation and reference mediation support the main security functions and ensure that these mechanisms are always invoked and cannot be bypassed.

1.5 TOE Description

1.5.1 Mac OS X TOE Definition

Apple distributes the Mac OS X product with its computer systems on DVDs or in DVD images. The distributed Mac OS X product contains many software components covering a large number of different functions.

Mac OS X is distributed as two different types:

- Mac OS X
- Mac OS X Server

The TOE covers both types of Mac OS X. The restrictions required by the evaluated configuration apply equally to both types. This results in the fact that the evaluated configuration is identical for both types, with the exception of the Mac OS X server providing additional GUI interfaces for managing network services.

The TOE is a subset of the distributed Mac OS X product; the subset consists of the software packages defined in section 1.5.4. The TOE includes the kernel, kernel extensions, system libraries and applications necessary to manage, support and configure the operating system.

Unless otherwise noted, all references to “Mac OS X” in the remainder of this document refer to the TOE and not to the product shipped by Apple.

The description of the logical and physical boundary in section 1.5.4 gives a precise definition of the scope of the evaluation.

The TOE version only covers the English language of the software as well as the guidance documentation.

1.5.2 Mac OS X Structure

Mac OS X is a UNIX-like operating system based on the Mach kernel and FreeBSD, which abstracts the complexity of UNIX and provides a user interface that fosters enhanced productivity and ease of use. Figure 1 provides an overview of the architecture of Mac OS X implemented in the TOE.

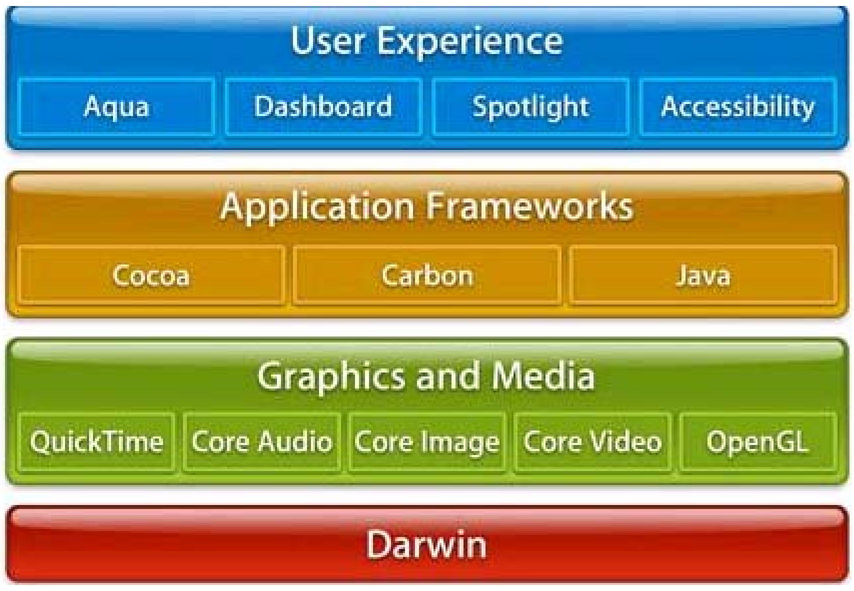

Figure 1 Mac OS X Architecture

The system components perform the following functions:

- User Experience – This layer provides the graphical interface to both users and administrators.
- Application Frameworks – This layer provides an application environment for users. An application environment consists of the frameworks, libraries, and services (along with associated APIs) necessary for the runtime execution of programs developed with those APIs. The application environments have dependencies on all underlying layers of system software. There are four application environments provided:
    1. Carbon – a set of programming interfaces derived from earlier Mac OS X APIs that have been modified to work with Mac OS X, especially its kernel environment.
    2. Cocoa – a native set of Mac OS X APIs that access Mac OS X features using an object framework.
    3. Java –development and runtime environments and an application framework that allow applications developed in Java to execute on Mac OS X.
    4. BSD Commands (not shown in the picture) – a native implementation of a command line BSD command environment.
- Graphics and Media – This layer contains the graphics and windowing environment of Mac OS X. This environment is responsible for screen rendering, printing, event handling, and low-level window and cursor management. It also holds libraries, frameworks, and background servers useful in the implementation of graphical user interfaces. In addition, this layer includes a number of Carbon managers that offer low-level services to all application environments. These services include cooperative and preemptive threading, resource management, memory management, and file-system operations.
- Darwin – The base operating system environment including the kernel is the lowest layer of system software. This environment provides essential operating-system functionality to the layers above it, such as:
    - Kernel: Pre-emptive multitasking

- Kernel: Real-time support guaranteeing low-latency access to processor resources for time-sensitive media applications
- Kernel: Virtual memory with memory protection and dynamic memory allocation
- Kernel: Symmetric multiprocessing
- Kernel: Multi-user access
- Kernel: File systems based on VFS (Virtual File System)
- Kernel: Device drivers
- Kernel: Networking
- Kernel: Basic threading packages
- Rosetta, the translator of PowerPC processor instructions to Intel x86 instructions to allow the execution of native PowerPC applications on Intel x86 hardware
- Shell and BSD/POSIX utilities

1.5.3 Darwin

The UNIX-like core of Mac OS X is called Darwin. Darwin integrates a number of technologies, including the Mach 3.0 kernel, kernel services based on BSD UNIX (Berkeley Software Distribution), high-performance networking facilities, and support for multiple integrated file systems. Further, Darwin's modular design allows developers to dynamically load extensions, such as device drivers, networking extensions, and new file systems.

Darwin provides an advanced memory protection and management system. Darwin ensures reliability by protecting applications with a robust architecture that allocates a unique address space for each application or process. The Mach portion of the kernel manages different virtual memory segments to separate simultaneous application environments.

Device drivers are created using an object-oriented programming framework called I/O Kit. Drivers created with I/O Kit easily acquire true plug-and-play, dynamic device management ("hot plugging"), and power management. I/O kit also provides hardware access to high-level application software. For network protocol developers, Darwin provides the Network Kernel Extension (NKE) facility. This allows developers to create networking modules and even entire protocol stacks that can be dynamically loaded and unloaded. NKEs also make it possible to automatically configure protocol stacks and easily monitor and modify network traffic. At the data-link and network layers, NKEs can also receive notifications of asynchronous events from device drivers.

1.5.4 Definition of the TOE Boundaries

The TOE includes both physical and logical boundaries. This section describes both the security functions provided by the TOE and the physical realization of the TOE.

1.5.4.1 Logical boundary

1.5.4.1.1 Functional Description

Mac OS X provides a rich set of security functions. Each security function is identified and described below:

- Audit – Mac OS X has the ability to audit user actions and store the records in an audit trail that is protected from unauthorized access. The administrator has the ability to select which events get audited and to sort and search the audit log after the records have been collected. The audit facility is flexible in permitting the administrator to decide if the system should overwrite old records or halt if the audit trail becomes full (Mac OS technically also permits to keep the system running once the audit trail is full, but any time a process wants to create an audit entry, this process is stopped – this state is non-recoverable).
- User Data Protection – Mac OS X provides a discretionary access control mechanism to protect user objects such as files, directories, and message queues. Access to these objects is mediated by the operating system and granted only if a set of rules is passed. In addition to controlling access, Mac OS X ensures that whenever a user is allocated a resource, that resource is clear of any previous information that it may have contained.
- Identification and Authentication – All users on Mac OS X are identified and authenticated before they can access any system service. Mac OS X maintains a user database with user name, group associations, and authentication information. Mac OS X supports password authentication in the evaluated configuration.

Based on the authentication information, a set of internal user identifiers are associated with the process spawned for the user by the login mechanism. These identifiers may change during the execution of the process based on the policy defined for the TOE.

- Residual Data Protection – Mac OS X ensures that all previously allocated memory is cleared before is it allocated to a user process. File system objects are created with all fields initialized at creation time, overwriting the existing information. Additionally, an end of file marker prevents users from accessing data beyond the current file boundary. Other objects that use memory are cleared upon allocation. This includes process addresses space and execution context, as well as IPC memory spaces.
- Secure Communication – Mac OS X can be accessed using the SSHv2 protocol, which provides cryptographically-protected network access. Confidentiality and integrity protection are provided by the SSHv2 protocol, which ensures that data is not modified or disclosed.
- Security Management – Mac OS X provides a rich set of administrative functions. Graphical tools are provided to manage user accounts, object access rights, and the audit trail.
- TOE Self Protection – Mac OS X has several features to protect the security functions. Mac OS X utilizes the security features of the hardware, including running the kernel in the most privileged state of the hardware. Memory protection and process isolation are provided to keep processes from interfering with each other and, more importantly, from interfering with the operating system. A set of diagnostic tools that can be run to ensure the correct operation of the hardware is also provided to the administrator.

1.5.4.1.2 TOE Software

The TOE is defined as the software installed from the DVDs or the DVD images for the distributed Mac OS X product following the evaluated configuration guide. The installed software, therefore, includes the software implementing all portions of the security functionality, as well as auxiliary software enabling usability of the TOE that does not interfere with the security functionality. It also installs software which is deactivated in the evaluated configuration and therefore considered to be unavailable.

Additional software may be installed by either the administrator or the user with the following restrictions:

- Applications owned by root and with the SUID bit set must not be installed.
- Applications owned by any group with group ID below 100 and the SGID bit set must not be installed.
- Applications executed with root privileges and providing services to untrusted users (such as additional services) must not be installed.
- Kernel extension must not be installed or even loaded.

1.5.4.2 Physical boundary

The evaluated hardware is one or more Apple laptops, Apple desktops, and Apple servers connected via an Ethernet with one or more Intel Pentium and Xeon-based processors with the EM64T extension running the TOE. The following devices may be attached:

- Display monitor
- Keyboard
- Mouse
- DVD drive
- Fixed disk drives
- Printer
- Audio adapter
- Network adapter

1.5.4.3 Configurations

The evaluated configuration is defined with the following constraints:

- The set of software packages forming the TOE must be installed during installation time in accordance with the installation instructions provided in the Common Criteria guidance document.

