Linux Security Modules (LSM)

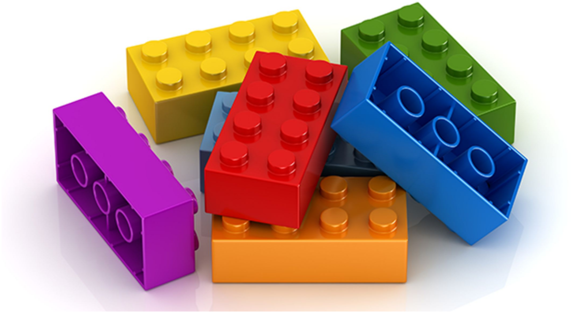
Florian Hantke, Bruno Labres, Eduardo Trevisan, Pedro Demarchi Gomes

Outline

- History
- LSM Design and Modules
- Examples
- Downsides
- Conclusion

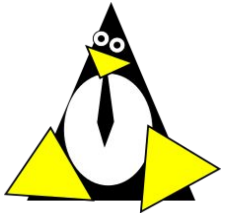
History

- Problem: Access Control Modules (ACMs) have failed to win acceptance into mainstream operating systems - security community cannot agree on one solution
- Problem: You need to patch the kernel to change the ACM
- 2001: NSA proposed to include SELinux in Linux 2.5
- Linus Torvalds rejected it seeing too many security projects in development
  - “Make it a module”



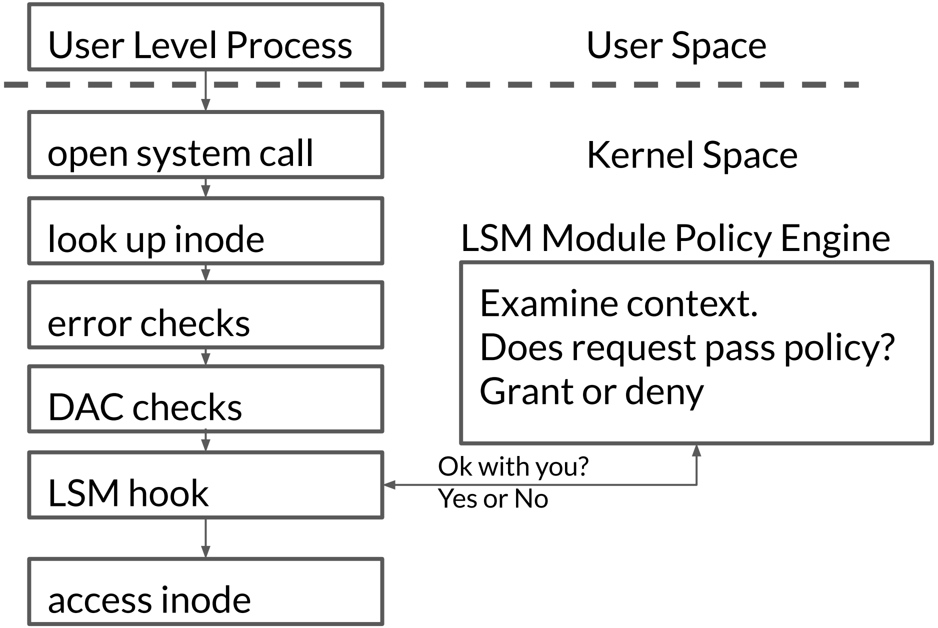
History

- Crispin Cowan et al proposed Linux Security Modules (LSM)
- LSM: Framework that allows the Linux kernel to support a variety of computer security models while avoiding favoritism toward any single security implementation.
- 2003: LSM is standard part of kernel since Linux 2.6
- AppArmor, SELinux, Smack, and TOMOYO Linux are the currently accepted modules in the official kernel.



