National Information Assurance Partnership

Common Criteria Evaluation and Validation Scheme Validation Report

Apple Computer Mac OS X Version 10.3.6

Report Number:	**CCEVS-VR-05-0086**
Dated:	**13 January 2005**
Version:	**1.0**

National Institute of Standards and Technology
Information Technology Laboratory
100 Bureau Drive
Gaithersburg, MD 20899

National Security Agency
Information Assurance Directorate
9800 Savage Road STE 6740
Fort George G. Meade, MD 20755-6740

Table of Contents

1. Executive Summary

This report documents the NIAP validators' assessment of the CCEVS evaluation of the Apple Mac OS X version 10.3.6 and Apple Mac OS X Server version 10.3.6. Mac OS X and Mac OS X Server enforce the same security functions, the only differences lie in performance; therefore, both will be referred to as Mac OS X throughout this report.

The evaluation for Mac OS X was performed by Science Applications International Corporation (SAIC) in the United States and was completed on 27 December 2004. The evaluation was conducted in accordance with the requirements of the Common Criteria for Information Technology Security Evaluation, version 2.1, Evaluation Assurance Level 3 (EAL3), and the Common Evaluation Methodology for IT Security Evaluation (CEM), Part 2, Version 1.0. The Target of Evaluation (TOE) also conforms to the Controlled Access Protection Profile (CAPP), version 1.d, October 8, 1999.

The TOE includes the Mac OS X operating system, supporting hardware, and those applications necessary to manage, support and configure the operating system. The TOE is a subset of the Mac OS X product as defined in the administrator guidance. 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.

Services outside this evaluation include: e-mail services; web server services; remote apple events; print sharing services; file sharing services; and classic programming support (Old Macintosh OS compatibility support). Mac OS X Server contains a watchdog timer to restart services and provide stability; however, this timer is disabled in the evaluated configuration.

In addition, Darwin, the open source UNIX-based core that Mac OS X is built upon, only supports the HFS+ filesystem in the evaluated configuration. Furthermore, while Mac OS X supports a wide range of protocols and network services, only TCP/IP and the NFS (Network Filesystem), DNS (Domain Name Service), and SSH (Secure Shell) services are supported in the evaluated configuration.

Mac OS X provides the following five security functions, which are described in Section 3 of this report:

- Security Audit
- User Data Protection
- Identification and Authentication
- Security Management
- Protection of the TOE Security Functions

SAIC is an approved NIAP Common Criteria Testing Laboratory (CCTL). The CCTL concluded that the Common Criteria assurance requirements for Evaluation Assurance Level 3 (EAL3) have been met and that the conclusions in its Evaluation Technical Report are consistent with the evidence produced.

This Validation Report is not an endorsement of the Mac OS X by any agency of the US Government and no warranty of the product is either expressed or implied.

1.1 Evaluation Details

Table 1 provides the required evaluation identification details.

Table 1. Evaluation Details

Item	Identification
Evaluation Scheme	US Common Criteria Evaluation and Validation Scheme (CCEVS)
Target of Evaluation	Apple Mac OS X version 10.3.6 and Apple Mac OS X Server version 10.3.6
EAL	EAL3
Protection Profile	Controlled Access Protection Profile (CAPP), version 1.d, October 8, 1999
Security Target	Apple Mac OS X Version 10.3.6 Security Target, Version 1.0, 13 December 2005
Developer	Apple Computer, Inc. 1 Infinite Loop; Cupertino, CA 95014
Evaluators	SAIC - Common Criteria Testing Laboratory 7125 Gateway Drive, Suite 300 Columbia, MD 21046
Validator	Catalina M. Gomolka Mitretek Systems, Inc., Falls Church, VA
Dates of Evaluation	June 2002 to 27 December 2004
Conformance Result	Part 2 conformant, Part 3 conformant, and EAL3 conformant
Common Criteria (CC) Version	Part 1: Introduction and General Model, Version 2.1, August 1999, CCIMB-99-031 Part 2: Security Functional Requirements, Version 2.1, August 1999, CCIMB-99-032 Part 3: Security Assurance Requirements, Version 2.1, August 1999, CCIMB-99-033
Common Evaluation Methodology (CEM) Version	Part 1: Introduction and General Model, Version 0.6, 97/01/11, CEM-97/017 Part 2: Evaluation Methodology, Version 1.0, August 1999, CEM-99/045 Part 2: Evaluation Methodology, Supplement: ALC_FLR - Flaw Remediation, Version 1.1, February 2002, CEM-2001/0015R
Evaluation Technical Report	Evaluation Technical Report for Apple Computer Mac OS X v10.3.6 and Mac OS X Server v10.3.6 Part 1 (Non-Proprietary) Version 1.0 December 2004 Evaluation Technical Report for Apple Computer Mac OS X v10.3.6 and Mac OS X Server v10.3.6 Part 2 (SAIC and Apple Proprietary) Version 1.0 December 2004