- Only local user databases for password based authentication are allowed to be used by the DirectoryService mechanism.
- The automated password quality checking mechanism is configured to meet the password quality constraints set forth in FIA_SOS.1.
- The root account is disabled for interactive login. Administrators use the 'sudo' application to gain root privileges.
- The initial configuration outlined in the Common Criteria guidance document representing the evaluated configuration must be achieved before users are allowed to interact with the TOE.
- Mac OS X supports different networking protocols, including IPv4 and IPv6. Only IPv4 is supported in the evaluated configuration.
- Mac OS X supports a wide range of protocols and network services. In the evaluated configuration, the TCP/IP protocol, the NFS client, and SSH services are supported. NFS is allowed to be used in the evaluated configuration, but is not covered by security functional claims in this ST.
- Darwin offers support for multiple file systems. In the evaluated configuration, the HFS+ file system is supported.
- For SSH, the allowed cryptographic mechanisms must be in line with the specification in FCS_COP.1. Specifically, the "none" cipher (no encryption), and the "none" keyed hash function (no HMAC) are not allowed in the evaluated configuration.
- Apple provides several Mac OS X software applications that are considered outside the scope of the defined TOE and thus not part of the evaluated configuration. These services may be executed under a non-root user ID without invalidating the evaluated configuration, but the evaluation does not make any security claims about these services. Services outside this evaluation include:
    - E-mail services
    - Web server services
    - Remote apple events
    - Print sharing services
    - File sharing services
    - Unencrypted base services, such as FTP or the r-utilities
    - Classic programming support (Old Macintosh OS compatibility support)
    - Mac OS X contains a watchdog timer to restart services and provide stability; however, this timer is disabled in the evaluated configuration.

1.5.4.4 TOE Environment

In a network of TOE systems, each TOE implements its security policy independent from the other systems. No synchronization function for these policies is provided. Therefore, a single user may have accounts on the individual systems with different user IDs, different roles, different security attributes and different user data. A synchronization mechanism may be employed by the administrator; however, such a mechanism is not part of the TOE and conflicts with the TOE requirements.

All systems connected to the network of the TOE must be managed by the same administrative authority subject to appropriate security policies that do not conflict with those specified in this ST. The TOE does not include any physical network components between network adapters of a connection. The ST assumes that any network connections, equipment, and cables are appropriately protected in the TOE security environment.

2 Conformance Claims

2.1 Common Criteria

The ST is [CC] Part 2 extended and Part 3 conformant.

2.2 Packages

The ST claims an Evaluation Assurance Level of EAL3 augmented by ALC_FLR.3.

2.3 Protection Profiles

This Security Target claims demonstrable conformance with the Controlled Access Protection Profile [CAPP]. This protection profile is listed on the CCEVS site of NIAP as a certified protection profile.

2.4 CAPP: Demonstration of Conformance

Due to the demonstrable conformance claimed by the ST, this section contains the rationale for the changes compared to the PP and demonstrates that the ST still meets all security requirements, objectives and TOE environmental constraints defined in the PP.

2.4.1 Changes to the SPD

The following aspects have been added to those defined for the Security Problem Definition in [CAPP]:

- Threats for the TOE have been added to allow more clear identification of the protection of resources and data maintained by the TOE.
- Assumptions have been clarified from [CAPP] explicitly explaining the requirements set forth by CAPP, because DAC requires the involvement of users to protect the user's data.

2.4.2 Changes to the Objectives

The following security objectives have been added to those defined in [CAPP]:

- O.COMPROT: This objective defines the ability of the TOE to connect with other trusted IT products via trusted channels. The addition of this objective defines additional security functionality without changing or limiting any other SFR in CAPP.
- The following security objectives for the TOE environment have been added:
  - OE.ADMIN
  - OE.INFO_PROTECT
  - OE.RECOVER
  - OE.SOFTWARE_IN
  - OE.PROTECT

  All objectives are required to appropriately cover the specific assumptions for the TOE environment. All objectives are related to physical and procedural security measures and therefore target the non-IT environment of the TOE. The objectives have been introduced to refine the claims specified by CAPP.

2.4.3 Changes to the SFRs

The list of SFRs defined by [CAPP] has been extended by the following SFRs:

- FMT_SMF.1 has been added due to updates to CC part 2 which were not applicable at the time of writing of CAPP. As the SFRs of FMT_MSA.1 and FMT_MTD.1 specified in CAPP depend on FMT_SMF.1 after the updates to CC part 2, FMT_SMF.1 was included.
- FCS_CKM.1, FCS_CKM.2, FCS_COP.1, FDP_UCT.1, FDP_UIT.1, and FTP_ITC.1 define the cryptographically protected communication channel of SSHv2.

- For FIA_UAU.1 and FIA_UID.1, the ST author chose to claim the hierarchically-higher SFRs of FIA_UAU.2 and FIA_UID.2, requiring that no interaction between a user and the TOE is allowed prior to authentication.
- FDP_ACC.1(2), FDP_ACF.1(2), FMT_MSA.1(2), FMT_MSA.3(2) have been added to cover the access control policy implemented for Mach ports heavily used in the TOE. This mechanism is independent from other functional mechanisms. The high-speed IPC mechanism of Mach ports allows the arbitrary transmission of data between different applications. This communication channel provides the lowest level of communication where the Mach port mechanism has no information about the contents of the message. As auditing these message transmissions would not give any useful information to the administrator, no audit logs are recorded. If audit shall be implemented, the applications utilizing the Mach port mechanism have to create audit records.
- FPT_AMT.1 has been replaced with FPT_TEE.1, as [CC] removed the former SFR and added the latter one. The ST author performed the operations on FPT_TEE.1 such that it most closely matches the old FPT_AMT.1.
- FPT_SEP.1 and FPT_RVM.1 have been replaced by ADV_ARC.1, based on the transitional changes from CC version 2.3 to 3.1.

2.4.4 Changes to the SARs

The definition of SARs specified in [CAPP] has been changed as follows:

- ALC_FLR.3 has been added, which has no dependency on any SAR specified in [CAPP]. Therefore, the addition of this SAR has no impact on the security assurance requirements stated in [CAPP].

3 Security Problem Definition

3.1 Introduction

The TOE Security Problem Definition consists of the threats to security, organizational security policies, and usage assumptions as they relate to Mac OS X. 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 [CAPP].

3.2 Threats

The assumed security threats are listed below.

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

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

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

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

The threat agents are assumed to originate from a well-managed user community in a non-hostile working environment, and hence the product protects against threats of security vulnerabilities that might be exploited in the intended environment for the TOE with medium level of expertise and effort. The TOE protects against straightforward or intentional breach of TOE security by attackers possessing a basic attack potential.

3.2.1 Threats countered by the TOE

T.UAUSER	An attacker (possibly, but not necessarily, an unauthorized user of the TOE) may impersonate an authorized user of the TOE. This includes the threat of an authorized user that tries to impersonate another authorized user without knowing the authentication information.
T.ACCESS	An authorized user may gain access to resources or perform operations for which no access rights have been granted.
T.COMPROT	An attacker (possibly, but not necessarily, an unauthorized user of the TOE) may intercept a communication link between the TOE and another trusted IT product to read or modify information transferred between the TOE and the other trusted IT product (which may be another instantiation of the TOE) using defined protocols (SSH) in a way that cannot be detected by the TOE or the other trusted IT product.

3.3 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.

3.4 Assumptions

This section contains assumptions regarding the security environment and the intended usage of the TOE. 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.

3.4.1 Physical Aspects

The TOE is intended for application in user areas that have physical control and monitoring. It is assumed that the following physical conditions will exist:

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

3.4.2 Personnel Aspects

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.
A.UTRAIN	Users are trained to use the security functionality provided by the system appropriately.
A.UTRUST	Users are trusted to accomplish some task or group of tasks within a secure IT environment by exercising complete control over their data.

3.4.3 Connectivity Aspects

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. CAPP-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.CONNECT	All connections to peripheral devices and all network connections not using the secured protocols SSH v2.0 reside within the controlled access facilities. CAPP-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 defines the security objectives of Mac OS X and its supporting environment. Security objectives reflect the stated intent to counter identified threats and/or comply with any organizational security policies identified. All of the identified threats and organizational policies are addressed under one of the categories below. All of the objectives were taken directly from the CAPP without modifications.

4.1 Security Objectives for the TOE

O.AUTHORIZATION	The TOE must ensure that only identified and authorized users gain access to the TOE and its resources.
O.DISCRETIONARY_ACCESS	The TSF must control access to resources based on the identity of users. The TSF must allow authorized users to specify which resources may be accessed by which users.
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.
O.COMPROT	The TSF must be designed and implemented in a manner that allows for establishing a trusted channel between the TOE and another trusted IT product that protects the user data transferred over this channel from disclosure and undetected modification.

4.2 Security Objectives for the TOE Environment

All security objectives listed in this section are targeted at the non-IT environment of the TOE.

OE.ADMIN	Those responsible for the administration of the TOE are competent and trustworthy individuals, capable of managing the TOE and the security of the information it contains.
OE.CREDEN	Those responsible for the TOE must ensure that all access credentials, such as passwords or other authentication information, are protected by the users in a manner which maintains IT security objectives.
OE.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.
OE.PHYSICAL	Those responsible for the TOE must ensure that those parts of the TOE critical to security policy are protected from physical attack, which might compromise IT security objectives.
OE.INFO_PROTECT	Those responsible for the TOE must establish and implement procedures to ensure that information is protected in an appropriate manner. In particular: • DAC protections on security-critical files (such as configuration files and authentication databases) must always be set up correctly. • Network and peripheral cabling must be approved for the transmittal of the most sensitive data held by the system. Such physical links are assumed to be adequately protected against threats to the confidentiality and integrity of the data transmitted unless one of the secure protocols provided by the TOE is used for the communication with another trusted entity. This requires that users are trained to perform those tasks properly and trustworthy to not deliberately misuse their access to information and pass it on to somebody that does not have the right to access the information.

OE.RECOVER	Those responsible for the TOE must ensure that procedures and/or mechanisms are provided to assure that after system failure or other discontinuity, recovery without a protection (i.e., security) compromise is obtained.
OE.SOFTWARE_IN	Those responsible for the TOE must ensure that the system is configured so that only an administrative user can introduce new trusted software into the system.
OE.PROTECT	Those responsible for the TOE must ensure that procedures and/or mechanisms exist to ensure that data transferred between servers is secured from disclosure and tampering when using communication links that are not protected by the use of the SSH v2.0 protocol. (Note that interruption of communication is not prevented by the use of those protocols. If protection against interruption of communication is required, adequate protection in the TOE environment must be established for all communication links.)

