Hands-on SELinux: A Practical Introduction

Security Training Course

Dr. Charles J. Antonelli
The University of Michigan
2012

Roadmap

- Day 1:
  - Why SELinux?
  - Overview of SELinux
  - Using SELinux
  - SELinux Permissive Domains
- Day 2:
  - SELinux Booleans
  - SELinux Policy Theory
  - SELinux Policy Praxis
  - SELinux audit2allow

SELinux Tools

- GUI
  - Configure SELinux
    sudo /usr/bin/system-config-selinux
    Applications| Other| SELinux Management
  - Interpret SELinux log errors
    /usr/bin/sealert
    Applications | System Tools | SELinux Troubleshooter
- Command line
  - semanage, setsebool, setenforce, getenforce, audit2allow, …
  - As always, **`man`** is your friend

Command-line Hints

1. man is your friend

```
man semanage
```

2. Use shell command history
3. Search for string `foo` in all files rooted in directory tree `bar`:

```
find bar -print0 | xargs grep -0 foo
```

SELinux Booleans

Booleans

- Allow policies to be changed at runtime
  - Fine-tune service access
  - Change service port numbers
    - Must be pre-defined
  - Greatly reduces need for new policy modules
  - Originally Boolean values only
    - Now extended beyond Boolean values

Example

- httpd_can_network_connect_db

  List all Booleans

```
getsebool -a
semanage boolean -l
```

  Set a Boolean, but not across reboot

```
setsebool httpd_can_network_connect_db on
```

  Set a Boolean permanently

```
setsebool -P httpd_can_network_connect_db on
```

Example

- `http_port_t`

```
semanage port -l
semanage port -a -t http_port_t -p tcp 1234
```

Booleans

- Command documentation

```
man getsebool
man setsebool
man semanage
```

Lab – httpd server

Goal: Observe and remove SELinux policy violations

- Start and stop httpd as installed

  systemctl status httpd.service
  sudo systemctl start httpd.service
  … observe default page
  sudo systemctl stop httpd.service

Lab – httpd server

- Create a new document directory

```
sudo mkdir /html
sudo touch /html/index.html
... maybe add some html
ls -ZaR /html
... observe types
```

Lab – httpd server

- Point DocumentRoot at the new directory

  sudo vi /etc/httpd/conf/httpd.conf
  … change DocumentRoot to /html

Lab – httpd server

- Start server

  sudo systemctl start httpd.service
  systemctl status httpd.service

- Navigate to /html

- Observe SELinux alert
  - Or run
    `sudo sealert -a /var/log/audit/audit.log`

Lab – httpd server

- Correct labeling

```
ls –ZaR /html
chcon -Rv –t httpd_sys_content_t /html
ls –ZaR /html
```

… what's the difference?

Lab – httpd server

- Navigate to /html
- Observe correct operation

Lab – httpd server

- The modified labels are not permanent
  - Will survive reboots
  - Will not survive filesystem relabels

