

LINUX SYSTEM ADMINISTRATION

Level 1

Lavatech Technology

Call: 096073 31234

Email: lavatech.training@gmail.com





YOU CAN

TOTALLY

DO THIS!

Contents

1	Services & systemctl	5
1.1	systemctl in detail	5
1.1.1	Introduction to systemd	6
1.1.2	Systemd units	7
1.1.3	systemctl to manage services	11
1.1.4	systemctl to manage targets	12
1.1.5	Selecting target at boot time	13
1.1.6	Recovering the root password	14
1.1.7	Troubleshooting boot issues	16
1.1.8	Practice	17
2	Firewall in Linux	23
2.1	firewalld in detail	23
2.1.1	Firewall overview	24
2.1.2	firewalld zones	25
2.1.3	firewall-cmd command	26
2.1.4	Practice	32

2.2	Advance firewall settings	32
2.2.1	Introduction to rich rules	33
2.2.2	Rich rules commands	34
2.2.3	Port forwarding	38
2.2.4	Practice	39
3	Domain Name Server	41
3.1	Introduction to DNS	41
3.1.1	What is /etc/hosts file?	42
3.1.2	Problem with /etc/hosts	43
3.1.3	What is DNS?	47
3.1.4	The DNS hierarchy	48
3.1.5	How domain names are assigned?	54
3.1.6	Practice	55
3.2	DNS concepts	55
3.2.1	Types of DNS server	56
3.2.2	8.8.8.8 DNS server	59
3.2.3	Zones in DNS	60
3.2.4	Forward & Reverse zone	61
3.2.5	Forward zone file	62
3.2.6	DNS RESOURCE RECORDS	65
4	Samba Server	67
5	NFS Server	69
6	Teaming and Bridging	71
7	ISCSI Server	73
8	Apache Server	75

Small steps every day.....

1. Services & systemctl

1.1 systemctl in detail

In this section, you are going to learn:

1. What is an Operating System (OS)?
2. Architecture of OS?
3. Types of OS

Finally, there will be a **small exercise** on these topics to check your knowledge.

So let's get started....



1.1.1 Introduction to systemd

- Process ID 1 is systemd, the new init system.
- **Advantages of systemd:**
 - Parallelization capabilities, which **increase the boot speed** of a system.
 - On-demand starting of daemons without requiring a separate service.
 - Tracking related processes together using Linux control groups.
- **What is a daemon?**

Daemons are processes that wait or run in the background performing various tasks.
- **What is a service?**

A service often refers to one or more daemons.

1.1.2 Systemd units

- A unit is a systemd object that performs or controls a particular task or action.
- Systemd uses units to:
 - start/stop/manage services
 - organize boot process
 - maintain tasks and processes
 - create sockets
 - mount file-system
 - initialize hardware
- A systemd unit consists of a name, type, and configuration file.
- Command to list available unit types:

```
# systemctl -t help
```

```
[root@server ~]# systemctl -t help
Available unit types:
service
socket
target
device
mount
automount
swap
timer
path
slice
scope
```

Figure 1.1: Sample output

- Command to list all units of a specific type:

```
# systemctl --type=service
# systemctl --type=socket
```



Location of systemd units file:

- /usr/lib/systemd/system/
- /etc/systemd/system/

Some common unit types are listed as follows:

- **Service units:**
 - Service units represent system services.
 - They have a **".service"** extension.
 - Used to start/stop/restart/reload daemons, such as a web server.
 - Eg: sshd.service unit file is at
/usr/lib/systemd/system/sshd.service
- **Socket units:**
 - What is a socket? A socket consists of the IP address of a system and the port number of a program within the system.



Figure 1.2: Socket

- A socket unit listen on an IP address and a port, and when something connects to it, the socket is handed over to the daemon it is made for.
- They have a **".socket"** extension.
- They represent interprocess communication (IPC) sockets.

- Eg: dbus.socket unit file is at
/usr/lib/systemd/system/dbus.socket
- **Target units:**
 - Targets are used for grouping and ordering units.
 - At different targets, different services, sockets, and other units are started.
 - They have a **".target"** extension.
 - Eg: multi-user.target unit file is at
/usr/lib/systemd/system/multi-user.target

1.1.3 systemctl to manage services

Command	Purpose
systemctl start service Eg: systemctl start sshd	Start a service/daemon.
systemctl status service Eg: systemctl status sshd	Display status of a service/daemon.
systemctl stop service Eg: systemctl stop sshd	Stop a service/daemon.
systemctl restart service Eg: systemctl restart sshd	Restart a service/daemon. The PID of daemon will change.
systemctl reload service Eg: systemctl restart sshd	Restart a service/daemon. The PID of daemon will change.
systemctl reload service Eg: systemctl restart sshd	Reload a service/daemon. The process ID of daemon will not change.
systemctl enable service Eg: systemctl enable sshd	Enables a service/daemon permanently, which means service/daemon will automatically start on system boot up.
systemctl disable service Eg: systemctl enable sshd	Disable a service/daemon permanently, which means service/daemon will not automatically start on system boot up.
systemctl list-dependencies service Eg: systemctl list-dependencies sshd	List units which are required and wanted by the specified unit.

1.1.4 systemctl to manage targets

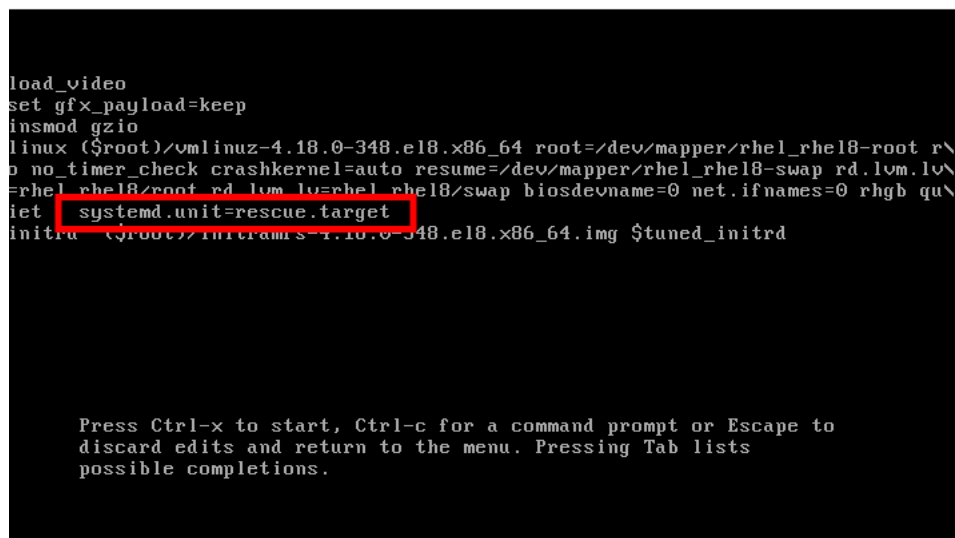
Systemd targets	Purpose
graphical.target	System supports multiple users, graphical and text-based logins.
multi-user.target	System supports multiple users, text-based logins only.
rescue.target	sulogin prompt, basic system initialization completed.
emergency.target	sulogin prompt, initramfs pivot complete and system root mounted on / read-only.

Commands to manage targets:

Command	Purpose
systemctl set-default target Eg: systemctl set-default graphical.target	Set a default target/runlevel so that the system boots into that specific target.
systemctl get-default	Get default target/runlevel.
systemctl isolate target Eg: systemctl isolate multi-user.target	Switch to a specific target instantly.

1.1.5 Selecting target at boot time

1. (Re)boot the system.
2. Interrupt the boot loader menu countdown by pressing any key.
3. Move the cursor to the entry to be started.
4. **Press e** to edit the current entry.
5. Move the cursor to the line that starts with **linux**. This is the kernel command line.
6. Append **systemd.unit=desired.target**.
eg: systemd.unit=multi-user.target



```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-4.18.0-348.el8.x86_64 root=/dev/mapper/rhel_rhel8-root r\
o no_timer_check crashkernel=auto resume=/dev/mapper/rhel_rhel8-swap rd.lvm.lv\
=rhel_rhel8/root rd.lvm.lv=rhel_rhel8/swap biosdevname=0 net.ifnames=0 rhgb qu\
iet systemd.unit=rescue.target
initrd ($root)/initramfs-4.18.0-348.el8.x86_64.img $tuned_initrd

Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists
possible completions.
```

Figure 1.3: Sample output

7. Press **Ctrl+x** to boot with these changes.

1.1.6 Recovering the root password

1. (Re)boot the system.
2. Interrupt the boot loader menu countdown by pressing any key.
3. Move the cursor to the entry to be started.
4. **Press e** to edit the current entry.
5. Move the cursor to the line that starts with `linux16`. This is the kernel command line.
6. Append **`rd.break`** (this will break just before control is handed from the `initramfs` to the actual system).
7. Press **`Ctrl+x`** to boot with these changes.

