Index

System Performance and Security

Computer security is often divided into three distinct master categories, commonly referred to as controls:

- Physical
- Technical
- Administrative

Vulnerability assessments may be broken down into one of two types:

- outside looking in
- inside looking around

System Performance and Security

- Security Policy: the People
  - Managing human activities (includes Security Policy maintenance)
  - Who is in charge of what?
  - Who makes final decision about (false) alarms?
  - When is law-enforcement notified?
- Security Policy: the System
  - Managing system activities
  - Regular system monitoring
  - Log to an external server in case of compromise
  - Monitor logs with logwatch
  - Monitor bandwidth usage inbound and outbound
  - Regular backups of system data

System Performance and Security

- Security in Practice
  - the system serves available resources - the system preserves available resources
  - Host only services you must, and only to those you must, so
    - "Do I need or know to host this?"
    - "Do they need or know to access this?"
    - "Is this consistent with past records of system behavior?"
    - "Have I applied all relevant security updates?"
    - Monitor system resources for vulnerabilities and poor performance

Security Levels

Focused on Red Hat Enterprise Linux but detailing concepts and techniques valid for all Linux systems, this slide details the planning and the tools involved in creating a secured computing host in network for the data center, workplace, and home.

- NetFilter (ipv4 and ipv6)
- SELinux
- libwrap
- xinetd
- PAM
- Application level
- Permissions level

What is NetFilter?

- Firewalls are a basic building block of network security.
- NetFilter is the set of kernel components that actually executes the firewall rules. It can FILTER, NAT and LOG.

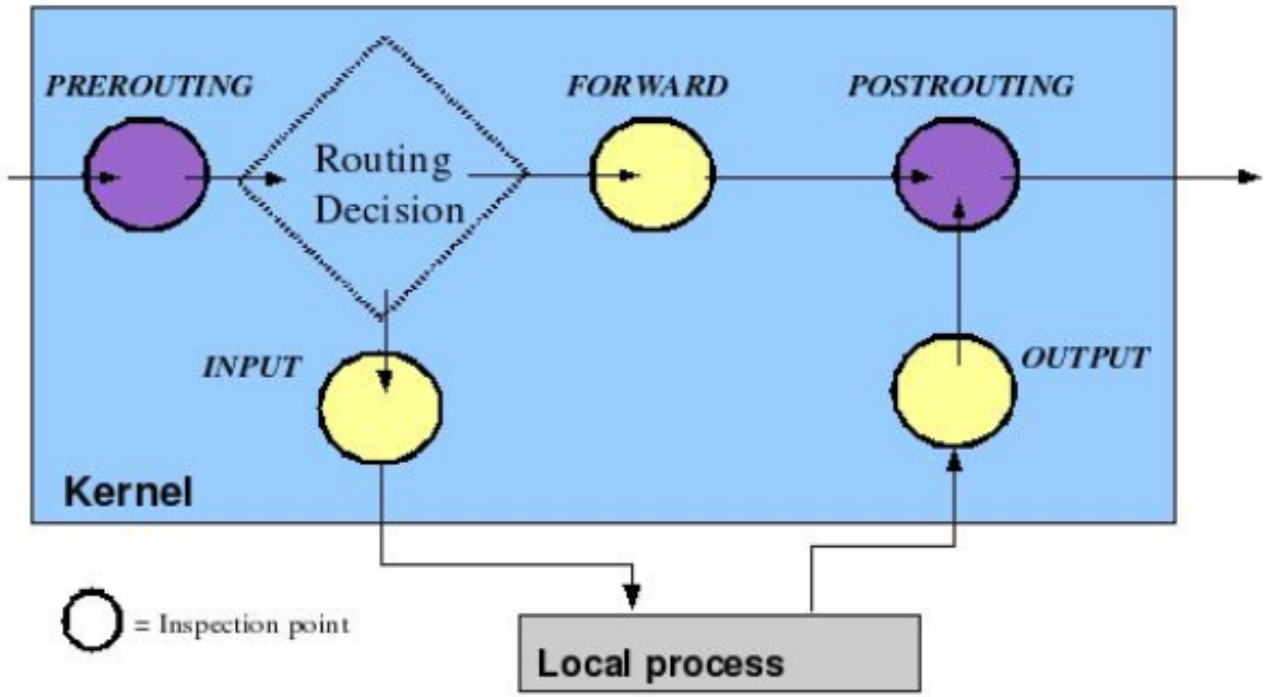


Figure: Netfilter Packet Flow

What is NetFilter?

