Tutorial Link https://course.testpad.chitkara.edu.in/tutorials/Networking and Security in Linux/63d75a8927750352bc2e8f13

TUTORIAL

Networking and Security in Linux

Topics

Basics of Computer Networking

Computer Network

An interconnection of multiple devices, also known as hosts, that are connected using multiple paths for the purpose of sending/receiving data or media. Computer networks can also include multiple devices/mediums which help in the communication between two different devices; these are known as **Network devices** and include things such as routers, switches, hubs, and bridges.

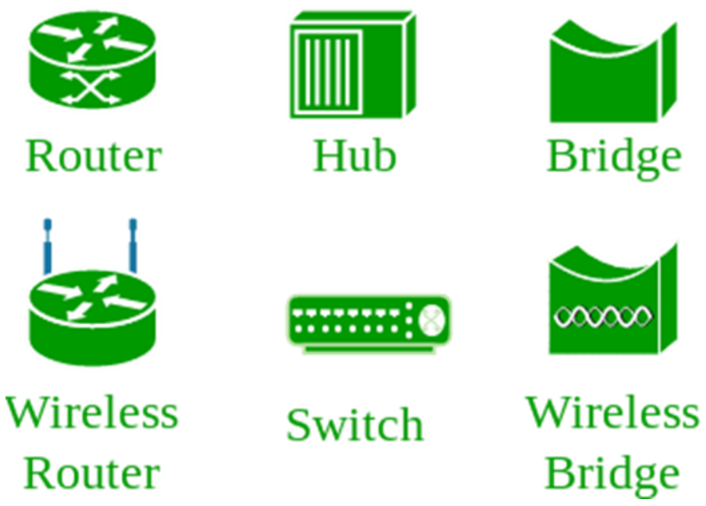


Image Source : https://www.geeksforgeeks.org/basics-computer-networking/

IP Address Basics

IP Address is a unique identifier for any host on the Network. You can assume like, if you enroll in your class, unique roll number will be assigned to you. Similarly, if any device joins a network, a unique roll number will be assigned to that device.

There are two standards for IP addresses: IP Version 4 (IPv4) and IP Version 6 (IPv6).

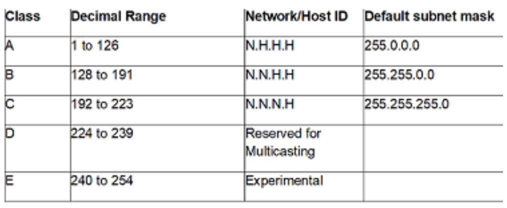
· IPv4 uses 32 binary bits to create a single unique address on the network. An IPv4 address is expressed by four numbers separated by dots. Each number is the decimal (base-10) representation for an eight-digit binary (base-2) number, also called an octet. For example: 192.168.1.12

· IPv6 uses 128 binary bits to create a single unique address on the network. An IPv6 address is expressed by eight groups of hexadecimal (base-16) numbers separated by colons, as in 2001:cdba:0000:0000:0000:0000:3257:9652. Groups of numbers that contain all zeros are often omitted to save space, leaving a colon separator to mark the gap (as in 2001:cdba::3257:9652)

· An IP address can be either dynamic or static. A static address is a permanently assigned address. Static IP addresses assigned by internet service providers are rare. You can assign static IPs to devices on your local network, but it can create network issues if you use it without a good understanding of TCP/IP. Dynamic addresses are the most common. They're assigned by the Dynamic Host Configuration Protocol (DHCP), a service running on the network. DHCP typically runs on network hardware such as routers or dedicated DHCP servers.

Network Classes And Subnet Mask

The organization which governs the internet has divided the IP addresses into different classes of the network. Each class is identified by its subnet mask. By the categorization of a default subnet mask, we can easily identify the class of an IP address of the network. The first octet of an IP address identifies the class of an IP address.



