Editing and Configuring Policies

Security Policy Development Primer for Security Enhanced Linux

(Module 13)

Changing a Policy

- Many ways to change/write a policy
- Much easier to modify the base policy
  - simple policy tweaks; e.g., add a user
  - disable/Enable policy program modules
  - modify an existing module
  - create a new module

Customizing Policy Modules

- Much of TE policy source is modularized
  - one module (.te file) per program
  - see ./policy/domains/programs modules
- Allows adding and removing unnecessary policy pieces
- Currently no strong dependency model
  - policy build errors
  - runtime error
  - may be difficult to track down

Removing a Policy Module

- All ./policy/domains/program/*.te are included in policy during policy build
- To remove a module
  - remove module.te file from program directory, or
  - ensure file does not end with .te
- Example using oav-update.te
  - remove oav-update.te
  - observe and resolve errors

Modifying an Existing Policy (ping)

```
type ping_t, domain, privlog;
role sysadm_r types ping_t;
role system_r types ping_t;
every_domain(ping_t)
```

```
type ping_exec_t, file_type, sysadmfile, exec_type;

domain_auto_trans(sysadm_t, ping_exec_t, ping_t)
domain_auto_trans(initrc_t, ping_exec_t, ping_t)
```

```
allow ping_t self:rawip_socket { create bind setopt getopt write read };
allow ping_t any_socket_t:rawip_socket sendto;
allow ping_t { self icmp_socket_t }:rawip_socket recvfrom;

allow ping_t ping_t:capability { net_raw setuid };
```

```
allow ping_t admin_tty_type:chr_file rw_file_perms;
ifdef(`gnome-pty-helper.te', `allow ping_t sysadm_gph_t:fd use;')
```

Changing the ping Policy

- How might we want to change ping?
- Can a normal user (user_t) ping?
- Does it make sense to allow a regular user to ping?
- What are the risks?

Changing ping.te

- Add
  - domain_auto_trans(user_t, ping_exec_t, ping_t)
- Oops, we need to also add
  - role user_r types ping_t;
- ...is that all we have to do? No
  - allow ping_t user_devpts_t:chr_file { rw_file_perms };
- Does it work now? It should.

Creating a Policy Module For ‘who’

- Only allow sysadm_r to run the `who’ command
- Policy requirements
  - create who_t domain/type
  - only allow sysadm_r access to who_t domain
  - allow sysadm_t to transition to who_t
  - protect system resources `who’ requires

‘who’ Module: the Beginning

- Create the module files (.te & .fc files)
- Create the types

```
# who.te
#DESC who command
type who_t, domain;
role sysadm_r types who_t;
type who_exec_t, file_type, exec_type;
```

- Assign labeling in the .fc file

```
# who.fc
/usr/bin/who  system_u:object_r:who_exec_t
```

‘who’ Module: Next step

- Add a domain transition for sysadm_t

```
type who_t, domain;
role sysadm_r types who_t;
type who_exec_t, file_type, exec_type;

domain_auto_trans(sysadm_t, who_exec_t,
      who_t)
```

- Build, load and test
  - chcon /usr/bin/who after loading policy

    chcon system_u:object_r:who_exec_t /usr/bin/who

`who' Module: part 3

- Allow common access permissions

```
type who_t, domain;
role sysadm_r types who_t;
type who_exec_t, file_type, exec_type;
domain_auto_trans(sysadm_t,
  who_exec_t,who_t)

every_domain(who_t)
```

- Build, load and test

`who' Module: part 4

- Access to tty

```
type who_t, domain;
role sysadm_r types who_t;
type who_exec_t, file_type, exec_type;
domain_auto_trans(sysadm_t, who_exec_t,
    who_t)
every_domain(who_t)
allow who_t admin_tty_type:chr_file
  { rw_file_perms };
```

Other who.te issues

- Restrict access to
  - /var/run/utmp
  - /var/log/wtmp

- Difficult to determine what domains also require access to these files.
- Exercise for the student! ☺

QUESTIONS?