At this point, a root shell will be presented, with the root file system for the actual system mounted read-only on `/sysroot`.

To recover the root password from this point, use the following procedure:

Remount `/sysroot` as read-write.

```
switch_root:/# mount -o remount,rw /sysroot
```

2. Switch into a `chroot` jail, where `/sysroot` is treated as the root of the file system tree.

```
switch_root:/# chroot /sysroot
```

3. Set a new root password:

```
sh-4.2# passwd root
```

4. Make sure that all unlabeled files (including `/etc/shadow` at this point) get relabeled during boot.

```
sh-4.2# touch /.autorelabel
```

5. Type **`exit`** twice. The first will exit the **`chroot`** jail, and the

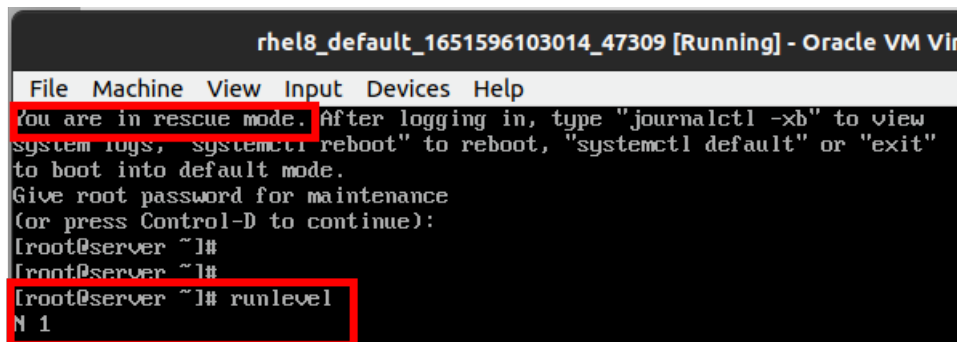
second will exit the initramfs debug shell.

At this point, the system will continue booting, perform a full SELinux relabel, then reboot again.

1.1.7 Troubleshooting boot issues

If your system is getting stuck during system boot, you can troubleshoot the system by booting it into rescue or emergency mode:

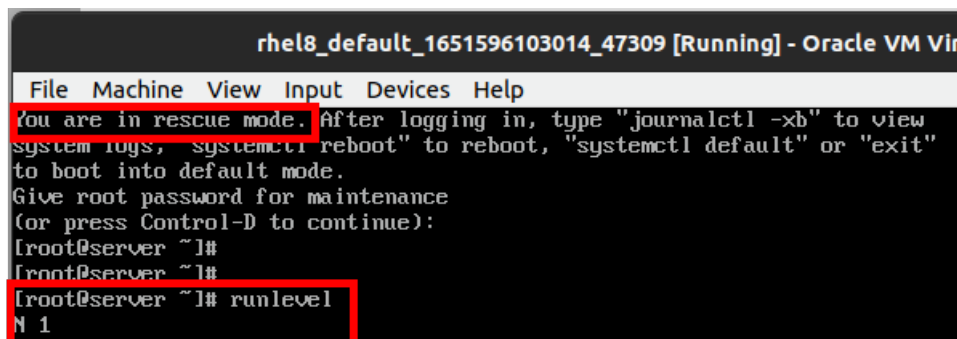
- Append either **systemd.unit=rescue.target** or **systemd.unit=emergency.target** to the kernel command line to boot the system into a rescue or emergency shell instead of starting normally.
- Both of these shells require the root password.
- The **emergency.target** keeps the root file system mounted read-only.



```
rhel8_default_1651596103014_47309 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
You are in rescue mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "exit"
to boot into default mode.
Give root password for maintenance
(or press Control-D to continue):
[root@server ~]#
[root@server ~]#
[root@server ~]# runlevel
N 1
```

Figure 1.4: Sample output

- The **rescue.target** waits for **sysinit.target** to complete first so that more of the system will be initialized, for example, logging, file systems, etc.



```
rhel8_default_1651596103014_47309 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
You are in rescue mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "exit"
to boot into default mode.
Give root password for maintenance
(or press Control-D to continue):
[root@server ~]#
[root@server ~]#
[root@server ~]# runlevel
N 1
```

Figure 1.5: Sample output

- Exiting from these shells will continue with the regular boot process.

1.1.8 Practice

- List all the available zones:

Syntax: `firewall-cmd --get-zones`

Eg:

```
[root@server ~]# firewall-cmd --get-zones
block dmz drop external home internal nm-shared public trusted work
```

Figure 1.6: Sample output

- Display the default zones:

Syntax: `firewall-cmd --get-default-zone`

Eg:

```
[root@server ~]# firewall-cmd --get-default-zone
public
```

Figure 1.7: Sample output

- Set the default zone:

Syntax: `firewall-cmd --set-default-zone=<zone>`

Eg:

```
[root@server ~]# firewall-cmd --set-default-zone=external
success
[root@server ~]#
[root@server ~]# firewall-cmd --get-default-zone
external
```

Figure 1.8: Sample output

- Display information for all zones (interfaces, sources, ports, services, etc).

Syntax: `firewall-cmd --list-all-zones`

- Display information for specific zone (interfaces, sources, ports, services, etc). If no "`--zone=`" option is provided, the default zone will be used.

Syntax: `firewall-cmd --list-all [-zone=<ZONE>]`

Eg:

```
[root@server ~]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: eth0 eth1
  sources:
  services: cockpit dhcpv6-client ssh
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Figure 1.9: Sample output

- Apply changes done by **firewall-cmd** command immediately:

Syntax: `firewall-cmd --reload`

- Allow traffic to <SERVICE>. If no `--zone=` option is provided, the default zone will be used.

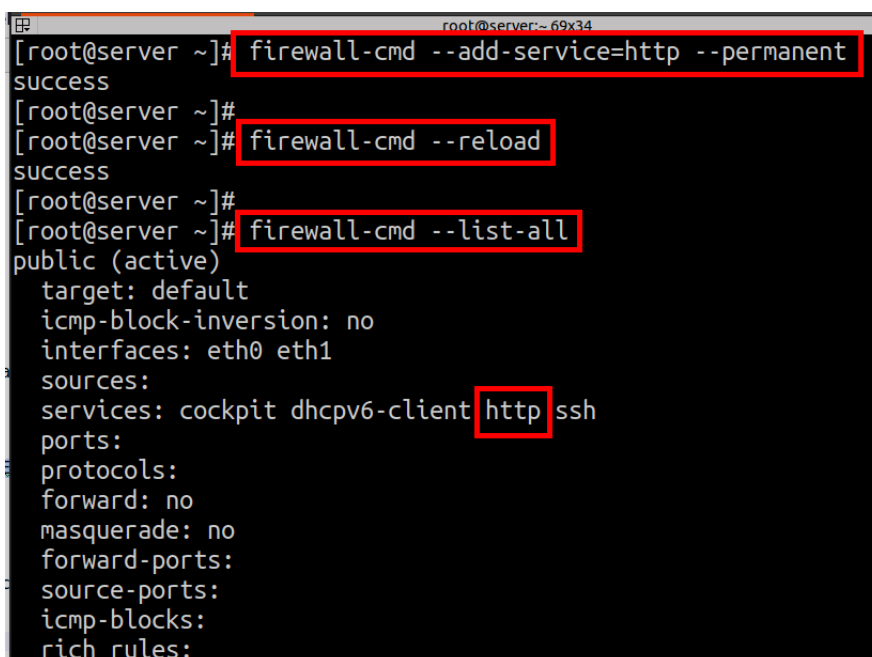
Syntax:

`firewall-cmd --add-service=<service> --permanent`
`[-zone=<ZONE>]`

`firewall-cmd --reload`

Eg:

```
# firewall-cmd --add-service=httpd --permanent
# firewall-cmd --reload
```



```
root@server:~ 69x34
[root@server ~]# firewall-cmd --add-service=http --permanent
success
[root@server ~]#
[root@server ~]# firewall-cmd --reload
success
[root@server ~]#
[root@server ~]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: eth0 eth1
  sources:
  services: cockpit dhcpv6-client http ssh
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Figure 1.10: Sample output

- Remove <SERVICE> from the allowed list for the zone. If no "--zone=" option is provided, the default zone will be used.

Syntax:

```
firewall-cmd  --remove-service=<service>  --permanent  
[-zone=<ZONE>]
```

```
firewall-cmd  --reload
```

Eg:

```
# firewall-cmd --remove-service=httpd --permanent  
# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --remove-service=http --permanent  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --reload  
success  
[root@server ~]#  
[root@server ~]#  
[root@server ~]# firewall-cmd --list-all  
public (active)  
  target: default  
  icmp-block-inversion: no  
  interfaces: eth0 eth1  
  sources:  
  services: cockpit dhcpv6-client ssh  
  ports:  
  protocols:  
  forward: no  
  masquerade: no  
  forward-ports:  
  source-ports:  
  icmp-blocks:  
  rich rules:
```

Figure 1.11: Sample output

- Allow traffic to the **<PORT/PROTOCOL>** port(s). If no "**-zone=**" option is provided, the default zone will be used.