- To guarantee permanence

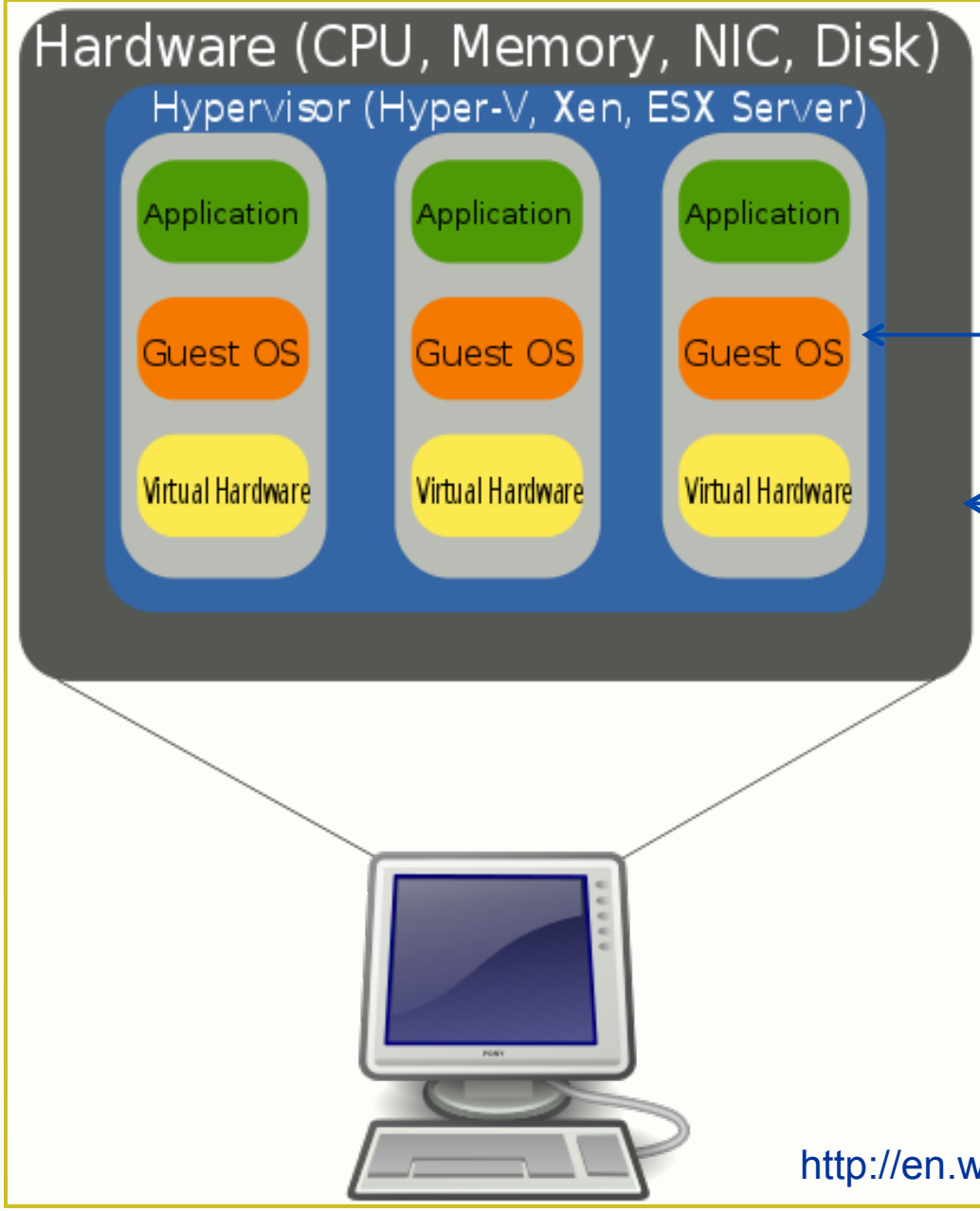
```
semanage fcontext –a –t
httpd_sys_content_t “/html(/.*)?”
```

Lab – vnc-server

Goal: install a VNC server on your guest and establish a connection to it from your host platform

- VNC allows you to access your Linux desktop from another (remote) IP address
- In this lab, we'll use your host platform as that remote IP address
- Although VNC use requires a separate password, it is not a secure protocol
  - So we'll use ssh to create a secure tunnel between your host and guest

Terminology



Guest, *i.e.*, VLE16

Host, *e.g.*, Windows

http://en.wikipedia.org/wiki/Platform_virtualization

Lab – vnc-server

1. Enable vnc-server on your guest
   - `wget http://www.umich.edu/~cja/SEL12/supp/INSTALL-vnc.sh`
   - `sh ./INSTALL-vnc.sh`
     - Should end with “vnc server running”
2. Obtain your guest’s IP address
   - `ifconfig`

     The IP address will be the contents of the `inet addr` field of the `eth`*N* entry listed, where *N* is a small integer

Lab – vnc-server

3. Install a VNC client on your *host* platform
   - Windows: (select the 32- or 64-bit full installer)
     http://www.uvnc.com:8080/downloads/ultravnc/92-ultravnc-1095.html
     Run the downloaded installer application (install Viewer only, keep all other defaults)
   - Mac OS X: (select cotv4-20b4.dmg)
     http://sourceforge.net/projects/cotvnc/
     Open the .dmg file to install.
   - Linux: `sudo yum install -y tigervnc`

Lab – vnc-server

4. Install an SSH client on your *host* platform
   (This step is needed only for Windows hosts)
   - Windows: We'll install PuTTY, a freely available SSH client: http://the.earth.li/~sgtatham/putty/latest/x86/putty-0.60-installer.exe

     Run the installer

Lab – vnc-server

5. Open an ssh tunnel to your guest from your *host* platform:

Linux & Mac OS X:

- `ssh -L 5901:localhost:5901 lab@guest.ip.addr`

`(Use your guest IP address from Step 2.)`

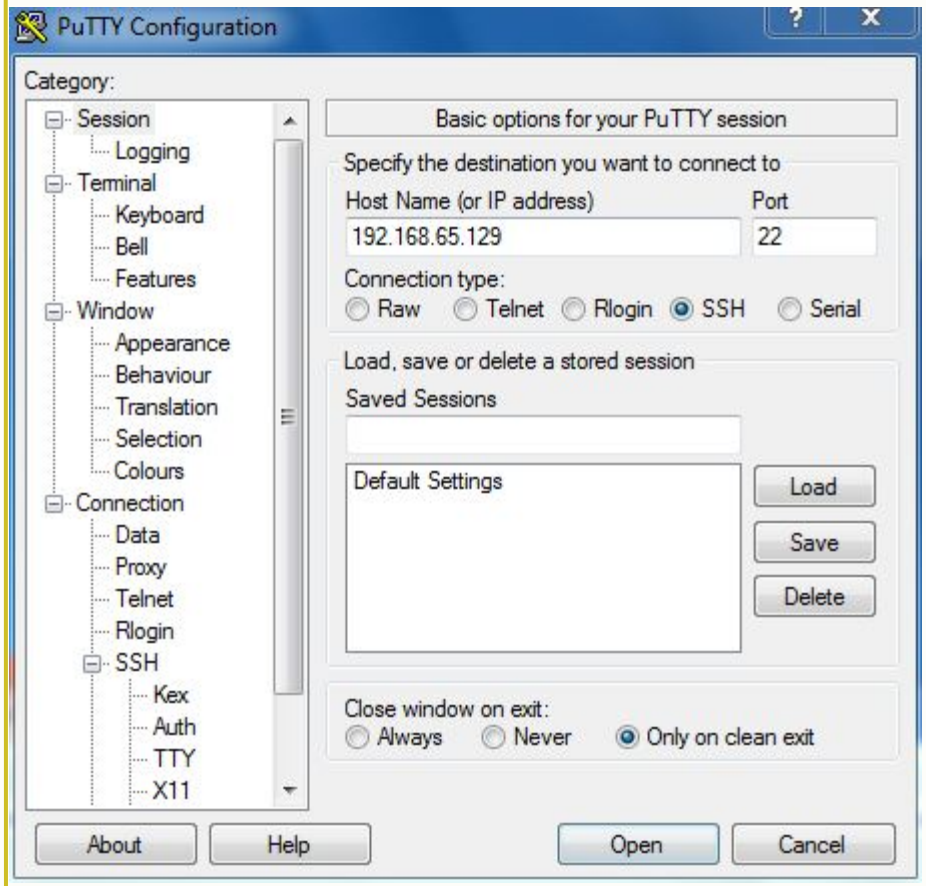
Lab – vnc-server

5. Open an ssh tunnel to your guest from your *host* platform:

Windows:

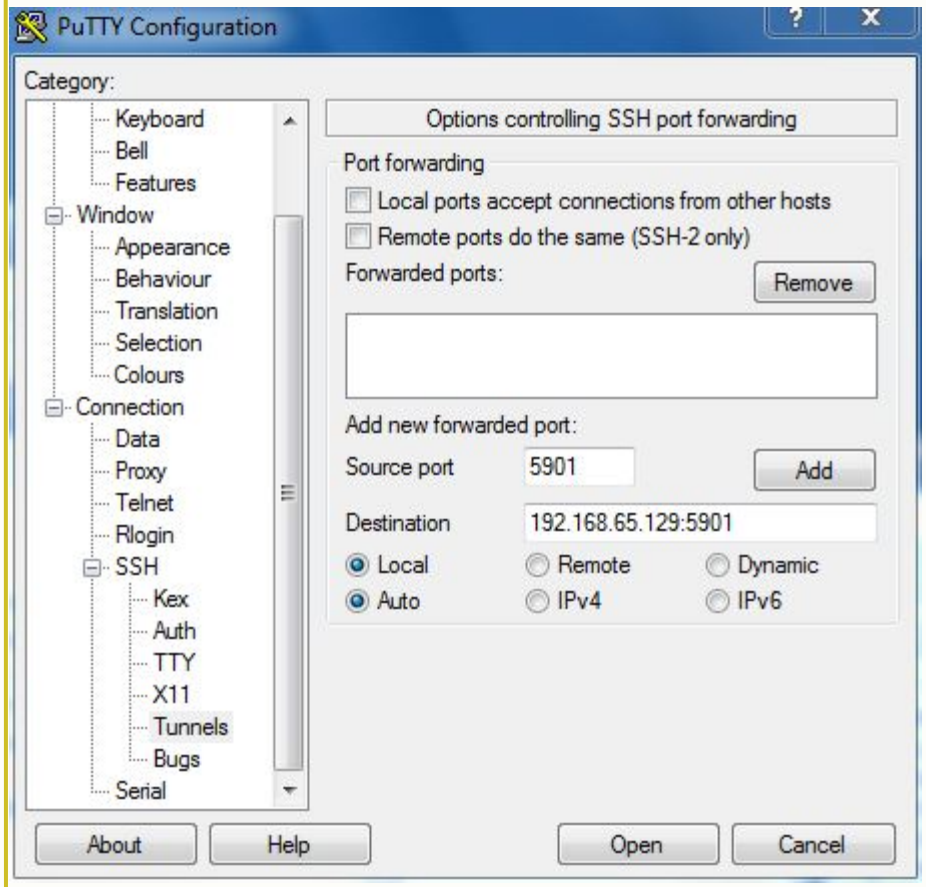
- Start PuTTY

Lab – vnc-server



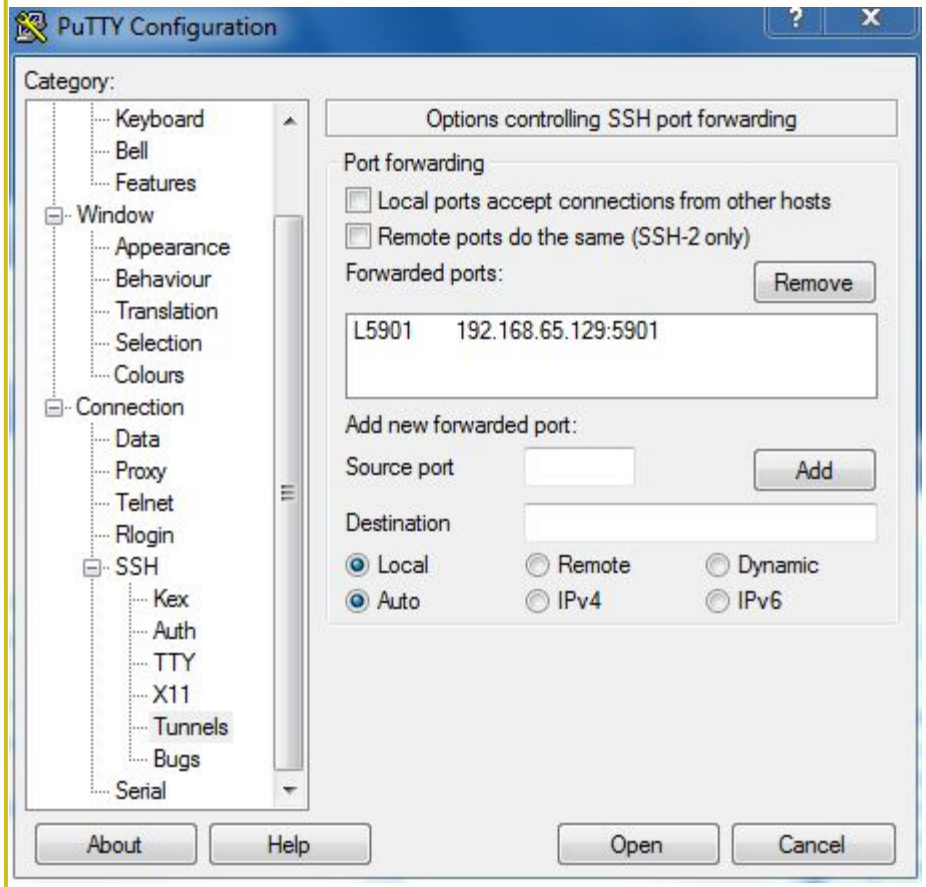
- Enter your guest IP address from Step 2 in the Host Name field.
- Then, in the Category box on the left, select Connection | SSH.
- Finally, expand the SSH menu item by clicking on its + icon, and select Tunnels.

