Working with SELinux on Android

Aayush Gupta

About Me

- Independent Contractor/ Freelancer
- Currently working at The Calyx Institute on CalyxOS
- Senior Staff, DevRel @ XDA Developers (Forums)
- FOSS Developer & Contributor

SELinux and Android

SELinux

- Optional feature of Linux kernel, based on the LSM framework
- Provides support to enforce access control security policies to enforce MAC
- Developed by NSA (USA), merged in Linux 2.6
- Red Hat and McAfee Corp. are some of the significant contributors

SELinux in Android

- NSA led project Security Enhancements (SE) for Android
- Introduced in Android 4.3, defaulting to permissive mode
- Required to be enforcing by Google CTS since Android 5.0

SELinux in Android

- Two modes, enforcing and permissive
- Default mode is enforcing
- Permissive mode is usually used by developers during development
- Denials are logged in the kernel buffer as well as logcat

```
avc: denied { write } for
comm="power@1.2-service"
name="double_tap" dev="proc"
ino=4026533160
scontext=u:r:hal_power_default:s0
tcontext=u:object_r:proc:s0 tclass=file
permissive=0
```

SELinux Policy

SELinux Policy

- Set of rules (permissions) stating which initiators can perform which action
- Android provides SELinux policy for AOSP components
- Downstream vendors provide their own policies for their components
- OS Compilation generates device-partitions specific SELinux policy
- All policies are compiled together into one when Android boots

SELinux Policy

Labelling initiator (not an app)

/path/to/initiator

u:object_r:name_you_want:s0

SELinux Policy

Labelling initiator (app)

user=_app
name=org.calyxos.systemupdater
domain=updater_app
type=app_data_file

SELinux Policy

Allowing permission

allow scontext tcontext:tclass permission;

SELinux Policy

Suppressing denial

dontaudit scontext tcontext:tclass permission;

Neverallow

- Overarching rule to mark specific rules that must not be generated
- Marked rules are considered to weaken the security of the system
- Doesn’t contains all possible scenarios

neverallow my_gallery
my_secret_passwords:{ dir file }
{ read write open };

Macros

- Shortens and simplifies the amount of code needed
- Several macros available to use
- Present in system/sepolicy repository of AOSP

Tools

- chcon
- audit2allow
- restorecon
- sepolicy-inject

Thank You!