Syntax:

```
firewall-cmd      --add-port=<PORT/PROTOCOL>      —  
permanent  [-zone=<ZONE>]
```

```
firewall-cmd  --reload
```

Eg:

```
# firewall-cmd --add-port=6060/tcp --permanent  
# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --add-port=6060/tcp --permanent  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --reload  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --list-all  
public (active)  
  target: default  
  icmp-block-inversion: no  
  interfaces: eth0 eth1  
  sources:  
  services: cockpit dhcpv6-client ssh  
  ports: 6060/tcp  
  protocols:  
  forward: no  
  masquerade: no  
  forward-ports:  
  source-ports:  
  icmp-blocks:  
  rich rules:
```

Figure 1.12: Sample output

- Remove the <PORT/PROTOCOL> port(s) from the allowed list for the zone. If no "-zone=" option is provided, the default zone will be used.

Syntax:

```
firewall-cmd    --remove-port=<PORT/PROTOCOL>    --  
permanent    [-zone=<ZONE>]
```

```
firewall-cmd    --reload
```

Eg:

```
# firewall-cmd --remove-port=6060/tcp --permanent  
# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --remove-port=6060/tcp --permanent  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --reload  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --list-all  
public (active)  
  target: default  
  icmp-block-inversion: no  
  interfaces: eth0 eth1  
  sources:  
  services: cockpit dhcpv6-client ssh  
  ports:  
  protocols:  
  forward: no  
  masquerade: no  
  forward-ports:  
  source-ports:  
  icmp-blocks:  
  rich rules:
```

Figure 1.13: Sample output

Small steps every day.....

2. Firewall in Linux

2.1 firewalld in detail

In this section, you are going to learn:

1. **Filesystem Hierarchy Standard (FHS)**
2. **Important directories in FHS**
3. **Some shortcuts to use in Linux OS**

Finally, there will be a **small exercise** on these topics to check your knowledge.

So let's get started....



2.1.1 Firewall overview

- A firewall is a security system that prevent unauthorized access to a computer network.
- **Firewalls guard traffic at a computer's ports**, where information is exchanged with external devices.

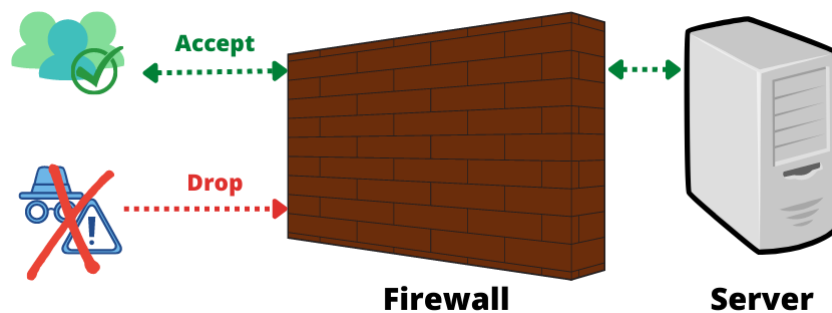


Figure 2.1: Firewall

- **firewalld** is the daemon for firewall in RHEL.
- Service unit for this daemon is **firewalld.service**.

2.1.2 firewalld zones

- The firewalld separates all incoming traffic into zones.
- Each zone have its own set of rules.
- Below are predefined zones with firewalld, having different usage:

Zone name	Default configuration
trusted	Allow all incoming traffic.
home	Reject incoming traffic except the ssh, mdns, ipp-client, samba-client, or dhcpv6-client services
internal	Reject incoming traffic except ssh, ipp-client, or dhcpv6-client services.
public	Reject incoming traffic ssh or dhcpv6-client services.
external	Reject incoming traffic except ssh service. Masqueraded IPv4 address of the outgoing network interface.
dmz	Reject incoming traffic except outgoing traffic for ssh service.
block	Reject all incoming traffic unless related to outgoing traffic.
drop	Drop all incoming traffic unless related to outgoing traffic (do not even respond with ICMP errors).

- The **public zone** is the default zone for network interfaces.

2.1.3 firewall-cmd command

- List all the available zones:

Syntax: `firewall-cmd --get-zones`

Eg:

```
[root@server ~]# firewall-cmd --get-zones
block dmz drop external home internal nm-shared public trusted work
```

Figure 2.2: Sample output

- Display the default zones:

Syntax: `firewall-cmd --get-default-zone`

Eg:

```
[root@server ~]# firewall-cmd --get-default-zone
public
```

Figure 2.3: Sample output

- Set the default zone:

Syntax: `firewall-cmd --set-default-zone=<zone>`

Eg:

```
[root@server ~]# firewall-cmd --set-default-zone=external
success
[root@server ~]#
[root@server ~]# firewall-cmd --get-default-zone
external
```

Figure 2.4: Sample output

- Display information for all zones (interfaces, sources, ports, services, etc).

Syntax: `firewall-cmd --list-all-zones`

- Display information for specific zone (interfaces, sources, ports, services, etc). If no "`--zone=`" option is provided, the default zone will be used.

Syntax: `firewall-cmd --list-all [-zone=<ZONE>]`

Eg:

```
[root@server ~]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: eth0 eth1
  sources:
  services: cockpit dhcpv6-client ssh
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Figure 2.5: Sample output

- Apply changes done by **firewall-cmd** command immediately:

Syntax: `firewall-cmd --reload`

- Allow traffic to <SERVICE>. If no `--zone=` option is provided, the default zone will be used.

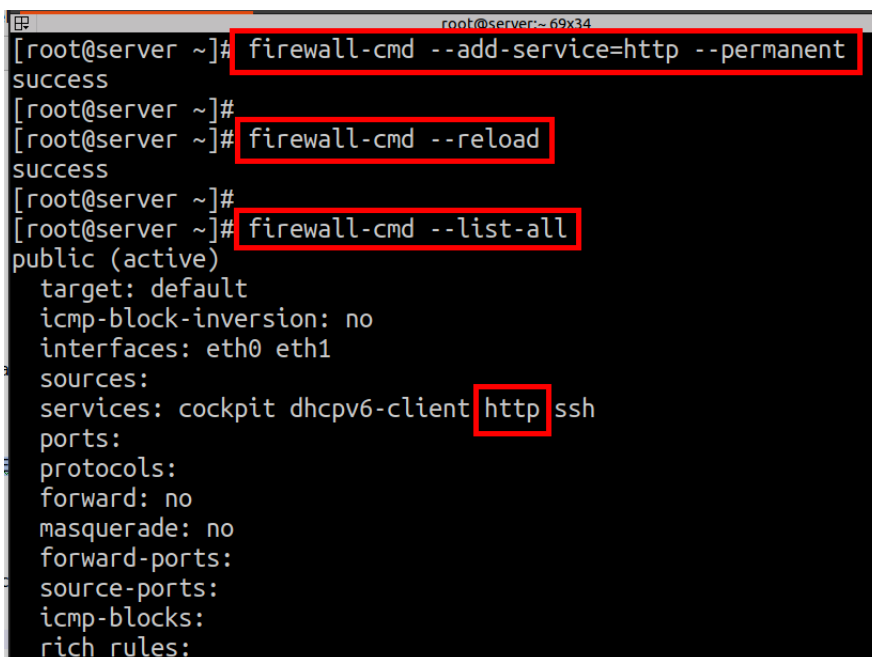
Syntax:

`firewall-cmd --add-service=<service> --permanent`
`[-zone=<ZONE>]`

`firewall-cmd --reload`

Eg:

```
# firewall-cmd --add-service=httpd --permanent
# firewall-cmd --reload
```

A terminal window screenshot showing the execution of firewall-cmd commands. The commands are highlighted with red boxes: 'firewall-cmd --add-service=http --permanent', 'firewall-cmd --reload', and 'firewall-cmd --list-all'. The output shows the firewall is in 'public (active)' state with various settings like target, interfaces, and services. The 'http' service is listed under services.