LSM Design

- LSM uses hooks in the kernel to call module
- The Module can grant or deny access
- Access is denied when first module denys access
- Change of modules without rebuild the kernel
- LSM is initialized and modules are loaded during kernel's boot sequence



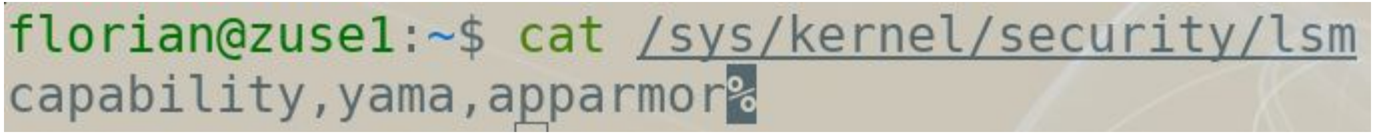
Origin Hooks

- Task Hooks (Process operations such as kill or setuid)
- Program Loading Hooks (During execve)
- Interprocess Communication Hooks (In existing ipcperms function)
- Filesystem Hooks (filesystem, inode and file)
- Network Hooks (socket-based protocols)
- Other Hooks (Kernel modules and System hooks)

LSMs

- Capabilities
- AppArmor
  - pathnames
- SELinux
  - complex
- Smack
  - simple; label based
- TOMOYO
  - end user intended; low adoption; trees of process invocation recording
- YAMA
  - miscellaneous DAC security enhancements

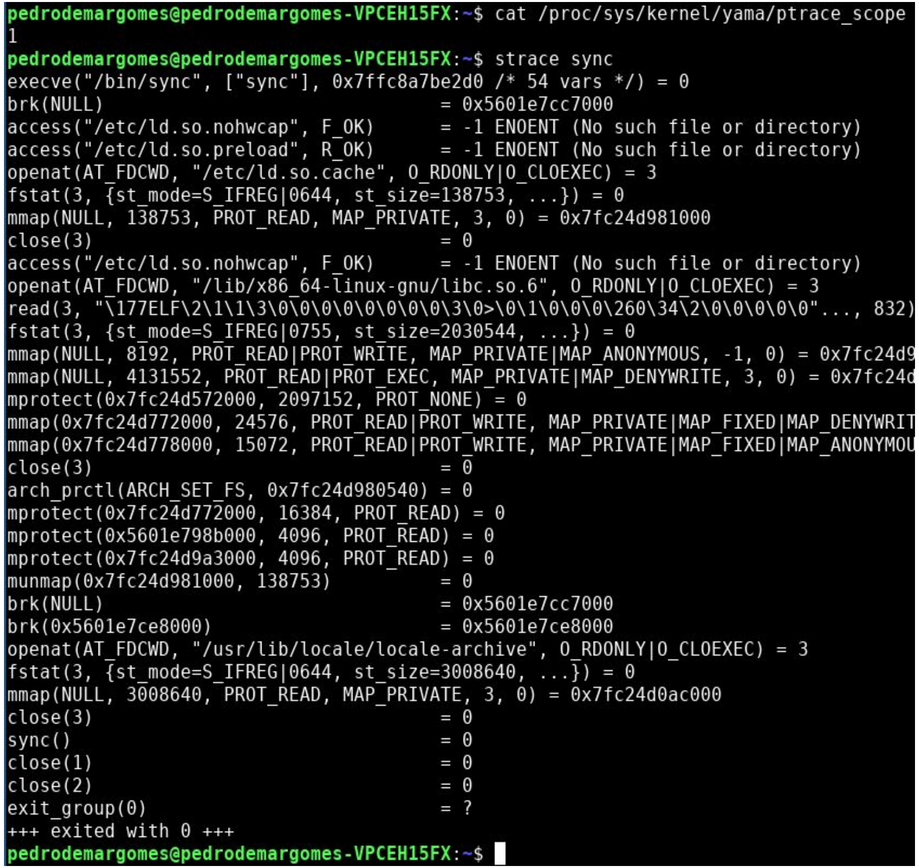
Example - todo mby list modules



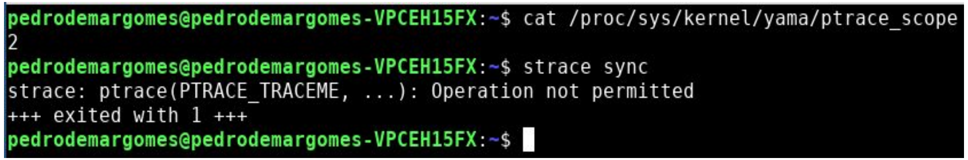
```
florian@zuse1:~$ cat /sys/kernel/security/lsm
capability,yama,apparmor%
```