Lab – vnc-server



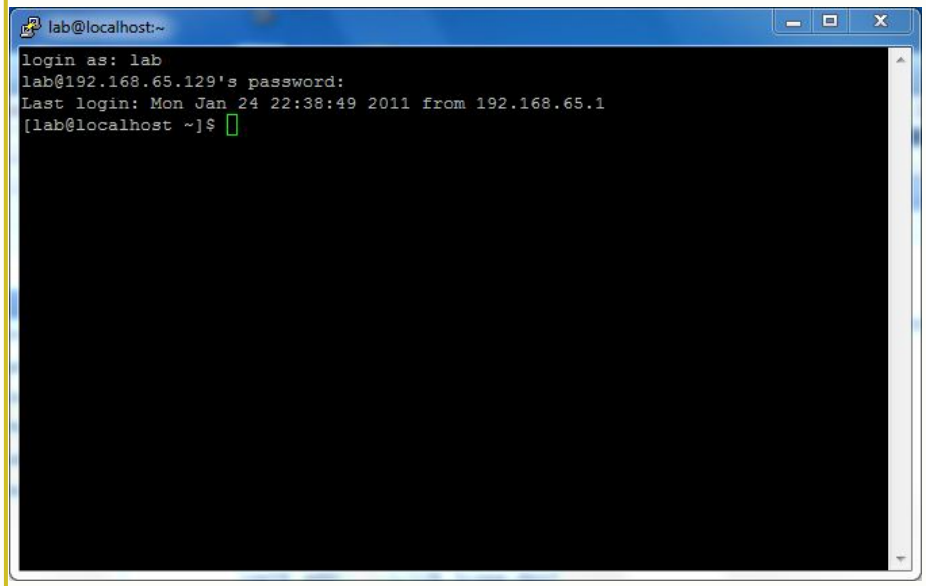
- Enter “5901” in the Source port field.
- Enter your guest IP address from Step 2 in the Destination field, followed by “:5901”.
- Then, click the Add button.

Lab – vnc-server



- Click the Open button.

Lab – vnc-server



- In the terminal window that appears, enter “lab”.
- When prompted, enter the password for the guest lab account.
- You should see a login banner and a shell prompt.
- You have (1) opened an SSH shell on your guest and (2) forwarded the VNC port (5901) from your host to your guest.

Lab – vnc-server

6. Connect to your guest using the VNC client on your host:
   - Windows:

     Start the application

     In the popup window, in the Server: box, enter:
     localhost:1

     `Connect`

     This attempt should fail with an SELinux security alert in your guest.

Lab – vnc-server

6. Connect to your guest using the VNC client on your host:
   - Mac OS X:

     Start the application
     `Connection | New Connection`

     In the popup window, enter:
     Host: localhost
     Display: 1
     Password: vle$vnc

     `Connect`

     This attempt should fail with an SELinux security alert in your guest.

Lab – vnc-server

6. Connect to your guest using the VNC client on your host:
   - Linux:
     `vncviewer localhost:1`

   This attempt should fail with an SELinux security alert in your guest.

Lab – vnc-server

7. Examine SELinux security alert

   Four ways to accomplish this:

   GUI:

   - Click the SELinux alert icon
   - `Applications | System Tools | SELinux Troubleshooter`
   - From the command line: `sealert`

   Plain text output:

   - From the command line:

```
sudo sealert -a /var/log/audit/audit.log | less
```

Lab – vnc-server

8. Examine Booleans

Command line:

- `sudo semanage boolean -l`
- `sudo semanage boolean -l | less`
- `sudo semanage boolean –l | grep ssh`

GUI:

- `System | Administration | SELinux Management`

Select Boolean
Filter by string, e.g., ssh
Check or uncheck desired Boolean(s)

Lab – vnc-server

9. Update Boolean

Command line:

- `sudo setsebool -P sshd_forward_ports 1`

GUI:

- System | Administration | SELinux Management

Lab – VNC server

10. Again, connect to your guest using the VNC client on your host

    This time you should see a popup asking for the VNC password. Enter VNC password: vle$vnc

    This attempt should succeed!