Remember:
A host-level firewall is one of the key building blocks of a secure infrastructure . The bad guys can't target what they can't talk to.

But, what is NetFilter?:

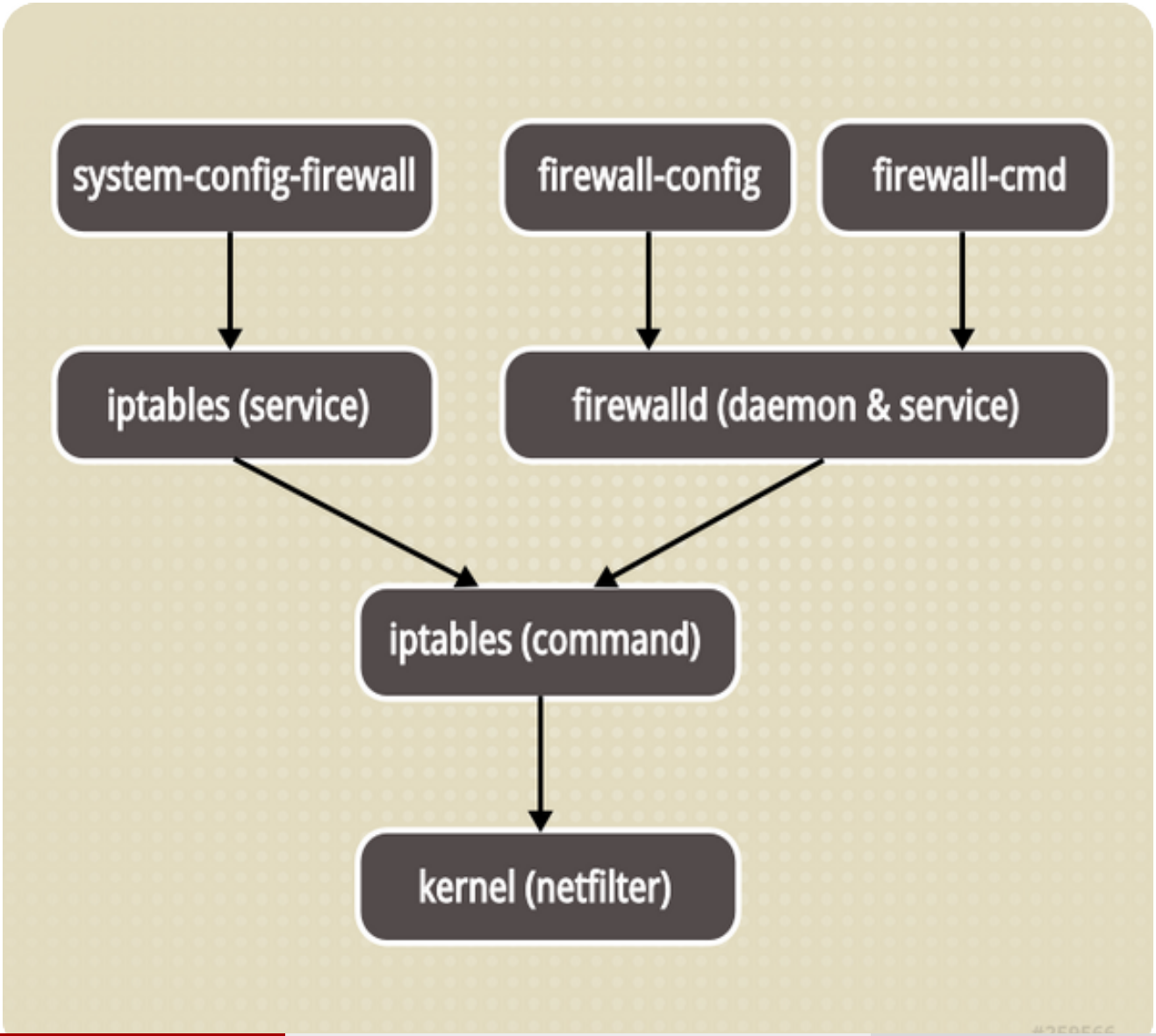
- The NetFilter package is included in the Linux 2.4.x kernel and later kernel series
- Filtering in the kernel: no daemon
- Asserts policies at layers 2, 3 and 4 of the OSI Reference Model
- Only inspects packet headers
- Consists of netfilter modules in kernel, and some daemon in the user space

