

IT-230

- Firewalls
- Zones and Configuration
- Open Services
- Open Ports
- SELinux Port Labels
- Practice



IT-230 • Linux Administration

Start `workstation` , `servera` , and `serverb` on your VDI



IT-230 • Linux Administration

Firewalls

Who can connect, and how

What a Firewall Does

Every firewall asks one question about a connection: **allow it**, or **refuse it**

- The answer comes from a list of rules
- A rule matches on what the connection looks like
 - Where it came **from**
 - Where it is going **to**
 - Which **port** it is asking for

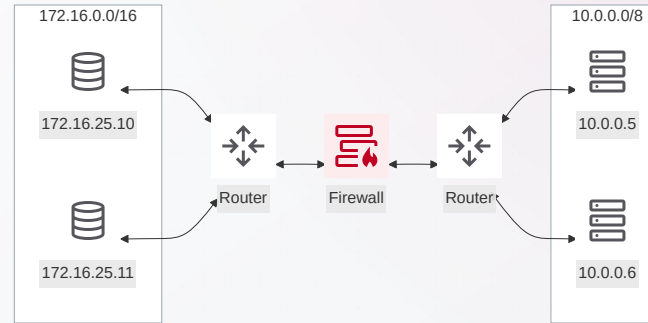
A Network Firewall

Stands between two networks and asks

Can this network talk to that network?

Rules have to name both ends

Each direction needs its own rule



Two rules, one per direction

	FROM	TO	PORT	ACTION
1				
2	10.0.0.0/8	172.16.0.0/16	3306	allow
3	172.16.0.0/16	10.0.0.0/8	any	deny

A Server Firewall

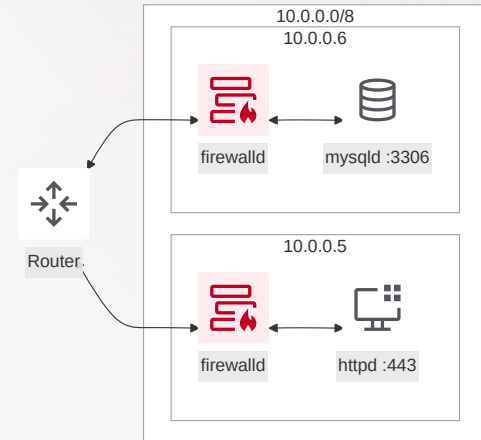
Stands between the network and this host and asks:

Can this network talk to me?

Only sees traffic addressed to itself

Host: 10.0.0.6				
1	FROM	TO	PORT	ACTION
2	10.0.0.0/8	me	3306	allow

Host: 10.0.0.5				
1	FROM	TO	PORT	ACTION
2	172.16.0.0/16	me	443	allow



Firewall

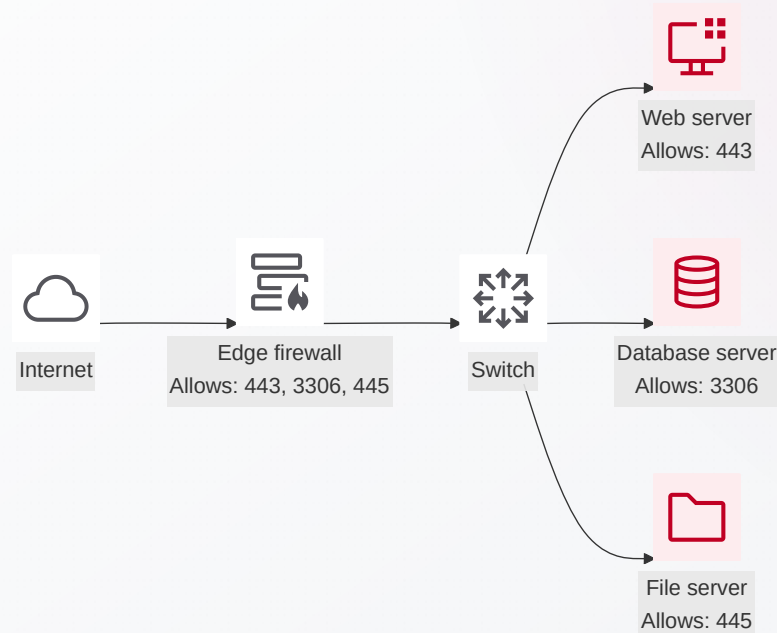
Source Network	Ports	Allowed / Blocked
192.168.1.0/24	20,21,22	Allowed
100.200.225.0/22	80,443	Allowed
100.200.225.18	80,443	Allowed
Any	25,587	Allowed
Any	Any	Blocked

Deny by Default

Anything not specifically allowed is denied

Why Multiple Firewalls?