2. Identification of the TOE

The Mac OS X Version 10.3.6 includes the Mac OS X operating system, supporting hardware, and those applications necessary to manage, support and configure the operating system. The TOE is a subset of the Mac OS X product as defined in the administrator guidance. 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. Services outside this evaluation include: e-mail services; web server services, remote apple events, print sharing services, file sharing services, and classic programming support (Old Macintosh OS compatibility support). Mac OS X Server contains a watchdog timer to restart services and provide stability; however, this timer is disabled in the evaluated configuration.

The following hardware platforms are included in the evaluated configuration:

- Mac OS X version 10.3.6
  - eMac G4
  - iMac G3
  - iMac G4
  - iMac G5
  - iBook G3
  - iBook G4
  - PowerBook G3
  - PowerBook G4
  - Power Mac G3
  - Power Mac G4 Cube
  - Power Mac G4 (Single processor)
  - Power Mac G4 Dual Processor
  - Power Mac G5 (Single processor)
  - Power Mac G5 Dual Processor
- Mac OS X Server version 10.3.6
  - Power Mac G4 (Single processor)
  - Power Mac G4 Dual Processor
  - Power Mac G5 (Single processor)
  - Power Mac G5 Dual Processor
  - Xserve G4 (Single processor)
  - Xserve G4 Dual Processor
  - Xserve G5 (Single processor)
  - Xserve G5 Dual Processor

The following is the software installed on the machines for testing of the TOE:

- Mac OS X v10.3.6 with Common Criteria Tools Package
- Mac OS X Server v10.3.6 with Common Criteria Tools Package

3. Security Policy

The Mac OS X provides the following five security functions:

- Security Audit
- User Data Protection
- Identification and Authentication
- Security Management
- Protection of the TOE Security Functions

3.1 Security Audit

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

3.1.1 Audit Generation

Mac OS X maintains all audit records in one or more audit files. The audit files are managed by the audit daemon. The audit daemon is called by both kernel-mode and user-mode processes to add audit records to the audit trail. When an audit record is created, the time stamp on the audit record is retrieved from the time daemon described in Section 3.5.4. The audit files are protected by discretionary access control (DAC) so that only administrators can access the audit trail.

Table 2 is a list of auditable events for which audit records may be generated.

Table 2. Auditable Events

Event
Start-up and shutdown of audit functions
Reading of information from the audit records
Unsuccessful attempts to read information from the audit records
All modifications to the audit configuration that occur while the audit collection functions are operating
Actions taken due to exceeding a threshold
Actions taken due to the audit storage failure
All requests to perform an operation on an object covered by the SFP (The identity of the object will also be included as part of the audit record)
Rejection or acceptance by the TSF of any tested secret All use of authentication mechanism
All use of the user identification mechanism, including the identity provided during successful attempts (The origin of the attempt will also be included as part of the audit record)
Success and failure of binding user security attributes to a subject

All modifications of the values of security attributes
Modifications of the default setting of permissive or restrictive rules
All modifications of the initial value of security attributes
All modifications to the values of TSF data (The new value of the TSF data will also be included in the audit record)
All attempts to revoke security attributes
Modifications to the group of users that are part of a role
Every use of the rights of a role (The role and the origin of the request will also be included in the audit record)
Execution of the tests of the underlying machine and the results of the test
Changes to the time

The following information is contained within each audit record:

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

The administrator has two options when configuring how the audit trail will be managed. The administrator can specify a rotation scheme among the audit files. If the last in a series of audit files is full, Mac OS X will go back to the first audit file to start writing and overwrite any previously recorded data. The second option is once all audit files are filled, Mac OS X will halt. When the audit trail reaches 80% capacity, a message is written to the syslog to alert the administrator that the audit trail is reaching capacity.

