AppArmor

Igor Vuk
ivuk84@gmail.com

Overview(1/6)

- http://wiki.apparmor.net
- GNU/Linux application security system
- Mandatory Access Control(MAC) framework
- Shipped by Mandriva, OpenSUSE, Ubuntu...
- AppArmor security policies define what system resources individual applications can access, and with what privileges
- Included in mainline Linux, as of 2.6.36

Overview(2/6)

- Protects systems from insecure/untrusted processes
  - Runs them in confinement
  - Restricts their ability to share files
  - Restricts their ability to communicate with other processes
- The restrictions are mandatory
- The restrictions apply to processes running with superuser privileges
- The protection stacks with DAC

Overview(3/6)

- kernel module and user-space tools licenced under GPL licence
- libapparmor is licenced under LGPL

Overview(4/6)

- It uses resources
  - Files
  - POSIX.1e capabilities
- No support for ulimit or other additional resources
- It controls access to those resources
  - Prevents processes from executing binaries
  - Prevents exercising privileges (i.e. acting on behalf of another user)
- Controls which files a process can access in which ways down to the individual file level

Overview(5/6)

- Protection is selective
  - AppArmor only confines processes for which policies(profiles) exist
- To confine an application
  - A new profile must be written
  - An existing profile must be modified
  - A new profile must be generated (via existing user-space tools)
- The application does not need to be modified

Overview(6/6)

- AppArmor is not based on labeling or label-based access/transition rules
- No objects in the file system are labeled
- Files are identified by name, not by label
  - A process is granted read access to /etc/shadow
  - # mv /etc/shadow /etc/shadow.old
  - # touch /etc/shadow
  - The process has read access to new /etc/shadow, not to /etc/shadow.old

The AppArmor security model(1/3)

- A file gets accessed via open(2), mkdir(2)...
- Kernel looks up the location of the object
- The result of the lookup is a pair of (dentry, vfsmount) kernel-internal objects
- AppArmor uses the (dentry, vfsmount) pair to compute the pathname of the file
- AppArmor checks if the current profile contains rules that match the pathname and if they allow the requested access
  - Accesses that are not explicitly allowed are denied

The AppArmor security model(2/3)

Symbolic links

- The pathname that AppArmor computes never contains symlinks
- If symlinks are used instead of directories(think /tmp) profiles must be adjusted accordingly

Namespaces

- Unclear at the time
- Creating new namespaces is bound to CAP_SYS_ADMIN capability

The AppArmor security model(3/3)

Mount

- Confined processes are denied access to mount(2) and umount(2)

The kernel NFS daemon

- Considered not suitable for AppArmor confinement

Profile permissions

- AppArmor differentiates between more permissions than UNIX does

r	Read.
w	Write.
ix	Execute and inherit the current profile.
px	Execute under a specific profile.
Px	Execute secure and under a specific profile.
ux	Execute unconfined.
Ux	Execute secure and unconfined.
m	Memory map as executable.
l	Link.

Table 1: File Access Permissions in Profiles

Profile {loading,replacement,removal}(1/4)

- Profiles must be loaded
- Profile sources consist of plain text
- Profiles contain a list of file access rules that may include wildcards

Profile {loading,replacement,removal}(2/4)

- AppArmor uses securityfs for configuration and reporting
- Profile {loading,replacement,removal} and AppArmor configuration is done via securityfs

Profile {loading,replacement,removal}(3/4)

```
# mount securityfs -t securityfs /sys/kernel/security
# cat /sys/kernel/security/apparmor/profiles
/usr/sbin/traceroute (complain)
/usr/sbin/tcpdump (enforce)
/usr/sbin/smbd (complain)
/usr/sbin/libvirtd (enforce)
...
```

Profile {loading,replacement,removal}(4/4)

- Profile loading is performed by apparmor_parser

```
# echo "/tmp/ls { /tmp/ls rm, }"|apparmor_parser
```

- Replacing an existing profile

```
# echo "/tmp/ls { /tmp/ls rm, }"|apparmor_parser --replace
```

Anatomy of a profile(1/3)

- man 5 apparmor.d
- Profiles are stored in /etc/apparmor.d/

Anatomy of a profile(2/3)

```
#include <tunables/global>
/bin/ping flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/consoles>
  #include <abstractions/nameservice>

  capability net_raw,
  capability setuid,
  network inet raw,

  /bin/ping mixr,
  /etc/modules.conf r,
}
```

Anatomy of a profile(3/3)

- /bin/ping is the name of the profile
- The profile will be automatically used
- flags=(complain) puts the profile into complain mode
- Complain mode can be enabled globally for all profiles

User-space tools

- aa-status
- enforce
- complain
- aa-autodep
- aa-logprof
- aa-genprof

Libvirt(1/5)

- http://wiki.apparmor.net/index.php/Libvirt
- AppArmor support for confining VMs as of 0.7.2
- The libvirtd process is confined with a lenient profile
  - It can launch VMs
  - It can change into another AppArmor profile and use virt-aa-helper to to manipulate AppArmor profiles
- virt-aa-helper can {add,remove,modify,load,unload} profiles in a restricted way

Libvirt(2/5)

- The system includes multiple files

/etc/apparmor.d/usr.sbin.libvirtd

/etc/apparmor.d/usr.lib.virt-aa-helper

/etc/apparmor.d/abstractions/libvirt-qemu

/etc/apparmor.d/libvirt/TEMPLATE

/etc/apparmor.d/libvirt/libvirt-<uuid>

/etc/apparmor.d/libvirt/libvirt-<uuid>.files

Libvirt(3/5)

- When a VM is started, libvirtd decides whether to ask virt-aa-helper to create a new profile or modify an existing one
- If no profile exists, libvirtd asks virt-aa-helper to generate the new base profile, based on /etc/apparmor.d/libvirt/TEMPLATE
- virt-aa-helper will determine what files are required for the guest to run, updates /etc/apparmor.d/libvirt/libvirt-<uuid>.files, then loads the profile

Libvirt(4/5)

- Before the VM is started, libvirtd transitions into the generated profile
- When the VM is terminated, virt-aa-helper unloads the profile

Libvirt(5/5)

- In general, you can forget about AppArmor and just use libvirt
- The guests are isolated from each other
- User-space protection for the host is provided
- The design offers a lot of flexibility

Some other stuff

- Be careful when writing profiles
- Check your #include files
- Pay attention to apps that do NOT have an AppArmor profile
- http://pad.lv/578922
- Watch your rmix-es :)

The End

Thank you for listening :)
Questions?