4.3 Security Objective Rationale

The following tables provide a mapping of security objectives to the Security Problem Definition, illustrating that each security objective covers at least one threat, assumption or policy and that each threat, assumption or policy is covered by at least one security objective.

4.3.1 Security Objectives Coverage

Table 1 Mapping objectives to threats, assumptions and policies

Objective	Threat / Policy
O.AUTHORIZATION	T.UAUSER, P.AUTHORIZED_USERS
O.DISCRETIONARY_ACCESS	T.ACCESS, P.NEED_TO_KNOW
O.AUDITING	P.ACCOUNTABILITY
O.RESIDUAL_INFORMATION	T.ACCESS, P.NEED_TO_KNOW
O.MANAGE	P.AUTHORIZED_USERS, P.NEED_TO_KNOW, T.UAUSER, P.ACCOUNTABILITY
O.ENFORCEMENT	P.AUTHORIZED_USERS, P.NEED_TO_KNOW, P.ACCOUNTABILITY
O.COMPROT	T.COMPROT, P.NEED_TO_KNOW

Table 2 Mapping objectives for the environment to threats, assumptions and policies

Env. Objective	Threat / Assumption / Policy
OE.ADMIN	A.MANAGE, A.NO_EVIL_ADM
OE.CREDEN	A.COOP
OE.INSTALL	A.MANAGE, A.NO_EVIL_ADM, A.PEER
OE.PHYSICAL	A.LOCATE, A.PROTECT, A.CONNECT
OE.INFO_PROTECT	A.PROTECT, A.UTRAIN, A.UTRUST, A.DISCRETIONARY_ACCESS
OE.RECOVER	A.MANAGE
OE.SOFTWARE_IN	P.NEED_TO_KNOW
OE.PROTECT	A.CONNECT

4.3.2 Security Objectives Sufficiency

T.UAUSER: The threat of impersonation of an authorized user by an attacker is sufficiently diminished by O.AUTHORIZATION requiring proper authorization of users gaining access to the TOE. O.MANAGE ensures that only administrative users (which are assumed to be trustworthy) have the ability to add new users or modify the attributes of users. Together, those objectives ensure that no unauthorized user can impersonate an authorized user.

T.ACCESS: The threat of an authorized user of the TOE accessing information resources without permission from the user responsible for the resource is removed by O.DISCRETIONARY_ACCESS requiring access control for resources and the ability for authorized users to specify the access to their resources. This ensures that a user can access a resource only if the requested type of access has been granted by the user responsible for the management of access rights to the resource. In addition, O.RESIDUAL_INFORMATION ensures that an authorized user cannot gain access to the information contained in a resource after the resource has been released to the system for reuse.

T.COMPROT: The threat of user data being compromised or modified without being detected is removed by O.COMPROT requiring the ability to set up an Inter-TSF trusted channel between the TOE and another trusted IT product that protects user data being transferred over this channel from disclosure and undetected modification.

A.PROTECT: The assumption on physical protection of all hard- and software as well as the network and peripheral cabling is covered by the objectives OE.INFO_PROTECT demanding the approval of network and peripheral cabling and OE.PHYSICAL requiring physical protection. Note: Physical protection of the network components and cabling is required by A.PROTECT, which may seem to be redundant to A.CONNECT. But A.CONNECT also addresses protection against passive wiretapping, which may be done without having physical access to a hardware component.

A.LOCATE: The assumption on physical protection of the processing resources of the TOE is covered by OE.PHYSICAL requiring physical protection.

A.MANAGE: The assumption on competent administrators is covered by OE.ADMIN requiring competent and trustworthy administrators and OE.INSTALL requiring procedures for secure distribution, installation and configuration of systems as well as OE.RECOVER requiring the administrator to perform all the required actions to bring the TOE into a secure state after a system failure or discontinuity.

A.NO_EVIL_ADM: The assumption on administrators that are neither careless nor willfully negligent or hostile is covered by OE.ADMIN requiring competent and trustworthy administrators and OE.INSTALL requiring procedures for secure distribution, installation and configuration of systems.

A.COOP: The assumption on authorized users to act in a cooperating manner is covered by the objective OE.CREDEN requiring the safe storage and non-disclosure of authentication credentials.

A.UTRAIN: The assumption on trained users is covered by OE.INFO_PROTECT which requires that users are trained to protect the data belonging to them.

A.UTRUST: The assumption on users to be trusted to protect data is covered by OE.INFO_PROTECT which requires that users are trusted to use the protection mechanisms of the TOE adequately to protect their data.

A.PEER: The assumption on the same management control and security policy constraints for systems with which the TOE communicates is covered by OE.INSTALL requiring procedures for secure distribution, installation and configuration of the networked system.

A.CONNECT: The assumption on controlled access to peripheral devices and protected internal communication paths is covered by OE.PROTECT for the protection of data transferred between systems and OE.PHYSICAL requiring physical protection.

P.AUTHORIZED_USERS: The policy demanding that users have to be authorized for access to the system is implemented by O.AUTHORIZATION and supported by O.MANAGE allowing the management of this function and O.ENFORCEMENT ensuring the correct invocation of the functions.

P.NEED_TO_KNOW: The policy to restrict access to and modification of information to authorized users which have a “need to know” for that information is implemented by O.DISCRETIONARY_ACCESS demanding an appropriate access control function that allows to define access rights down to the granularity of an individual user and O.COMPROT protecting user data during transmission to another trusted IT product. It is supported by O.RESIDUAL_INFORMATION ensuring that resources do not release such information during reuse and by OE.SOFTWARE_IN preventing users other than administrative users from installing new software that might affect the access control functionality. O.MANAGE allows administrative and normal users (for the files they own) to manage these functions. O. ENFORCEMENT ensures that the functions are invoked and operate correctly.

P.ACCOUNTABILITY: The policy to provide a means to hold users accountable for their activities is implemented by O.AUDITING providing the TOE with such functionality. O.MANAGE allows the management of this function and O.ENFORCEMENT ensuring the correct invocation of the function.

5 Extended Components Definition

The component Note 1 was included as stated in [CAPP]; it is defined in [CAPP] and therefore not defined in this chapter.

6 Security Requirements

6.1 TOE Security Functional Requirements

This section specifies the security functional requirements (SFRs) for the TOE. Functional requirements components in this ST were drawn from [CAPP] as well as from CC Part 2. The CAPP uses Part 2 of the CC. Some functional requirements are extensions to those found in the CC. This section organizes the SFRs by CC class.

All operations except iterations which are performed in this ST are marked in **bold** within each of the requirements. Other operations, already performed in the PP the ST claims compliance with, are not marked. Iterations are marked by an extension of the SFR definition.

6.1.1 Security Audit (FAU)

6.1.1.1 Audit Data Generation (FAU_GEN.1)

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the auditable events listed in column “Event” of Table 3 Auditable Events. This includes all auditable events for the basic level of audit, except FIA_UID.2’s user identity during failures.

Component	Event	Details – audit events (see audit_class(5), audit_event(5))
FAU_GEN.1	Start-up and shutdown of audit functions	ad
FAU_GEN.2	None	
FAU_SAR.1	Reading of information from the audit records.	fa, fc, fd, fm, fr, fw for the audit trail file
FAU_SAR.2	Unsuccessful attempts to read information from the audit records.	fa, fc, fd, fm, fr, fw for the audit trail file
FAU_SAR.3	None	
FAU_SEL.1	All modifications to the audit configuration that occur while the audit collection functions are operating.	fa, fc, fd, fm, fr, fw for the audit configuration
FAU_STG.1	None	
FAU_STG.3	Actions taken due to exceeding of a threshold	audit_warn(5)
FAU_STG.4	Actions taken due to the audit storage failure.	audit_warn(5)
FCS_CKM.1(1)	None	
FCS_CKM.1(2)	None	
FCS_CKM.2(1)	None	
FCS_CKM.2(2)	None	
FCS_COP.1	None	
FDP_ACC.1(1)	None	
FDP_ACC.1(2)	None	
FDP_ACF.1(1)	All requests to perform an operation on an object covered by the SFP.	cl, ex, fa, fc, fr, fd, fm, fr, fw, io, ipc, pc
FDP_ACF.1(2)	None	
FDP_RIP.2	None	
Note 1	None	
FDP_UCT.1	None	
FDP_UIT.1	None	
FIA_ATD.1	None	
FIA_SOS.1	Rejection or acceptance by the TSF of any tested secret.	ad, lo

Component	Event	Details – audit events (see audit_class(5), audit_event(5))
FIA_UAU.2	All use of the authentication mechanism.	lo
FIA_UAU.7	None	
FIA_UID.2	All use of the user identification mechanism, including the identity provided during successful attempts.[1]	lo
FIA_USB.1	Success and failure of binding user security attributes to a subject (e.g. success and failure to create a subject).	lo
FMT_MSA.1(1)	All modifications of the values of security attributes.	ad
FMT_MSA.1(2)	None	
FMT_MSA.3(1)	Modifications of the default setting of permissive or restrictive rules. All modifications of the initial value of security attributes.	ad
FMT_MSA.3(2)	None	
FMT_MTD.1(1)	All modifications to the values of TSF data.	ad
FMT_MTD.1(2)	All modifications to the values of TSF data.	ad
FMT_MTD.1(3)	All modifications to the values of TSF data.	ad
FMT_MTD.1(4)	All modifications to the values of TSF data.	ad
FMT_MTD.1(5)	All modifications to the values of TSF data.	ad
FMT_REV.1(1)	All attempts to revoke security attributes.	ad
FMT_REV.1(2)	All modifications to the values of TSF data.	ad
FMT_SMF.1	None	
FMT_SMR.1	Modifications to the group of users that are part of a role.	ad
FMT_SMR.1	Every use of the rights of a role. (Additional / Detailed).	ad
FPT_TEE.1	Execution of the tests of the underlying machine and the results of the test.	ex
FPT_STM.1	Changes to the time.	ad
FTP_ITC.1	None	

Table 3 Auditable Events

Application Note: The SFRs of FPT_RVM.1 and FPT_SEP.1 have been removed from the table, as they have been eliminated with CC version 3.1. Also, FPT_AMT.1 has been replaced with FPT_TEE.1 due to the change of requirements in the current CC.