```
root@server:~ 69x34
[root@server ~]# firewall-cmd --add-service=http --permanent
success
[root@server ~]#
[root@server ~]# firewall-cmd --reload
success
[root@server ~]#
[root@server ~]# firewall-cmd --list-all
public (active)
target: default
icmp-block-inversion: no
interfaces: eth0 eth1
sources:
services: cockpit dhcpv6-client http ssh
ports:
protocols:
forward: no
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
```

Figure 2.6: Sample output

- Remove <SERVICE> from the allowed list for the zone. If no "--zone=" option is provided, the default zone will be used.

Syntax:

```
firewall-cmd  --remove-service=<service>  --permanent  
[-zone=<ZONE>]
```

```
firewall-cmd  --reload
```

Eg:

```
# firewall-cmd --remove-service=httpd --permanent  
# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --remove-service=http --permanent  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --reload  
success  
[root@server ~]#  
[root@server ~]#  
[root@server ~]# firewall-cmd --list-all  
public (active)  
  target: default  
  icmp-block-inversion: no  
  interfaces: eth0 eth1  
  sources:  
  services: cockpit dhcpv6-client ssh  
  ports:  
  protocols:  
  forward: no  
  masquerade: no  
  forward-ports:  
  source-ports:  
  icmp-blocks:  
  rich rules:
```

Figure 2.7: Sample output

- Allow traffic to the **<PORT/PROTOCOL>** port(s). If no "**-zone=**" option is provided, the default zone will be used.

Syntax:

```
firewall-cmd      --add-port=<PORT/PROTOCOL>      --  
permanent  [-zone=<ZONE>]
```

```
firewall-cmd  --reload
```

Eg:

```
# firewall-cmd --add-port=6060/tcp --permanent  
# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --add-port=6060/tcp --permanent  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --reload  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --list-all  
public (active)  
  target: default  
  icmp-block-inversion: no  
  interfaces: eth0 eth1  
  sources:  
  services: cockpit dhcpv6-client ssh  
  ports: 6060/tcp  
  protocols:  
  forward: no  
  masquerade: no  
  forward-ports:  
  source-ports:  
  icmp-blocks:  
  rich rules:
```

Figure 2.8: Sample output

- Remove the <PORT/PROTOCOL> port(s) from the allowed list for the zone. If no "-zone=" option is provided, the default zone will be used.

Syntax:

```
firewall-cmd    --remove-port=<PORT/PROTOCOL>    --  
permanent    [-zone=<ZONE>]
```

```
firewall-cmd    --reload
```

Eg:

```
# firewall-cmd --remove-port=6060/tcp --permanent  
# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --remove-port=6060/tcp --permanent  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --reload  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --list-all  
public (active)  
  target: default  
  icmp-block-inversion: no  
  interfaces: eth0 eth1  
  sources:  
  services: cockpit dhcpv6-client ssh  
  ports:  
  protocols:  
  forward: no  
  masquerade: no  
  forward-ports:  
  source-ports:  
  icmp-blocks:  
  rich rules:
```

Figure 2.9: Sample output

2.1.4 Practice

2.2 Advance firewall settings

In this section, you are going to learn:

1. **Basic commands in Linux OS**
2. **Advance commands in Linux OS**

Finally, there will be a **small excerise** on these topics to check your knowledge.

So let's get started....



2.2.1 Introduction to rich rules

- firewalld rich rules allows custom firewall rules.
- Eg: Allow connections to a service from a single IP address, instead of all IP addresses in a zone.
- The basic syntax of a rich rule can be expressed by the following block:

```
rule
[source]
[destination]
service|port|protocol|icmp-block|masquerade|forward-port
[log]
[audit]
[accept|reject|drop]
```

2.2.2 Rich rules commands

- Display all rich rules for the specified zone, or the default zone if no zone is specified.

Syntax:

```
firewall-cmd --list-rich-rules
```

- Add <RICH RULE> to the specified zone, or the default zone if no zone is specified.

Syntax:

```
firewall-cmd --permanent --zone=<ZONE> --add-rich-rule='<RULE>'
firewall-cmd --reload
```

- Remove <RICH RULE> to the specified zone, or the default zone if no zone is specified.

Syntax:

```
firewall-cmd --permanent --zone=<ZONE> --remove-rich-rule='<RULE>'
firewall-cmd --reload
```

Examples:

1. Reject all traffic from a "192.168.0.11/32" IP address in default zone:

Eg:

```
# firewall-cmd --permanent --add-rich-rule='rule
family=ipv4 source address=192.168.0.11/32 reject'

# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --permanent --add-rich-rule='rule family=ipv4
source address=192.168.0.11/32 reject'
success
[root@server ~]# firewall-cmd --reload
success
[root@server ~]# firewall-cmd --list-all
public (active)
target: default
icmp-block-inversion: no
interfaces: eth0 eth1
sources:
services: cockpit dhcpv6-client ssh
ports:
protocols:
forward: no
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
    rule family="ipv4" source address="192.168.0.11/32" reject
```

Figure 2.10: Sample output

2. Allows port 8080 for a specific IP address "192.168.0.11/32":

Eg:

```
# firewall-cmd --add-rich-rule='rule family="ipv4"
source address="192.168.0.11" port port=8080
protocol=tcp accept'

# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --permanent --add-rich-rule='rule family="ipv4"
source address="192.168.0.11" port port=8080 protocol=tcp accept'
success
[root@server ~]#
[root@server ~]# firewall-cmd --reload
success
[root@server ~]#
[root@server ~]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: eth0 eth1
  sources:
  services: cockpit dhcpv6-client ssh
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
    rule family="ipv4" source address="192.168.0.11" port port="8080" prot
ocol="tcp" accept
```

Figure 2.11: Sample output

3. Accept all TCP packets on ports 7900, up to and including port 7905, in the vnc zone for the 192.168.1.0/24 subnet.

Eg:

```
# firewall-cmd --permanent --zone=vnc --  
add-rich-rule='rule family=ipv4 source ad-  
dress=192.168.1.0/24 port port=7900-7905 proto-  
col=tcp accept'  
  
# firewall-cmd --reload
```

```
[root@server ~]# firewall-cmd --permanent --zone=external --add-rich-rule='rule fami  
ly=ipv4 source address=192.168.1.0/24 port port=7900-7905 protocol=tcp accept'  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --reload  
success  
[root@server ~]#  
[root@server ~]# firewall-cmd --list-all --zone=external  
external  
target: default  
icmp-block-inversion: no  
interfaces:  
sources:  
services: ssh  
ports:  
protocols:  
forward: no  
masquerade: yes  
forward-ports:  
source-ports:  
icmp-blocks:  
rich rules:  
rule family="ipv4" source address="192.168.1.0/24" port port="7900-7905" pro  
tocol="tcp" accept
```

Figure 2.12: Sample output

2.2.3 Port forwarding

- Port forwarding is a form of **NAT** (Network Address Translation).
- With port forwarding, traffic of a server is forwarded to a different port on the same machine, or to a port on a different machine.
- This is used “**hide**” a server behind another machine, or to provide access to a service on an alternate port.
- Eg:
 - Ssh server with IP address "192.168.0.108" can configure rich rule such that traffic coming from client address "192.168.0.106" at port 2020 is forwarded to port 22 on ssh server.

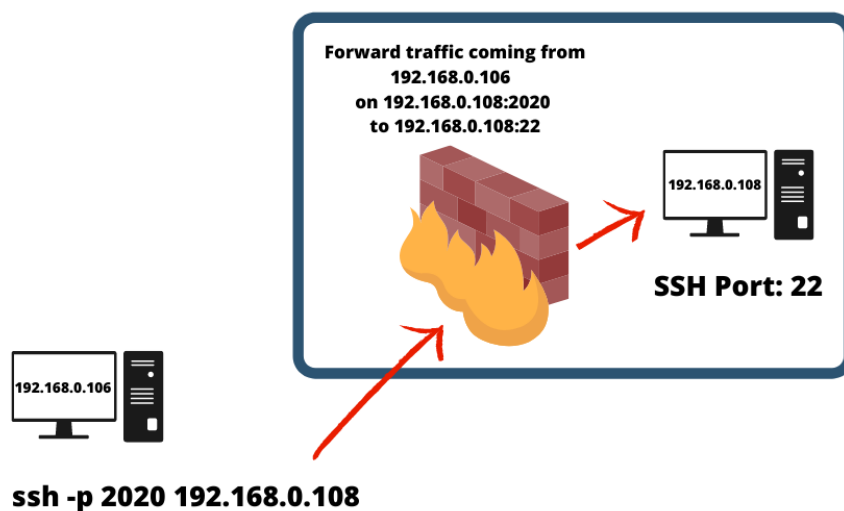



Figure 2.13: Port forwarding

```
# firewall-cmd --permanent --add-rich-rule 'rule family=ipv4 source address=192.168.0.106/24 forward-port port=2020 protocol=tcp to-port=22'

# firewall-cmd --reload
```

2.2.4 Practice

Don't fear failure. 
Not failure, but low aim is the crime.

3. Domain Name Server

3.1 Introduction to DNS

In this section, you are going to learn:

1. What is vi/vim editor?
2. Using vim editor:
 - How to move around in the file?
 - How to searching for text in the file?
 - How to save or not save the file?
 - Other related function related to file editing

Finally, there will be a **small exercise** on these topics to check your knowledge.

So let's get started....



3.1.1 What is /etc/hosts file?

- Every machine on internet is identified by a numerical address.
- It will be very difficult to memorize the numerical address of all the machines in the network.
- To solve the problem, historically, every machine in the network used **/etc/hosts** file where the name to address mapping was done.
- Structure of **/etc/hosts** file:

IP-address	FQDN
------------	------

Eg:

```
# cat /etc/hosts
192.168.2.4  server.example.com.
192.168.2.6  client.example.com.
```

3.1.2 Problem with `/etc/hosts`

- Every machine on internet needed to be update with newly added server entries in `/etc/hosts` file.
- There was no kind of notification available for clients to know a new entry has been added.
- A single `/etc/hosts` file became large and very large, making it difficult to handle.

Solution: DNS server or nameservers

- During the mid 1970's, **nameservers** came into place.
- Idea behind **nameservers** was to solve the problem of resolving hostname to numbers.
- Nameserver, are also known as a **DNS server**.
- DNS servers are mentioned in `/etc/resolv.conf` file.

/etc/resolv.conf file

- Lists nameservers that are used for DNS resolution.
- This file contains a lines specifying the "**search domains**" and up to **three IP addresses of DNS server**.
- Eg: Below resolv.conf have two **search domains** and **three DNS servers**:

```
# cat /etc/resolv.conf
search us.mydomain.com mydomain.com
nameserver 192.168.154.3
nameserver 192.168.154.4
nameserver 10.216.106.3
```

- If you are using DHCP, this file is automatically populated with DNS record issued by DHCP server.

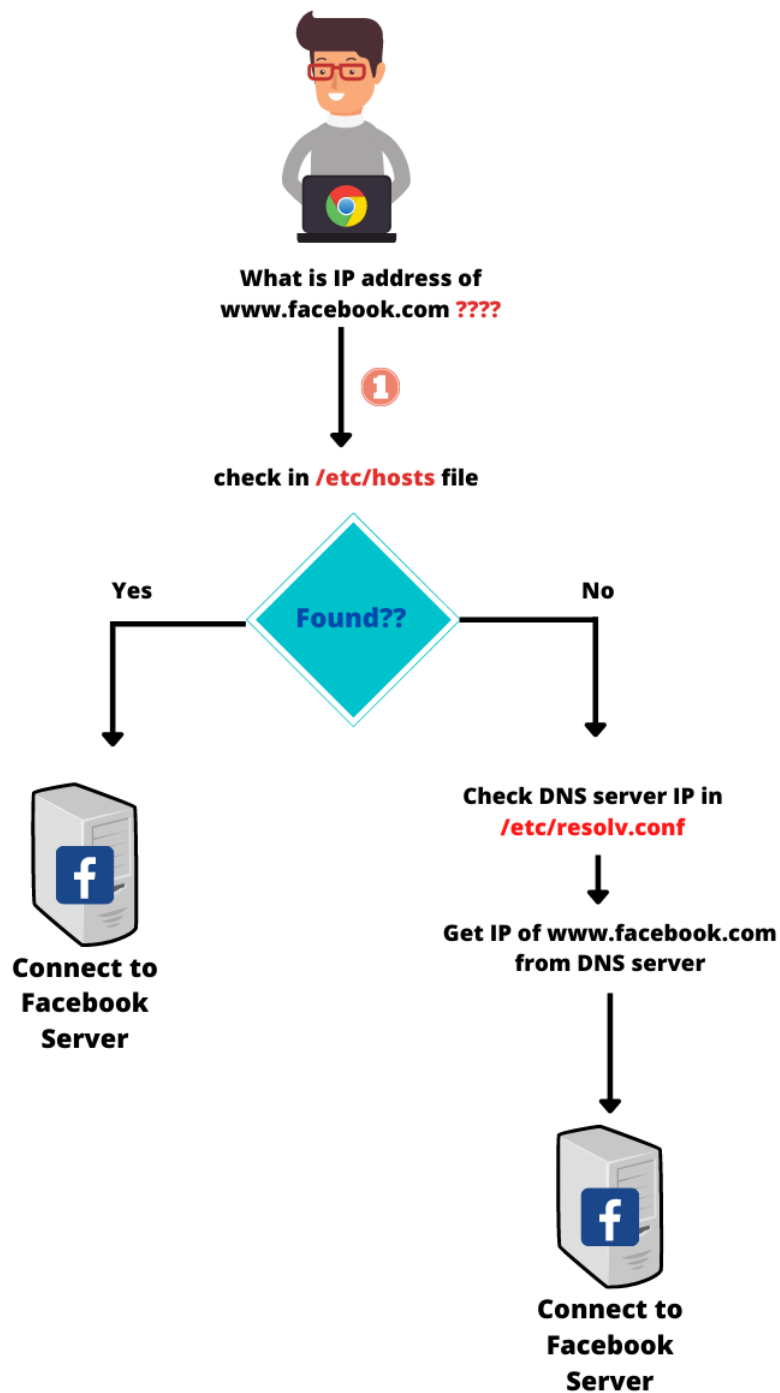
/etc/nsswitch.conf file

- This file defines, "Who should be consulted first for resolution of domain name, a DNS server or a /etc/hosts file?"
- Eg: Configuration setting "**hosts: files dns**" in this file means:
 - **/etc/hosts** file will be checked first for resolution.
 - If domain is still un-resolvable, DNS will then be consulted.

```
[root@server ~]# cat /etc/nsswitch.conf | grep -v ^#  
  
passwd:    sss files systemd  
group:     sss files systemd  
netgroup:  sss files  
automount: sss files  
services:  sss files  
  