NetFilter-firewalld Basic Concepts

But, what about NetFilter in Red Hat Enterprise Linux 7?

- In Red Hat Enterprise Linux 7 a new method of interacting with netfilter has been introduced **firewalld**
- firewalld (from the firewalld package) is a modern system to manage NetFilter
- firewalld daemon can do on-the-fly firewall changes
- Other applications (like NetworkManager) can interface with firewalld.

NetFilter-firewalld Basic Concepts



NetFilter-firewalld Basic Concepts

How firewalld works?

- firewalld separates all incoming traffic into zones. Traffic routed into zones based on source address, in first place, and incoming interface, in second place (first rule that matches wins). Default zone used when no source match.
- Different zones can have different access rules.
- Unmatched traffic sent to the default zone, by default the public zone, but can be changed.
- All pre-defined zones documented in firewalld.zones(5) manual page.

firewalld Basic Concepts

Three methods of configuration:

- Configuration files in /etc/firewalld
- firewall-config graphical tool
- firewall-cmd command-line tool

But, before:

[root@rht ~]# systemctl mask iptables ip6tables ebtables
[root@rht ~]# systemctl status firewalld

NetFilter-firewalld Basic Concepts

In this talk, we see the command line tool that supports all firewall features. firewall-cmd permits

- A **runtime configuration**, it is not permanent and will only be restored for a reload. After restart or stop of the service or a system reboot, these options will be gone.
- A **permanent configuration**, the configuration is stored in config files and will be restored with every machine boot or service reload or restart.
- Query modes!!!

General Rules in NetFilter

First,

[root@rht ~]# firewall-cmd - -get-zones

[root@rht ~]# firewall-cmd - -get-default-zone

[root@rht ~]# firewall-cmd - -get-active-zones

[root@rht ~]# firewall-cmd - -list-all-zones

[root@rht ~]# firewall-cmd - -list-all [- -zone=<ZONE>]

General Rules in NetFilter

Next, to set a rule in runtime:

[root@rht ~]# firewall-cmd - -set-default-zone=<**zone**>

[root@rht ~]# firewall-cmd - -add-source=<**cidr**> [- -zone=<**zone**>]

[root@rht ~]# firewall-cmd - -remove-source=<**cidr**> [- -zone=<**zone**>]

[root@rht ~]# firewall-cmd - -add-interface=<**int**> [- -zone=<**zone**>]

[root@rht ~]# firewall-cmd - -change-interface=<**int**> [--zone=<**zone**>]

General Rules in NetFilter

[root@rht ~]# firewall-cmd - -add-service=<**service**>

[root@rht ~]# firewall-cmd - -remove-service=<**service**>

[root@rht ~]# firewall-cmd - -add-port=<**pto/prot**>

[root@rht ~]# firewall-cmd - -remove-port=<**pto/prot**>

After, setting a rule in runtime we can to it permanent with the same command but adding at the end - **-permanent**

Rich Rules

Rich Rules give administrators more fine-grained control over their firewall.
Two types:

- Direct Rules (require detailed knowledge of iptables syntax)
- Rich Language Rules

Rule ordering per zone, first match always wins, and stop processing, except for logging which continues processing.:

1. Port-forwarding and masquerading rules
2. Logging rules
3. Deny rules
4. Allow rules

Working with Rich Rules

[root@rht ~]# firewall-cmd - -list-rich-rules [- -zone=<**zone**>]