Class	Decimal Range	Network/Host ID	Default subnet mask
A	1 to 126	N.H.H.H	255.0.0.0
B	128 to 191	N.N.H.H	255.255.0.0
C	192 to 223	N.N.N.H	255.255.255.0
D	224 to 239	Reserved for Multicasting	
E	240 to 254	Experimental	

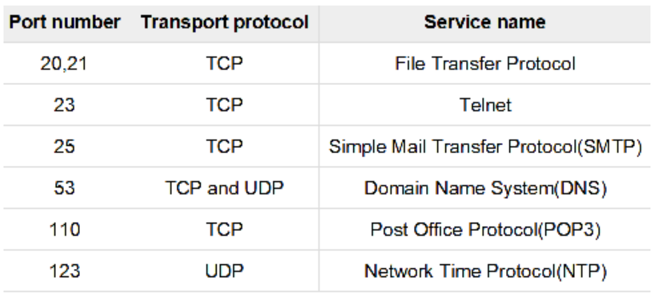
Here when we convert 255.0.0.0 to binary as 11111111.00000000.00000000.00000000 so we can represent this subnet mask as /8. Similarly for class B as /16 and Class C as /24.

Networking Ports

A network port which is provided by the Transport Layer protocols of Internet Protocol suite, such as Transmission Control Protocol (TCP) and User Diagram Protocol (UDP) is a number which serving endpoint communication between two computers.

Internet Assigned Numbers Authority (IANA) is responsible for managing the uses of these ports. There are three categories for ports by IANA –

- 0 to 1023 – well known ports or system ports.
- **1024 to 49151** – registered ports assigned by IANA to a specific service upon application by a requesting entity.
- **49152 to 65 535** – dynamic (private, high) ports range from 49,152 to 65,535. Can be used by private or customer service or temporal purposes.



Port number	Transport protocol	Service name
20,21	TCP	File Transfer Protocol
23	TCP	Telnet
25	TCP	Simple Mail Transfer Protocol(SMTP)
53	TCP and UDP	Domain Name System(DNS)
110	TCP	Post Office Protocol(POP3)
123	UDP	Network Time Protocol(NTP)

Fig: Some well-known ports

Various Network Commands and Tools

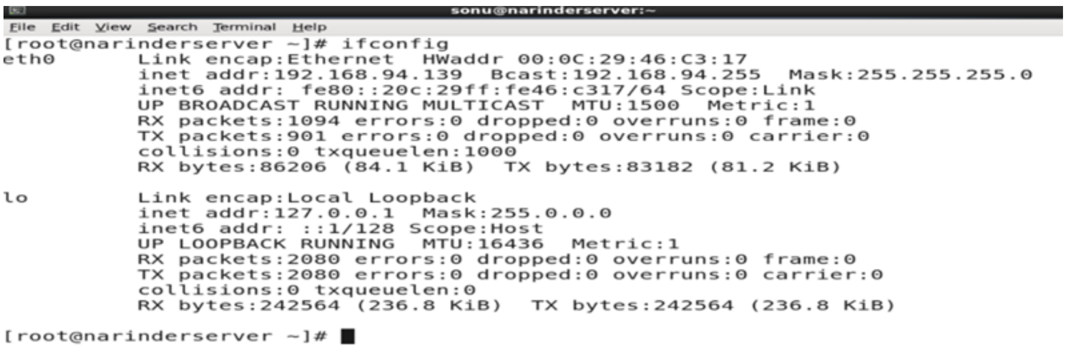
ifconfig

The interface configurator in Linux is referred to as ifconfig. It's one of the most fundamental network inspection instructions.

Ifconfig is used to configure an interface with an IP address, as well as to enable and disable it. It's also where the route and network interface are displayed.