Key points

- SELinux prevented sshd on your guest from connecting port 5901 from your host to port 5901 on your guest
- We told SELinux to permanently allow this connection by finding the right Boolean
- Your guest never unblocked firewall port 5901

SELinux Policy Theory

SELinux policy
Overview

- Behavior of processes is controlled by policy
- A base set of policy files define the system policy
- Additional installed software may specify additional policy
  - This policy is added to the system policy on installation

SELinux policy
Six easy pieces

- Type enforcement (TE) attributes
- TE type declarations
- TE transition rules
- TE change rules (not used much)
- TE access vector rules
- File context specifications

TE attributes

- Files named *.te
- Attributes identify sets of types with similar properties
  - SELinux does not interpret attributes
- Format:
  - <attribute> <name>
- Examples:
  - `attribute logfile;`
  - `attribute privuser;`

TE type declarations

- Files named *.te
- Defines type names, with optional aliases and attributes
- Format:
  - type <name> [alias <aliases>] [atttributes]
- Examples:
  - `type mailman_log_t, file_type, sysadmfile, logfile;`
  - `type man_t alias catman_t;`

TE transition rules

- Files named *.te
- Specifies allowed type transitions
- Format:
  - type_transition <source> <action> <target>

- Example:
  - `type_transition mysqld_t mysql_db_t:sock_file mysqld_var_run_t;`
    When a process running in the mysqld_t domain accesses a socket labeled with the mysql_db_t type, transition to the mysqld_var_run_t domain.

TE change rules

- Files named *.te
- Specifies the new type to use when relabeling, based on process domain, object type, and object class
- Format:
  - type_change <source> <action> <target>

- Example:
- `type_change rssh_t server_ptynode:chr_file rssh_devpts_t;`
  - *When running in the rssh_t domain, relabel the associated terminal device as a user terminal*

TE access vector rules

- Files named *.te
- Specifies the set of permissions based on a type pair and an object security class.
- Format:
  - <kind> <source> <target> <securityclass>
    <kind> is one of:
    - allow – allow requested access
    - auditallow – allow and log access
    - dontaudit – don't allow and don't log
    - neverallow – stop compilation of policy

TE access vector rules

- Examples
  - allow initrc_t acct_exec_t:file { getattr read execute };
    Processes running in the initrc_t domain have get-attribute, read, and execute access to files of type account_exec_t
  - dontaudit traceroute_t { port_type -port_t }:tcp_socket name_bind;
    Processes running in the traceroute_t domain do not log the denial of a request for name_bind permission on a tcp_socket for all types associated to the port_type attribute (except port_t)
  - auditallow ada_t self:process execstack;
    Processes runnin the the ada_t domain logs the granting of a request to execute code located on the process stack. Note*: a separate rule must exist to grant this permission.*
  - neverallow ~can_read_shadow_passwords shadow_t:file read;
    No subsequent allow rule can permit the shadow password file to be read, except for those rules associated with the can_read_shadow_passwords attribute. Note*: this rule is intended to be used during the compilation of policy files, not to protect a running system.*

File context specifications

- Files named *.fc
- Defines default contexts for files
- Format:
  - <name-re> [file-type][security-context]
- Examples:
  - `/bin/login -- system_u:object_r:login_exec_t:s0`
  - `/var/tmp/logcheck -d system_u:object_r:logrotate_tmp_t`
  - `/etc/tripwire(/.*)? system_u:object_r:tripwire_etc_t`

SELinux Policy Praxis

Lab – examine policy sources

- Download policy sources from web page
  - `rpm -ihv http://www-personal.umich.edu/~cja/SEL12/supp/selinux-policy-3.10.0-75.fc16.src.rpm`
  - `tar zxf serefpolicy-3.10.0.tgz`
  - `cd ~/rpmbuild/SOURCES/serefpolicy-3.10.0`
  - `sudo make install-src`
  - `cd /etc/selinux/refpolicy/src/policy/policy`

Lab – examine policy sources

- Raw Audit Messages :

type=AVC msg=audit(1331774736.845:64): avc: denied { execheap } for pid=1989 comm="selsmash" scontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tcontext=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 tclass=process

type=SYSCALL msg=audit(1331774736.845:64): arch=i386 syscall=mprotect success=no exit=EACCES a0=81fb000 a1=1000 a2=7 a3=0 items=0 ppid=1928 pid=1989 auid=1000 uid=0 gid=1000 euid=0 suid=0 fsuid=0 egid=1000 sgid=1000 fsgid=1000 tty=pts0 ses=2 comm=selsmash exe=/home/cja/selsmash/selsmash subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 key=(null)

