Fun with SELinux

Writing SELinux Policy| Permissive Domains | sVirt

Presented by
Eduard Benes, Miroslav Grepl
ebenes@redhat.com
mgrepl@redhat.com

Today's Topics

1. **Show process of writing policy**
   - understanding basic of SElinux
   - using SELinux tools
   - Permissive Domains
2. **Real examples**
   - creating & testing portreserve policy
   - how to solve real bug (vncserver)
3. **sVirt**
   - introduction and short demo

Known basics?

- the most important part of SELinux
  - type enforcement language
  - everything on a selinux system has a type - processes, files
- security context
  - system_u:object_r:**etc_t**:s0
- policy decision
  - security context labels are used to make access control decisions between processes and objects

Known basics?

- using security context

  # id -Z

  root:staff_r:staff_t

  #cat /etc/shadow

  cat: /etc/shadow: Permission denied

  - audit log records the following:

    avc: denied { **read** } for pid=13653 exe=/bin/cat

    name=shadow dev=hda6 ino=1361441 scontext=root:staff_r:**staff_t**

    tcontext=system_u:object_r:**shadow_t** tclass=**file**

Known basics?

- policy rules

  COMMAND SOURCETYPE TARGETTYPE:CLASS PERMS;

  allow staff_t etc_t:file { open read getattr ioctl lock};

  dontaudit staff_t shadow_t:file { open read getattr ioctl lock};

  - class – file, dir , sock_file
  - perms – macros can be used

    define(`**r_file_perms**', `{ open read getattr lock ioctl }

Setup environment

- Remove portreserve policy
  - semodule -r portreserve.pp
- Fix lables
  - restorecon -R -v $portreserve_files
- Default initrc_t domain
  - unconfined domain
  - for process started by init system
  - process without policy

SELinux process transition

- Transation
  - without transition using service script
    - initrc_t -> *bin_t* -> initrc_t
  - with transition using service script
    - initrc_t ->*portreserve_exec_t* -> portreserve_t
  - run directly – no transition!!
    - unconfined_t->*portreserve_exec_t*->unconfined_t

=> SELinux is all about labels

Generating initial policy

- Using sepolgen or sepolgen-gui
  - give you policy files

```
# sepolgen -t 0 `which portreserve`
Created the following files in:
./
```

portreserve.te # Type Enforcement file

- Contains all the rules used to confine your application

portreserve.fc # File Contexts file

- Contains the regular expression mappings for on disk file contexts

portreserve.if # Interface file

- Contains the interfaces defined for other confined applications, to interact with your confined application

portreserve.sh # Setup Script

Generating initial policy

- Install policy
  - using script
    - # sh portreserve.sh
  - using Makefile
    - # make -f /usr/share/selinux/deve/Makefile
    - # semodule -i portreserve.pp
    - # restorecon -R -v $files
- Do some checks
  - # semodule -l | grep portreserve
  - # ps -eZ | grep portre
  - # ausearch -m avc -ts recent

Permissive Domains

- initial policies are running as permissive domains
  - # permissive domain portreserve_t
- checks are performed but not enforced
- users don't have to switch to permissive mode globally
- we can catch AVC messages
  - # ausearch -m avc -ts recent | grep portreserve
- make domain permissive
  - # semanage permissive -a httpd_t

Complete our policy

- ausearch, audit2allow tools
  - # ausearch -m avc -ts today | grep portreserve | audit2allow -R
- compile and load rules
  - # ausearch -m avc -ts today | grep portreserve | audit2allow -R >> portreserve.te
  - # make -f /usr/share/selinux/devel/Makefile
  - # semodule -i portreserve.pp
- test it without permissive domain
  - # sed -i s/^permissive/#permissive/ portreserve.te

Real bug - vncserver

- new policies for new unconfined services/apps?
  - are not always necessary
    - spamc_t domain type treat a lot of spam apps
    - does not make sense creating new policy for each spam apps
  - policy has many types to use
    - for example vncserver
      - runnig as initrc_t -> causes issues
      - new domain -> would end up as unconfined domain
      - we use unconfined_exec_t for vncserver binary

