

NAME

AppArmor – kernel enhancement to confine programs to a limited set of resources.

DESCRIPTION

AppArmor is a kernel enhancement to confine programs to a limited set of resources. AppArmor's unique security model is to bind access control attributes to programs rather than to users.

AppArmor confinement is provided via *profiles* loaded into the kernel via **apparmor_parser** (8), typically through the `/etc/init.d/apparmor` SysV initscript, which is used like this:

```
# /etc/init.d/apparmor start
# /etc/init.d/apparmor stop
# /etc/init.d/apparmor restart
```

AppArmor can operate in two modes: *enforcement*, and *complain or learning*:

- *enforcement* – Profiles loaded in enforcement mode will result in enforcement of the policy defined in the profile as well as reporting policy violation attempts to syslogd.
- *complain* – Profiles loaded in `complain` mode will not enforce policy. Instead, it will report policy violation attempts. This mode is convenient for developing profiles. To manage `complain` mode for individual profiles the utilities **aa-complain** (8) and **aa-enforce** (8) can be used. These utilities take a program name as an argument.

Profiles are traditionally stored in files in `/etc/apparmor.d/` under filenames with the convention of replacing the `/` in pathnames with `.` (except for the root `/`) so profiles are easier to manage (e.g. the `/usr/sbin/nscd` profile would be named `usr.sbin.nscd`).

Profiles are applied to a process at **exec** (3) time (as seen through the **execve** (2) system call): once a profile is loaded for a program, that program will be confined on the next **exec** (3). If a process is already running under a profile, when one replaces that profile in the kernel, the updated profile is applied immediately to that process. On the other hand, a process that is already running unconfined cannot be confined.

AppArmor supports the Linux kernel's `securityfs` filesystem, and makes available the list of the profiles currently loaded; to mount the filesystem:

```
# mount -tsecurityfs securityfs /sys/kernel/security
$ cat /sys/kernel/security/apparmor/profiles
/usr/bin/mutt
/usr/bin/gpg
...
```

Normally, the initscript will mount `securityfs` if it has not already been done.

AppArmor also restricts what privileged operations a confined process may execute, even if the process is running as root. A confined process cannot call the following system calls:

```
create_module(2) delete_module(2) init_module(2) ioperm(2)
iopl(2) ptrace(2) reboot(2) setdomainname(2)
sethostname(2) swapoff(2) swapon(2) sysctl(2)
```

ERRORS

When a confined process tries to access a file it does not have permission to access, the kernel will report a message through audit, similar to:

```
audit(1386511672.612:238): apparmor="DENIED" operation="exec"
parent=7589 profile="/tmp/sh" name="/bin/uname" pid=7605
comm="sh" requested_mask="x" denied_mask="x" fsuid=0 ouid=0

audit(1386511672.613:239): apparmor="DENIED" operation="open"
parent=7589 profile="/tmp/sh" name="/bin/uname" pid=7605
comm="sh" requested_mask="r" denied_mask="r" fsuid=0 ouid=0
```

```
audit(1386511772.804:246): apparmor="DENIED" operation="capable"
parent=7246 profile="/tmp/sh" pid=7589 comm="sh" pid=7589
comm="sh" capability=2 capname="dac_override"
```

The permissions requested by the process are described in the `operation=` and `denied_mask=` (for files – capabilities etc. use a slightly different log format). The “name” and process id of the running program are reported, as well as the profile name including any “hat” that may be active, separated by “//”. (“Name” is in quotes, because the process name is limited to 15 bytes; it is the same as reported through the Berkeley process accounting.)

For confined processes running under a profile that has been loaded in complain mode, enforcement will not take place and the log messages reported to audit will be of the form:

```
audit(1386512577.017:275): apparmor="ALLOWED" operation="open"
parent=8012 profile="/usr/bin/du" name="/etc/apparmor.d/tunables/"
pid=8049 comm="du" requested_mask="r" denied_mask="r" fsuid=1000 ouid=0
```

```
audit(1386512577.017:276): apparmor="ALLOWED" operation="open"
parent=8012 profile="/usr/bin/du" name="/etc/apparmor.d/tunables/"
pid=8049 comm="du" requested_mask="r" denied_mask="r" fsuid=1000 ouid=0
```

If the userland `auditd` is not running, the kernel will send audit events to `klogd`; `klogd` will send the messages to `syslog`, which will log the messages with the KERN facility. Thus, REJECTING and PERMITTING messages may go to either `/var/log/audit/audit.log` or `/var/log/messages`, depending upon local configuration.

DEBUGGING

AppArmor provides a few facilities to log more information, which can help debugging profiles.

Enable debug mode

When debug mode is enabled, AppArmor will log a few extra messages to `dmesg` (not via the audit subsystem). For example, the logs will tell whether environment scrubbing has been applied.

To enable debug mode, run:

```
echo 1 > /sys/module/apparmor/parameters/debug
```

Turn off deny audit quieting

By default, operations that trigger deny rules are not logged. This is called *deny audit quieting*.

To turn off deny audit quieting, run:

```
echo -n noquiet > /sys/module/apparmor/parameters/audit
```

Force audit mode

AppArmor can log a message for every operation that triggers a rule configured in the policy. This is called *force audit mode*.

Warning! Force audit mode can be extremely noisy even for a single profile, let alone when enabled globally.

To set a specific profile in force audit mode, add the `audit` flag:

```
profile foo flags=(audit) { ... }
```

To enable force audit mode globally, run:

```
echo -n all > /sys/module/apparmor/parameters/audit
```

If `auditd` is not running, to avoid losing too many of the extra log messages, you will likely have to turn off rate limiting by doing:

```
echo 0 > /proc/sys/kernel/printk_ratelimit
```

But even then the kernel ring buffer may overflow and you might lose messages.

Else, if `auditd` is running, see [auditd\(8\)](#) and [auditd.conf\(5\)](#).

FILES

/etc/init.d/apparmor
/etc/apparmor.d/
/var/lib/apparmor/
/var/log/audit/audit.log
/var/log/messages

SEE ALSO

apparmor_parser (8), **aa_change_hat** (2), **apparmor.d** (5), **aa-autodep** (1), **clean** (1), **auditd** (8), **aa-unconfined** (8), **aa-enforce** (1), **aa-complain** (1), and <<https://wiki.apparmor.net>>.