[root@rht ~]# firewall-cmd - -query-rich-rule=[<**rich-rule**>]

[root@rht ~]# firewall-cmd - -add-rich-rule=‘rule family=ipv4 source address=172.25.0.10/32 service name=”http” accept ’

[root@rht ~]# firewall-cmd - -remove-rich-rule=‘rule family=ipv4 source address=172.25.0.10/32 service name=”http” accept ’

Working with Rich Rules

General rule structure, '*rule*[*family* = "*rulefamily* "]'

[*source address* = "*address* " [*invert* = *True*]]

[*destination address* = "*address* "[*invert* = *True*]]

service | port | protocol | icmp | masquerd | forward-port

[*log*]

[*audit*]

[*accept*|*reject*|*drop*]

Rich Rule Examples

[root@rht ∼]# firewall-cmd - -add-rich-rule=‘rule family=ipv4 source address=172.25.0.10/32 service name=”http” accept ’

[root@rht ∼]# firewall-cmd - -add-rich-rule=‘rule family=ipv4 destination address=192.168.0.11/32 accept ’

[root@rht ∼]# firewall-cmd - -add-rich-rule=‘rule family=ipv4 source address=192.168.0.12/32 reject ’

[root@rht ∼]# firewall-cmd - -add-rich-rule=‘rule service name=ftp limit value=2/m accept ’

[root@rht ∼]# firewall-cmd - -add-rich-rule=‘rule protocol value=icmp drop ’

[root@rht ∼]# firewall-cmd - -add-rich-rule=‘rule family=ipv4 source address=192.168.1.0/24 port port=7900-7905 protocol=tcp accept ’

Logging with rich rules Examples

Logging to syslog

Notation:
log [*prefix* = "*prefix text*"] [*level* = "*log level*"]
[*limit value* = "*rate*/*duration*"]

[root@rht ~]# firewall-cmd --add-rich-rule='rule protocol value=icmp **log prefix="icmp " level="notice" limit value="3/m"** drop '

