



# Middleware MAC for Android

Stephen Smalley  
Trusted Systems Research  
National Security Agency



# Motivation

- Many attacks on Android can occur entirely at the middleware layer.
- Not directly visible to kernel except as legitimate IPC calls.
- Cannot be fully addressed via SE Android kernel layer MAC (SELinux).



# Android Permissions

- Android permission model is more like DAC than MAC.
- Subject to whims/mistakes of user.
- Prone to privilege escalation due to flawed and malicious apps.
  - Collusion, confused deputy attacks.



# SELinux for Userspace?

- SELinux userspace object manager approach.
  - e.g. D-BUS, Xorg, SE-Postgres
- Successfully applied to init property service and ZygoteConnection in Android.
- Problematic for Android middleware.



# Problems

- Binder IPC, not socket IPC.
- Saving and restoring caller identity.
- checkPermission API compatibility.
- App-defined permissions.
- Implications for SELinux policy.



# Middleware MAC (MMAC)

- Separate MAC mechanism(s) at middleware layer.
- Only interaction with kernel layer MAC (SELinux) is selecting security contexts.
- Allows SELinux policy to remain small, simple, and fixed.



# Install-time MAC

- Install-time check of app permissions against MAC policy configuration.
- Ability to select SELinux security contexts based on app certificate, package name.
- Can enforce organizational restrictions.
- Can disable even pre-installed apps.



# setool

- Tool for generating install-time MAC policy stanzas.
- `setool --keys`
- `setool --whitelist`





# Sample signer stanza

```
<!-- Platform dev key with AOSP -->  
<signer signature="...b1b357" >  
  <allow-all />  
  <seinfo value="platform" />  
</signer>
```



# Sample signer stanza #2

```
<!-- Media dev key in AOSP -->
```

```
<signer signature="...a1a81" >
```

```
  <allow-permission name="android.permission.  
ACCESS_ALL_DOWNLOADS" />
```

```
  <allow-permission name="android.permission.  
ACCESS_CACHE_FILESYSTEM" /> ...
```

```
  <seinfo value="media" />
```

```
</signer>
```



# Sample package stanza

```
<package name="com.android.browser" >
```

```
  <allow-permission
```

```
name="android.permission.ACCESS_COARSE_LOCATION"/>
```

```
  <allow-permission
```

```
name="android.permission.ACCESS_DOWNLOAD_MANAGER"/>
```

```
  ...
```

```
</package>
```



# Sample default stanza

```
<default>
```

```
  <seinfo value="default" />
```

```
  <deny-permission  
name="android.permission.CALL_PHONE" />
```

```
  <deny-permission  
name="android.permission.CAMERA" />
```

```
  ....
```

```
</default>
```



# Permission Revocation

- Reduce permissions of installed apps based on a MAC policy configuration.
- Based on CyanogenMod mechanism, but policy-driven.
- Caveat: Most apps do not handle permission revocation well.



# Sample revoke stanza

```
<package name="com.android.browser">  
    <revoke-permission  
name="android.permission.ACCESS_COARSE_L  
OCATION" />  
    <revoke-permission  
name="android.permission.ACCESS_FINE_LOCA  
TION" />  
    </package>
```



# Tag Propagation

- A form of taint tracking.
  - Tag apps based on permissions.
  - Propagate tags on IPC.
  - Block unauthorized IPC.
- Caveats:
  - Taint explosion is common.
  - System apps pose problems.



# Defining a tag

```
<tag name="sensitive_data">  
    <permission  
name="android.permission.ACCESS_FINE_LOCATION" />  
    <permission  
name="android.permission.READ_SMS" />  
    <permission  
name="android.permission.RECORD_AUDIO" />  
    ...  
</tag>
```





# Defining another tag

```
<tag name="sinks">
```

```
  <permission  
name="android.permission.INTERNET" />
```

```
  <permission  
name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

```
  ...
```

```
</tag>
```



# Defining a policy

```
<policy name="sensitive_data">  
  <tag name="sensitive_data" />  
  <tag name="sinks" />  
</policy>  
  
<dont-propagate app-  
name="com.android.launcher" />  
  
<dont-propagate app-  
name="com.android.inputmethod.latin" />
```



# Flask for Middleware

- Apply Flask architecture to Android middleware.
- Similar to SELinux userspace approach.
- But with a middleware policy server and separate middleware security contexts.



# What's Next?

- Upstream install-time MAC.
- Further development of permission revocation and tag propagation.
- Implementation of middleware Flask.



# Questions?

- <http://selinuxproject.org/page/SEAndroid>
- SELinux mailing list:
  - [selinux@tycho.nsa.gov](mailto:selinux@tycho.nsa.gov)
- NSA SE Android team:
  - [seandroid@tycho.nsa.gov](mailto:seandroid@tycho.nsa.gov)
- My email:
  - [sds@tycho.nsa.gov](mailto:sds@tycho.nsa.gov)