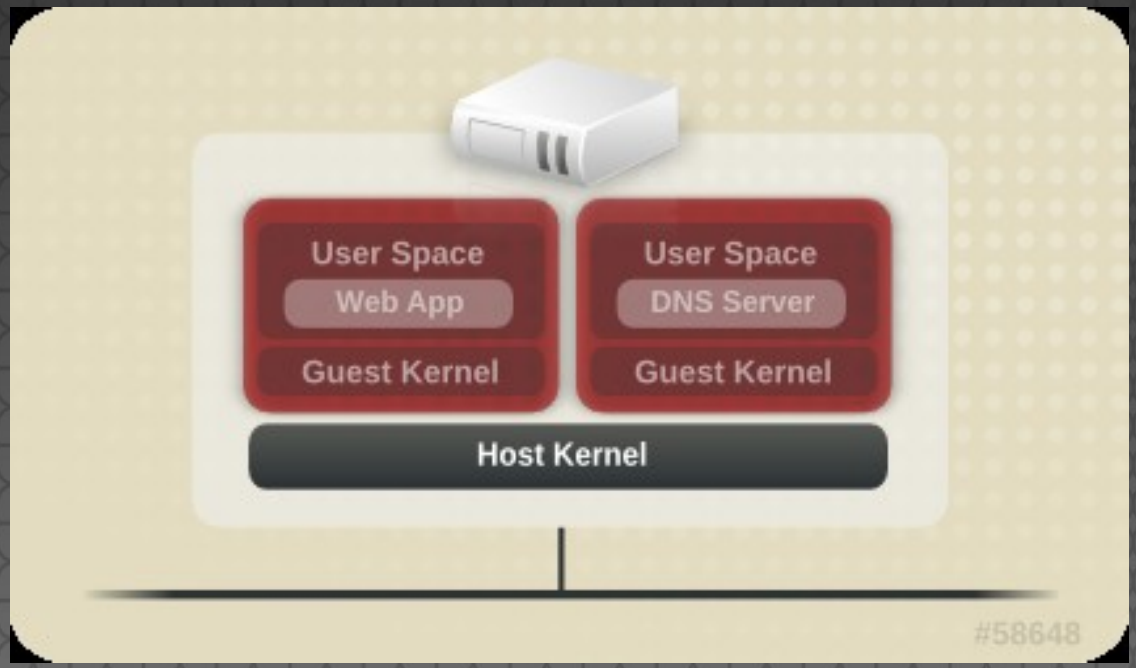
Restore your environment

- load the default policy using semodule
  - # semodule -r portreserve.pp -i /usr/share/selinux/targeted/portreserve.pp.bz2
- fix labels using restorecon
  - # for files in `rpm -ql portreserve | grep -E "(etc|bin|log|lib|run)"`;do restorecon -R -v $files;done;
- remove permissive domain using semanage
  - # semanage permissive -d httpd