- List of active security modules
- Order, in which checks are made

YAMA



```
pedrodemargomes@pedrodemargomes-VPCEH15FX:~$ cat /proc/sys/kernel/yama/ptrace_scope
1
pedrodemargomes@pedrodemargomes-VPCEH15FX:~$ strace sync
execve("/bin/sync", ["sync"], 0x7ffc8a7be2d0 /* 54 vars */) = 0
brk(NULL)                               = 0x5601e7cc7000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=138753, ...}) = 0
mmap(NULL, 138753, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fc24d981000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\260\34\2\0\0\0\0\0"..., 832)
fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fc24d9
mmap(NULL, 4131552, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fc24d
mprotect(0x7fc24d572000, 2097152, PROT_NONE) = 0
mmap(0x7fc24d772000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRIT
mmap(0x7fc24d778000, 15072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOU
close(3)                                = 0
arch_prctl(ARCH_SET_FS, 0x7fc24d980540) = 0
mprotect(0x7fc24d772000, 16384, PROT_READ) = 0
mprotect(0x5601e798b000, 4096, PROT_READ) = 0
mprotect(0x7fc24d9a3000, 4096, PROT_READ) = 0
munmap(0x7fc24d981000, 138753)          = 0
brk(NULL)                               = 0x5601e7cc7000
brk(0x5601e7ce8000)                     = 0x5601e7ce8000
openat(AT_FDCWD, "/usr/lib/locale/locale-archive", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=3008640, ...}) = 0
mmap(NULL, 3008640, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fc24d0ac000
close(3)                                = 0
sync()                                  = 0
close(1)                                = 0
close(2)                                = 0
exit_group(0)                           = ?
+++ exited with 0 +++
pedrodemargomes@pedrodemargomes-VPCEH15FX:~$
```



```
pedrodemargomes@pedrodemargomes-VPCEH15FX:~$ cat /proc/sys/kernel/yama/ptrace_scope
2
pedrodemargomes@pedrodemargomes-VPCEH15FX:~$ strace sync
strace: ptrace(PTRACE_TRACEME, ...): Operation not permitted
+++ exited with 1 +++
pedrodemargomes@pedrodemargomes-VPCEH15FX:~$
```

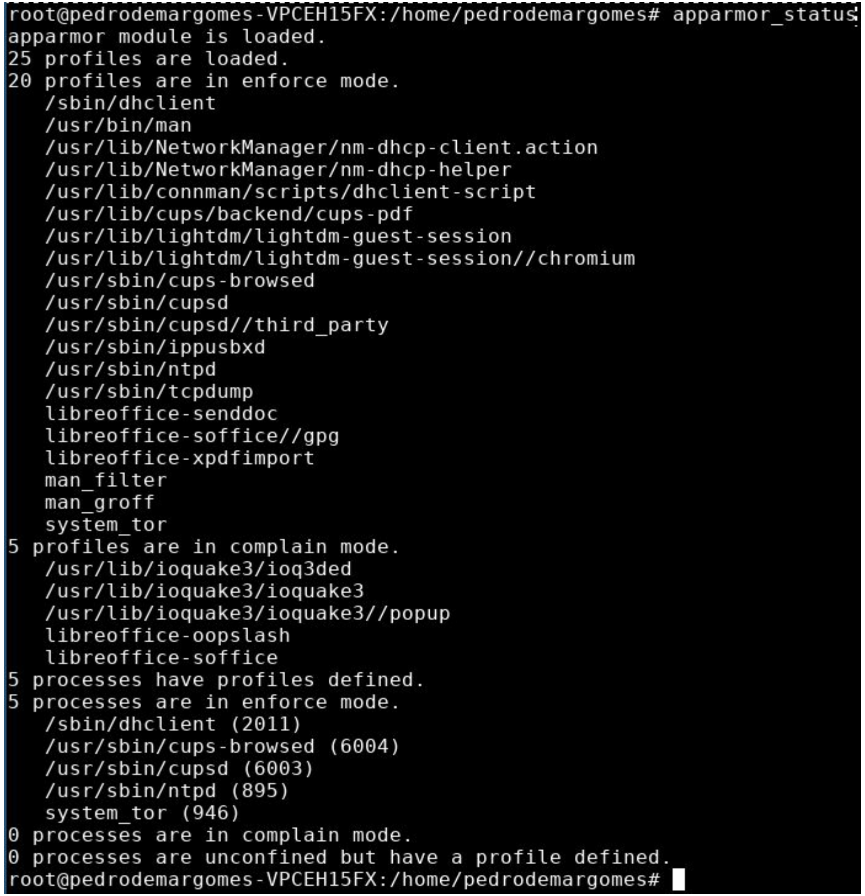
kernel.yama.ptrace_scope = 0: All processes can be debugged, as long as they have same uid. This is the classical way of how ptracing worked.