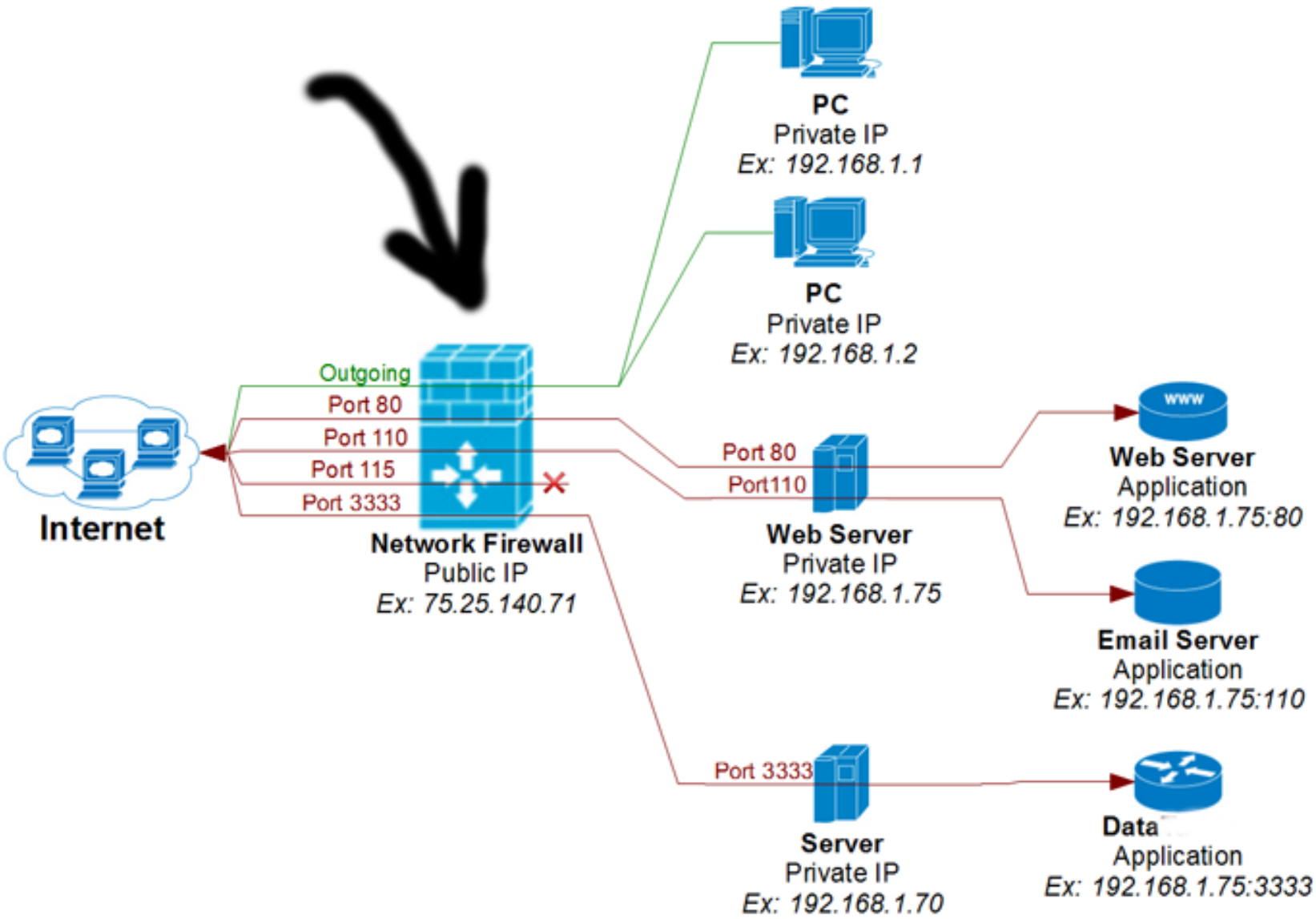
Logging to the kernel audit subsystem

[root@rht ~]# firewall-cmd - -add-rich-rule='rule family=ipv4 source address=192.168.1.0/24 port port=7900-7905 protocol=tcp **audit limit value="1/h"** accept '

Network Address Translation

- firewalld supports two types of NAT
  - masquerading
  - port-forwarding
- Can be configured with regular commands and rich rules.
- Masquerading
  - Used to hide an entire subnet behind a single host.
  - Outgoing packets are rewritten to appear to come from the firewall.
  - Answers to those packets are send back to the original client.
- Port Forwarding
  - Port forwarding forwards traffic addressed to a single port to a different port, or (a different port on) a different machine.
  - When sending traffic to a different machine, answers from that machine must be routed back through the firewall, for example by setting the firewall as default gateway or a custom route.

Network Address Translation



Network Address Translation, Examples of Rules

Masquerade

[root@rht ~]# firewall-cmd [- -zone=<**zone**>] - -add-masquerade

[root@rht ~]# firewall-cmd [- -zone=<**zone**>] - -add-rich-rule=‘rule family=ipv4 source address=192.168.0.0/24 masquerade ’

