Setting up Web Server

Introduction to Web Server

A Web server is a system that delivers content or services to end users over the Internet. A Web server consists of a physical server, server operating system (OS) and software used to facilitate HTTP communication. A Web server is also known as an Internet server.
In 1989 Tim Berners-Lee proposed a new project to his employer CERN (European Center for Particle Physics), with the goal of easing the exchange of information between scientists by using a hypertext system. The project resulted in Berners-Lee writing two programs in 1990:

- A browser called WorldWideWeb.
- The world's first web server, later known as CERN httpd, which ran on NeXTSTEP

A Web Server is a Computer or Combination of computers, which is connected through internet or intranet to serve the clients quests, coming from their web browser. It is a large repository of web pages which transfer to the client in response to their request. The client requests to the server through protocol such as FTP, HTTP, SMTP etc. for their own specific use. Every web server has a unique IP address and domain name which identifies that machine on the network. A server contains the server software installed on it, which manages the client request and response them.

- The collection of all our web pages is called **our web site**
- To let others view our web pages, we must **publish our web site**
- To publish our work, we must **copy our site to a web server**
- Our own PC can act as a web server if it is **connected to a network**
- Most common is to use an **Internet Service Provider (ISP)**

The primary function of a web server is to store, process and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP). Pages delivered are most frequently HTML documents, which may include images, style sheets and scripts in addition to text content. They can also be found embedded in devices such as printers, routers, webcams and serving only a local network. The web server may then be used as a part of a system for monitoring and/or administering the device.

The Web server usually has a simpler job: to accept Hyper Text Transfer Protocol (HTTP) requests and send a response to the client. However, this job can get much more complex, executing functions such as:

- Performing access control based on file permissions, username/password pairs, and hostname/IP address restrictions.
- Parsing a document (Substituting appropriate values for any conditional fields with in the document) before sending it to the client.
- Spawning a Common Gateway Interface (CGI) script or custom Application Programming Interface (API) program to evaluate the contents of a submitted form presenting a dynamically created document, or accessing a database.
- Logging any successful accesses, failures, and errors.

Leading Web servers include Apache (the most widely-installed Web server), Microsoft's Internet Information Server (IIS) and nginx (pronounced *engine X*) from NGNIX. Other Web servers include Novell's NetWare server, Google Web Server (GWS) and IBM's family of Domino servers.

The Apache Web Server

The Apache Web Server (also referred to as simply "Apache") is one of the most popular open-source HTTP Servers that exists today. It's a powerful, secure, and fully featured web server that can be found hosting anything from personal web sites to corporate domains.

Apache is developed and maintained by the Apache Software Foundation, which consists of a decentralized team of developers. The software is produced under the Apache license, which makes it free and open source. Apache is available for a range of operating systems, including UNIX, Linux, Novell Netware, Windows, Mac OS X, Solaris, and FreeBSD.

The Apache HTTP server is a software (or program) that runs in the background under an appropriate operating system, which supports multi-tasking, and provides services to other applications that connect to it, such as client web browsers. It was first developed to work with Linux/Unix operating systems, but was later adapted to work under other systems, including Windows and Mac. The Apache binary running under UNIX is called *HTTPd* (short for HTTP daemon), and under win32 is called *Apache.exe*.

The main features of the Apache Web Server include:

- The stability and rapid development cycle associated with large group of cooperative volunteer programmers.
- Full source code, downloadable at no charge.
- Access control based on client host name/IP address or username/password combinations.
- Support for server-side scripting as well as CGI scripts.
- A custom API that enables external modules to be used by the server daemon.

Apache installation

There are three different ways in which we can install apache on our RHEL/CentOS machines:

1. **Using RPM** – We need rpm package for it. Once you obtain the rpm package for apache, we can install it using the following command:
   $ rpm –ivh httpd-2.4.x-1.i686.rpm
2. **Using YUM**– Yum is the easiest way to install apache on your system. We don't need to download any package when using yum, just use the following:
   $ yum install httpd
3. **Using source package**– This process is a bit complex but provide us with various options to modify our apache installation as we are using source packages to compile & install the apache package. Firstly we need to download source package (Download from here), we will then extract it, compile it & then install it.