shadow:    files sss  
hosts:     files dns myhostname  
  
aliases:   files  
ethers:    files  
gshadow:   files  
networks:  files dns  
protocols: files  
publickey: files  
rpc:       files
```

Figure 3.1: nsswitch file with hosts entry

- Note that DNS servers are mentioned in **/etc/resolv.conf**.

Figure 3.2: Working of `/etc/nsswitch.conf`

3.1.3 What is DNS?

- A DNS server, is also known as a **nameserver**.
- DNS stands for **Domain Name System**.
- It resolve (translate) **hostnames to IP addresses** and vice versa.

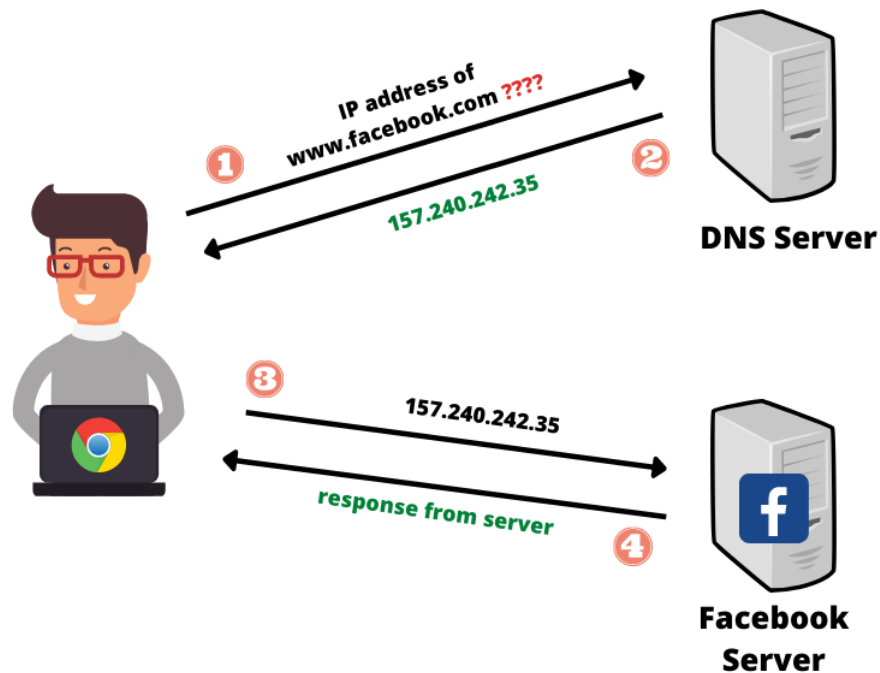


Figure 3.3: DNS server

3.1.4 The DNS hierarchy

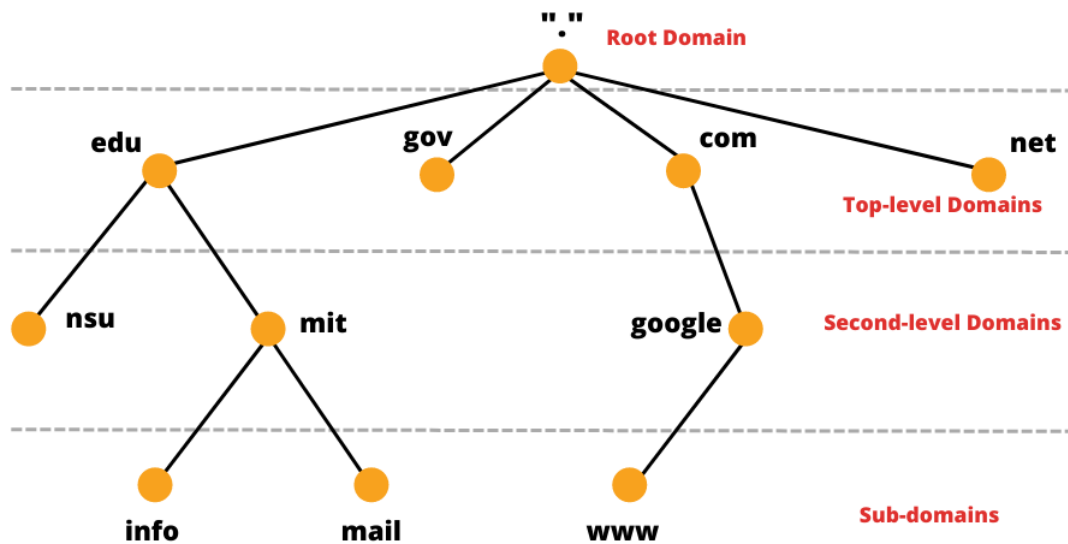


Figure 3.4: DNS Hierarchy

- **What is domain?**
 - A domain name is an easy-to-remember address used to access websites.
- A domain consist of:
 - The **root-level domain** at the top
 - The **top-level domains**(TLD) underneath it
 - Followed by **second-level domains**
 - Finally **sub-domains**.

Let's see each of these in detail

- **What is root-level domain?**

- Root domain is the highest hierarchical level of the Internet.
- When any DNS server is asked for an information which it does not have, the first thing that DNS server does is asking one of the (.)root name server.

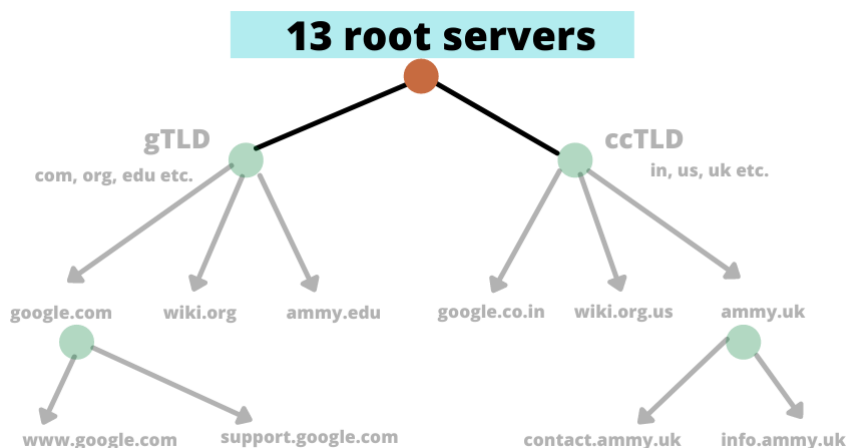


Figure 3.5: 13 root servers

- There are 13 root name servers as follows:
 - * a.root-servers.net.
 - * b.root-servers.net.
 - * c.root-servers.net.
 - * d.root-servers.net.
 - * e.root-servers.net.
 - * f.root-servers.net.
 - * g.root-servers.net.
 - * h.root-servers.net.
 - * i.root-servers.net.
 - * j.root-servers.net.
 - * k.root-servers.net.
 - * l.root-servers.net.
 - * m.root-servers.net.

- What is TLD?

- TLDs tell users the general purpose of the service behind the domain name.
- Types of TLD:
 - * Generic TLDs (gTLDs) like **.com**, **.edu**, **.net**, etc.
 - * Country code TLDs (ccTLDs) like **.us**, **.uk**, **.ru**, **.in**, etc.

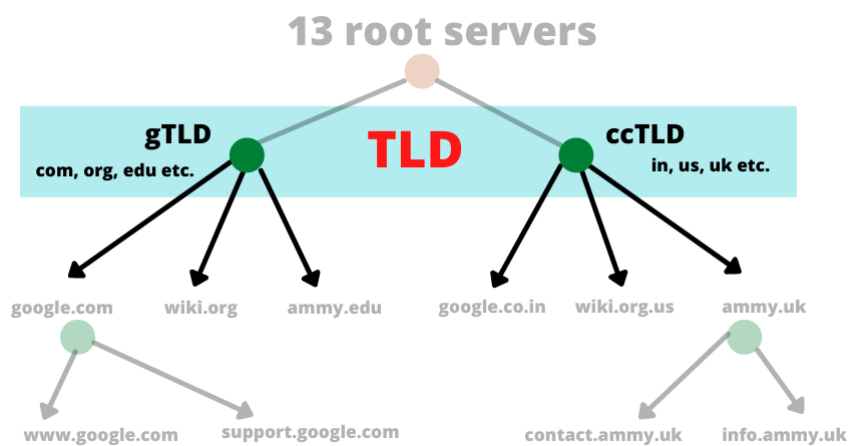


Figure 3.6: TLD types

- List of TLDs:
<https://data.iana.org/TLD/tlds-alpha-by-domain.txt>

- **What is second level domain?**
 - This is where domain holders put the brand name, project name, organization name or other familiar identifier for users.

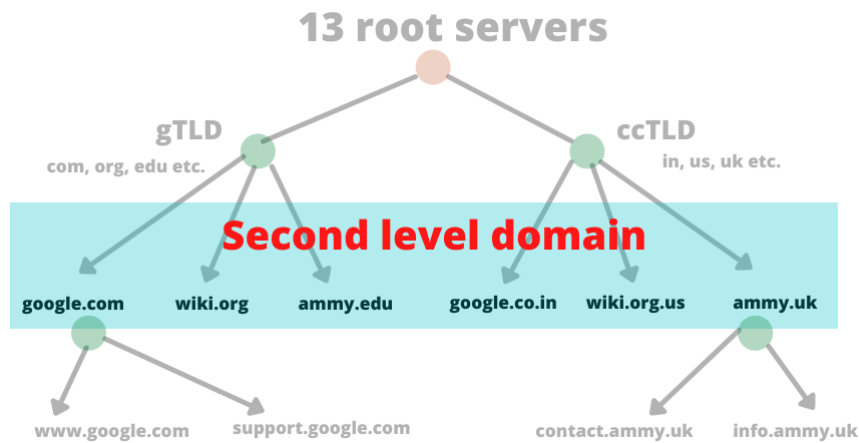


Figure 3.7: Second level domain

- Eg: For amazon.com, **amazon** is second level domain.

- **What is subdomain?**
 - It is additional information added to the front of website's domain name.
 - It organizes the content for website.

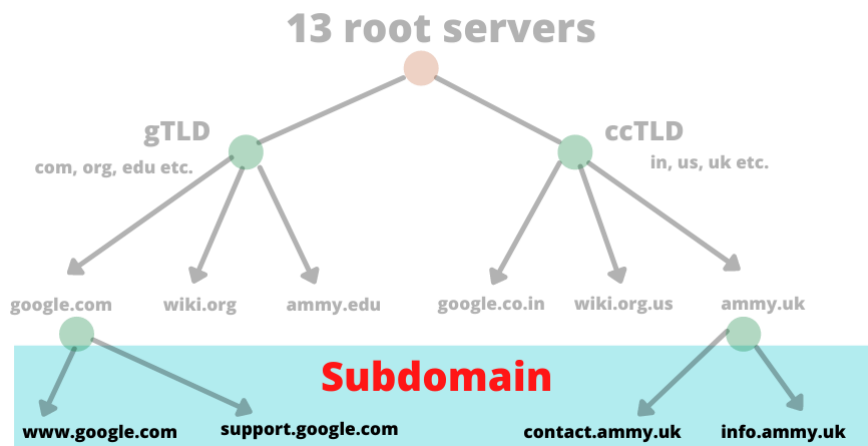


Figure 3.8: Subdomain

Command examining the DNS query structure

- **dig +trace domainname:** Understand the entire flow of the query.

```
# dig +trace google.com
```

Eg:

```
1 [root@localhost ~]# dig +trace example.com
2
3 ; <<> DiG 9.3.6-P1-RHEL5-9.3.6-16.P1.el5 <<> +trace example.com
4 ;; global options: printcmd
5 .                76094 IN      NS      b.root-servers.net. 1
6 .                76094 IN      NS      e.root-servers.net.
7 .                76094 IN      NS      j.root-servers.net.
8 .                76094 IN      NS      g.root-servers.net.
9 .                76094 IN      NS      i.root-servers.net.
10 .               76094 IN      NS      h.root-servers.net.
11 .               76094 IN      NS      c.root-servers.net.
12 .               76094 IN      NS      m.root-servers.net.
13 .               76094 IN      NS      l.root-servers.net.
14 .               76094 IN      NS      f.root-servers.net.
15 .               76094 IN      NS      d.root-servers.net.
16 .               76094 IN      NS      k.root-servers.net.
17 .               76094 IN      NS      a.root-servers.net.
18 ;; Received 497 bytes from 172.16.140.34#53(172.16.140.34) in 0 ms
19
20 com.             172800 IN      NS      a.gtld-servers.net. 2
21 com.             172800 IN      NS      g.gtld-servers.net.
22 com.             172800 IN      NS      d.gtld-servers.net.
23 com.             172800 IN      NS      m.gtld-servers.net.
24 com.             172800 IN      NS      f.gtld-servers.net.
25 com.             172800 IN      NS      l.gtld-servers.net.
26 com.             172800 IN      NS      h.gtld-servers.net.
27 com.             172800 IN      NS      b.gtld-servers.net.
28 com.             172800 IN      NS      c.gtld-servers.net.
29 com.             172800 IN      NS      e.gtld-servers.net.
30 com.             172800 IN      NS      j.gtld-servers.net.
31 com.             172800 IN      NS      i.gtld-servers.net.
32 com.             172800 IN      NS      k.gtld-servers.net.
33 ;; Received 489 bytes from 192.228.79.201#53(b.root-servers.net) 1 34 ms
34
35 example.com.     172800 IN      NS      a.iana-servers.net. 3
36 example.com.     172800 IN      NS      b.iana-servers.net.
37 ;; Received 165 bytes from 192.5.6.30#53(a.gtld-servers.net) 2 6 ms
38
39 example.com.     172800 IN      A       192.0.43.10
40 example.com.     172800 IN      NS      b.iana-servers.net.
41 example.com.     172800 IN      NS      a.iana-servers.net. 3
42 ;; Received 93 bytes from 199.43.132.53#53(a.iana-servers.net) 0 ms
```

Figure 3.9: dig command output

Understanding dig command output:

- **/etc/resolv.conf:** Your local dns server (in **/etc/resolv.conf**), provides with address of the **13 dns root servers**.
- **Box 1:** Dig command selects one root server from the 13, to fetch the address of the **.com TLD servers**.
- **Box 2:** The root server selected will reply with the **.com TLD server** addresses.
- **Box 3:** The **COM GTLD** server replied with IP address of "example.com" which is **192.0.43.10**.

3.1.5 How domain names are assigned?

- The Internet Corporation for Assigned Names and Numbers (ICANN) is the ultimate authority for domain-name assignments.
- **Buying a domain name**
 - First check if domain you want is available here or not:
<https://lookup.icann.org/en>
 - If it is available, you cannot "buy a domain name".
 - You pay for the right to use a domain name for one or more years.
 - You can renew your right after it's duration is expired.
 - In simple words, you rent a second-level domain name.
- **How many subdomains are possible for domain name?**
 - Owner of domain can create unlimited number of subdomains.
 - Eg: utexas.edu can have second-level domain like:
 - * gslis.utexas.edu
 - * computerstore.utexas.edu
- **How big can be a domain name?**
 - Each subdomain can have its own subdomains.
 - Eg: example.subdomains.in.the.domain.utexas.edu.
 - There can be **maximum 127 subdomains** in a single domain name.
 - Maximum size of each subdomain is **63** characters.

3.1.6 Practice

3.2 DNS concepts

Welcome

Welcome guys! We are glad to have you here.

3.2.1 Types of DNS server

- **Authoritative nameserver:**
 - An authoritative nameserver has the original settings for domain name.
 - This is where the domain administrator has configured the DNS records for a domain.

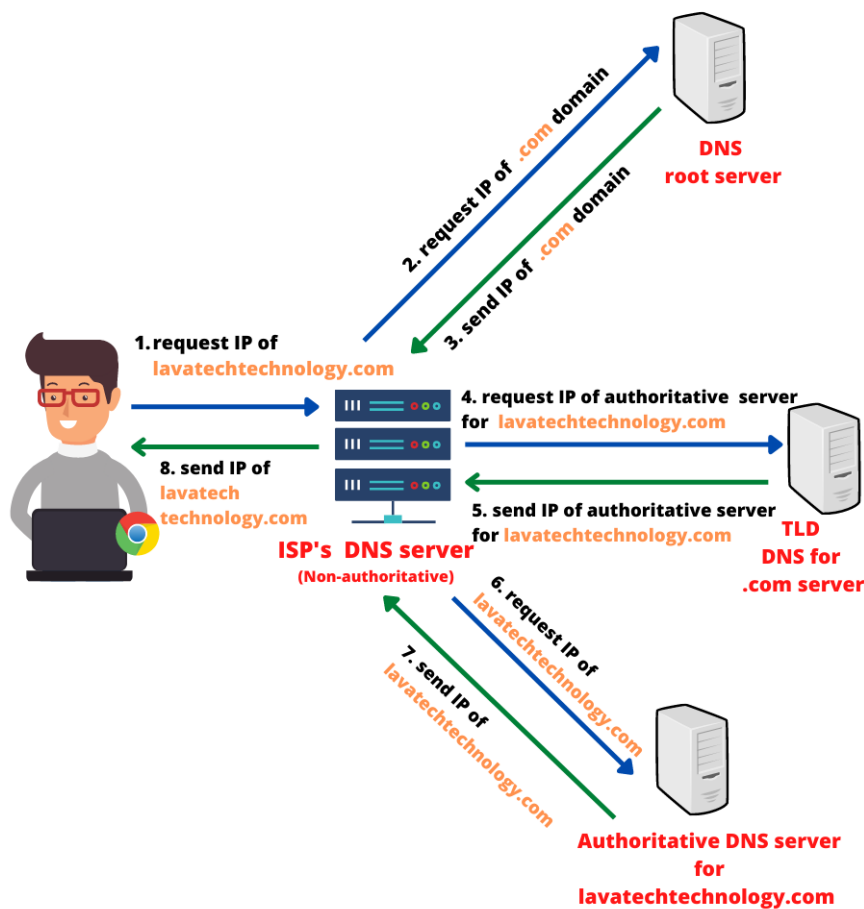


Figure 3.10: DNS Hierarchy

- Find authoritative DNS server for any domain:

Syntax: `dig +short NS <domain name>`

Eg:

```
# dig +short NS lavatechtechnology.com
ns06.domaincontrol.com.
ns05.domaincontrol.com.
```

or

Syntax: `host -t ns <domain name>`

Eg:

```
# host -t ns lavatechtechnology.com
lavatechtechnology.com      name      server
ns06.domaincontrol.com.
lavatechtechnology.com      name      server
ns05.domaincontrol.com.
```

- Using **nslookup** (Name Server lookup) command, confirm if the domain name is resolved using the authoritative nameserver.

Syntax: `nslookup <domain-name> <nameserver>`

```
@lavatech:~$ dig +short NS lavatechtechnology.com
ns06.domaincontrol.com.
ns05.domaincontrol.com.
@lavatech:~$
@lavatech:~$ nslookup lavatechtechnology.com ns06.domaincontrol.com.
Server:      ns06.domaincontrol.com.
Address:     173.201.70.3#53

Name:   lavatechtechnology.com
Address: 116.73.251.14
```

Figure 3.11: Sample output

- **Non-authoritative nameserver:**
 - Non-authoritative name servers do not contain original source files of domain's zone.
 - They have a **cache file** for the domains that is constructed from all the DNS lookups done previously.
 - If a DNS server responded for a DNS query which doesn't have original file is known as a Non-authoritative answer.
 - Find non-authoritative DNS server for any domain:

Syntax: `nslookup <domain-name>`

Eg:

```
# nslookup lavatechtechnology.com
```

```
@lavatech:~$ nslookup lavatechtechnology.com
Server:      127.0.0.53
Address:     127.0.0.53#53
Non-authoritative answer:
Name:   lavatechtechnology.com
Address: 116.73.251.14
```

Figure 3.12: Sample output