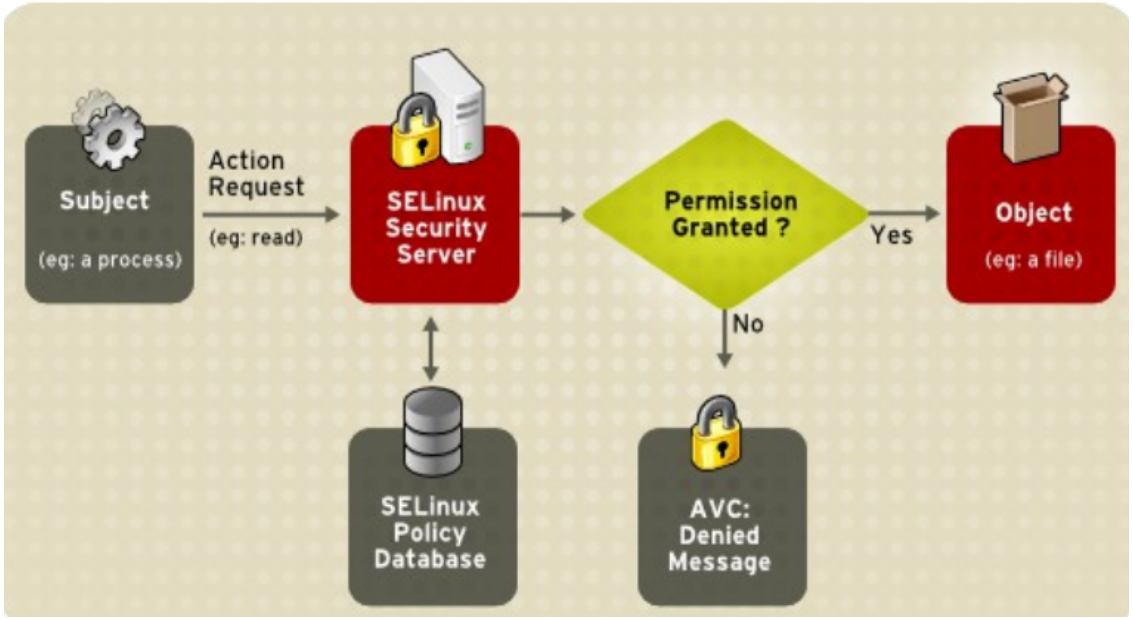
Port Forward

[root@rht ~]# firewall-cmd --add-forward-port=port=513:proto=tcp:toport=80

[root@rht ~]# firewall-cmd --add-rich-rule=‘rule family=ipv4 source address=172.25.*X*.10/32 forward-port port=443 protocol=tcp to-port=22’

What is SELinux?

- SELinux is a **LABELING SYSTEM**
- Every Process has a **LABEL**
- Every File, Directory, System object has a **LABEL**
- Policy rules control access between labeled processes and labeled objects
- The Kernel enforces the rules



What is SELinux?

- SELinux is a **labeling System**
- If the labels are wrong, SELinux will generate issues.
- Solution? Fix your labels.
- If your files are not labeled correctly access might be denied
- Alternative paths for confined domains? SELinux needs to KNOW.
- Example: http files in */srv/myweb* instead of */var/www/html*? Tell SELinux:
  [root@rht ~]# semanage fcontext -a -t httpd_sys_content_t '*/srv/myweb*(/.*)?'
  [root@rht ~]# restorecon -R */srv/myweb*

What is SELinux?

- SELinux File labels definitions stored in

  $$/etc/selinux/targeted/contexts/file\_context.*$$

- semanage fcontext command is used to change default labeling
- File labels are stored in the inode Xattrs
- restorecon command to apply labels to the file system objects

What is SELinux?

- The selinux-policy-devel package has SERVICE_selinux man
- man httpd_selinux lists contexts and booleans for the Apache application
- **setroubleshoot** will tell you if there is a wrong context or-and a boolean available for an AVC
- Logs: */var/log/audit/audit.log*

What is SELinux?

What is SELinux trying to tell you?

1. You have something wrong with your labels
2. You changed the system defaults but did not tell SELinux about it
3. Applications or SELinux has bugs that have not been fixed yet
4. You could be COMPROMIZED!!!

Selinux and Ports

SELinux Port Labeling

- SELinux labels everything, including network ports.
- Can stop rogue services from starting on the ports used by another service.