3.1.2 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 graphical 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 and success or failure of the audit event.
- *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.

3.2 User Data Protection

The Mac OS X enforces a discretionary access control (DAC) policy on processes acting on the behalf of users, files, directories, named pipes, symbolic links, unnamed pipes, shared memory segment, processes, SysV/Posix semaphores, notifications, BSD locks, at jobs, crontab files, and print queue entries, and all operations among subjects and objects covered by the DAC policy. The DAC policy refers to a policy that allows authorized users and authorized administrators to control access to objects on the basis of individual user identity or membership in a group. The user data protection security function also provides residual information protection. This section describes both aspects of the security function.

3.2.1 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.

3.2.2 DAC Attributes

DAC decisions are based upon the attributes of subjects and objects. The DAC relevant attributes of a subject are:

- Real UID – UID established at logon
- Saved UID – UID saved when a setuid performed
- Saved GID - GID saved when a setgid performed
- PID – Process identifier
- Effective UID – Currently active UID
- Real GID - GID established at logon
- Effective GID - Currently active GID
- Group list – List of groups to which user belongs
- Process umask – Default permissions

The DAC-specific attributes of the named objects depend upon the type of object. The following list identifies the object types followed by their associated DAC attributes.

- File System Objects (files, directories, named pipes, symbolic links, unnamed pipes)
  - Owner
  - Owning group
  - Permission bits
- Process Objects
  - Real UID
  - Effective UID

  - Saved UID
  - PID
  - Real GID
- Interprocess Communication (IPC) Mechanisms (shared memory segments, SysV/Posix semaphores, notifications, BSD locks)
  - Owner's UID
  - Owning GID
  - Creator's UID
  - Creator's GID
  - Permission bits
- Non-kernel objects (at jobs, crontab files, and print queue entries)
  - Owner

3.2.3 DAC Algorithm

DAC checks are made using either permissions bit checks or simple comparisons. This section describes how permission bit checking occurs. Simple comparisons are not described as they are straightforward and will be identified when they are used.

Permission bits divide permissions into three categories and users into three relative groups. The three categories of permissions are read, write, and execute. They are denoted as "r" for read, "w" for write, and "x" for execute. The three relative groups are the owner of the file, the owner's group, and every other user. When a check is performed against the permission bits, the owner bits are checked first, followed by the group bits, and concluded with the other bits. The first set of permissions matched is the permission granted.

3.2.4 DAC Policies

3.2.4.1 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. In order to access a file system object, a subject must pass the permissions bit check. Additionally, in order to read data from an object with a pathname, a subject must have execute/search access to each directory in an object's pathname, and read access to the file. In order to write data into an object, a subject must have execute/search access to each directory in the pathname, and write access to the file. Directory objects have a special bit, called a *sticky bit*. When the sticky bit is set, only the owner of the directory may delete anything within the directory.

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

Access checks are performed slightly differently on symbolic links. The access checks are stored on the pathname stored in the symbolic link, not on the link itself.

3.2.4.2 Processes

A process can always read its own attributes. It can change its real UID and effective UID if it is root (UID=0) or if the requested value is equal to its real or saved UID. A process can also change its real GID, effective GID, and group list if it is root.

There is one instance where processes access one another. One process can always read the attributes of another process if it can name the target process using its PID.

3.2.4.3 IPC Mechanisms

Each IPC object includes a structure that contains: (a) the creator's effective UID and GID, (b) an owner effective UID and GID, and (c) a set of permission bits for creator and owner, creator's group and owner's group, and others. The owner UID and GID is initially set to the creator's UID and GID. Access is granted according to the permission bits.

3.2.4.4 Non-Kernel Objects

All non-kernel objects, at jobs, crontab files, and print queue entries, have their owner set to the user that submitted the request. All access is checked against the owner of the object. Only the owner or root may access any non-kernel object.

3.2.4.5 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.

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 affect 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.

3.2.5 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.

3.3 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 the user identifier, group memberships, authentication data, and security-relevant roles. The following features are provided by Mac OS X: logon, user attribute management, and password and account management. Each of these is described in the following sections.

3.3.1 Logon Process

The logon process ensures that before a user uses any TSF-mediated functions, that user is identified and authenticated to Mac OS X. Users can either log onto the system locally or via a network service. When users log on at a local terminal, they are presented with a graphical interface requesting their user name and password. The user name is echoed back to the screen while the password is obscured from the user with dots. Authentication can take place using local passwords.