Lab – examine policy sources

SELinux audit2allow

audit2allow

- Generates SELinux policy “allow” rules from logs of denied operations
  - Creates installable policy modules
  - A brute-force tool for removing protection
    - You must examine the generated policy modifications carefully
  - Warns if Booleans already exist that achieve the same purpose

audit2allow

- Examples

```
audit2allow -a -m localpol >localpol
audit2allow -a -M localpol
semodule -i localpol.pp
semodule -r localpol.pp
```

audit2allow

- Commands

```
man audit2allow
man semodule
```

Lab – install LogAnalyzer

Goal: install a web application that summarizes system log messages

1. Enable httpd (Apache web server)
   - `sudo yum install php`
   - `sudo service httpd start`
2. Download LogAnalyzer
   - `wget` http://download.adiscon.com/loganalyzer/loganalyzer-3.0.4.tar.gz
   - `tar zxf loganalyzer-3.0.4.tar.gz`
   - `cd loganalyzer-3.0.4`
   - `less Install`

Lab – modify a policy

3. Configure LogAnalyzer
   - `sudo cp -r src/* /var/www/html`
   - `sudo touch /var/www/html/config.php`
   - `sudo chmod 666 /var/www/html/config.php`
4. Install LogAnalyzer
   - Browse to http://localhost/
   - Click the word “here” in the Critical Error Notice
   - What happened?

Lab – modify a policy

5. Generate and install modified SELinux policy
   - `sudo grep http /var/log/audit/audit.log | audit2allow -m lapol >lapol.te`
   - `checkmodule -M -m -o lapol.mod lapol.te`
   - `semodule_package -o lapol.pp -m lapol.mod`
   - `sudo semodule -i lapol.pp`
6. Change the context of the DocumentRoot
   - `sudo chcon -hR -t httpd_sys_script_rw_t /var/www/html`
7. Give Apache access to the system log
   - `sudo setfacl -m u:apache:r /var/log/messages`

Lab – modify a policy

8. Install LogAnalyzer, again
   - Browse to http://localhost/
   - Click the word “here” in the Critical Error Notice
   - Accept all defaults except:
     - Step 7 – Set Syslog file to `/var/log/messages`
9. Revoke un-needed privileges
   - `sudo chmod 644 /var/www/html/config.php`
   - `sudo restorecon -R /var/www/html`

Lab – modify a policy

10. Run LogAnalyzer!
    - Browse to http://localhost/

11. When done with lab:
    - `sudo setfacl -b /var/log/messages`

End Day 2

References

- P. A. Loscocco, S. D. Smalley, P. A. Muckelbauer, R. C. Taylor, S. J. Turner, and J. F. Farrell, "The inevitability of failure: the flawed assumption of security in modern computing environments," *Proceedings of the 21st National Information Systems Security Conference, pp 303–314, Oct. 1998.* *http://csrc.nist.gov/nissc/1998/proceedings/paperF1.pdf*
- Ray Spencer, Stephen Smalley, Peter Loscocco, Mike Hibler, Dave Andersen, and Jay Lepreau, "The Flask Security Architecture: System Support for Diverse Security Policies," Proceedings of the 8th USENIX Security Symposium, Washington D.C., August 1999.
- Loscocco, P. and S. Smalley, "Integrating Flexible Support for Security Policies into the Linux Operating System," Proceedings of the FREENIX Track, Usenix Technical Conference, June 2001.
- Trent Jaeger, Reiner Sailer, and Xiaolan Zhang, "Analyzing Integrity Protection in the SELinux Example Policy," Proc. 12th Usenix Security Symposium, Washington DC, August 2003.
- Fedora Project Documentation Team, "Fedora 11 Security-Enhanced Linux User Guide," Linux Documentation Library, http://www.linbrary.com/.
- D. E. Bell and L. J. La Padula, "Secure computer systems: Mathematical foundations and model," Technical Report M74-244, MITRE Corporation, Bedford, MA, May 1973.
- Bill McCarty, "SELinux: NSA's Open Source Security Enhanced Linux," O'Reilly Media, 2005.
- Richard Petersen, "Fedora 14 Desktop Handbook," Surfing Turtle Press, 2011.
- http://wiki.centos.org/HowTos/SELinux
- http://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-selinux.html