When using ifconfig, the following basic information is displayed:

- IP address
- MAC address
- MTU(Maximum Transmission Unit)



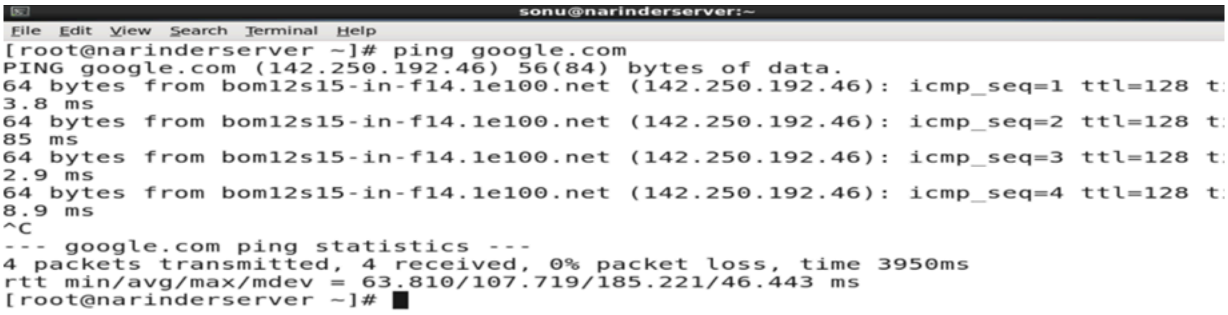
```
[root@narinderserver ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:46:C3:17
          inet addr:192.168.94.139  Bcast:192.168.94.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe46:c317/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1094 errors:0 dropped:0 overruns:0 frame:0
          TX packets:901 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:86206 (84.1 KiB)  TX bytes:83182 (81.2 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2080 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2080 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:242564 (236.8 KiB)  TX bytes:242564 (236.8 KiB)

[root@narinderserver ~]# █
```

ping

Linux ping is one of the most used network troubleshooting commands. It basically checks for the network connectivity between two nodes. ping stands for **Packet Internet Groper**.

The ping command sends the ICMP echo request to check the network connectivity. It keeps executing until it is interrupted.

Use **Ctrl+C Key** to interrupt the execution.



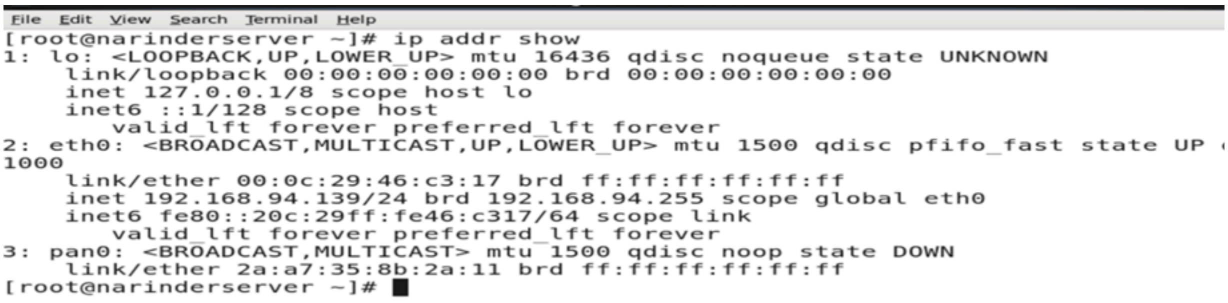
```
[root@narinderserver ~]# ping google.com
PING google.com (142.250.192.46) 56(84) bytes of data.
64 bytes from bom12s15-in-f14.1e100.net (142.250.192.46): icmp_seq=1 ttl=128 t
3.8 ms
64 bytes from bom12s15-in-f14.1e100.net (142.250.192.46): icmp_seq=2 ttl=128 t
85 ms
64 bytes from bom12s15-in-f14.1e100.net (142.250.192.46): icmp_seq=3 ttl=128 t
2.9 ms
64 bytes from bom12s15-in-f14.1e100.net (142.250.192.46): icmp_seq=4 ttl=128 t
8.9 ms
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3950ms
rtt min/avg/max/mdev = 63.810/107.719/185.221/46.443 ms
[root@narinderserver ~]#
```

ip command

This command, which replaces the configure command, has a variety of uses. This command will give you information about the system's specified interface. Second, the command can track the query's progress, and it can also be used to configure Ethernet and IP addresses.

