Linux Security Modules
Is there more than SELinux?

May 2025 - OLUG

By Aaron Grothe

Introduction

If you have questions/comments please feel free to ask them anytime. You don't have to hold them until the end of the talk.

If there are other resources similar to these that you think might be useful to people please let the group know.

Hopefully this will be an interactive and productive session.

Links for the topics are at the end of the presentation.

Disclaimer

When you start playing around with Linux Security Modules, it may result in a unbootable system.

Incremental changes, and backups are your friend.

You have been warned

How Many Linux Security Modules exist?

- How many Linux Security Modules are there?

I'll get you started with the two big ones

AppArmor
SELinux

How about the rest???

List of Linux Security Modules

1. SELinux (Security-Enhanced Linux)
2. AppArmor
3. Capabilities
4. Yama (Yet Another Mandatory Access Control Architecture)
5. Smack (Simplified Mandatory Access Control Kernel)
6. TOMOYO Linux
7. LoadPin
8. Landlock
9. Integrity Policy Enforcement (IPE)
10. SafeSetID
11. Lockdown

Size of Linux Security Modules

This is a rough approximation of the size of each of the security modules

Using Linux-6.14.5

cd security
find <lsm> -type f -print | xargs wc -l

This is just a quick approximation of each

Size of Linux Security Modules

1. SELinux (Security-Enhanced Linux) - 28454
2. AppArmor - 21303
3. Capabilities - Not a separate system
4. Yama (Yet Another Mandatory Access Control Architecture) - 504
5. Smack (Simplified Mandatory Access Control Kernel) - 9687
6. TOMOYO Linux - 12622
7. LoadPin - 491
8. Landlock - 4700
9. Integrity Policy Enforcement (IPE) - 3496
10. SafeSetID - 735
11. Lockdown - 221

Break into Categories

We'll break the LSMs down into the following categories

- General purpose MAC systems - Major Systems
- Additional capabilities
- Future

Which LSMs ship with Distros

To see which LSMs are available on your system

% cat /sys/kernel/security/lsm

Ubuntu 24.04

lockdown,capability,landlock,yama,apparmor

Ubuntu 25.04

lockdown,capability,landlock,yama,apparmor,ima,evm

Which LSMs ship with Distros

Rocky 9.4 - RHEL 9.4

lockdown,capability,yama,selinux,bpf

Debian 12.10

lockdown,capability,landlock,yama,apparmor,tomoyo,bpf

Debian Testing

lockdown,capability,landlock,yama,apparmor,tomoyo,bpf,ipe,ima,evm

General Purpose MAC Systems

What is MAC (Mandatory Access Control)?

On a typical computer the owner of a file or resource has full capabilities and can control access to it.

E.g. chmod 0777 ~/file.txt

This gives anybody on the system the capability to read/write/execute the file

This is Discretionary Access Control (DAC)

General Purpose MAC Systems

What is MAC (Mandatory Access Control)?

Mandatory Access Control adds an additional layer on it. E.g. with MAC you set a system so even the owner of a file can't make changes to it that would violate the MAC

This is one of the major differences between a regular O/S and one that is certified under EAL-4 for government use

General Purpose MAC Systems

The following are the General Purpose MAC systems

- SELinux - Used by RHEL systems mostly
- AppArmor - Used by Ubuntu/Debian
- Smack - Embedded systems (was part of Automotive Grade Linux, replaced by SELinux)
- Tomoyo Linux - behavior-based MAC system, trainable system

Theoretically - you can stack Major LSMs, but you're probably going to have a bad day trying

General Purpose MAC Systems

There have been a lot of papers/presentations about each of the above.

So we'll talk more about the second set of LSMs

These have come a long way. In the old days if you hit any article on howtoforge and half the RHEL articles had as their first or second step: disable selinux

Additional LSMs

- Yama (Yet Another Mandatory Access Control Architecture)
- LoadPin
- Landlock
- Integrity Policy Enforcement (IPE)
- SafeSetID
- Lockdown

Yama (Yet Another Mandatory Access Control Architecture)

Yama is an LSM that adds onto the DAC of Linux

It does this primarily through locking down access through the ptrace call.

On a regular system all of a user's processes can see the memory and running state of other processes. E.g. if Chromium was compromised it could look at memory/running state for the user's bash shells, ssh sessions, etc.

Yama (Yet Another Mandatory Access Control Architecture)

Yama has 4 running states

0 - classic ptrace: doesn't change any behaviour
1 - restricted ptrace: can only get descendants information
2 - admin-only attach: process needs cap_sys_ptrace priv to attach
3 - no attach: no process can use ptrace, once set this can't be changed

To see what state your system is running in do the following

% cat /proc/sys/kernel/yama/ptrace_scope

Yama (Yet Another Mandatory Access Control Architecture)

To change value on running system

% sudo sysctl -w kernel.yama.ptrace_scope=<value>

Note on modern Ubuntu and Fedora this is now set to 1, so you might not need to make any change

Yama (Yet Another Mandatory Access Control Architecture)

To set at startup

% sudo nano /etc/sysctl.conf

Add the following line

kernel.yama.ptrace_scope = <value>

Can dynamically change via

% sudo sysctl -p

Yama (Yet Another Mandatory Access Control Architecture)

To set at startup

% sudo nano /etc/sysctl.conf

Add the following line

kernel.yama.ptrace_scope = <value>

Can dynamically change via

% sudo sysctl -p

Yama (Yet Another Mandatory Access Control Architecture).