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

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

- [1] This meets the interpretation I-410 as it only requires auditing of the subject identity on successful login attempts.

b) The additional information specified in the “Details” column of Table 3 Auditable Events.[2]

6.1.1.2 User Identity Association (FAU_GEN.2)

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

6.1.1.3 Audit Review (FAU_SAR.1)

FAU_SAR.1.1	The TSF shall provide authorized administrators with the capability to read all audit information from the audit records.
FAU_SAR.1.2	The TSF shall provide the audit records in a manner suitable for the user to interpret the information.

6.1.1.4 Restricted Audit Review (FAU_SAR.2)

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.
Application Note:	The discretionary access control mechanism supported by proper permission settings ensure that only authorized administrators are able to access the audit records.

6.1.1.5 Selectable Audit Review (FAU_SAR.3)

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

a) **Audit** user identity;

b) Effective user ID;

c) Real user ID;

d) Effective group ID;

e) Real group ID;

f) Date and time of audit event;

g) Event type (based on name or number);

h) Pathname;

i) Message queue ID;

j) Semaphore ID;

k) Shared memory ID.

Application Note: The selection of the reviewed audit information is performed with the auditreduce tool.

6.1.1.6 Selective Audit (FAU_SEL.1)

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) Success and failure;

c) Event type.

Application Note: The selection of the generated audit information is performed with the audit_control file specifying which audit events defined in the audit_class file together with the audit_event and audit_user files are to be audited.

[2]The FAU_GEN.1 requirement derived from the CAPP is a valid refinement of I-0407 as all elements are addressed as defined in the interpretation.

6.1.1.7 Guarantees of Audit Data Availability (FAU_STG.1)

FAU_STG.1.1	The TSF shall protect the stored audit records from unauthorized deletion.
FAU_STG.1.2	The TSF shall be able to prevent modifications to the audit records.
Application Note:	Protection of the audit records is achieved using DAC and appropriate permissions on the audit log files.

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

FAU_STG.3.1	The TSF shall generate an alarm to the authorized administrator if the audit trail exceeds **the configured minimum level of free disk space for the file system the audit log file resides in as defined in the audit_control file**.
Application Note:	When the free disk space falls below the percentage level specified with the minfree configuration option, a warning is generated. This warning causes the audit_warn file, which is a shell script, to be executed.

6.1.1.9 Prevention of Audit Data Loss (FAU_STG.4)

FAU_STG.4.1 The TSF shall be able to prevent auditable events, except those taken by the authorized administrator, and **one of the following audit system configurations:**

- **continuation of regular operation and overwriting the old audit records,**
- **suspension of processes when requesting an auditable action (which is currently a non-recoverable state),**
- **or halting the system**

if the audit trail is full.

6.1.2 Cryptographic Support (FCS)

6.1.2.1 Cryptographic Key Generation for SSH symmetric keys (FCS_CKM.1)(1)

FCS_CKM.1.1 The TSF shall generate **symmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm **defined by the SSH version 2.0 standard with [SSHTRANS]** and **the following** specified cryptographic key sizes

a) 128 bit,

b) 168 bit,

c) 192 bit, and

d) 256 bit

that meet the following: **generation of session keys as defined in the SSH version 2.0 standard**.

6.1.2.2 Cryptographic Key Generation for SSH DSS keys (FCS_CKM.1)(2)

FCS_CKM.1.1 The TSF shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm **defined in FIPS 186-2** and specified cryptographic key sizes **1024 bit** that meet the following: **generation of asymmetric public/private key pairs as defined in FIPS 186-2**.

6.1.2.3 DH Cryptographic Key Distribution for SSH (FCS_CKM.2) (1)

FCS_CKM.2.1 The TSF shall distribute **symmetric** cryptographic keys in accordance with a specified cryptographic key distribution method that meets the following: **diffie-hellman-group1-sha1, diffie-hellman-group14-sha1 specified by SSH version 2.0 protocol defined with [SSHTRANS]; diffie-hellman-group-exchange-sha1 defined by [RFC4419]**.

6.1.2.4 DSS Public Cryptographic Key Distribution for SSH (FCS_CKM.2) (2)

FCS_CKM.2.1 The TSF shall distribute **public parts of asymmetric** cryptographic keys in accordance with a specified cryptographic key distribution method **of cryptographic certificates for public DSS keys** that meets the following: **SSH version 2.0 protocol defined with [SSHTRANS]**.

6.1.2.5 SSH Cryptographic Operation (FCS_COP.1)

FCS_COP.1.1 The TSF shall perform **encryption and decryption** in accordance with **the following** specified cryptographic algorithm and cryptographic key sizes:

a) 3DES in CBC mode with 168 bit key size;

b) AES in CBC mode with 128 bit, 192 bit or 256 bit key size;

c) Blowfish in CBC mode with 128 bit;

d) Arcfour in CBC mode with 128 bit key size ;

e) CAST in CBC mode with 128 bit key size;

f) HMAC-SHA1 with 160 bit hash size; and

g) HMAC-MD5 with 128 bit hash size

that meet the following: **SSH version 2.0 Transport Layer Protocol with the aforementioned cipher suites defined with [SSHTRANS]**.

6.1.3 User Data Protection (FDP)

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

FDP_ACC.1.1 The TSF shall enforce the **Discretionary Access Control Policy** on
Subjects: processes acting on the behalf of users,
Objects:

a) File system objects: files, directories, named pipes, symbolic links, unnamed pipes, character devices, block devices, unix domain sockets;

b) IPC kernel objects: SysV/POSIX shared memory, SysV message queues, SysV/POSIX semaphores;

c) IPC user space objects: BSD notifications

and all operations among subjects and objects covered by the DAC policy.

Application Note: The BSD notifications are implemented with the notify(3) mechanism. A daemon process allows other processes to register where other applications can submit notifications to the registered processes. The notify(3) mechanism utilizes kernel IPC mechanisms, but enforces the access control independently.

6.1.3.2 Mach Access Control Policy (FDP_ACC.1) (2)

FDP_ACC.1.1 The TSF shall enforce the **Mach Access Control Policy** on
Subject: tasks acting on behalf of users
Objects: Mach ports
Operations: sending and receiving messages.

Application Note: The Mac OS X kernel maintains two different representations of one process: the Mach task and the BSD process. The TOE ensures that both representations always point to the same acting entity. To avoid confusion with the TOE guidance, the ST uses the term “task” when referring to Mach mechanisms and “process” when referring to BSD mechanisms.

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

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

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

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

- **File system objects: ACLs comprising of Access Control Entries (ACE) with each ACE specifying the user (either UID or UUID) or group (either GID or UUID) it applies to and one or more of the permissions specified in** Table 4 File permission bits using ACLs**;**
- **File system objects: Unix permissions comprising of object owner ID, owning group ID and permission bits (read, write, execute, search, savetext, none) individually assigned to the different IDs or to all others;**
- **IPC kernel objects: owner ID, owning group ID, creator ID, creator group ID, permission bits (read, write, modify, none) individually assigned to the different IDs or to all others;**
- **IPC user space objects: notification keys and permission bits (read/receive, write/post, none)**

Application Note: The UUID is a 128-bit representation of the UID or GID provided to help administrators manage coherent and consistent rule sets for multiple user databases. The UUID is generated based on the UID/GID of the user by libraries while this UUID is resolved. Therefore, there is no separate user data store holding the UUID.

Application Note: Notification keys for the notify(3) mechanism are strings, although developers are encouraged to avoid namespace collisions by choosing keys in "Reverse-ICANN" style notation, e.g. "com.foo.bar.baz", using a prefix that should be unique to the respective software.

Bit	Bit meaning for file operation	Bit meaning for directory operation
read	Open file for read	List directory contents
write	Open file for write	Add a file entry to the directory
execute	Execute file	Search through the directory
delete	Delete file	Delete directory
append	Append to file	Add subdirectory to directory
delete child	N/A	Remove a file or subdirectory entry from the directory
read attributes	Read basic attributes	Read basic attributes
write attributes	Write basic attributes	Write basic attributes
read extended	Read extended (named) attributes	Read extended (named) attributes
write extended	Write extended (named) attributes	Write extended (named) attributes
read permissions	Read file permissions (ACL)	Read directory permissions (ACL)
write permissions	Write file permissions (ACL)	Write directory permissions (ACL)
take ownership	Take ownership	Take ownership

Table 4 File permission bits using ACLs

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

- **File system objects: Access is granted if one of the following is true, provided the subject possess search permission for every directory element of the path name**
  - **i. If the effective UID of the subject (for ACLs, also the UUID derived from the effective UID is considered) is identical to the owner ID of the object or the UID/UUID of the ACE, and the object's owner or ACE permissions indicate that the operation required accesses are allowed;**
  - **ii. If the effective UID of the subject is not identical to the owner ID of the object, and effective GID or the supplemental GIDs of the subject (for ACLs, also the UUID derived from the effective or supplemental GID is considered) is identical to the object's owning group or the GID/UUID of the ACE, and the object's**

group or ACE permissions indicate that the operation required accesses are allowed; or

iii. **If neither the effective UID nor the effective GID nor the supplemental GIDs of the subject are identical to the object's owner or object's owning group, the object's world UNIX permissions indicate that the operation required accesses are allowed.**

- **IPC kernel objects: Access is granted if one of the following is true:**

  i. **If the effective UID of the subject is identical to the owner ID or the creator ID of the object, and the object's owner permissions indicate that the operation required accesses are allowed;**

  ii. **If the effective UID of the subject is not identical to the owner ID of the object, and effective GID or the supplemental GIDs of the subject is identical to the object's owning group or the group ID of the ACE, and the object's group or ACE permissions indicate that the operation required accesses are allowed; or**

  iii. **If neither the effective UID nor the effective GID nor the supplemental GIDs of the subject are identical to the object's owner or object's owning group, the object's world UNIX permissions indicate that the operation required accesses are allowed (for SysV shared memory, the world write permission is not applied).**