kernel.yama.ptrace_scope = 1: only a parent process can be debugged.

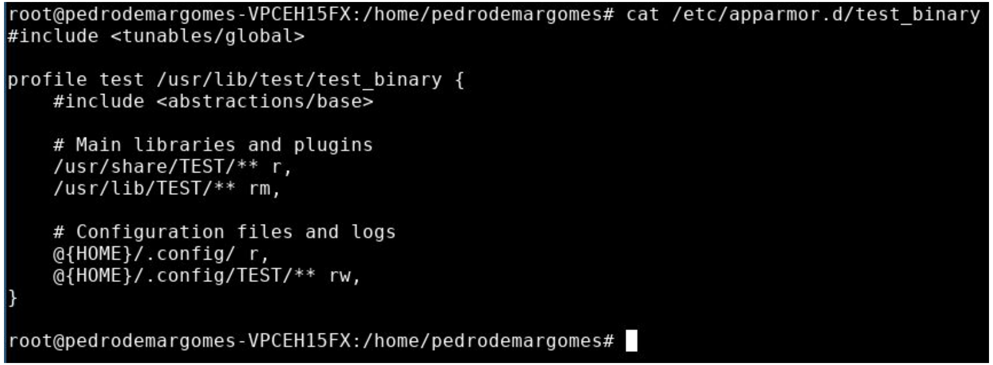
kernel.yama.ptrace_scope = 2: Only admin can use ptrace, as it required CAP_SYS_PTRACE capability.

kernel.yama.ptrace_scope = 3: No processes may be traced with ptrace. Once set, a reboot is needed to enable ptracing again.

AppArmor



```
root@pedrodemargomes-VPCEH15FX:/home/pedrodemargomes# apparmor_status
apparmor module is loaded.
25 profiles are loaded.
20 profiles are in enforce mode.
   /sbin/dhclient
   /usr/bin/man
   /usr/lib/NetworkManager/nm-dhcp-client.action
   /usr/lib/NetworkManager/nm-dhcp-helper
   /usr/lib/connman/scripts/dhclient-script
   /usr/lib/cups/backend/cups-pdf
   /usr/lib/lightdm/lightdm-guest-session
   /usr/lib/lightdm/lightdm-guest-session//chromium
   /usr/sbin/cups-browsed
   /usr/sbin/cupsd
   /usr/sbin/cupsd//third_party
   /usr/sbin/ippusbxd
   /usr/sbin/ntpd
   /usr/sbin/tcpdump
   libreoffice-senddoc
   libreoffice-soffice//gpg
   libreoffice-xpdfimport
   man_filter
   man_groff
   system_tor
5 profiles are in complain mode.
   /usr/lib/ioquake3/ioq3ded
   /usr/lib/ioquake3/ioquake3
   /usr/lib/ioquake3/ioquake3//popup
   libreoffice-oopslash
   libreoffice-soffice
5 processes have profiles defined.
5 processes are in enforce mode.
   /sbin/dhclient (2011)
   /usr/sbin/cups-browsed (6004)
   /usr/sbin/cupsd (6003)
   /usr/sbin/ntpd (895)
   system_tor (946)
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
root@pedrodemargomes-VPCEH15FX:/home/pedrodemargomes# 
```



```
root@pedrodemargomes-VPCEH15FX:/home/pedrodemargomes# cat /etc/apparmor.d/test_binary
#include <tunables/global>

profile test /usr/lib/test/test_binary {
    #include <abstractions/base>

    # Main libraries and plugins
    /usr/share/TEST/** r,
    /usr/lib/TEST/** rm,

    # Configuration files and logs
    @{HOME}/.config/ r,
    @{HOME}/.config/TEST/** rw,
}

root@pedrodemargomes-VPCEH15FX:/home/pedrodemargomes# 
```