When a user requests some network services, the user must be authenticated. Network services either trust the identity of the client process or perform authentication on their own. The client process ssh passes an identity, which is trusted on the remote server, and is not required to re-authenticate unless the authentication fails; if authentication fails, the user is prompted for a password. Other network services/protocols in the evaluated configuration are unauthenticated. Those unauthenticated services/protocols are: TCP/IP, NFS, and DNS. The TCP/IP protocol relies on higher level services to perform authentication, NFS performs access checks based on the credentials of the requesting user, and DNS does not require authentication as it simply responds to name requests.

3.3.2 User Subject Binding

A process is used to associate users with subjects within the TOE. A process is an abstraction for a running program. A process's 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 and file descriptors. The following security relevant attributes are associated with a process:

- Real UID
- Saved UID
- Saved group identifier (GID)
- PID
- Effective UID
- Real GID
- Effective GID
- Group list
- Process umask.

There are three ways that a new subject can be created: a logon, a fork or exec, or a batch job is activated. 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 or exec, the calling process is created and 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.

In most cases, the security attributes of a subject are derived from its creator; however, using a setuid or setgid program, or issuing the su command are exceptions. When executing a setuid or setgid program, the effective UID or GID, respectively, of the process will be changed. The new effective ID is taken from the file owner of the file being executed. Successful execution of the su program allows the subject to take on the targeted identity.

3.3.3 Password and Account Management

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

- User name
- User ID
- Password
- Groups

The group information indicates if a user is part of the admin group. Being a member of the admin group makes a user an administrator, the only defined role on the system. There are two cases where users are permitted to perform certain role-like functions; those are users authorized by the DAC Policy to modify object security attributes, and users are authorized to modify their own authentication data. In these two special cases, users do not have to be the member of any group. All information is stored in files protected by DAC. Additionally, the password data is stored in encrypted format. Only the administrator has access to update the user account information. Changes to the authentication database take affect the next time a user logs on. 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 affect.

Passwords are initially assigned by the administrator and can be changed at any time by the administrator. Users are permitted to change their own passwords by supplying their current password and selecting a new password. The new password must have at least five characters. All characters are significant.

3.4 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.

3.5 Protection of the TOE Security Functions

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.

3.5.1 Abstract Machine Testing

Since hardware and firmware are included in the TOE, both have tests to demonstrate their security features operate as claimed. 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. Apple also provides a set of hardware diagnostics to test that the hardware is operating correctly. The tests are run during system start-up and are called power-on self tests (POST).

3.5.2 Reference Mediation

The MAC OS X architecture is based on 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.

3.5.3 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 mode (i.e., kernel-mode) provided by the evaluated processor architectures. A system call trap is 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., printing) and Administrator tools 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.

3.5.4 Time

The system time is maintained and exported by the kernel using the time daemon. This system time is used in the audit trail to maintain an accurate accounting of when audit events occurred. Additionally the network time daemon and network time protocol (NTP) are used to synchronize clocks among networked computers.

4. Assumptions and Clarification of Scope

This section describes the security aspects of the environment in which Mac OS X is expected to operate.

4.1 Usage Assumptions

This section contains assumptions regarding the security environment and the intended usage of Mac OS X. Mac OS X 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. Mac OS X is intended for use in areas that have physical control and monitoring. Table 3 identifies the specific conditions that are assumed to exist in an environment where Mac OS X is employed.

Table 3. Assumptions

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

5. Architectural Information

Mac OS X is built upon an open source, UNIX-based core called Darwin. Darwin integrates a number of technologies, including the Mach 3.0 kernel, operating system services based on BSD UNIX, and networking facilities.

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 kernel augments standard virtual memory semantics with the abstraction of memory objects. This enables Mac OS X to manage separate application environments simultaneously.

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 configure protocol stacks automatically and easily monitor and modify network traffic. At the data-link and network layers, they can also receive notifications of asynchronous events from device drivers.