- **IPC user space objects: Access is granted if one of the following is true:**

  i. **If the effective UID of the subject is identical to the owning ID defined for the key in /etc/notify.conf, and the owner permissions indicate that the operation required accesses are allowed;**

  ii. **If the effective UID of the subject is not identical to the owning ID defined for the key in /etc/notify.conf, and effective GID or the supplemental GIDs of the subject is identical to the owning group ID defined for the key in /etc/notify.conf, and the group permissions indicate that the operation required accesses are allowed; or**

  iii. **If neither the effective UID nor the effective GID nor the supplemental GIDs of the subject are identical to the owning ID or owning group ID defined for the key in /etc/notify.conf, the key's world permissions indicate that the operation required accesses are allowed.**

Application Note: The configuration file /etc/notify.conf may contain access control rules covering ranges of keys by specifying a common prefix covering the intended range of keys in the access control rule.FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules:

a) **File system objects: A subject with the effective UID of 0 is allowed to read and write any object regardless of the object's permission settings. A subject with the effective UID of 0 is allowed to execute files and search directories if the execute/search permission is granted to at least one user. A subject with the effective UID equal to the owner ID of the object covered with an ACL is granted read and write regardless of the object's permission settings.**

b) **IPC kernel objects: A subject with the effective UID of 0 is allowed to perform any access method for objects regardless of the object's permission settings. If the effective UID of the subject is identical to the owner ID or the creator ID of the object and the subject's requested access mode is modify, access is granted regardless of the object's permission setting.**

c) **IPC user space objects: none.**

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

a) **File system objects: A file stored in a directory with the savetext bit set in the directory permission bits can only be deleted by the owner provided the owner of the file has write access to the directory.**

b) **IPC kernel objects: If the effective UID of the subject is different to the owner ID or the creator ID of the object and the subject's requested access mode is modify, access is denied.**

c) **IPC kernel objects: For the semop operation on SysV semaphores, the caller must possess write permissions for the targeted semaphore to perform the operation.**

d) **IPC user space objects: none.**

Application Note: ACLs take precedence over permission bits as defined in section 7.3.1.2.1.3.

6.1.3.4 Mach Access Control Functions (FDP_ACF.1) (2)

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

a) **Subject attributes: task ID**

b) **Object attributes: port rights**

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

- **Receive right (reading of a message) for the Mach port is granted if the task has a receive right for the Mach port.**
- **Send right (submitting a message) for the Mach port is granted if the task has a send right for the Mach port.**

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

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the **following rules: none.**

Application Note: Each port has one receive right, the owner of which can receive messages on that port. Each port also has one or more send rights; the owners of the send rights can send messages to the port.

6.1.3.5 Object Residual Information Protection (FDP_RIP.2)

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.

6.1.3.6 Subject Residual Information Protection (Note 1)

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.

6.1.3.7 Basic Data Exchange Confidentiality (FDP_UCT.1)

FDP_UCT.1.1 The TSF shall enforce the **Discretionary Access Control Policy** to be able to **transmit and receive** objects in a manner protected from unauthorised disclosure.

Application Note: Using the cryptographically protected protocol of SSHv2, confidentiality of the transmitted data can be ensured. Discretionary Access Control still restricts access to the data which the user is able to transmit.

6.1.3.8 Data Exchange Integrity (FDP_UIT.1)

FDP_UIT.1.1 The TSF shall enforce the **Discretionary Access Control Policy** to be able to **transmit and receive** user data in a manner protected from **modification and insertion** errors.

FDP_UIT.1.2 The TSF shall be able to determine on receipt of user data, whether **modification or insertion** has occurred.

Application Note: Using the cryptographically protected protocol of SSHv2, integrity of the transmitted data can be ensured. Discretionary Access Control still restricts access to the data which the user is able to transmit.

6.1.4 Identification and Authentication (FIA)

6.1.4.1 User Attribute Definition (FIA_ATD.1)

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) Security-relevant roles; and

e) **no other attributes.**

Application Note: As mentioned in the application note for FDP_ACC.1(1), the UUID is a 128-bit value generated from the user ID or group ID every time the UUID is resolved.

6.1.4.2 Verification of Secrets (FIA_SOS.1)

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

a) For each attempt to use the authentication 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.

6.1.4.3 Authentication (FIA_UAU.2)

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

6.1.4.4 Protected Authentication Feedback (FIA_UAU.7)

FIA_UAU.7.1 The TSF shall provide only obscured feedback to the user while the authentication is in progress.

6.1.4.5 Identification (FIA_UID.2)

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

6.1.4.6 User-Subject Binding (FIA_USB.1)

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) **no other attributes.**

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

a) After successful authentication, the audit UID, the real UID, and the effective UID of the newly spawned process for the user have to be set to the UID specified for the user entry of the successfully authenticated user.

b) **After successful authentication, the real GID, the effective GID, and the supplemental GIDs of the newly spawned process for the user have to be set to the IDs specified for the user entry of the successfully authenticated user.**

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

a) **The real and effective UID 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 that has the SUID permission bit set, the effective UID associated with the subject shall be changed to that of the owner of the file;**

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

6.1.5 Security Management (FMT)

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

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:

a) **File system objects: the object owner or authorized administrator;**

b) **IPC kernel objects: the object owner, the creator or authorized administrator;**

c) **IPC user space objects: the authorized administrator.**

6.1.5.2 Management of Object Security Attributes (FMT_MSA.1) (2)

FMT_MSA.1.1 The TSF shall enforce the **Mach Access Control Policy** to restrict the ability to **transfer** the security attributes **of send and receive right** to **the owner of the respective right**.

Application Note: The Mach port rights can be transferred between tasks by being attached to a message.

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

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

6.1.5.4 Static Attribute Initialization (FMT_MSA.3) (2)

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

FMT_MSA.3.2 The TSF shall allow **nobody** to specify alternative initial values to override the default values when an object or information is created.

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

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

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

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

6.1.5.7 Management of User Attributes (FMT_MTD.1) (3)

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.

6.1.5.8 Initialization of Authentication Data (FMT_MTD.1) (4)

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

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

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.

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

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; and

b) The revocation or modification of security attributes of a user is enforced during the next time the user logs in.

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

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 policy.

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) No other rules.

Application Note: Similar to other UNIX operating systems, the TOE implements a delayed access revocation as explained by the application note to this SFR in [CAPP]. For files, the revoke(2) system call can be used to invalidate all file descriptors applicable for a specified file system object.

6.1.5.12 Specification of Management Functions (FMT_SMF.1)

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

a) Audit management;

b) User attribute management;

c) Object attribute management; and

d) Authentication data management.

6.1.5.13 Security Roles (FMT_SMR.1)

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 to modify their own authentication data; and

d) no other roles.

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

6.1.6 Protection of the TSF (FPT)

6.1.6.1 Reliable Time Stamps (FPT_STM.1)

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

6.1.6.2 Testing of External Entities (FPT_TEE.1)

FPT_TEE.1.1 The TSF shall run a suite of tests **during initial start-up, periodically or at the request of an authorized administrator** to check the fulfillment of **the security assumptions provided by the abstract machine that underlies the TSF**.

FPT_TEE.1.2 If the test fails, the TSF shall **notify the authorized administrator**.

Application Note: The test suite can be periodically executed when utilizing the batch job facilities provided by the TOE for a delayed execution (such as cron).

6.1.6.3 Inter-TSF trusted channel (FTP_ITC.1)

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

FTP_ITC.1.2 The TSF shall permit the **TSF or the remote trusted IT product** to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for **communication channels that use the SSHv2 protocol offered as a service by the TOE.**

6.2 Security Requirements Rationale

This section provides the rationale for the internal consistency and completeness of the security functional requirements defined in this Security Target.

6.2.1 Internal Consistency of Requirements

The mutual support and internal consistency of the components selected for this Security Target is described in this section.

The functional components were selected from CC components defined in part 2 of the Common Criteria, except the SFR of “Note 1” which was derived from [CAPP]. The use of component refinement was accomplished in accordance with CC guidelines.

Multiple instantiation of identical or hierarchically-related components was used to clearly state the required functionality that exists in this Security Target.

The following rationale demonstrates the internal consistency of the functional requirements.

6.2.1.1 Audit

The functional requirements for auditing were completely derived from [CAPP]. Please refer to this protection profile for the rationale.

6.2.1.2 Identification and Authentication

The functional requirements for identification and authentication were completely derived from [CAPP]. Please refer to this protection profile for the rationale.

The selection of the hierarchical SFRs of FIA_UAU.2 and FIA_UID.2 over the SFRs of FIA_UAU.1 and FIA_UID.1 specified in [CAPP] does not invalidate the rationale given there.

6.2.1.3 User Data Protection

The functional requirements for the Discretionary Access Control Policy were completely derived from [CAPP]. Please refer to this protection profile for the rationale.

The following rationale applies to the Mach Access Control Policy: FDP_ACC.1(2) requires the enforcement of the Mach Access Control Policy on Mach ports. The access control rules for this policy are defined with FDP_ACF.1(2). The management of the Mach port rights is defined with FMT_MSA.1(2) with the default values defined in FMT_MSA.3(2). Revocation of Mach port rights is enforced as defined for FDP_REV.1. No identification or authentication information is required as the Mach port rights are defined during creation time of these ports and the rights are distributed to the tasks that are given permissions to use the port by the creator. The policy is also supported by the residual data protection functionality FDP_RIP.2 preventing unauthorized access to residual user data by clearing the data objects prior to allocation of the object.

6.2.1.4 Residual Data Protection

The functional requirements for the Residual Data Protection were completely derived from [CAPP]. Please refer to this protection profile for the rationale.

6.2.1.5 Secure Communication

The TOE provides the SSHv2 communication protocol utilizing cryptographic mechanisms to protect data transmitted by the channel. The cryptographic mechanisms are used to ensure confidentiality and integrity protection of the user data transmitted through the SSHv2 channel, covering FDP_UCT.1 and FDP_UIT.1. Therefore, the protocol is able to provide a trusted inter-TSF communication channel required by FTP_ITC.1. The cryptographic key management is covered by the different iterations of FCS_CKM.1 as well as the different iterations of FCS_CKM.2. The cryptographic operation of the communication protocol is defined by FCS_COP.1.