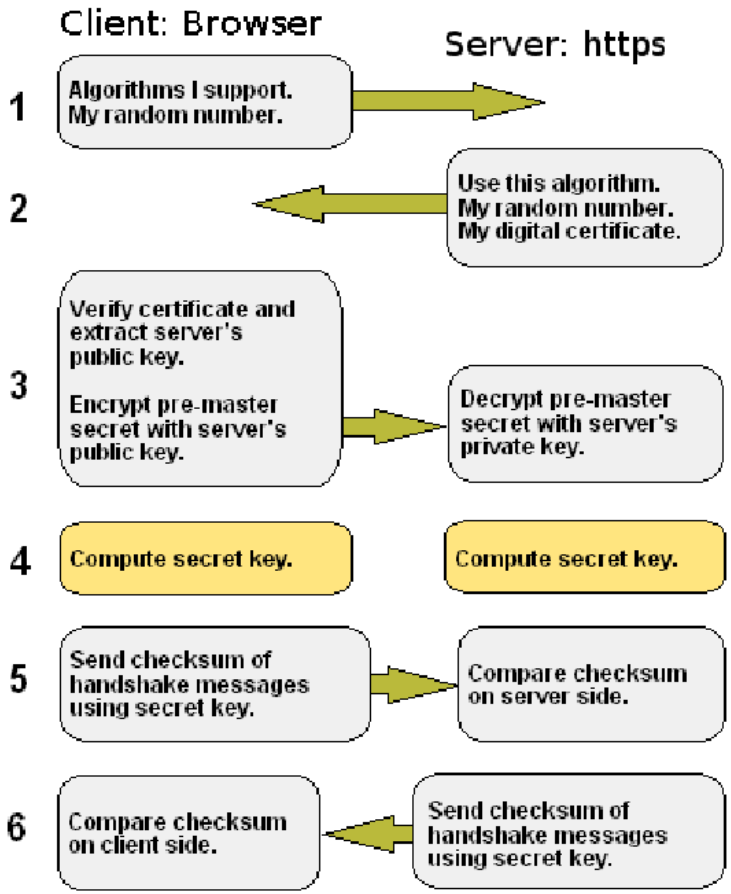
Managing SELinux Port Labeling

- List with **semanage port -l**
- View only local changes with **semanage port -l -C**

Selinux and Ports

- Assign a context to a port with
  [root@rht ~]# semanage port -a -t type -p protocol portnum
- For example,
  [root@rht ~]# semanage port -a -t http_port_t -p tcp 8088
- ports can only have one label associated with them.
- Remove with -r instead of -a
- Modify label with -m

HTTPS: how it work



ISET: https

The four basic steps:

- **Install** Get certificate, add mod_ssl to httpd, configure a virtual host for TLS.
- **Start**

  [root@rht ~]# systemctl start httpd
- **Enable**

  [root@rht ~]# systemctl enable httpd
- **Text** Open a browser and test

Firewalld and SELinux in https

1. Install,
   [root@rht ~]# yum install httpd mod_ssl
2. See file:
   /etc/httpd/conf.d/ssl.conf
3. Configuring, change the defaults.
   [root@rht ~]# mkdir -p */srv/webapp/www*
   [root@rht ~]# vim */srv/webapp/www/index.html*
   [root@rht ~]# restorecon -Rv */srv/*

Firewalld and SELinux in https

3.cont Configuring, change the defaults.

[root@rht ~]# vim /etc/httpd/conf.d/ssl.conf

...

<VirtualHost *:4433>

ServerName server0.example.com

SSLEngine On

SSLProtocol all -SSLv2 -SSLv3

SSLCipherSuite HIGH:MEDIUM:!aNull:!MD5

SSLHonorCipherOrder on

SSLCertificateFile /etc/pki/tls/certs/server0.crt

SSLCertificateKeyFile /etc/pki/tls/private/server0.key

SSLCertificateChainFile /etc/pki/tls/certs/example − ca.crt

DocumentRoot /srv/webapp/www

</VirtualHost>

<Directory /srv/wwwX/www>

Require all granted

</Directory>

Firewalld and SELinux in https

4. Firewall,
   [root@rht ∼]# sudo firewall-cmd - -permanent --add-port=4433
   [root@rht ∼]# sudo firewall-cmd - -reload
5. Selinux and ports,
   [root@rht ∼]# semanage port -l | grep -w http_port_t
   [root@rht ∼]# semanage port -a -t http_port_t -p tcp 4433


danke
謝謝
ngiyabonga
teşekkür ederim
спасибо
dank je
thank you
gracias
tapadh leat
mochchakkeram
bedankt
hvala
mauruuru
dziękuję
go raibh maith agat
sagolun
sukriya
kop khun krap
arigatô
takk
dakujem
grazie
obrigado
terima kasih
мерси
GRACIES
감사합니다
merci
ευχαριστώ