While Darwin offers support for multiple file systems, only the HFS+ filesystem is supported in the evaluated configuration. Darwin also supplies the following advanced functionality:

- Preemptive and cooperative multitasking via the Mach kernel
- Symmetric multiprocessing (SMP) augmented by support for multi-threading
- Real-time support guaranteeing low-latency access to processor resources for time-sensitive media applications

While Mac OS X supports a wide range of protocols and network services, only TCP/IP and the NFS (Network Filesystem), DNS (Domain Name Service), and SSH (Secure Shell) services are supported in the evaluated configuration.

Figure 1 provides an overview of the architecture of Mac OS X.

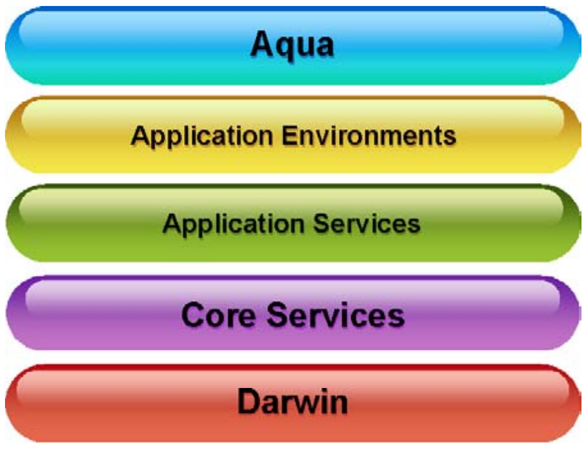

Figure 1. Mac OS X Architecture

The system components perform the following functions:

- *Aqua* – This provides the graphical interface to both users and administrators.

- *Application Environments* – 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 five application environments provided:

  1. Carbon – a set of programming interfaces derived from earlier Mac OS APIs that has 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. Classic - provides a compatibility environment that allows legacy applications to execute on Mac OS X. The Classic environment is essentially a virtual machine that allows legacy applications to execute in a simulated environment where older APIs are translated to Mac OS X APIs. This environment is not included in the evaluated configuration since older versions of the operating system are not included in the evaluation.
  4. Java – provides development and runtime environments and an application framework that allow applications developed in Java to execute on Mac OS X.
  5. BSD Commands – a native implementation of a command line BSD command environment.

- *Application Services* -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.

- *Core Services* - The Core Services 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 kernel environment is the lowest layer of system software. The kernel environment provides essential operating-system functionality to the layers above it, such as:
  - Preemptive multitasking
  - Advanced virtual memory with memory protection and dynamic memory allocation
  - Symmetric multiprocessing
  - Multi-user access
  - File systems based on VFS (Virtual File System)
  - Device drivers
  - Networking
  - Basic threading packages

6. Documentation

The following is a list of the evaluation evidence, each of which was issued by the developer (and sponsor).

6.1 Design documentation

Document	Version	Date
Component Specifications:		
Administrative Tools	7	27 December 2004
Daemons	1.06	29 November 2004
Filesystem	2.06	8 October 2004
Hardware	3	13 December 2004
I/O Kit	5	11 January 2005
BSD/System V Inter-Process Communication	8	24 November 2004
Memory Management	4	8 June 2004
Microkernel	12	2 September 2004
Networking	6	19 November 2004
Process Management	3.6	6 September 2004
Security Framework	1.3	12 May 2004
User Interface	1.0	29 April 2004
SSH	0.05	16 December 2004
DNS	0.02	15 March 2004
High Level Design Overview	5	14 December 2004

6.2 Guidance documentation

Document	Version	Date
Common Criteria Configuration and Administration	1.0	12 January 2005

6.3 Configuration Management documentation

Document	Version	Date
Apple Configuration Management Plan	0.5	16 December 2004

6.4 Delivery and Operation documentation

Document	Version	Date
Apple Delivery Procedures	0.5	23 November 2004

6.5 Life Cycle Support documentation

Document	Version	Date
Apple Life Cycle Manual	0.4	20 December 2004

6.6 Test documentation

Document	Version	Date
Apple Computer Mac OS X Version 10.3.6 Test Plan	0.6	27 December 2004
Component Test Specifications:		
Admin Tools (GUI)	2	30 November 2004
Admin Tools (CLI) Test	2	19 November 2004
Daemons Test	3	13 November 2004
File System Test	7	13 November 2004
NFS Filesystem Test	3	12 November 2004
Hardware Test	1	13 November 2004
I/O Kit Test	3	22 November 2004
IPC Test	4	12 November 2004
Memory Management Test	3	11 November 2004
Microkernel Test	5	19 November 2004
Networking Test	4	12 November 2004
Process Management Test	6	27 December 2004
Security Framework Test	3	12 November 2004
User Networking Test	3	13 November 2004