6.2.1.6 Security Management

The functional requirements for the security management of the above mentioned functions were completely derived from [CAPP] with the exception of the management SFRs covering the Mach Access Control Policy. Please refer to this protection profile for the rationale.

The consistency of the Mach Access Control Policy SFRs, including its management SFRs is already analyzed above.

6.2.1.7 TOE Self Protection

The functional requirements for the TOE self-protection were completely derived from [CAPP]. Please refer to this protection profile for the rationale.

Due to the compliance of the ST with CC version 3.1, the SFRs of FPT_RVM.1 and FPT_SEP.1 were replaced with the SAR of ADV_ARC.1.

6.2.2 Security Requirements Coverage

The following table shows that each security functional requirement addresses at least one objective.

Table 5 Mapping Security Functional Requirements to Objectives

SFR	Objectives
FAU_GEN.1	O.AUDITING
FAU_GEN.2	O.AUDITING
FAU_SAR.1	O.AUDITING
FAU_SAR.2	O.AUDITING
FAU_SAR.3	O.AUDITING
FAU_SEL.1	O.AUDITING
FAU_STG.1	O.AUDITING
FAU_STG.3	O.AUDITING
FAU_STG.4	O.AUDITING
FCS_CKM.1(1)	O.COMPROT
FCS_CKM.1(2)	O.COMPROT
FCS_CKM.2(1)	O.COMPROT
FCS_CKM.2(2)	O.COMPROT

SFR	Objectives
FCS_COP.1	O.COMPROT
FDP_ACC.1(1)	O.DISCRETIONARY_ACCESS
FDP_ACC.1(2)	O.DISCRETIONARY_ACCESS
FDP_ACF.1(1)	O.DISCRETIONARY_ACCESS
FDP_ACF.1(2)	O.DISCRETIONARY_ACCESS
FDP_RIP.2	O.RESIDUAL_INFORMATION
Note 1	O.RESIDUAL_INFORMATION
FDP_UCT.1	O.COMPROT
FDP_UIT.1	O.COMPROT
FIA_ATD.1	O.AUTHORIZATION, O.DISCRETIONARY_ACCESS
FIA_SOS.1	O.AUTHORIZATION
FIA_UAU.2	O.AUTHORIZATION
FIA_UAU.7	O.AUTHORIZATION
FIA_UID.2	O.AUTHORIZATION
FIA_USB.1	O.AUTHORIZATION, O.DISCRETIONARY_ACCESS
FMT_MSA.1(1)	O.DISCRETIONARY_ACCESS, O.MANAGE
FMT_MSA.1(2)	O.DISCRETIONARY_ACCESS, O.MANAGE
FMT_MSA.3(1)	O.DISCRETIONARY_ACCESS, O.MANAGE
FMT_MSA.3(2)	O.DISCRETIONARY_ACCESS
FMT_MTD.1(1)	O.AUDITING, O.MANAGE
FMT_MTD.1(2)	O.AUDITING, O.MANAGE
FMT_MTD.1(3)	O.MANAGE
FMT_MTD.1(4)	O.MANAGE
FMT_MTD.1(5)	O.AUTHORIZATION, O.MANAGE
FMT_REV.1(1)	O.MANAGE
FMT_REV.1(2)	O.DISCRETIONARY_ACCESS, O.MANAGE
FMT_SMF.1	O.MANAGE
FMT_SMR.1	O.MANAGE, O.AUDITING
FPT_TEE.1	O.ENFORCEMENT
FPT_STM.1	O.AUDITING
FTP_ITC.1	O.COMPROT

O.AUDITING

The events to be audited are defined in [FAU_GEN.1], and are associated with the identity of the user that caused the event [FAU_GEN.2]. Authorized administrators are provided the capability to read the audit records [FAU_SAR.1] and to be able to select which audit information they want to review [FAU_SAR.3], while all other users are denied access to the audit records [FAU_SAR.2]. The administrator must have the capability to specify which audit records are generated [FAU_SEL.1]. The TOE prevents the audit log from being modified or deleted [FAU_STG.1], and ensure that the audit log is not lost due to resource shortage [FAU_STG.3, FAU_STG.4]. The TOE provides a time stamp for use by the audit facility [FPT_STM.1]. Management mechanisms of the audit facility are provided [FMT_MTD.1(1), FMT_MTD.1(2)]. The TOE maintains the notion of an administrative role which is allowed to manage the audit facility [FMT_SMR.1].

O.AUTHORIZATION

The TSF must ensure that only authorized users gain access to the TOE and its resources. Users authorized to access the TOE have to use an identification and authentication process [FIA_UID.2, FIA_UAU.2]. To ensure authorized access to the TOE, authentication data is protected [FIA_ATD.1, FIA_UAU.7, FMT_MTD.1(5)]. Proper authorization for subjects acting on behalf of users is also ensured [FIA_USB.1]. The appropriate strength of the authentication mechanism is ensured [FIA_SOS.1].

O.DISCRETIONARY_ACCESS

The TSF must control access to resources based on the identity of users which are allowed to specify which resources they want to access.

The discretionary access control policy, as well as the Mach access control policy must have a defined scope of control [FDP_ACC.1(1), FDP_ACC.1(2)]. The rules for the different access control policies are be defined [FDP_ACF.1(1), FDP_ACF.1(2)]. The management of the different access control policies is specified [FMT_MSA.1(1), FMT_MSA.1(2)], including the revocation of access rights [FMT_REV.1(2)]. The default values for the different access control policies are defined [FMT_MSA.3(1), FMT_MSA.3(2)]. The security attributes of subjects used to enforce the different access control mechanisms must be defined [FIA_ATD.1, FIA_USB.1].

O.RESIDUAL_INFORMATION

Residual information associated with objects in the TOE must be cleared of prior to reuse of the object [FDP_RIP.2] and before a resource/object is given to a subject [Note 1].

O.COMPROT

The TOE provides a cryptographically-protected communication channel between itself and another trusted IT product [FTP_ITC.1], which prevents user data disclosure [FDP_UCT.1] and undetected modification [FDP_UIT.1]. The cryptographic functions supporting that secure communication channel are: key generation [all iterations of FCS_CKM.1], key distribution [all iterations of FCS_CKM.2] and cryptographic protocol definition [FCS_COP.1].

O.MANAGE

The TOE provides the capability to manage all other security functions [FMT_SMF.1]. To ensure secure management, the TOE implements separation of roles [FMT_SMR.1]. Management tools for the audit facility are provided [FMT_MTD.1(1), FMT_MTD.1(2)], as well as for the user management [FMT_MTD.1(3), FMT_MTD.1(4), FMT_MTD.1(5), FMT_REV.1(1)], and object management [FMT_REV.1(2)], FMT_MSA.1(1), FMT_MSA.1(2)]. Also, the default values for the Discretionary Access Control Policy can be managed [FMT_MSA.3(1)].

O.ENFORCEMENT

The TOE provides tools to the administrator which allow the verification of the underlying hardware for correct behavior [FPT_TEE.1].

6.2.3 Security Requirements Dependency Analysis

The following table shows the dependencies between the different security functional requirements and if they are resolved in this Security Target.

Table 6 Dependencies between Security Functional Requirements

Security Functional Requirement	Dependencies	Resolved (reference to SFR in case of ambiguities)
FAU_GEN.1	FPT_STM.1	Yes
FAU_GEN.2	FAU_GEN.1 FIA_UID.1	Yes FIA_UID.2
FAU_SAR.1	FAU_GEN.1	Yes
FAU_SAR.2	FAU_SAR.1	Yes
FAU_SAR.3	FAU_SAR.1	Yes
FAU_SEL.1	FAU_GEN.1	Yes
FAU_STG.1	FAU_GEN.1	Yes
FAU_STG.3	FAU_STG.1	Yes
FAU_STG.4	FAU_STG.1	Yes
FCS_CKM.1(1)	FCS_CKM.2 or FCS_COP.1 FCS_CKM.4	No
FCS_CKM.1(2)	FCS_CKM.2 or FCS_COP.1 FCS_CKM.4	No
FCS_CKM.2(1)	FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 FCS_CKM.4	No FCS_CKM.1(1)
FCS_CKM.2(2)	FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 FCS_CKM.4	No FCS_CKM.1(2)
FCS_COP.1	FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 FCS_CKM.4	No FCS_CKM.1(1)
FDP_ACC.1(1)	FDP_ACF.1	Yes FDP_ACF.1(1)
FDP_ACF.1(1)	FDP_ACC.1 FMT_MSA.3	Yes FDP_ACC.1(1), FMT_MSA.3(1)
FDP_ACC.1(2)	FDP_ACF.1	Yes FDP_ACF.1(2)

Security Functional Requirement	Dependencies	Resolved (reference to SFR in case of ambiguities)
FDP_ACF.1(2)	FDP_ACC.1 FMT_MSA.3	Yes FDP_ACC.1(2), FMT_MSA.3(2)
FDP_RIP.2	No dependencies	Yes
Note 1	No dependencies	Yes
FDP_UCT.1	FDP_ACC.1 or FDP_IFC.1 FTP_ITC.1 or FTP_TRP.1	Yes FDP_ACC.1(1)
FDP_UIT.1	FDP_ACC.1 or FDP_IFC.1 FTP_ITC.1 or FTP_TRP.1	Yes FDP_ACC.1(1)
FIA_ATD.1	No dependencies	Yes
FIA_SOS.1	No dependencies	Yes
FIA_UAU.2	FIA_UID.1	Yes FIA_UID.2
FIA_UAU.7	FIA_UAU.1	Yes FIA_UAU.2
FIA_UID.2	No dependencies	Yes
FIA_USB.1	FIA_ATD.1	Yes
FMT_MSA.1(1)	FDP_ACC.1 or FDP_IFC.1 FMT_SMR.1 FMT_SMF.1	Yes FDP_ACC.1(1)
FMT_MSA.1(2)	FDP_ACC.1 or FDP_IFC.1 FMT_SMR.1 FMT_SMF.1	Yes FDP_ACC.1(2)
FMT_MSA.3(1)	FMT_MSA.1 FMT_SMR.1	Yes FDP_MSA.1(1)
FMT_MSA.3(2)	FMT_MSA.1 FMT_SMR.1	Yes FDP_MSA.1(2)
FMT_MTD.1(1)	FMT_SMR.1 FMT_SMF.1	Yes
FMT_MTD.1(2)	FMT_SMR.1 FMT_SMF.1	Yes
FMT_MTD.1(3)	FMT_SMR.1 FMT_SMF.1	Yes
FMT_MTD.1(4)	FMT_SMR.1 FMT_SMF.1	Yes
FMT_MTD.1(5)	FMT_SMR.1 FMT_SMF.1	Yes
FMT_REV.1(1)	FMT_SMR.1	Yes
FMT_REV.1(2)	FMT_SMR.1	Yes
FMT_SMF.1	No dependencies	Yes
FMT_SMR.1	FIA_UID.1	Yes FIA_UID.2
FPT_TEE.1	No dependencies	Yes
FPT_STM.1	No dependencies	Yes
FTP_ITC.1	No dependencies	Yes