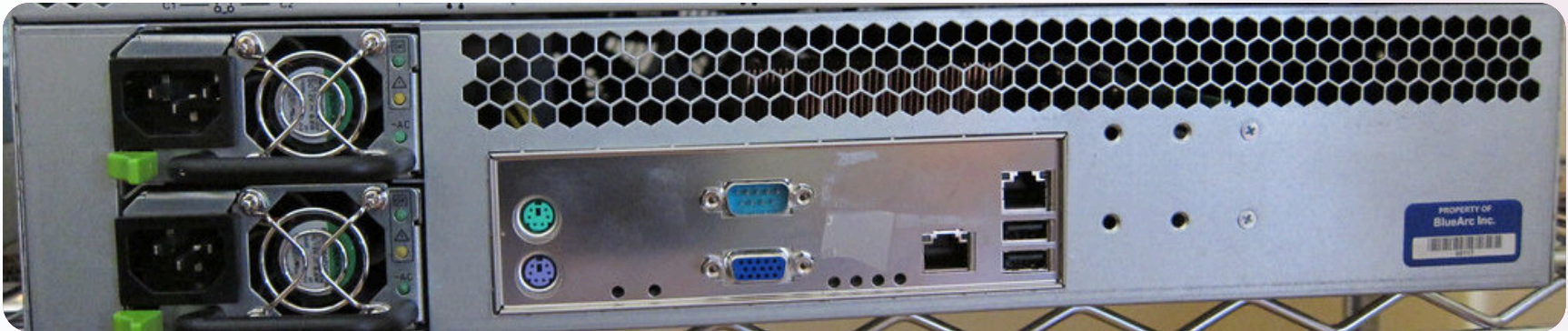
If we have network Firewalls why do we also need firewalls on the server?



Provide **defense in depth** and prevent **lateral movement**

Ports on the Back of the Server

One server, two network interfaces, and each one can sit in a different zone



Photograph by [ChrisDag](#), [CC BY 2.0](#), via Flickr

Interface Names

- The name describes where the card is, not what it does
- Different hardware gives a different name, so look yours up with `nmcli device status`

Environment	Interface to use	Also present
SCC lab	<code>enp0s8</code>	<code>enp0s3</code> , the NAT link
Red Hat Academy lab	<code>ens3</code>	<code>tun0</code> , the VPN
Slides in this deck	<code>enp1s0</code>	none



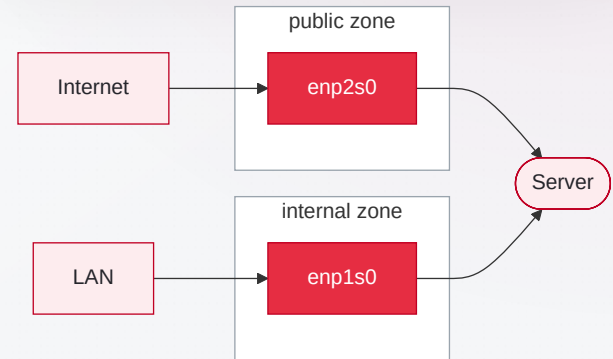
Leave the others alone

Recommend leaving the **"Also present"** interfaces alone when working on assignments

Traffic Flow

- Traffic arrives on an **interface**
 - enp1s0
 - enp2s0
- The interface belongs to a **Firewalld zone**
 - public
 - internal
- The zone decides what is **allowed**
 - 22
 - 80 and 443
 - 3306

Zone	Allowed
public	443
internal	22, 80, 3306



RHEL Firewall Components

- **firewalld**
 - The dynamic manager you configure, and the "front end"
- **nftables**
 - Classifies packets and applies the rules
- **netfilter**
 - The kernel framework that filters, translates, and forwards packets