6.7 Vulnerability Assessment documentation

Document	Version	Date
Apple Computer Mac OS X Version 10.3.6 Strength of Function Analysis	2.0	21 December 2004
Apple Computer Mac OS X and Mac OS X Server Vulnerability Analysis	0.4	19 November 2004

6.8 Security Target

Document	Version	Date
Apple Computer Mac OS X v10.3.6 and Mac OS X Server v10.3.6 Security Target	1.0	13 December 2004

7. IT Product Testing

This section describes the testing efforts of the vendor and the Evaluation Team.

7.1 Vendor Testing

The vendor's approach to testing the Mac OS X security functions is based on testing the components identified in the design evidence. These tests demonstrate the security-relevant behavior of Mac OS X at the interfaces identified in the functional specification and defined in the high-level design. The functional and high-level designs are presented as a set of component specifications. Each such specification describes the purpose and functionality of the component, identifies and describes its interfaces that are part of the TSFI, maps the interfaces to security functions, and describes the test cases for each interface. Testing was therefore based on exercising each of the TSFIs and demonstrating that the specified security behavior (in terms of checks, effects and errors) has been implemented.

The vendor's tests consisted primarily of automated test suites that exercise the majority of the TSFIs. These were supplemented by manual tests to exercise aspects of the administrator and user GUIs and the CLI.

The vendor addressed test depth by testing the security checks and effects of each interface with detailed knowledge about the check or effect. Specific knowledge of security checks and effects is identifiable in the component specifications. In addition, each test procedure description explains how it achieves depth in its Variation Overview section.

Tests were organized according to component, and to security-relevant interface within each component, with individual test cases corresponding to those described in the component specifications. Each test procedure specification also includes instructions for running its tests, elaborating on the general instructions provided in the Test Plan. Each test procedure specification also detailed the expected results from running the tests.

An automated test suite was used to execute the specified test cases and report the results for each test. Test results were then viewed using a graphical user interface by opening the results file in a web browser and navigating to the component's test result pages.

7.2 Evaluator Testing

The Evaluation Team executed all automated vendor test procedures per the evaluated configuration as described in the Apple Computer Mac OS X version 10.3.6 Test Plan. The tests were run against the TOE versions identified below:

G5 Processor:

- PowerMac

G4 Processor:

- iMac
- PowerBook
- PowerMac
- Xserve

The Evaluation Team also developed and executed its own set of tests in order to ensure that security policies were properly enforced. Tests were developed for the Audit, Identification & Authentication, and TOE Protection Mechanisms security functions. The Evaluation Team's goal was to gain additional confidence in Apple's test results and to provide independent confirmation of those results. No Independent Tests were developed for the User Data Protection and Security Management security functions. The Evaluation Team determined that the test procedures used by the vendor were sufficient.

All tests were successful, with the actual results matching the expected results.

8. Evaluated Configuration

The evaluated configuration consisted of the components identified below:

- *Hardware Component:*
  - Mac OS X version 10.3.6: eMac G4, iMac G3, iMac G4, iMac G5, iBook G3, iBook G4, PowerBook G3, PowerBook G4, Power Mac G3, Power Mac G4 Cube, Power Mac G4 (single processor), Power Mac G4 Dual Processor, Power Mac G5 (single processor), Power Mac G5 Dual Processor
  - Mac OS X Server version 10.3.6: Power Mac G4 (single processor), Power Mac G4 Dual Processor, Power Mac G5 (single processor), Power Mac G5 Dual Processor, Xserve G4 (single processor), Xserve G4 Dual Processor, Xserve G5 (single processor), Xserve G5 Dual Processor

- *Software Component:*
  - Mac OS X v10.3.6 with Common Criteria Tools Package
  - Mac OS X Server v10.3.6 with Common Criteria Tools Package

To obtain additional details about the evaluated configuration, please refer to the *Common Criteria Configuration and Administration v1.0.*

The Validator would like to note that the Evaluation Team has not conducted tests on the G3 systems. The developer included the G3 systems in the set of configurations and has provided full test results for each claimed configuration. The Evaluation Team reviewed the test results for all claimed configurations and confirmed that all actual results matched the expected results.

