NAME

mac - Mandatory Access Control

SYNOPSIS

options MAC

DESCRIPTION

Introduction

The Mandatory Access Control, or MAC, framework allows administrators to finely control system security by providing for a loadable security policy architecture. It is important to note that due to its nature, MAC security policies may only restrict access relative to one another and the base system policy; they cannot override traditional UNIX security provisions such as file permissions and superuser checks.

Currently, the following MAC policy modules are shipped with FreeBSD:

Name	Description	Labeling	Load time
mac_biba(4)	Biba integrity policy	yes	boot only
mac_bsdextended(4)	File system firewall	no	any time
mac_ddb(4)	ddb(4) interface restrictions	no	any time
mac_ifoff(4)	Interface silencing	no	any time
mac_ipacl(4)	IP Address access control	no	any time
mac_lomac(4)	Low-Watermark MAC policy	yes	boot only
mac_mls(4)	Confidentiality policy	yes	boot only
mac_ntpd(4)	Non-root NTP Daemon policy	no	any time
mac_partition(4)	Process partition policy	yes	any time
mac_portacl(4)	Port bind(2) access control	no	any time
mac_priority(4)	Scheduling priority policy	no	any time
mac_seeotheruids(4)	See-other-UIDs policy	no	any time
mac_test(4)	MAC testing policy	no	any time

MAC Labels

Each system subject (processes, sockets, etc.) and each system object (file system objects, sockets, etc.) can carry with it a MAC label. MAC labels contain data in an arbitrary format taken into consideration in making access control decisions for a given operation. Most MAC labels on system subjects and objects can be modified directly or indirectly by the system administrator. The format for a given policy’s label may vary depending on the type of object or subject being labeled. More information on the format for MAC labels can be found in the maclabel(7) man page.

MAC Support for UFS2 File Systems

By default, file system enforcement of labeled MAC policies relies on a single file system label (see *MAC Labels*) in order to make access control decisions for all the files in a particular file system. With some policies, this configuration may not allow administrators to take full advantage of features. In order to enable support for labeling files on an individual basis for a particular file system, the "multilabel" flag must be enabled on the file system. To set the "multilabel" flag, drop to single-user mode and unmount the file system, then execute the following command:

tunefs -l enable *filesystem*

where *filesystem* is either the mount point (in fstab(5)) or the special file (in */dev*) corresponding to the file system on which to enable multilabel support.

Policy Enforcement

Policy enforcement is divided into the following areas of the system:

File System
File system mounts, modifying directories, modifying files, etc.

KLD
Loading, unloading, and retrieving statistics on loaded kernel modules

Network
Network interfaces, bpf(4), packet delivery and transmission, interface configuration (ioctl(2), ifconfig(8))

Pipes
Creation of and operation on pipe(2) objects

Processes
Debugging (e.g. ktrace(2)), process visibility (ps(1)), process execution (execve(2)), signalling (kill(2))

Sockets
Creation of and operation on socket(2) objects

System
Kernel environment (kenv(1)), system accounting (acct(2)), reboot(2), settimeofday(2), swapon(2), sysctl(3), nfsd(8)-related operations

VM

mmap(2)-ed files

Setting MAC Labels

From the command line, each type of system object has its own means for setting and modifying its MAC policy label.

Subject/Object	Utility
File system object	setfmac(8), setfsmac(8)
Network interface	ifconfig(8)
TTY (by login class)	login.conf(5)
User (by login class)	login.conf(5)

Additionally, the su(1) and setpmac(8) utilities can be used to run a command with a different process label than the shell's current label.

Programming With MAC

MAC security enforcement itself is transparent to application programs, with the exception that some programs may need to be aware of additional errno(2) returns from various system calls.

The interface for retrieving, handling, and setting policy labels is documented in the mac(3) man page.

SEE ALSO

mac(3), mac_biba(4), mac_bsdextended(4), mac_ddb(4), mac_ifoff(4), mac_ipacl(4), mac_lomac(4), mac_mls(4), mac_none(4), mac_ntpd(4), mac_partition(4), mac_portacl(4), mac_priority(4), mac_seeotheruids(4), mac_stub(4), mac_test(4), login.conf(5), maclabel(7), getfmac(8), getpmac(8), setfmac(8), setpmac(8), mac(9)

Mandatory Access Control, *The FreeBSD Handbook*, https://docs.FreeBSD.org/en/books/handbook/mac/.

HISTORY

The **mac** implementation first appeared in FreeBSD 5.0 and was developed by the TrustedBSD Project.

AUTHORS

This software was contributed to the FreeBSD Project by Network Associates Labs, the Security Research Division of Network Associates Inc. under DARPA/SPAWAR contract N66001-01-C-8035 ("CBOSS"), as part of the DARPA CHATS research program.

BUGS

While the MAC Framework design is intended to support the containment of the root user, not all attack

channels are currently protected by entry point checks. As such, MAC Framework policies should not be relied on, in isolation, to protect against a malicious privileged user.