IT-230 • Linux Administration

Assign the Interface to a Zone

Firewalld Predefined Zones

Ten zones ship with `firewalld`, and each one is an allow list you can customize

Zone	Starts out allowing
<code>trusted</code>	Everything
<code>internal</code>	<code>ssh</code> , <code>mdns</code> , <code>samba-client</code> , <code>dhcpv6-client</code>
<code>public</code>	<code>ssh</code> , <code>dhcpv6-client</code> , and nothing else
<code>dmz</code>	<code>ssh</code> only
<code>block</code>	Nothing, and refuses out loud
<code>drop</code>	Nothing, and stays silent

`home` and `work` start as near copies of `internal`, and `external` adds masquerading

View the Zones

```
student@servera:~  
1 #^ 1. Every zone that exists  
2 student@servera:~$ sudo firewall-cmd --get-zones  
3 block dmz drop external home internal nm-shared public trusted work  
4 #^ 2. The zones actually in use  
5 student@servera:~$ sudo firewall-cmd --get-active-zones  
6 public (default)  
7   interfaces: enp1s0  
8 #^ 3. Where an unassigned interface lands  
9 student@servera:~$ sudo firewall-cmd --get-default-zone  
10 public
```

Nobody assigned `enp1s0` to `public` ; it is there because `public` is the default

Inspect the Active Configuration

`firewall-cmd --list-all` reports everything one zone allows

```
student@servera:~$ sudo firewall-cmd --list-all
public (default, active)
  interfaces: enp1s0
  services: cockpit dhcpv6-client ssh
  ports:
```

Individual ports, empty until you add one

Move an Interface to Another Zone

```
1  #^ 1. Move it
2  student@servera:~$ sudo firewall-cmd --zone=dmz --change-interface=enp1s0
3  success
4  #^ 2. Two zones are active now, but the interface has moved to dmz
5  student@servera:~$ sudo firewall-cmd --get-active-zones
6  dmz
7      interfaces: enp1s0
8  public (default)
9  #^ 3. A shorter list of services applies
10 student@servera:~$ sudo firewall-cmd --list-all --zone=dmz
11 dmz (active)
12 ...output omitted...
13 interfaces: enp1s0
14 services: ssh
```

public allowed **cockpit dhcpv6-client ssh**, and **dmz** allows **ssh** alone

firewall-cmd Options

Option	What it does
<code>--get-zones</code>	Every zone that exists
<code>--get-active-zones</code>	Zones in use
<code>--get-default-zone</code>	Print the default zone
<code>--set-default-zone=ZONE</code>	Change it, everywhere
<code>--change-interface=IFACE</code>	Move an interface

Option	What it does
<code>--list-all</code>	What one zone allows
<code>--add-service=NAME</code>	Allow a service
<code>--add-port=PORT/PROTO</code>	Allow one port
<code>--reload</code>	Reload the permanent config
<code>--runtime-to-permanent</code>	Keep the runtime config

Exercise: Move an Interface to Another Zone

GOAL

Move `servera` 's interface into `dmz` , notice the impacts to what is allowed

ENVIRONMENT

Host: `servera` , from `workstation`

WORKFLOW

1. List the zones that exist, the ones in use, and the default
2. Inspect what the active zone allows
3. Move `enp1s0` into `dmz`
4. Confirm the interface moved and the service list is shorter
5. Run `firewall-cmd --reload`
6. Look again, and see what is left

Exercise: Move an Interface to Another Zone

```
#####  
Exercise: Move an Interface to Another Zone  
Requirements  
  Host: servera, from workstation  
  Move servera's interface into dmz, notice the impacts to what is allowed  
Steps  
  1. List the zones  
  2. Inspect what the active zone allows  
  3. Move enpl1s0 into dmz  
  4. Confirm the interface moved and the service list is shorter  
  5. Reload the firewall  
  6. Look again  
#####  
[student@workstation ~]$ █
```



00:00



Written exercise



IT-230 • Linux Administration

Permanent Config

vs

Runtime Config

Where Did the Change Go?

