OPERATING SYSTEM SECURITY GUEST LECTURE

MANDATORY ACCESS CONTROL

SECURITY ENHANCED LINUX (SELINUX)

PATRICK UITERWIJK
PUITERWIJK@REDHAT.COM / PATRICK.UITERWIJK.ORG
GPG KEY: 4096R/0X9AB51E50

MANDATORY ACCESS CONTROL

A system implements Mandatory access control if the protection state can only be modified by trusted administrators via trusted software.

- SELinux is an implementation of a Mandatory Access Control system, thus it cannot be disabled by a user.
- This means that even if your process runs as root, SELinux will be in effect, and may block you from doing stuff.
- Resources here is meant in the broad sense of the word: SELinux protects files, processes, network sockets, and more.

GENERAL MECHANISMS AVAILABLE IN SELINUX

- Type enforcement
- User/role access control
- Multi-level security/Multi-category security (MLS/MCS)

Will mostly talk about type enforcement, MLS/MCS is most often used for "Three Letter Agencies".

POLICIES

SELinux uses policies to define what is allowed.

THE TWO TYPICAL POLICIES:

1. Targeted: Primarily type enforcement
2. Multi-Level Security/Multi-Category Security (MLS/MCS)

Will mostly talk about type enforcement.

SECURITY CONTEXTS

Every resource gets a SELinux security context in the form of user:role:type:range.

Files have extended attributes (ls -Z):

```
-rw-rw-r--. puiterwijk puiterwijk unconfined_u:object_r:user_home_t:s0 2015-01-05-selinu
```

Other resources have their context in the kernel:

Processes (`ps -AZ`):

```
system_u:system_r:init_t:s0 systemd
```

Network sockets (`netstat -tulpnZ`):

```
tcp6 :::80 :::* LISTEN 9804/httpd system_u:system_r:httpd_t:s0
```

Users (`id -Z`):

```
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

SECURITY CONTEXT SOURCES

How do resources get their contexts?

- Processes: get the parent context

Can be configured with transitioning rules (later)

- Other resources: transitioning rules
- Files: get the upper directory context

Can be modified:

```
chcon -t httpd_user_content_t index.html
```

Or configured with file context rules:

```
HOME_DIR/public_html(/.+)? gen_context(system_u:object_r:httpd_user_content_t,s0)
```

TYPE ENFORCING: RULE TYPES

- Where the Unix file permissions protect users' and the system files from modification by unintended users, SELinux type enforcing protects resources from modification or usage by unintended processes.

Multiple type enforcing rule types exist:

- Type definitions
- Allow rules
- Transitioning rules
- Tunables (booleans)

TYPE ENFORCING: ALLOW RULES

Type enforcement allow rules define `what security context` is allowed to perform `which actions` against `specific resources`.

Example rule:

```
allow httpd_t httpd_config_t:dir list_dir_perms;
```

This allows:

- Processes in the security context of htpd_t
- to list files
- in directories with security context httpd_config_t

TYPE ENFORCING: TRANSITIONS

- Processes need to transition from one context to another to get their correct permissions
- Transition rules define which contexts may transition to which other contexts under which conditions
- Example:

```
allow initrc_domain daemon : process transition;
```

Will make sure init_t -> httpd_exec_t -> httpd_t is allowed, and happens.

After transition, there is no going back except when reverse transition exists.

TYPE ENFORCING: TUNABLES/BOOLEANS

- Some rules only make sense in specific setups (httpd_can_network_connect)
- Tunable policy rules allow to use the same policy in multiple situations
- getsebool/setsebool to get/set

TYPE ENFORCING: UNCONFINED

- Unconfined contexts have almost all permissions
- Unconfined is the default for processes without policy rules
- Some specific things are still blocked

WHAT IS TYPE ENFORCING USEFUL FOR?

Remember:

- Discretionary Access Control: user could chmod 777 and undo all security
- Root is omnipotent: root can do everything
- Type enforcing doesn't care who runs the process (to a certain extend), just what process is being run
- SELinux helps protect against further impact after a service is compromised

PROTECTION EXAMPLE: SHELLSHOCK APACHE

- CVE-2014-6271, CVE-2014-6277, et al
- Possibility to run random commands on remote apache host
- Impact seriously limited by SELinux:

Writing to disk, could we be used to write a rootkit:

```
sesearch -A -s httpd_t -c file -p write
```

Connections from apache to outside, could we be used as a spam bot:

```
sesearch -A -s httpd_sys_script_t -p name_connect -C | grep -v ^D
```

Connections from outside to apache:

```
sesearch -A -s httpd_sys_script_t -p name_bind -C | grep -v ^D
```

TYPE ENFORCING: SUMMARY

- Type enforcing only looks at the types, and uses rules defined in the policy.
- Type enforcing can seriously limit attack vectors.
- If you have a system that offers SELinux (Fedora, CentOS, probably more), make sure to enable it!
- To enable: `setenforce 1`

```
[puiterwijk@bofh ~]$ getenforce
Enforcing
```

USERS/ROLES

Remember: Contexts are of type: **user:role**:type:range

- Type enforcing prevents processes access to resources they should not touch
- The **user** and **role** are used to define what processes the user can run
- SELinux user is not a specific user, but rather a class of users: user_u, guest_u, staff_u, ...
- SELinux semanage login manages unix user -> SELinux user mapping

ROLES

- SELinux users can have multiple roles, in which case switching is possible: `newrole -r sysadm_r`
- Roles define what commands the user is allowed to execute.

Example:

```
sudo_role_template(staff, staff_r, staff_t)
```

MLS/MCS

Remember: Contexts are of type: user:role:type:**range**

Range has the syntax: levellow-levelhigh:categories

- Level: value from s0-s15, follows Bell-LaPadula-model
(No read up, no write down)

Write or read-level must "dominate" all levels of the user.

- Categories: c0-c1023, adds categories
(Only access to specified categories)

Must have all categories in order to read file, written files have all categories.

- For more information, check last lecture

SELINUX STATES

If you have a kernel with the SELinux module enabled, SELinux is in one of three states:

1. Disabled, SELinux is disabled and will not do anything
2. Permissive, SELinux will read the policy and alert, but not actually block anything
3. Enforcing, SELinux will read the policy file and deny all actions not allowed in the policy

State can be checked with `getenforce` or `sestatus`.

SUMMARY

- SELinux is a MAC whose rules are defined by policies
- Rules are enforced by the kernel
- Multiple types of rules: type enforcing, users/roles and MLS/MCS
- Very useful to limit impact of compromised service
- Also useful to determine who may read or write which files (MLS/MCS)

LAST BUT NOT LEAST

If you have a system with SELinux available, do NOT disable it, but learn how to use it!

http://stopdisablingselinux.com/

In case of questions, feel free to reach out to me (see http://patrick.uiterwijk.org/contact/ for info).