This command can also be used to set up or delete routing entries, display cache entries, acquire IP address information, manage networks, and display current network positions, among other things.

Syntax : ip a or ip addr



```
[root@narinderserver ~]# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
1000
    link/ether 00:0c:29:46:c3:17 brd ff:ff:ff:ff:ff:ff
    inet 192.168.94.139/24 brd 192.168.94.255 scope global eth0
    inet6 fe80::20c:29ff:fe46:c317/64 scope link
       valid_lft forever preferred_lft forever
3: pan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN
    link/ether 2a:a7:35:8b:2a:11 brd ff:ff:ff:ff:ff:ff
[root@narinderserver ~]#
```

dig command

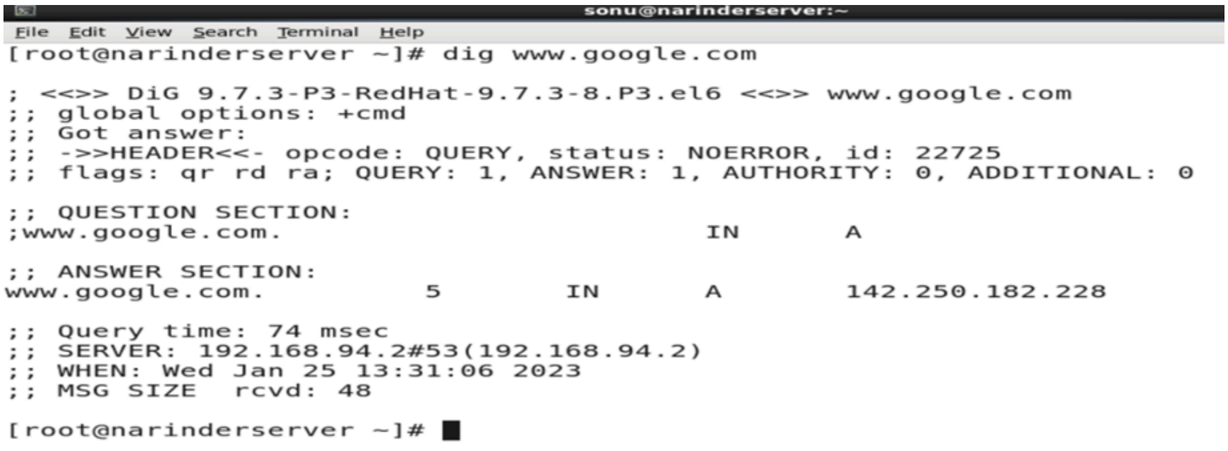
Dig is a versatile tool that stands for domain information groper. dig is a command-line utility for querying DNS name servers on Linux.

Dig executes DNS lookups and displays the results from the primary and secondary nameservers that were queried.

The dig tool is used by the majority of DNS administrators to troubleshoot DNS-related issues because to its flexibility, ease of use, and clarity of output when compared to other tools.

Syntax : dig @**server name type**

- server - It is the name of the name server or IP address whom we want to query.
- name - It determines the name of the resource record.
- type - It specifies the required type of queries such as ANY, A, MX, SIG, and more.



```
[root@narinderserver ~]# dig www.google.com

; <<>> DiG 9.7.3-P3-RedHat-9.7.3-8.P3.el6 <<>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22725
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         5       IN      A       142.250.182.228

;; Query time: 74 msec
;; SERVER: 192.168.94.2#53(192.168.94.2)
;; WHEN: Wed Jan 25 13:31:06 2023
;; MSG SIZE  rcvd: 48

[root@narinderserver ~]# █
```

netstat

Network statistics is the name of the Linux command netstat. It shows data on open sockets, routing tables, and connection information, as well as other interface statistics.