Comment

The SFRs of FCS_CKM.1, FCS_CKM.2, and FCS_COP.1 all depend on FCS_CKM.4 requiring the functionality of a key destruction mechanism. The TOE does not implement an explicit key destruction mechanism. The key destruction for the symmetric keys is implicitly performed by the residual information protection mechanism, which ensures that memory and disk space is cleared before it is reassigned to another subject or object. Concerning the long-term public-private key pairs, the key destruction is performed by deleting the file. As already mentioned, the residual information protection mechanism would clear the disk blocks containing the deleted information of the key material prior to reassignment to subjects or objects. As the TOE generates all required keys itself, all other dependencies are satisfied.

There are no unresolved dependencies between security assurance requirements. This is because the evaluation assurance level EAL3 has been defined such that no unresolved dependencies exist. The additional assurance component ALC_FLR.3 has no dependencies and, therefore, there are no unresolved dependencies for assurance components.

6.3 TOE Security Assurance Requirements

The target evaluation assurance level for the product is EAL3 [CC] augmented by ALC_FLR.3, which is seen as appropriate for a controlled environment where attackers only have a basic attack potential.

7 TOE Summary Specification

This chapter describes the Mac OS X security functions and associated assurance measures.

7.1 Audit

The audit security function provides for the generation and management of audit. Both of these areas are described in this section.

7.1.1 Audit Generation

Mac OS X maintains all audit records in the equivalent to the Basic Security Module (BSM). The audit log is managed by the audit daemon. The audit daemon only retrieves its audit records from the kernel via the /dev/audit device, and stores them in the audit trail. When an audit record is created, the time stamp for the audit record is controlled and updated by the time daemon.

Mac OS X provides a wide set of audit events to include those listed for FAU_GEN.1. Each audit record contains the following information:

- Date
- Time
- Event type
- User Identifier (UID)
- Process ID (PID)
- Success or failure

In addition to the information listed, for some audit records, additional information is provided as identified the table specified for FAU_GEN.1.

The administrator has several options for configuring the management of the audit trail. The administrator can specify a rotation scheme among the audit files. If the last in a series of audit files fills up the disk, Mac OS X will go back to the first audit file to start writing and overwrite any previously recorded data. The second option is that once all audit files are filled, Mac OS X will halt. When the audit trails reaches a configured percentage value of the remaining capacity of the hard disk, a shell script is executed with can be used to alert the administrator in many different ways that the audit trail is reaching capacity.

7.1.2 Audit Protection

The kernel collects all audit information originating from within the kernel and sends it via /dev/audit to the user space audit daemon. This audit daemon stores the information in a file on hard disk. Only administrators can access /dev/audit.

Audit events generated by user space applications use a system call to send the audit message to the kernel. The kernel then handles them the same way as kernel-internal generated audit records by sending them back to user space via /dev/audit. The system call is restricted to authorized administrators.

The audit log files are protected by discretionary access control (DAC) so that only administrators can access the audit trail.

7.1.3 Audit Management

The audit trail is protected so that only the administrator can access and manage the audit trail. The administrator has a number of tools to manage the audit trail. The audit tools provide the following capabilities:

- Audit maintenance – the administrator can create, clear, and delete the audit trail.
- Audit selection – the administrator can select which audit events to record and can refine the selection based on user identity, success or failure of the audit event and the events listed in FAU_SAR.3.
- Audit review – the administrator can review the contents of the audit trail. The audit trail viewing tool permits the administrator to search and sort the audit trail based on user identity.

The Audit security function satisfies the following security requirements FAU_GEN.1, FAU_GEN.2, FAU_SAR.1, FAU_SAR.2, FAU_SAR.3, FAU_SEL.1, FAU_STG.1, FAU_STG.3, FAU_STG.4, FMT_MTD.1(1), FMT_MTD.1(2).

7.2 Identification and Authentication

The identification and authentication security function provides logon features, user attribute management, and password and account management. Each one is described in this section.

7.2.1 Logon Processing

Users can either log onto the system locally or via network. The following list explains each type of login mechanism provided by the TOE:

- Console login: the loginwindow application ensures that before a user accesses any TSF-mediated functions, that user is identified and authenticated to Mac OS X. The loginwindow application provides a graphical interface requesting a user name and password. The user name is echoed back to the screen, while the password is obscured from the user with dots.
- Login via SSH: The client process of ssh passes an identity and authentication credentials on to the ssh server. The server validates these credentials based on the configured authentication mechanisms. Logons with username / password only are supported.

Logon processing is performed by the DirectoryService mechanism which provides a common interface to the user databases. DirectoryService allows access to different types of user databases with a single API to calling application wanting to identify and authenticate users.

The TOE also provides the PAM (pluggable authentication module) mechanism which allows specifying different identification and authentication procedures for different calling applications. The above mentioned applications providing identification and authentication to users are linked to PAM to identify and authenticate users. The DirectoryService is called by the PAM framework by including the authentication module interfacing with DirectoryService in the PAM configuration for these applications.

Upon successful identification and authentication, a new process is spawned with the owning user ID derived from the authentication credentials. This new process starts a session for the user and can be used to interactively work with the TOE. Additional applications may be started for the session, either automatically from the initially spawned process or manually by the user.

Other network services/protocols in the evaluated configuration are unauthenticated. Those unauthenticated services/protocols are: TCP/IP, and NFS. The trust in these unauthenticated services is established as follows:

- The TCP/IP protocol relies on higher-level services to perform authentication. The IP packet is accepted by the TOE only if an application listens on a socket and interprets the data transmitted with the IP packet. If there is no application listening, the TOE rejects the packet.
- NFS performs access checks based on the credentials of the requesting user. Therefore, the NFS server implicitly trusts the NFS client system to have performed identification and authentication. Also, the NFS server trusts the NFS client system to transmit the proper user ID of the user causing the file access request.

7.2.2 User Subject Binding

Processes are the subjects within the TOE and are associated with users based on the IDs maintained for the process. A process is an abstraction for a running program. A process' resources include a virtual address space, threads, and file descriptors. In Mac OS X, a process is based on one Mach task and one or more Mach threads. Every thread within a process has access to the address space, file descriptors, Mach ports and other resources. There are a several security relevant attributes associated with a subject:

- Real UID
- Saved UID
- Effective UID
- Audit UID
- Real GID
- Saved GID

- Effective GID
- Supplemental group list
- PID
- Process umask

There is only one way that a new subject can be created: using fork system call. All applications which need to spawn a new process use this system call to create a new process. When a logon occurs, a process is created on behalf of the requesting user, and attributes are assigned at creation. When a running process issues a fork, a copy of the calling process is created and it is assigned a new PID. Lastly, when a batch job is activated, a new process is created on behalf of the requesting user with the attributes of the requestor.

The security attributes of a subject are derived from the parent process calling the fork system call. However, using a SUID or SGID program, the effective UID or GID, respectively, of the process will be changed. The new effective UID is taken from the file owner and the effective GID is taken from the owning group of the file being executed.

The audit UID is set during login time and is not changed by any SUID program execution or setuid()/setreuid() system call.

7.2.3 Account and Password Management

User account information is stored in the user database. The following information is stored for each user:

- User name
- User ID
- Password
- Groups

The groups information indicates if a user is part of the admin group. Being a member of the admin group makes a user an administrator. Users belonging to this group are entitled to use the sudo command, which allows the caller to execute any command with administrative privileges.

Users are authorized to modify their own authentication data (i.e. their passwords). Users are permitted to change their own passwords by supplying their current password and selecting a new password. Password quality rules are enforced during the changing of the users password. These rules are set to values meeting the requirements in FIA_SOS.1 by the evaluated configuration. All information is stored in files protected by DAC against read or write access by untrusted users. Additionally, the password data is stored in hashed format. Only the administrator has access to update the user account information other than the authentication data. Changes to the authentication database take effect the next time a user authenticates. If the administrator needs to immediately revoke a user’s account attributes, the user must be forced to log off so the changes can take effect.

Passwords for user accounts are initially assigned by the administrator and can be changed at any time by the administrator.

The Identification and Authentication security function satisfies the following security requirements: FIA_ATD.1, FIA_SOS.1, FIA_UAU.2, FIA_UAU.7, FIA_UID.2, FIA_USB.1, FMT_MTD.1(3), FMT_MTD.1(4), FMT_MTD.1(5), FMT_REV.1(1).

7.3 User Data Protection

The user data protection security function provides discretionary access control (DAC), as well as the access control protection implemented for the Mach messaging.

7.3.1 Discretionary Access Control

The DAC mechanism is used to control access between subjects and named objects. DAC is a mechanism that controls access based on user and object identities. This section first identifies the attributes that are used when making DAC decisions and then describes the DAC policy for each type of object.

7.3.1.1 DAC Attributes

DAC decisions are based upon the attributes of subjects and objects. The DAC relevant attributes of a subject are defined by the user-subject binding.

The DAC-specific attributes of the named objects depend upon the type of object.

7.3.1.2 DAC Policies

7.3.1.2.1 File System Objects

File system objects follow the DAC algorithm described in this section with one exception; administrators are always granted permission to file system objects, with the exception of the execute/search right as specified in FDP_ACF.1(1). In order to access a file system object, a subject must pass the permissions bit check.