The G3 system is a subset of the G4, therefore the Evaluation Team did not deem it necessary to conduct team testing on a G3 system.

9. Results of the Evaluation

The Mac OS X satisfies all of the EAL3 assurance requirements against which it was evaluated. The EAL3 assurance requirements include the following:

Table 4. EAL3 Assurance Components

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

The Security Target provides a detailed description of how Mac OS X meets each of the listed components.

10. Validation Comments/Recommendations

The Validator determined that the evaluation and all of its activities were performed in accordance with the CC, the CEM and CCEVS practices.

The Validator agrees that the CCTL presented appropriate rationales to support the Evaluation Results presented in Section 5 of the ETR, Part 1, and the Conclusions presented in Section 6 of the ETR, volume 1.

The Validator, therefore, concludes that the evaluation and the Pass results for the TOE identified below is complete and correct:

Mac OS X v10.3.6 and Mac OS X Server v10.3.6

The Validator would like to acknowledge that the TOE is a subset of the Mac OS X product as defined in the administrator guidance. 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.

Services outside the evaluation configuration include: e-mail services; web server services; remote apple events; print sharing services; file sharing services; and classic programming support (Old Macintosh OS compatibility support). Also, Mac OS X Server contains a watchdog timer to restart services and provide stability; however, this timer is disabled in the evaluated configuration.

In addition, the open source UNIX-based core that Mac OS X is built upon only supports the HFS+ filesystem in the evaluated configuration. Furthermore, while Mac OS X supports a wide range of protocols and network services, only TCP/IP and the NFS (Network Filesystem), DNS (Domain Name Service), and SSH (Secure Shell) services are supported in the evaluated configuration.

11. Security Target

The Security Target is entitled, Apple Computer Mac OS X v10.3.6 and Mac OS X Server v10.3.6 Security Target, version 1.0, 13 December 2004.

12. List of Acronyms

API	Application Program Interface
BSD	Berkeley Software Distribution
CAPP	Controlled Access Protection Profile
CCEVS	Common Criteria Evaluation and Validation Scheme
CCIMB	Common Criteria Interpretations Management Board
CCTL	Common Criteria Testing laboratory
CEM	Common Evaluation Methodology
CLI	Command Line Interface
DAC	Discretionary Access Control
DNS	Domain Name Service
EAL3	Evaluation Assurance Level 3
GID	Saved Group Identifier
GUI	Graphical User Interface
HFS	Hierarchical Filesystem
IPC	Interprocess Communication
NFS	Network Filesystem
NIAP	National Information Assurance Partnership
NKE	Network Kernel Extension
NTP	Network Time Protocol
PID	Process Id
POST	Power-on Self Tests
SAIC	Science Applications International Corporation
SFP	Security Function Policy
SMP	Symmetric Multiprocessing
SSH	Secure Shell
TOE	Target of Evaluation
TSF	TOE Security Function
TSFIs	TOE Security Function Interfaces
UID	User Identifier
VFS	Virtual Filesystem

13. Bibliography

The following documents were used in compiling this Validation Report:

- Common Criteria for Information Technology Security Evaluation, Version 2.1, August 1999:
    - Part 1: Introduction and General Model
    - Part 2: Security Functional Requirements
    - Part 3: Annexes
    - Part 4: Security Assurance Requirements

- Common Evaluation Methodology for Information Technology Security:
    - Part 1: Introduction and General Model, Version 0.6, 11 January 1997
    - Part 2: Evaluation Methodology, Version 1.0, August 1999

- Apple Computer Mac OS X v10.3.6 and Mac OS X Server v10.3.6 Security Target, version 1.0, 13 December 2004

- Evaluation Technical Report for Apple Computer Mac OS X v10.3.6 and Mac OS X Server v10.3.6 Part 2 (SAIC and Apple Proprietary), Version 1.0, 27 December 2004

- Evaluation Technical Report for Apple Computer Mac OS X v10.3.6 and Mac OS X Server v10.3.6 Part 1 (Non-Proprietary), Version 1.0, 27 December 2004

- Evaluation Team Test Plan for Apple Computer Mac OS X 10.3.6 and Mac OS X Server v10.3.6 – ETR Part 2 Supplement (SAIC and Apple Proprietary), Version 1.0, 27 December 2004