It also has the capability of displaying all socket connections (including TCP, UDP). It also shows sockets that are waiting to be connected, in addition to those that are already connected. Network and system administrators will find it useful.

Syntax : netstat

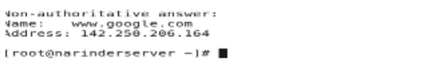
nslookup

You can use nslookup command to acquire answers to DNS-related questions. It can also be used to obtain the following information:

- Specific systems' IP addresses
- MX records for domains
- NS servers for domains

It is capable of working in both interactive and non-interactive modes.

Syntax : nslookup **<domainName>**



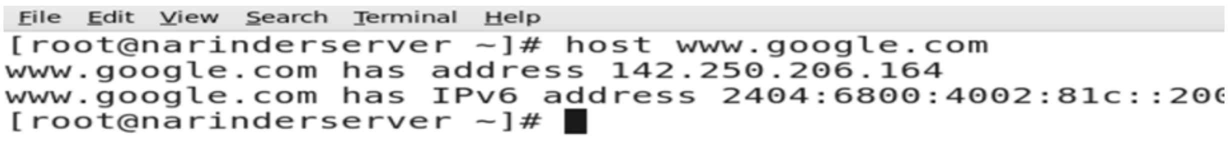
```
[root@narinderserver ~]# nslookup www.google.com
Server:         192.168.94.2
Address:        192.168.94.2#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.250.206.164

[root@narinderserver ~]# █
```

host

In Linux, host command can be used to find information about a domain name's IP address or vice versa. With this command, you can acquire extensive information on any domain using a variety of arguments. This command is mostly used to execute a DNS lookup (Domain Name System).

Syntax : host **<name>**

name: The name can be a domain name or an IP address (Ipv4 or Ipv6).\



```
[root@narinderserver ~]# host www.google.com
www.google.com has address 142.250.206.164
www.google.com has IPv6 address 2404:6800:4002:81c::20
[root@narinderserver ~]# █
```

Configuring firewall rules with iptables

What is IPTables

IPTables can be used for personal computing or can also be applied to the entire network.
Using **IPTables**, we will be defining a set of rules by which we can monitor, allow or block incoming or outgoing network packets.

While discussing **IPTables**, we must understand 3 terms: **Tables**, **Chains**, and **Rules**. As these are the important parts

Tables in IPTables

There are 5 types of **tables** in IPTables and each has different rules applied. So let's start with the most common table "**Filer**".

Filter Table – This is the default and main table while using **IPTables**. It means whenever you won't mention any specific table while applying rules, they will be applied to the filter table. As its name suggests, the role of the Filter table is to decide whether the packages should be allowed to reach their destination or deny their request.

NAT (Network Address Translation) – As its name suggests, this table allows users to determine the translation of network addresses. The role of this table is to determine whether to modify and how to modify the source and destination of the packet address.

Mangle Table – This table allows us to modify the IP headers of packets. For example, you can adjust **TTL** to either lengthening or shorting network hops that the packet can sustain. Similarly, other IP headers can also be modified according to your preference.

RAW Table – The main use of this table is to track connections as it provides a mechanism for marking packets to view packets as a part of an ongoing session.

Security Table – Using the Security table, users can apply internal **SELinux** security context marks on network packets.

For the most use cases, the last 2 types (**RAW** and **Security**) of the table don't have much to do and only the first 3 options are counted as main tables.

Chains in IPTables

They behave at points in the route of the network where we can apply rules. In IPTables, we 5 types of **chains** and we will discuss each of them. Keep in mind that not each type of chain is available for each type of table.

Pre-routing – This chain is applied to any incoming packet once it is entered the network stack and this chain is processed even before any routing decision has been made regarding the final destination of the packet.

Input Chain – It is the point where a packet enters the network stack.

Forward Chain – It is the point where the packet has been forwarded through your system.

Output Chain – The output chain is applied to the packet when it originated through your system and goes out.