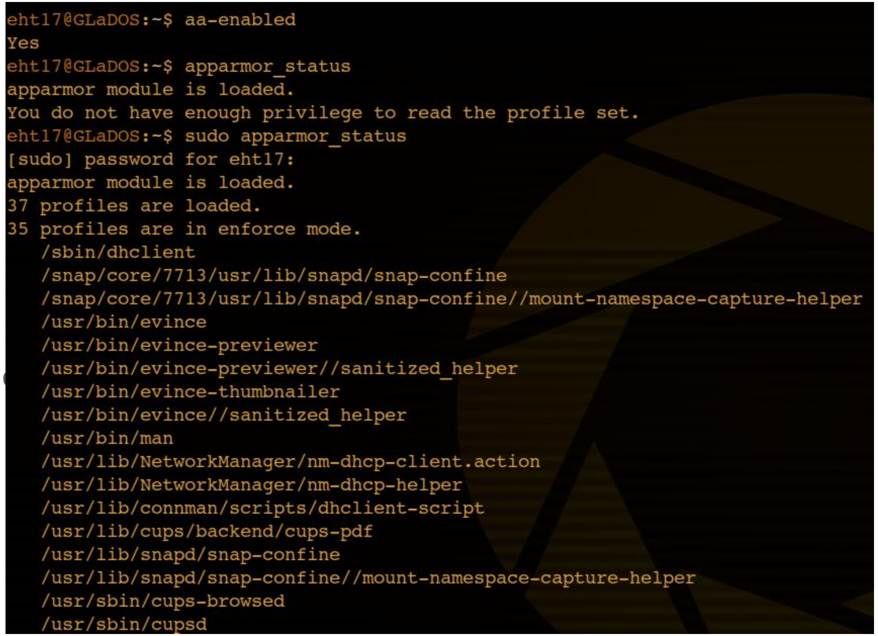
- Profiles are described at /etc/apparmor.d/
- Variables begin with @, and are defined at the included files(tunables/global in this case).
- This permissions cannot exceed the permissions defined by DAC.

Downsides

- Overhead
- Stateless Calls
- Not so many hooks
- Too much work to port
- Rootkits can use it too

Conclusion

- Modularity
- Allow support for MAC policies
- Supplements the default DAC rather than
- Only adds restrictive behavior
- Allows some forms of “abuse” to bypass th



```
eht17@GLaDOS:~$ aa-enabled
Yes
eht17@GLaDOS:~$ apparmor_status
apparmor module is loaded.
You do not have enough privilege to read the profile set.
eht17@GLaDOS:~$ sudo apparmor_status
[sudo] password for eht17:
apparmor module is loaded.
37 profiles are loaded.
35 profiles are in enforce mode.
   /sbin/dhclient
   /snap/core/7713/usr/lib/snapd/snap-confine
   /snap/core/7713/usr/lib/snapd/snap-confine//mount-namespace-capture-helper
   /usr/bin/evince
   /usr/bin/evince-previewer
   /usr/bin/evince-previewer//sanitized_helper
   /usr/bin/evince-thumbnailer
   /usr/bin/evince//sanitized_helper
   /usr/bin/man
   /usr/lib/NetworkManager/nm-dhcp-client.action
   /usr/lib/NetworkManager/nm-dhcp-helper
   /usr/lib/connman/scripts/dhclient-script
   /usr/lib/cups/backend/cups-pdf
   /usr/lib/snapd/snap-confine
   /usr/lib/snapd/snap-confine//mount-namespace-capture-helper
   /usr/sbin/cups-browsed
   /usr/sbin/cupsd
```

Sources

- Linux Security Modules: General Security Support for the Linux Kernel
- https://www.kernel.org/doc/html/v4.15/admin-guide/LSM/index.html
- https://www.kernel.org/doc/htmldocs/lsm/
- https://grsecurity.net/lsm
- https://www.rsbac.org/documentation/why_rsbac_does_not_use_lsm

Vielen Dank!

Fragen???