The reload put `enp1s0` back in `public`

- **Runtime**

- What the firewall is enforcing right now
- Gone at the next `--reload` or `reboot`

- **Permanent**

- What the firewall loads when it starts
- Survives reboots

`--change-interface` wrote to the runtime only, so the reload threw it away

Making Changes Permanent

Test, Then Keep

```
1 root@servera:~# firewall-cmd --add-service=http
2 success
3 root@servera:~# firewall-cmd --runtime-to-permanent
4 success
```

- Test change on runtime config
- Then make it permanent
- Safer

Write, Then Load

```
1 root@servera:~# firewall-cmd --permanent --add-service=http
2 success
3 root@servera:~# firewall-cmd --reload
4 success
```

- Make the change permanently
- Then load it to running config
- More dangerous



IT-230 • Linux Administration

Open Services

Some Predefined Services

Firewalld has **services** so you don't have to memorize ports

Service	Ports	What it is
ssh	22/tcp	Remote login
http	80/tcp	Web server
https	443/tcp	Web server, encrypted
cockpit	9090/tcp	The web console
dns	53/tcp, 53/udp	Name server

The three already open on **servera** are **cockpit**, **dhcpv6-client**, and **ssh**

Firewalld Services

A service is a name for the ports one program needs

```
student@servera:~$ sudo firewall-cmd --get-services
cockpit dhcp dhcpv6-client dns dns-over-tls docker-registry factorio ftp git grafana http http3 https imap libvirt mosh
minecraft mountd mqtt mysql nfs nfs3 ntp opentelemetry pop3 rpc-bind samba smtp ssh syncthing syslog telnet vnc-server
...output omitted...
```

Builtin definitions live in `/usr/lib/firewalld/services/`

```
/usr/lib/firewalld/services/http.xml
<service>
  <short>WWW (HTTP)</short>
  <description>HTTP is the protocol used to serve Web pages. ...</description>
  <port protocol="tcp" port="80"/>
</service>
```

User-created services go in `/etc/firewalld/services/`

Exercise: Allow HTTP Traffic

GOAL

Install a web server on `servera` and let `workstation` reach it

ENVIRONMENT

Host: `servera` , from `workstation`

WORKFLOW

1. Install `httpd` and write a page
2. Enable and start the service
3. Request the page from `workstation` and watch it fail
4. Inspect what the zone allows
5. Allow the `http` service
6. Request the page again
7. Keep the change with `--runtime-to-permanent`

Exercise: Allow HTTP Traffic

```
#####  
Exercise: Allow HTTP Traffic  
Requirements  
  Host: servera, from workstation  
  Install a web server on servera and let workstation reach it  
Steps  
  1. Install httpd and write a page  
  2. Enable and start the service  
  3. Request the page from workstation  
  4. Inspect what the zone allows  
  5. Allow the http service  
  6. Request the page again  
  7. Keep the change  
#####  
[student@workstation ~]$ █
```



00:00



Written exercise



IT-230 • Linux Administration

Open Ports

Non-standard Ports

- Only one process can listen on a port
- A server running several web services has to move the extras somewhere else

```
student@servera:~$ sudo firewall-cmd --add-port=8888/tcp
success
student@servera:~$ sudo firewall-cmd --list-all
public (default, active)
...output omitted...
services: cockpit dhcpv6-client http ssh
ports: 8888/tcp
```

A port is written `port/protocol`, so `8888/tcp` rather than `8888`

Specifying a Port in a URL

`http://hostname:port/path`

Part	Example
Default	<code>http://servera</code>
With port	<code>http://servera:8888</code>
With path	<code>http://servera:8888/index.html</code>

With no port, the browser uses the default for the scheme: **80** for **http** and **443** for **https**

Exercise: Configure Apache to Listen on 8888

GOAL

Move the web server on `servera` to port 8888, and find out what stops it

ENVIRONMENT

Host: `servera` , from `workstation`

Prerequisite exercise: Allow HTTP Traffic

WORKFLOW

1. Allow `8888/tcp` in the firewall, permanently
2. Point Apache's `Listen` line at 8888
3. Restart `httpd` and watch it fail
4. Confirm SELinux is the cause, then put it back
5. Read what `sealert` recommends

Exercise: Configure Apache to Listen on 8888

```
#####  
Exercise: Configure Apache to Listen on 8888  
Requirements  
Host: servera, from workstation  
Prerequisite exercise: Allow HTTP Traffic  
Move the web server to port 8888 and find out what stops it  
Steps  
1. Allow 8888/tcp in the firewall  
2. Point Apache's Listen line at 8888  
3. Restart httpd and watch it fail  
4. Confirm SELinux is the cause  
5. Read what sealert recommends  
#####  
[student@workstation ~]$ █
```