Post-routing – This is the complete opposite of the pre-routing chain and is applied to forwarded or outgoing packets once the routing decision has been made.

Rules in IPTables

Rules are nothing but the set or individual commands by which users manipulate network traffic. Once each chain will come into action, the packet will be checked against defined rules.

If one rule does not satisfy the condition, it will be skipped to the next one and if it satisfies the condition, the next rule will be specified by the value of the target.

Each rule has two components: the **matching component** and the **target component**.

· **Matching Component** – They are different conditions to define rules which can be matched by protocol, IP address, port address, interfaces, and headers.

· **Target Component** – This is an action that will be triggered once the conditions are satisfied.

Installing IPTables Firewall in Linux

Most probably, IPTables comes by default in Linux, but if not so, You can use below command to install :

```
$ sudo dnf install iptables-services
```

If you are using Firewalld, you'll have to disable it before proceeding to installation. To stop the firewall completely, you'll have to utilize the following commands:

```
$ sudo systemctl stop firewalld
                                                              $ sudo systemctl disable firewalld
                                    $ sudo systemctl mask firewalld
```

Once you install **IPTables**, you can enable the firewall by given commands:

```
$ sudo systemctl enable iptables
                                                              $ sudo systemctl start iptables
```

To monitor the state of the **IPTable** service, you can use the given command:

```
$ sudo systemctl status iptables
```

IPTables Command in Linux

```
# iptables -t {type of table} -options {chain points} {condition or matching component} {action}
```

Table

The first part is `-t` where we can choose from any 5 available table options and if you remove the `-t` part from the command, it will use **filter** table as it is the default table type.

Chain Options

The second part is for the **chain**. Here you can choose from different chain point options and those options are given below:

· `-A` – Adds a new rule to the chain at the end of the chain.

· `-C` – Checks for rule whether it satisfies the chain's requirement.

· `-D` – Allows users to delete an existing rule from the chain.

· `-F` – This will remove each rule defined by the user.

· `-I` – Allows users to add a new rule at the specified position.

· `-N` – Creates an entirely new chain.

· `-v` – When used with the list option, brings detailed information.

· `-X` – Deletes chain.

Matching Component

Matching options are the condition for checking the chain's requirement. You can choose from various options and some of them are given below:

```
Protocols -p
                                                               Source IP -
s
                                                  Destination IP -
d
                                              IN interface -
I
                                               OUT interface -o
```

For **TCP**, they are as follows:

```
-sport                                                   -
dport                                                       --tcp-flags
```

Action Component

Now, if we consider the action part, the available options depend on the type of table such as **NAT**, and the mangle table has more options compared to others. By using action, you can also target a specific table or chain.

The most used action is `Jump (-j)` which will give us several options such as:

- **ACCEPT** – It is used to accept packets and end traversing.
- **DROP** – It is used to drop packets and end traversing.
- **REJECT** – This is quite similar to DROP but it sends rejected packets to the source.
- **RETURN** – This will stop traversing of the packet in the sub chain and will send the specific packet to the superior chain with no effect.

Once we are done with syntax, we will show you how you can use **IPTables** including basic configuration.

Listing Current Set of IPTables Rules

If you want to check what is passing through your **Firewall** by default, listing a current set of rules is a perfect way. To list applied rules, use the given command:

```
$ sudo iptables -L
```

Allow/Deny Network Traffic on Specific Ports

If you want to allow **HTTPS** network traffic, we will have to allow port no **443** by utilizing the given command:

```
$ sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
```

Similarly, you can also disable **HTTP** web traffic by given command:

```
$ sudo iptables -A INPUT -p tcp --dport 80 -j REJECT
```

Explanation of used command options:

- `-p` is used for checking specified protocol and in our case it's TCP.
- `--dport` is used for specifying the destination port.
- `-j` is used for taking action (accept or drop).

Control Traffic with IP Address

Yes, you can also control network traffic from an IP address. Not just one or two but also control the range of IP addresses and we will show you how.