```
@lavatech:~$ nslookup google.com
Server:      127.0.0.53
Address:     127.0.0.53#53
Non-authoritative answer:
Name:   google.com
Address: 142.250.199.174
Name:   google.com
Address: 2404:6800:4009:82e::200e
```

Figure 3.13: Sample output

• :

3.2.2 8.8.8.8 DNS server

- Can I use an 8.8 8.8 DNS?
- This is the public DNS server of Google and basically means that Google is the provider of the DNS and is responsible for the maintenance of the service. This DNS can be used by anyone on the internet.

3.2.3 Zones in DNS

What is DNS zone?

- The DNS is broken up into many different zones.
- **DNS zones encompass all records for a domain.**
- Eg: A zone for xzy.com would contain records such as www.xzy.com, support.xyz.com etc.

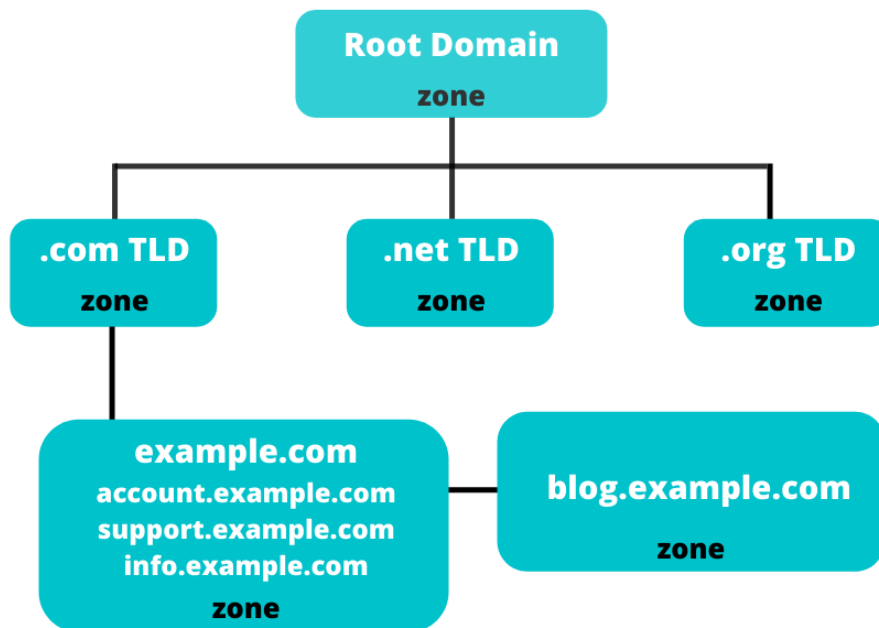


Figure 3.14: DNS zone

What is zone file?

- A DNS zone file is a text file that maps domain names and IP addresses.
- They are human-readable and editable.

3.2.4 Forward & Reverse zone

Forward Lookup zone

- Converts a **domain name to an IP address** or another name.
- This zone contains all the records of domain names to their IP addresses.

Reverse Lookup zone

- Converts a **an IP address to domain name**.
- This zone contains all the records of IP addresses to their domain names.

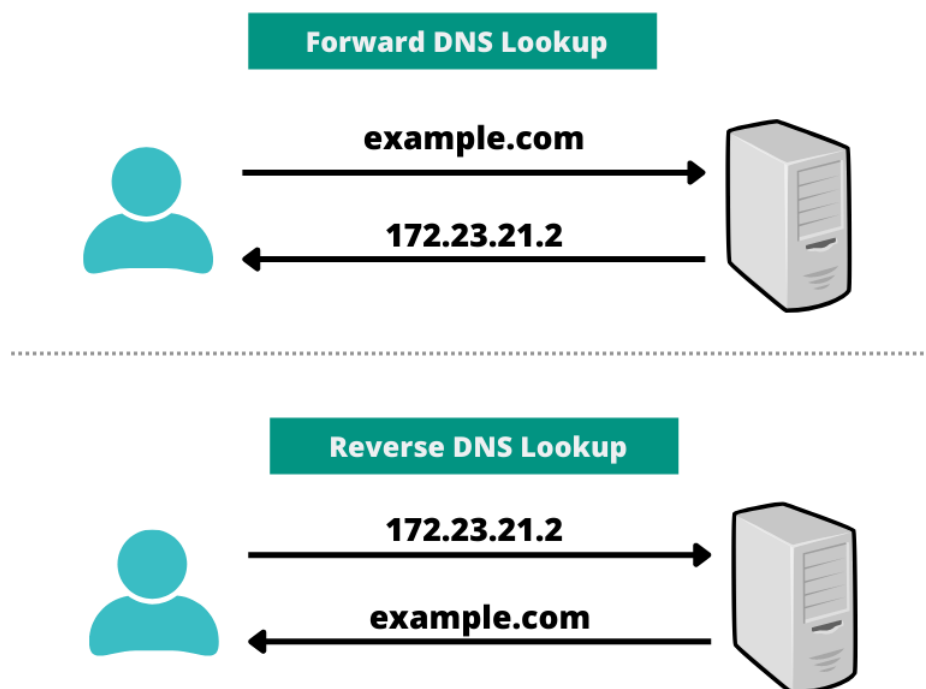



Figure 3.15: Forward & Reverse DNS Lookup

3.2.5 Forward zone file

```
[root@server ~]# cat /var/named/forward.zone
$TTL 1D
@ IN SOA @ rname.invalid. (
                                0      ; serial
                                1D      ; refresh
                                1H      ; retry
                                1W      ; expire
                                3H )    ; minimum
@ IN NS server.student.com.
server IN A 192.168.0.101
client IN A 192.168.0.103
```

- \$TTL – DNS TTL (time to live) is a setting that tells the DNS resolver how long to cache a query before requesting a new one.

```
[root@server ~]# cat /var/named/forward.zone
$TTL 1D
@      IN SOA  @ rname.invalid. (
                                0      ; serial
                                1D     ; refresh
                                1H     ; retry
                                1W     ; expire
                                3H )   ; minimum
@      IN     NS      server.student.com.
server IN     A       192.168.0.101
client IN     A       192.168.0.103
```

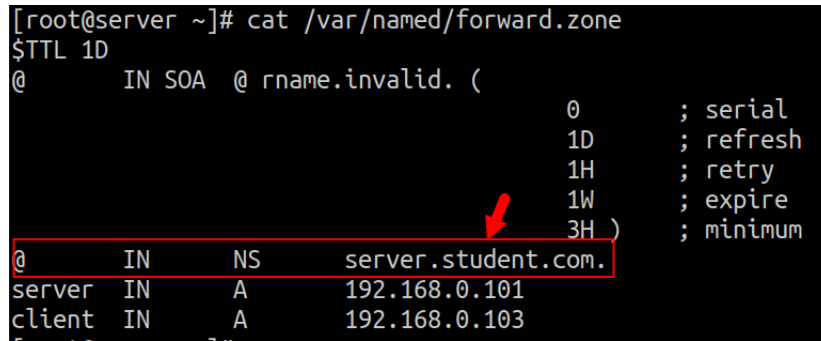


Syntax:

```
@      IN      SOA      @      hostmaster-email      (
serial-number
time-to-refresh
time-to-retry
time-to-expire
minimum-TTL)
```

- @ means domain-name
- IN: Stands for Internet
- SOA: The DNS ‘start of authority’ (SOA) record stores important information about a domain or zone such as the email address of the administrator, when the domain was last updated, and how long the server should wait between refreshes.
-

```
[root@server ~]# cat /var/named/forward.zone
$TTL 1D
@      IN SOA  @ rname.invalid. (
                                0      ; serial
                                1D     ; refresh
                                1H     ; retry
                                1W     ; expire
                                3H     ; minimum
)
@      IN  NS   server.student.com.
server IN  A    192.168.0.101
client IN  A    192.168.0.103
```



Syntax:

```
@      IN      SOA      @      hostmaster-email      (
serial-number
time-to-refresh
time-to-retry
time-to-expire
minimum-TTL)
```

- The DNS ‘start of authority’ (SOA) record stores important information about a domain or zone such as the email address of the administrator, when the domain was last updated, and how long the server should wait between refreshes.

3.2.6 DNS RESOURCE RECORDS

What is DNS zone?

- The DNS is broken up into many different zones.
- **DNS zones encompass all records for a domain.**
- Eg: A zone for xzy.com would contain records such as www.xzy.com, support.xyz.com etc.

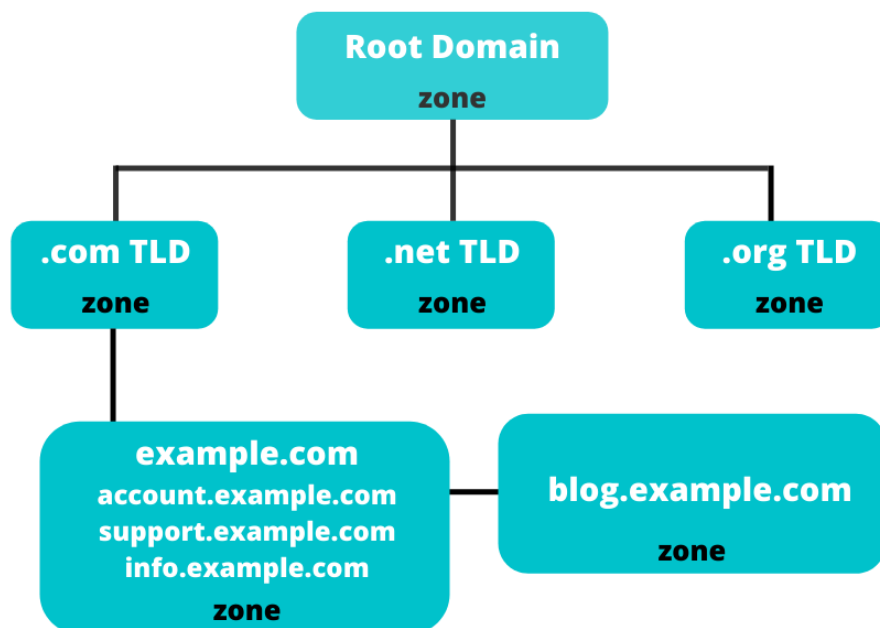


Figure 3.16: DNS zone

What is zone file?

- A DNS zone file is a text file that maps domain names and IP addresses.
- They are human-readable and editable.

If not now, when?

4. Samba Server

Most difficult thing is to START..... 

5. NFS Server

START where you are.
USE what you have.
DO what you can.



6. Teaming and Bridging

Don't focus on the **pain** -> 

Focus on **progress** -> 

- Dwayne Johnson

7. ISCSI Server



**Whatever you are,
be a good one!**



- Abraham Lincoln

8. Apache Server