The DAC mechanism for file system objects covers the two algorithms which are listed in the following sections.

7.3.1.2.1.1 Permission Bits

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 owning group, and every other user. When a check is performed against the permission bits, the owner bits are checked first, followed by the group bits, and then the other bits. The first set of permissions matched is the permission granted.

Directory objects have a special bit, called a *sticky bit*. When the sticky bit is set, only the owner of the file system object may delete the object.

After a subject gains access to a file system object, the subject receives a file descriptor. In the file descriptor is a pointer to the file object and the permission granted. Whenever the subject makes future requests for the same object, the file descriptor is checked to determine that the subject is attempting to use only the permissions granted during the access check. In the case of unnamed pipes, no name exists in the namespace, so all access occurs via file descriptors. An unnamed pipe is only accessible through the file descriptors given to the process that created it and the creating process' descendants. Because it has no name in the file system namespace, it has no pathname through which it can be accessed by unrelated processes, which causes DAC to be not applicable.

Access checks are performed slightly differently on symbolic links. The access permissions are stored with the file system object referenced by the symbolic link, not on the link itself.

7.3.1.2.1.2 Access Control Lists

An ACL consists of an ordered list of access control entries (ACEs), each of which associates a user or group with a set of permissions and specifies whether each permission is allowed or denied. ACLs have a fine-grained set of permissions. The list of permissions allowed to be used for an ACE is given in [MACSEC], section “ACLs”.

The owner of a file using an ACL has certain unrevokable permissions (read and write permissions), regardless of the contents of the ACL.

The evaluation algorithm for ACLs is described in [MACSEC], section “ACLs”.

7.3.1.2.1.3 Consistency of file permissions

Due to the fact that two permission algorithms are provided to protect file system objects, both mechanisms must be interpreted consistently.

ACLs are validated first. If the end of the access validation of the ACL associated with the file system object is reached without finding all of the required permissions, and if the object also has permission bits, then the system checks the unsatisfied permissions against the BSD permissions. If these are sufficient to grant all required permissions, the request is allowed. If the permission requested has no BSD equivalent (such as “take ownership”), then it is considered still outstanding and the request is denied.

7.3.1.2.2 IPC Mechanisms

Each IPC kernel object includes a structure that contains the permission and ownership information. The set of permission bits maintained for creator and owner, creator's group and owner's group, and others can allow either read or write access to the IPC object. The owner UID and GID is initially set to the creator's UID and GID.

Each IPC user space object (implemented with the notify(3) mechanism) is associated with the permission and ownership information applicable to that object in the /etc/notify.conf configuration file. The set of permission bits maintained for owning ID, owning group ID, and others can allow either read or write access to the IPC object. The access control rules are maintained independent from the IPC objects. Therefore, IPC objects can be created and removed independently from the access control rules. If no access control rule applies to an IPC object, access is denied.

Access is granted according to the permission bits.

7.3.1.2.3 Default DAC

Only the owner or the administrator can modify the access control attributes associated with an object. 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. The various IDs associated with objects are taken from the creating subject's attributes.

With ACLs, newly created files and subdirectories can inherit permissions from their enclosing directory. Each ACE on a directory can contain any combination of the following inheritance flags:

- Inherited (this ACE was inherited).
- File Inherit (this ACE should be inherited by files created within this directory).
- Directory Inherit (this ACE should be inherited by directories created within this directory).
- Inherit Only (this ACE should not be checked during authorization).
- No Propagate Inherit (this ACE should be inherited only by direct children; that is, the ACE should lose any Directory Inherit or File Inherit bit when inherited).

Additional information on ACL inheritance can be found in [MACSEC] section "ACLs".

If a subject changes the permissions of an object, the changes take affect on the next access check against the object. So, if a subject has a file descriptor open for an object and attempts to use the file descriptor, the old permissions will remain in effect until the subject closes the object. If the subject closes the object and then attempts to re-open it, the new permissions would then be enforced.

7.3.2 Mach IPC Access Control

Mach security is based on ports and port rights. A Mach port is an endpoint of a communication channel between a client who requests a service and a server that provides the service. Mach ports are unidirectional; a reply to a service request must use a second port.

7.3.2.1 Mach Messaging Attributes

A port has a set of associated port rights, which are owned by tasks. A port right specifies which task can use that port. Each port has one receive right, the owner of which can receive messages on that port. Each port also has one or more send rights; the owners of the send rights can send messages to the port.

7.3.2.2 Mach Messaging Access Control Algorithm

Tasks owning a send right for a particular port are allowed to send messages to that port.

Tasks owning a receive right for a particular port are allowed to receive messages from that port.

7.3.2.3 Mach Messaging Access Control Policies

Send or receive rights can be transferred between tasks by being attached to a message. However, unless the port right name is sent correctly, the receiving task will not be able use the right. The only way to transmit a port right between two tasks is by sending a Mach message and attaching the right name to that message using the correct syntax and message structure.

When a Mach port is created, the creator receives the send and receive rights.

When a task is created, Mach returns a port right name to the creator of the task that references a send right for the port (the receive right for a task port is always owned by the kernel). Messages can be sent to this port to start and stop the task, kill the task, manipulate the task's address space, and so forth. Therefore, whoever owns a send right for a task's port effectively owns the task and can manipulate the task's state without regard to BSD security policies or any higher-level security policies.

The User Data Protection security function satisfies the following security requirements: FAU_SAR.2, FDP_ACC.1(1), FDP_ACC.1(2), FDP_ACF.1(1), FDP_ACF.1(2), FIA_USB.1, FMT_MSA.1, FMT_MSA.3, FMT_REV.1(2). As ADV_ARC.1 covers the property of the TOE that it can appropriately protect itself, DAC and Mach IPC access control supports the protection of the TOE against untrusted users.

7.4 Residual Data Protection

Mac OS X ensures that all previously allocated memory is cleared before it is allocated to a user process. File system objects are initialized at creation time, overwriting the existing information in the reused disk blocks. Additionally, an end of file marker prevents users from accessing data beyond the current file boundary. Other objects that use memory are cleared upon allocation. This includes process addresses space and execution context, as well as IPC memory spaces.

The Residual Data Protection security function satisfies the following security requirements: FDP_RIP.2, Note 1.

7.5 Secure Communication

Using cryptographic services, the TOE provides a network communication channel protecting the transported information against disclosure and undetected and unauthorized modification. The protocol of SSHv2 is used to provide protection of communication against the threats mentioned in the Security Problem Definition.

All other communication channels using different protocols are not protected against these threats and must therefore be secured by environmental constraints.

With the protocol of SSHv2, a secure remote communication between the TOE and another trusted IT product over an insecure network can be achieved.

The TOE provides an implementation of:

- Secure Shell Transport Layer protocol version 2 specified in [SSHTRANS]
- Secure Shell Authentication protocol specified in [SSHAUTH] chapter 8.
- Generation of asymmetric keys as defined in FIPS 186-2

The TOE provides the implementation of the server as well as the client side for the mentioned protocols. Therefore, the TOE is able to act as the server as well as the client for the secured network communication.

The server side provided by the TOE allows remote clients to connect via SSHv2 and initiate a session after performing identification and authentication.

The following cryptographic mechanisms:

- Encryption: the ciphers listed in FCS_COP.1 are supported by the TOE out of the list of ciphers stated in [SSHTRANS] section 6.3,
- Keyed hash functions: the keyed hash functions listed in FCS_COP.1 are supported by the TOE out of the list of ciphers stated in [SSHTRANS] section 6.4.
- Key exchange: the key exchange mechanisms listed in FCS_CKM.2(1) and FCS_CKM.2(2) are supported by the TOE out of the list of key exchange mechanisms stated in [SSHTRANS] section 6.5.

If the SSH server identifies an integrity error in a communication, the connection is closed.

The Secure Communication security function satisfies the following security requirements: all iterations of FCS_CKM.1, all iterations of FCS_CKM.2, FCS_COP.1, FDP_UCT.1, FDP_UIT.1, and FTP_ITC.1.

7.6 Security Management

Mac OS X supports security management by providing an administrator to manage the security functions. The administrator is authorized to perform all management functions including establishing and maintaining user accounts, modifying access rights, and managing the audit trail. The administrator account is realized via the use of a group. Members of the admin group are considered to be administrators.

The Security Management security function satisfies the following security requirements: FMT_MSA.1, FMT_MSA.3, FMT_MTD.1 (all iterations), FMT_REV.1(2), FMT_SMF.1, FMT_SMR.1.

7.7 TOE Self Protection

The TOE protection mechanisms security function provides abstract machine testing, reference mediation, domain separation, and reliable time stamps. Each function is described in this section.

7.7.1 Abstract Machine Testing

The TOE provides a testing facility to demonstrate the correct operation of the security features of the underlying hardware and firmware. These evaluation test suites include tests for memory protection and processor privileged instructions. The tests are made available to the administrator to run at the administrator's request.

7.7.2 Reference Mediation

The MAC OS X architecture is based on a kernel-mode architecture. 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 processes. When untrusted processes access system resources in user-mode, they do so through well–defined server interfaces so that the security policy is enforced on the untrusted processes.

7.7.3 Domain Separation

The software portion of the TOE uses several execution domains to protect itself from external interference and tampering. The kernel runs in the privileged model (i.e., kernel-mode) provided by the evaluated processor architectures. A system call trap is a software interrupt operation that causes a context switch from user-mode into kernel-mode. In kernel-mode, a process can only execute the kernel-defined code sequence that follows from a system call.

All TOE trusted servers (e.g., the SSH daemon), administrator tools and user programs execute in a process which is protected through the address space isolation mechanisms of the kernel. Each process is allocated a separate address space that is not shared unless specific access is granted.

7.7.4 Time

The system time is maintained by the kernel. This system time is used in the audit trail to maintain an accurate accounting of when audit events occurred. The kernel maintained system clock is initialized during boot using the time information of the hardware clock.

The TOE Self Protection Mechanism security function satisfies the following security requirements: FPT_TEE.1, FPT_STM.1. As ADV_ARC.1 covers the property of the TOE that it can appropriately protect itself, the TOE self-protection mechanism supports the protection of the TOE against untrusted users.

8 Abbreviations