Yama is being used by multiple systems and some cloud vendors to increase security

It can have some impact, but its behaviour by default is pretty safe

Loadpin

Loadpin was initially developed by Google to protect ChromeOS

Quite simply all the kernel (modules, firmware, kexec images, security policy) most all come from the same filesystem used for loading.

Idea is that if you have a read-only filesystem you can set it so it harder to muck with the system. APTs, rootkits, have a harder time getting into your system

Loadpin

While you can turn this on and off dynamically it is dangerous and not recommended. You should boot into the system with it enabled preferably.

In /etc/default/grub

Add the following to to the menu

GRUB_CMDLINE_LINUX_DEFAULT="... loadpin.enforce=1 ..."

Loadpin

While you can turn this on and off dynamically it is dangerous and not recommended. You should boot into the system with it enabled preferably.

In /etc/default/grub

Add the following to to the menu

GRUB_CMDLINE_LINUX_DEFAULT="... loadpin.enforce=1 ..."

Loadpin.

Loadpin will probably gain additional ground in the future as immutable distributions continue to evolve.

To enable on a system where it isn't enabled, you'll need to configure the kernel and add the additional options.

Landlock

The landlock LSM allows you to write programs that can create their own security policies.

You can restrict what privileges a program can have, and these are inherited by the processes children.

E.g. you have a program that reads PDFs. You can set it so it can only access the pdf file and a temporary directory if necessary. Preventing it from accessing any other resources

This is kind of similar to the way that OpenBSD broke up its processes years ago, to reduce privs required by ssh, smtp, etc.

Landlock.

Landlock is a cool idea, the userspace tools are evolving.

Someone should write a wrapper for landlock that will allow you to easily restrict privs for programs. Something like firejail.

It would allow you to run programs with minimal changes.

This exists and is called Landrun - https://github.com/Zouuup/landrun

Lockdown

Lockdown is an LSM that restricts what the root user can do on the system

It does this in two modes

Integrity
Confidentiality

Lockdown

Integrity

- Disables loading of unsigned kernel modules
- Can't exec into another kernel unless it is signed
- Restricts access to devices such as /dev/mem, /dev/kmem
- Turns off some debugging features
- Stops changes to ACPI tables
- Stops unencrypted suspend/hibernation

Lockdown

Confidentiality - Does everything in Integrity

- Additional restrictions to /dev/mem, /dev/kmem, and /dev/port
- Prevents reading traffic on serial ports by root
- Disable access to /proc/kcore - which is image of memory
- Restrict BFP to read kernel memory directly

Lockdown

How to tell what state you're currently in

% cat /sys/kernel/security/lockdown

[none] integrity confidentiality

This says it set to none

Lockdown

While you can turn this on dynamically it is not recommended. You should boot into the system with it enabled preferably.

In /etc/default/grub

Add the following to to the menu for confidentiality

GRUB_CMDLINE_LINUX_DEFAULT="lsm=lockdown,confidentiality"

Lockdown.

You can also dynamically turn the lockdown LSM on as follows

```
sudo sh -c "echo integrity > /sys/kernel/security/lockdown"

# or

sudo sh -c "echo confidentiality > /sys/kernel/security/lockdown"
```

Once, set you have to reboot to turn it off

In the future if your system has Secureboot enabled you might transparently get lockdown enabled and activated,

SafeSetID.

SafeSetID restricts the usage of setuid, and setgid and related system calls.

Example of configuration for SafeSetID

echo "1000:1001" > /sys/kernel/security/safesetid/uid_allowlist_policy

Will allow user 1000 to switch to userid 1001

Hornet.

Right now the Hornet LSM is going through the kernel acceptance process on the Linux Kernel mailing list

It is an interesting LSM for several reasons

- Its primary author is Microsoft
- It is designed to require signatures for eBPF extended Berkeley Packet Filter programs

eBPF can do some very interesting stuff to your system. That is another talk.

Documentation

There are two primary sources of documentation for LSMs

The Linux kernel Documentation/security/lsm.rst provides a lot of good information about these

The security directory has the source code for the modules

So???

Linux Security Modules continue to evolve and improve

New modules in the future will probably tap into hardware built into new PCs such as TPM chips. There has been some work in this area

A lot of security innovations/experiments for Linux are taking place in Linux Security Modules

Summary

That's what I've got for tonight.

Thanks for listening.

Any Questions???