sVirt

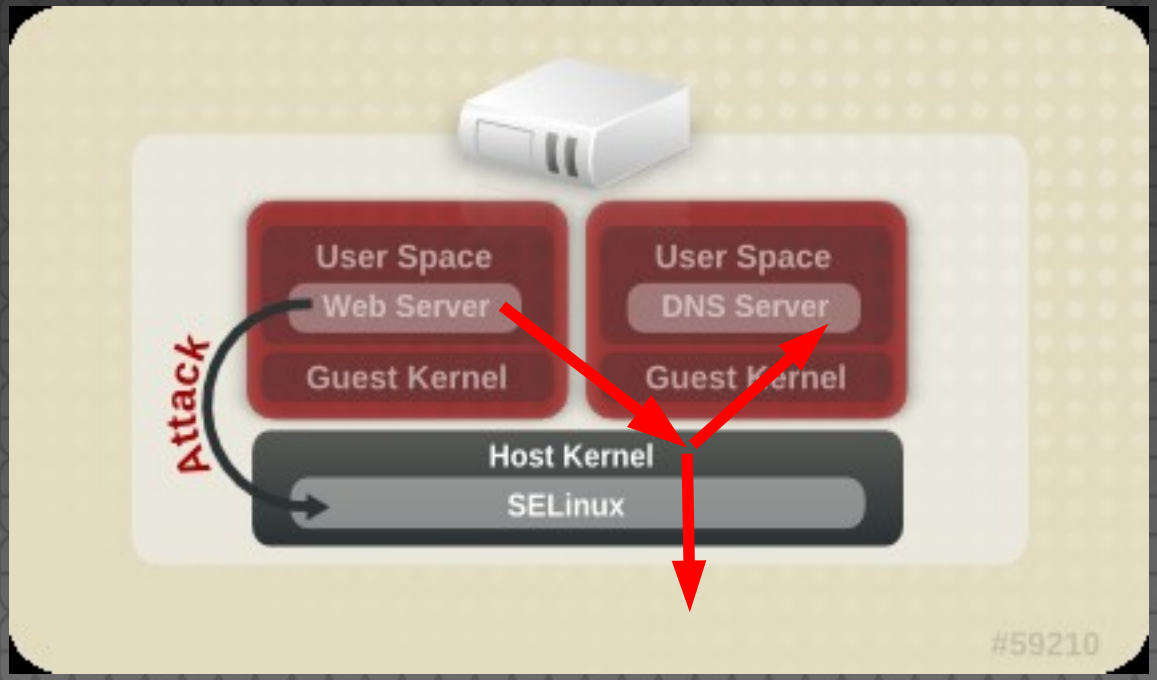
Virtualized Environment

- KVM/Qemu virtualization
- Several OS instances running within a single host kernel and physical host ... clouds



Security and Virtualization

- Hypervisor vulnerabilities
  - Attack host
  - Attack other running guests
  - Not theoretical ... but real



Before sVirt

- “Just” SELinux
- All VMs run in the same security context
- No isolation between individual VMs

Use cases – What does it provide?

- Isolation
  - “Physical” (similar level of isolation)
  - Desktop apps (online banking, PDF/Office)
- Protection
  - Host (untrusted VM guests, grid/cloud)
  - Guest (host flaws)
  - Data (imagine a cloud ... Pepsi vs. Cola)
- Test environment (all the above ;-) ...)

sVirt Labeling

- sVirt is transparent under typical use
- VM process is **labeled** and runs with assigned context (MCS) –> isolation
- Disk images are labeled with a context to match the processes

```
$ ps axZ | grep c411
system_u:system_r:svirt_t:s0:c411,c476 ... /usr/libexec/qemu-kvm -S
$ sudo ls -Z /var/lib/libvirt/images/test1.img
... qemu qemu system_u:object_r:svirt_image_t:s0:c411,c476 test1.img
```

```
$ ps axZ | grep c333
system_u:system_r:svirt_t:s0:c333,c666 ... /usr/libexec/qemu-kvm -S
$ sudo ls -Z /var/lib/libvirt/images/test2.img
... qemu qemu system_u:object_r:svirt_image_t:s0:c333,c666 test2.img
```

Labels - Overview

Type	SELinux Context	Description
VM Processes	system_u: system_r: **svirt_t:MCS1**	MCS1 is a randomly selected MCS field. Currently approximately 500,000 labels are supported. (*running process*)
VM Image	system_u: object_r: **svirt_image_t:MCS1**	Only svirt_t processes with the same MCS fields are able to read/write these image files and devices. (*image for running VM*)
VM Shared Read/Write Content	system_u: object_r: **svirt_image_t:s0**	All svirt_t processes are allowed to write to the svirt_image_t:s0 files and devices. (*disk shared between multiple guests*)
VM Shared Shared Read Only content	system_u: object_r: **virt_content_t:s0**	All svirt_t processes are able to read files/devices with this label. (*readonly CD-ROMs*)
VM Image	system_u: object_r: **virt_image_t:s0**	System default label used when an image exits. No svirt_t virtual processes are allowed to read files/devices with this label. (*powered off VM*)

Demo- lunchbox

Final notes & Questions?

- Do I use it? ... YES ;-)
  - sVirt enabled by default
  - Configured in `/etc/libvirt/qemu.conf`

```
security_driver="none|selinux"
```

Xguest

How to secure your wife/GF

Demo – Kiosk mode

- Provided by xguest package
  - One “example” of SELinux Confined Users
  - Allows users to log in and use Firefox to browse Internet websites
  - All changes are lost at logout

```
$ rpm -qi xguest
 ...
Description :
Installing this package sets up the xguest user
to be
used as a temporary account to switch to or as a
kiosk
user account. The account is disabled unless
SELinux is
```

The End

Summary

- Keep it enforcing ... ;-)
  - use Permissive Domains
  - check labels
    - matchpathcon tool

Links

- http://danwalsh.livejournal.com/
- http://blogs.fedoraproject.org/wp/mgrepl/
- http://people.fedoraproject.org/~dwalsh/SELinux/
- http://docs.fedoraproject.org/en-US/Fedora/13/html-single/Security-Enha
- http://fedoraproject.org/wiki/Features/SVirt_Mandatory_Access_Control
- http://www.redhat.com/f/pdf/summit/dwalsh_350_secure_virt.pdf

Questions?

Contact:
ebenes@redhat.com
mgrepl@redhat.com