**Docker Basic Commands:**

|  |  |
| --- | --- |
| Command | Description |
| ***docker run hello world*** | To create a Docker container, download the ‘hello world’ image, by typing the following command in the terminal |
| ***docker images*** | For checking the number of images on your system, use the following command |
| ***docker search <image>*** | For searching an image in the Docker Hub |
| ***docker attach*** | Attach local standard input, output, and error streams to a running container |
| ***docker build*** | Build an image from a Dockerfile |
| ***docker builder*** | Manage builds |
| ***docker checkpoint*** | Manage checkpoints |
| ***docker commit*** | Create a new image from a container’s changes |
| ***docker config*** | Manage Docker configs |
| ***docker container*** | Manage containers |
| ***docker cp*** | Copy files/folders between a container and the local filesystem |
| ***docker create*** | Create a new container |
| ***docker deploy*** | Deploy a new stack or update an existing stack |
| ***docker diff*** | Inspect changes to files or directories on a container’s filesystem |
| ***docker engine*** | Manage the docker engine |
| ***docker events*** | Get real time events from the server |
| ***docker exec*** | Run a command in a running container |
| ***docker export*** | Export a container’s filesystem as a tar archive |
| ***docker history*** | Show the history of an image |
| ***docker image*** | Manage images |
| ***docker images*** | List images |
| ***docker import*** | Import the contents from a tarball to create a filesystem image |
| ***docker info*** | Display system-wide information |
| ***docker inspect*** | Return low-level information on Docker objects |
| ***docker kill*** | Kill one or more running containers |
| ***docker load*** | Load an image from a tar archive or STDIN |
| ***docker login*** | Log in to a Docker registry |
| ***docker logout*** | Log out from a Docker registry |
| ***docker logs*** | Fetch the logs of a container |
| ***docker manifest*** | Manage Docker image manifests and manifest lists |
| ***docker network*** | Manage networks |
| ***docker node*** | Manage Swarm nodes |
| ***docker pause*** | Pause all processes within one or more containers |
| ***docker plugin*** | Manage plugins |
| ***docker port*** | List port mappings or a specific mapping for the container |
| ***docker ps*** | List containers |
| ***docker pull*** | Pull an image or a repository from a registry |
| ***docker push*** | Push an image or a repository to a registry |
| ***docker rename*** | Rename a container |
| ***docker restart*** | Restart one or more containers |
| ***docker rm*** | Remove one or more containers |
| ***docker rmi*** | Remove one or more images |
| ***docker run*** | Run a command in a new container |
| ***docker save*** | Save one or more images to a tar archive (streamed to STDOUT by default) |
| ***docker search*** | Search the Docker Hub for images |
| ***docker secret*** | Manage Docker secrets |
| ***docker service*** | Manage services |
| ***docker stack*** | Manage Docker stacks |
| ***docker start*** | Start one or more stopped containers |
| ***docker stats*** | Display a live stream of container(s) resource usage statistics |
| ***docker stop*** | Stop one or more running containers |
| ***docker swarm*** | Manage Swarm |
| ***docker system*** | Manage Docker |
| ***docker tag*** | Create a tag TARGET\_IMAGE that refers to SOURCE\_IMAGE |
| ***docker top*** | Display the running processes of a container |
| ***docker trust*** | Manage trust on Docker images |
| ***docker unpause*** | Unpause all processes within one or more containers |
| ***docker update*** | Update configuration of one or more containers |
| ***docker version*** | Show the Docker version information |
| ***docker volume*** | Manage volumes |
| ***docker wait*** | Block until one or more containers stop, then print their exit codes |

**IP routing:** It is the process of sending packets from a host on one network to another host on a different remote network.

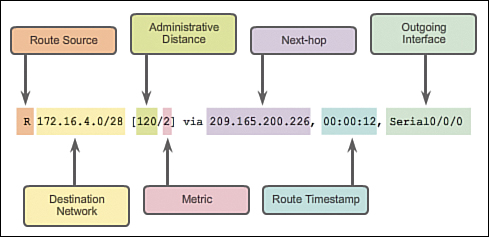
This process is done by routers. Routers examine the destination IP address of a packet, determines next hop address and forward the packet.

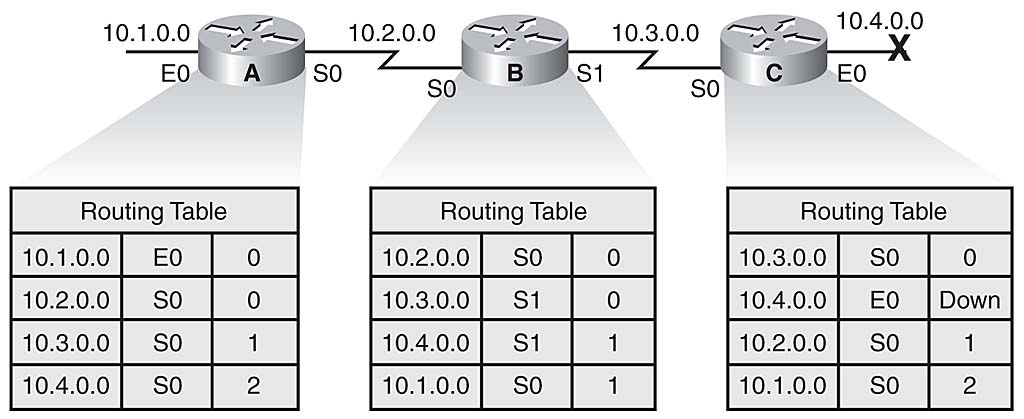
Each router maintains a routing table and stores it in RAM. A routing table is used by routers to determine the path to the destination network. Each routing table consists of the following entries:

* network destination and subnet mask – specifies a range of IP addresses.
* remote router – IP address of the router used to reach that network.
* outgoing interface – outgoing interface the packet should go out to reach the destination network

There are three different methods for populating a routing table:

* directly connected subnets
* using static routing
* using dynamic routing

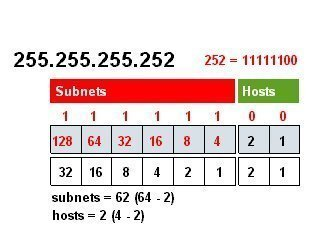


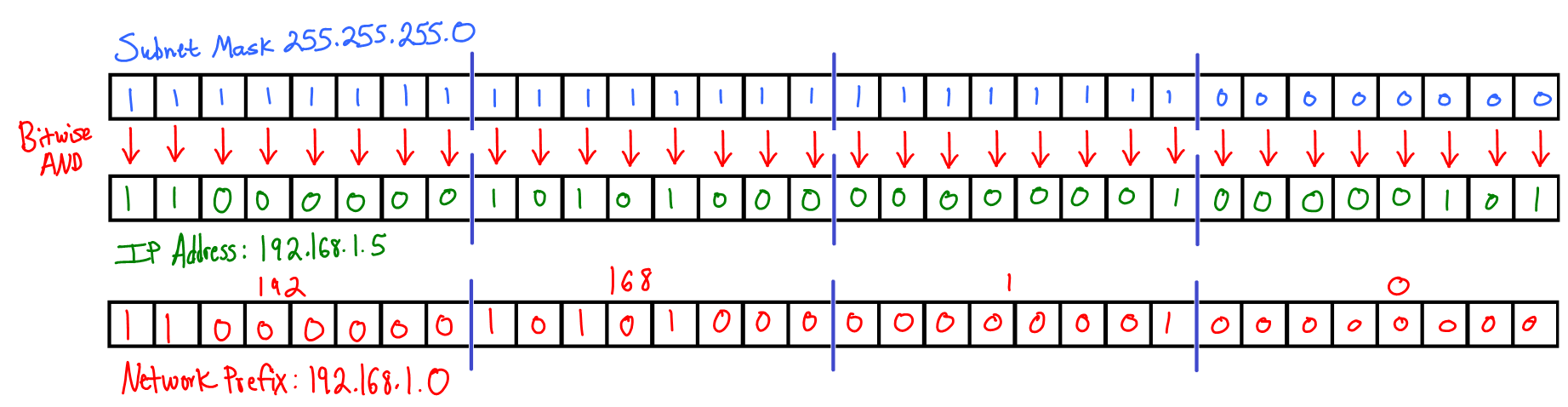


**Subnet Mask:**

A subnet mask is a number that defines a range of IP addresses that can be used in a network. (It is not something you wear on your head to keep subnets out.) Subnet masks are used to designate subnetworks, or subnets, which are typically local networks LANs that are connected to the Internet. Systems within the same subnet can communicate directly with each other, while systems on different subnets must communicate through a router. Therefore, subnetworks can be used to partition multiple networks and limit the traffic between them.

A subnet mask hides, or "masks," the network part of a system's IP address and leaves only the host part as the machine identifier. A common subnet mask for a Class C IP address is 255.255.255.0. Each section of the subnet mask can contain a number from 0 to 256, just like an IP address. Therefore, in the example above, the first three sections are full, meaning the IP addresses of computers within the subnet mask must be identical in the first three sections. The last section of each computer's IP address can be anything from 0 to 255. For example, the IP addresses 10.0.1.201 and 10.0.1.202 would be in the same subnet, while 10.0.2.201 would not. Therefore, a subnet mask of 255.255.255.0 allows for close to 256 unique hosts within the network (since not all 256 IP addresses can be used).





Find out how to give permissions to a folder through Chain modification?

***chmod permission-mask folder-path***

Basic commands: for file transfer, edit file, save, create, read and write file?

File transfer: ***ftp, nc***

Edit, save, create, read and write: ***vi***

How to install any software in Linux platform?

***apt install***

***apt-get install ${packagename}***

How to kill a process using PID?

***kill pid***

***ps aux | grep chrome***

where ps – process status, ‘a’ – list the processes of all users, ‘u’ ­­­­­– detailed info

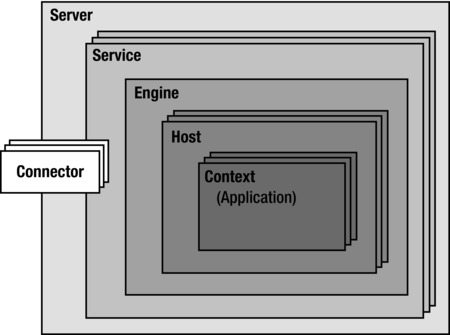
How to create certificates (there is an open source for ssl where we can create ssl)?

***“let’s encrypt”*** gives free certificates

***https://www.javatpoint.com/linux-tutorial***

**Tomcat Server:**

Tomcat is a Servlet and JSP Server serving Java technologies. Tomcat is a servlet container and also offers a web server. A servlet, at the end, is a Java class. JSP files (which are similar to PHP, and older ASP files) are generated into Java code (HttpServlet), which is then compiled to .class files by the server and executed by the Java virtual machine. Tomcat's web server is quite good, able to handle most small and medium web site needs.



https://howtodoinjava.com/tomcat/tomcats-architecture-and-server-xml-configuration-tutorial/