00:00



Written exercise



IT-230 • Linux Administration

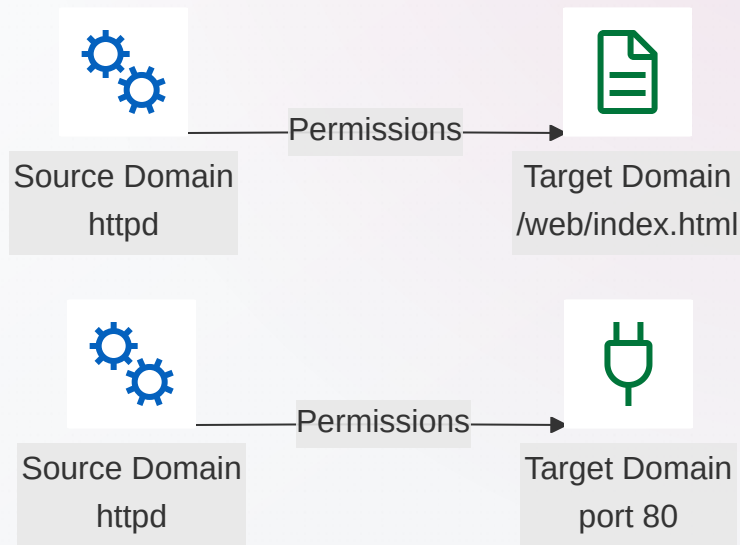
SELinux Port Labels

Ports are targets too

Guess Who's Back, Back Again. SELinux

Last week the **target** was always a file, it can also be a port

- Source Domain
 - Usually a process
- Target Domain
 - A file or directory
 - **Or a port**



SELinux Policy allows the **httpd** process to access ports **80** and **443**

SELinux Ports

- SELinux controls what a process is allowed to reach
- A port is a target, the same way a file is
- A target carries a label, and the policy says which labels `httpd` may bind to

```
student@servera:~$ sudo semanage port -l | grep ^http
http_cache_port_t      tcp      8080, 8118, 8123, 10001-10010
http_cache_port_t      udp      3130
http_port_t            tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
```

8888 is not on the `http_port_t` line, so `httpd` was refused when it asked for it

Labeling a Port

```
semanage port -a -t http_port_t -p tcp 8888
```

The port itself

- It applies at once, with no `restorecon`, because no file is being relabeled
- `-m` changes a label that already exists, and `-d` removes one you added
- `semanage port -l -C` lists your changes alone

Exercise: Configure SELinux to Allow HTTP on 8888

GOAL

Label port 8888 so `httpd` can bind to it, then reach the page from `workstation`

ENVIRONMENT

Host: `servera`, from `workstation`

Prerequisite exercise: Configure Apache to Listen on 8888

WORKFLOW

1. Inspect the ports `httpd` is allowed to bind to
2. Label 8888 with `http_port_t`
3. Start `httpd` and request the page from `workstation`
4. Put `servera` back the way you found it

Exercise: Configure SELinux to Allow HTTP on 8888

```
#####  
Exercise: Configure SELinux to Allow HTTP on 8888  
Requirements  
  Host: servera, from workstation  
  Prerequisite exercise: Configure Apache to Listen on 8888  
  Label port 8888 so httpd can bind to it  
Steps  
  1. Inspect the ports httpd may bind to  
  2. Label 8888 with http_port_t  
  3. Start httpd and request the page from workstation  
  4. Put servera back the way you found it  
#####  
[student@workstation ~]$
```



00:00



Written exercise



IT-230 • Linux Administration

Network Security Practice

The whole path in one pass

Exercise: Serve a Web Page on a Custom Port

GOAL

Serve a page from `workstation` on port 8484 and reach it from `servera`

ENVIRONMENT

Host: `workstation` , from `servera`

WORKFLOW

1. Install `httpd` and write a page
2. Point Apache's `Listen` line at 8484
3. Open `8484/tcp` in the firewall, permanently
4. Label 8484 for SELinux
5. Start `httpd`
6. Request the page from `servera`
7. Put `workstation` back the way you found it

Exercise: Serve a Web Page on a Custom Port

```
#####  
Exercise: Serve a Web Page on a Custom Port  
Requirements  
Host: workstation, from servera  
Serve a page from workstation on port 8484 and reach it from servera  
Steps  
1. Install httpd and write a page  
2. Point Apache's Listen line at 8484  
3. Open 8484/tcp in the firewall  
4. Label 8484 for SELinux  
5. Start httpd  
6. Request the page from servera  
7. Put workstation back the way you found it  
#####  
[student@servera ~]$ █
```



00:00



Written exercise