NAME

mac_mls - Multi-Level Security confidentiality policy

SYNOPSIS

To compile MLS into your kernel, place the following lines in your kernel configuration file:

```
options MAC
options MAC_MLS
```

Alternately, to load the MLS module at boot time, place the following line in your kernel configuration file:

```
options MAC
```

and in loader.conf(5):

```
mac_mls_load="YES"
```

DESCRIPTION

The **mac_mls** policy module implements the Multi-Level Security, or MLS model, which controls access between subjects and objects based on their confidentiality by means of a strict information flow policy. Each subject and object in the system has an MLS label associated with it; each subject's MLS label contains information on its clearance level, and each object's MLS label contains information on its classification.

In MLS, all system subjects and objects are assigned confidentiality labels, made up of a sensitivity level and zero or more compartments. Together, these label elements permit all labels to be placed in a partial order, with confidentiality protections based on a dominance operator describing the order. The sensitivity level is expressed as a value between 0 and 65535, with higher values reflecting higher sensitivity levels. The compartment field is expressed as a set of up to 256 components, numbered from 1 to 256. A complete label consists of both sensitivity and compartment elements.

With normal labels, dominance is defined as a label having a higher or equal active sensitivity level, and having at least all of the same compartments as the label to which it is being compared. With respect to label comparisons, "lower" is defined as being dominated by the label to which it is being compared, and "higher" is defined as dominating the label to which it is being compared, and "equal" is defined as both labels being able to satisfy the dominance requirements over one another.

Three special label values exist:

Label	**Comparison**
mls/low	dominated by all other labels
mls/equal	equal to all other labels
mls/high	dominates all other labels

The "mls/equal" label may be applied to subjects and objects for which no enforcement of the MLS security policy is desired.

The MLS model enforces the following basic restrictions:

- Subjects may not observe the processes of another subject if its clearance level is lower than the clearance level of the object it is attempting to observe.

- Subjects may not read, write, or otherwise observe objects without proper clearance (e.g. subjects may not observe objects whose classification label dominates its own clearance label)

- Subjects may not write to objects with a lower classification level than its own clearance level.

- A subject may read and write to an object if its clearance level is equal to the object's classification level as though MLS protections were not in place.

These rules prevent subjects of lower clearance from gaining access information classified beyond its clearance level in order to protect the confidentiality of classified information, subjects of higher clearance from writing to objects of lower classification in order to prevent the accidental or malicious leaking of information, and subjects of lower clearance from observing subjects of higher clearance altogether. In traditional trusted operating systems, the MLS confidentiality model is used in concert with the Biba integrity model (mac_biba(4)) in order to protect the Trusted Code Base (TCB).

Label Format

Almost all system objects are tagged with an effective, active label element, reflecting the classification of the object, or classification of the data contained in the object. In general, object labels are represented in the following form:

mls/*grade*:*compartments*

For example:

```
mls/10:2+3+6
mls/low
```

Subject labels consist of three label elements: an effective (active) label, as well as a range of available labels. This range is represented using two ordered MLS label elements, and when set on a process, permits the process to change its active label to any label of greater or equal integrity to the low end of the range, and lesser or equal integrity to the high end of the range. In general, subject labels are represented in the following form:

> mls/*effectivegrade*:*effectivecompartments*(*lograde*:*locompartments*-
> *higrade*:*hicompartments*)

For example:

```
mls/10:2+3+6(5:2+3-20:2+3+4+5+6)
mls/high(low-high)
```

Valid ranged labels must meet the following requirement regarding their elements:

> *rangehigh* >= *effective* >= *rangelow*

One class of objects with ranges currently exists, the network interface. In the case of the network interface, the effective label element references the default label for packets received over the interface, and the range represents the range of acceptable labels of packets to be transmitted over the interface.

Runtime Configuration

The following sysctl(8) MIBs are available for fine-tuning the enforcement of this MAC policy.

security.mac.mls.enabled	Enables the enforcement of the MLS confidentiality policy. (Default: 1).
security.mac.mls.ptys_equal	Label pty(4)s as "mls/equal" upon creation. (Default: 0).
security.mac.mls.revocation_enabled	Revoke access to objects if the label is changed to a more sensitive level than the subject. (Default: 0).

IMPLEMENTATION NOTES

Currently, the **mac_mls** policy relies on superuser status (suser(9)) in order to change network interface MLS labels. This will eventually go away, but it is currently a liability and may allow the superuser to bypass MLS protections.

SEE ALSO

mac(4), mac_biba(4), mac_bsdextended(4), mac_ddb(4), mac_ifoff(4), mac_lomac(4), mac_none(4), mac_partition(4), mac_portacl(4), mac_seeotheruids(4), mac_test(4), maclabel(7), mac(9)

HISTORY

The **mac_mls** policy module 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 Laboratories, 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.