To allow a specific IP address, utilize the given command structure:

```
$ sudo iptables -A INPUT -s 69.63.176.13 -j ACCEPT
```

Similarly, to drop packets from specific IP, you are required to use the given command structure:

```
$ sudo iptables -A INPUT -s 192.168.0.27 -j DROP
```

If you want, you can also control the range of IP addresses by utilizing the given command structure:

```
$ sudo iptables -A INPUT -m range --src-range 192.168.0.1-192.168.0.255 -j REJECT
```

Delete User Defined Rule

Sometimes we may end up making mistakes while creating rules and the best way to overcome those mistakes is to delete them. Deleting defined rules are the easiest process throughout this guide and to delete them, first, we will have to list them.

To list defined rules with numbers, use the given command:

```
$ sudo iptables -L --line-numbers
```

To delete rules, we will have to follow the given command structure:

Let’s suppose I want to delete the 10th rule from **INPUT** so I will be using the given command:

To cross-check whether we have successfully removed rule, we have to list rules by given command:

Save Defined IPTables Rules

You might be wondering why we have to save rules where they are working fine after applying them? The issue is once your system reboots, all the defined rules which are not saved will be deleted so it is crucial for us to them.

To save rules in **RHEL**-based distros:

```
$ sudo /sbin/service iptables save
```

To save rules in **Debian** derivatives:

```
$ sudo /sbin/iptables–save
```

Encrypting and decrypting files with GPG

What are GPG keys

GPG stands for **GNU Privacy Guard**. It uses the concept of **Asymmetric encryption**. Let's see how asymmetric encryption works and how is it different from **Symmetric encryption** which we generally use.

In **Symmetric encryption**, there is only one key, generally known as password, which we use to encrypt/decrypt the files. Now the problem here is, how will you share the same password over the network to the sender/receiver. This problem is solved in Asymmetric Encryption.

In **Asymmetric encryption**, there is a pair of keys, one public and one private. The owner can share the public key with anyone whosoever wants to send the files in an encrypted format. That encrypted file is then sent back to the owner and that can only be decrypted by the corresponding private key.

GPG Use Cases

- **Encryption:** Used to encrypt files. We can some content to someone and you don't want anyone in the middle to read it.
- **Signing Commits:** Helpful for proving your identity. For eg, you can use this GPG key to sign your commits in Github, to basically verify that you're the one actually done it.
- **Encrypting Passwords:** Very helpful if you use a command line password utility like a password manager called pass. It uses your GPG keys to handle the encryption for all your secrets you want to store in that password manager.

Install GPG

For Ubuntu/any Debian based distributions

```
sudo apt install gnupg
```

For CentOS/ RHEL based distributions

```
sudo yum install gnupg
```

Verify Installation

```
gpg --version
```

GPG Commands

Generate a new key with default configuration (Quick Key Generation)

```
gpg --generate-key
## It will prompt you for the following
# Real Name
# Email Address
# Passphase
```

Generate a new key with your own configuration (Full Key Generation)

```
gpg --full-generate-key
## It will prompt you for the following
# Key Encryption Type
# Key Size
# Key Expiry
# Real Name
# Email Address
# Comment
# Passphrase
```

List all GPG public keys

```
gpg --list-keys
```

List all GPG private key pairs

```
gpg --list-secret-keys
```

Export Public Key in ASCII Format

```
## Output to STDOUT
gpg --armor --export <email-Id>

## Output to a file
gpg --armor --export --output <file.txt> <email-Id>
```

Encrypt a file for a specific user using GPG (Using Asymmetric Encryption)

```
gpg --encrypt --recipient <recipient-user-email> <file-name>
```

Encrypt a file using GPG (Uses Symmetric Encryption)

```
gpg --symmetric <file-name>
## It will prompt for a password
```

Decrypt that encrypted file (For Both, Symmetric and Asymmetric Encryption)

```
gpg --decrypt <encrypted-file>
```