Configuring the Apache Server

The primary file for configuring our Apache server is httpd.conf. All Apache configuration files are plain-text files and can be edited with our favorite text editor. Some of individual module, scripts and other services related to Apache, such as perl, php, ssl, webalizer, and mysql have individual configuration files and they are contained in /etc/httpd/conf.d directory.
The httpd.conf file is the primary configuration file for the Apache Web server. It contains options that pertain to general to the general operation of the server the default filename (/etc/httpd/conf/httpd.conf) can be overridden bye the –f filename command-line argument to the httpd daemon or te ServerConfigFile directive.

Apache is also has a major advantage, that it can support multiple website hosting on a single server. There are actually two types of hosting :-

1. **IP address based hosting**– For IP based hosting, we need to have a different IP for every website that we are hosting. These IPs are then attached to a single or multiple NICs.
2. **Name based Virtual hosting**– Name based hosting is used to host multiple virtual websites using a single IP address:

   Step 1: We will first configure a website test1.com using the name based hosting. Firstly open the apache configuration file which is **/etc/httpd/conf/httpd.conf** & add the following lines to the bottom of file:

   ```
   $ vi /etc/httpd/conf/httpd.conf

   <VirtualHost 192.168.0.120:80>
   ServerAdmin webmaster@test1.com
   DocumentRoot /var/www/html/test1.com
   ServerName www.test1.com
   ErrorLog logs/www.test1.com-error_log
   CustomLog logs/www.test1.com-access_log common
   </VirtualHost>
   ```

   Step 2: Next, search for the “*NameVirtualHost*” in the same file & uncomment it by removing ‘*#*’ from the starting of the line & add the IP of the server. So it should look like:

   ```
   NameVirtualHost 192.168.0.120:80
   ```

   Now save the file & exit.

   Step 3: We will now create ‘index.html’ file for the websites but we will firstly create a directory for the website,

   ```
   $ mkdir /var/www/html/test1.com
   ```

   Now add some content to index.html to identify the site,

   ```
   $ vi /var/www/html/test1.com/index.html
           <html>
           <head>
           <title>This is test1.com</title>
           </head>
   ```

   Step 4: We will now restart our server to implement changes & will then check our sites

   ```
   $ systemctl restart httpd.service or
   $ service httpd restart
   ```

   Step 5: Now, open your browser & enter the website url:

   www.test1.com

   You should now see the content of webpage.

Starting the Apache Web Server

Installing Apache on Linux does require a bit of programming skills (though it is not too difficult). Installing it on a Windows platform is straight forward, as you can run it through a graphical user interface.

If Apache wasn’t installed during the Fedora/RHEL installation, we can install it later form the DVD. Here’s a quick way to get our Apache server going.

1. Make sure that Apache is installed by typing the following from a Terminal window.

   ```
   $ rpm –qa | grep httpd
   ```

2. A valid host name is recommended if it is a public Apache server. If we don't have a real, fully qualified domain we can edit the */etc/http/conf/httpd.conf* file and define the Server Name as our computer's IP address.
3. Add the administrative e-mail address where someone can contact you in case an error is encountered with your server. In the */etc/httpd/conf/httpd.conf* file, the default administrative address appears as follows:
   serverAdmin root@localhost
   change the *root@localhost* to the email address of your Apache Administrator.
4. Start the httpd server. As root user, type the following:
   # service httpd start
5. To have httpd start every time we boot our system, run the command as root user *#chkconfig httpd on*
6. To make sure that the Web server is working open firefox and tye the following into the location box and press enter.
   http://lovalhost/
7. We should see the Test Page for the Apache Web server. To access this page from another computer, we will need to enter our Aapche server's host name or IP address.
8. The test Page is actually an error condition, indicating that we haven't added any content to our web site yet. To get started, we can add an index.html file that contains our home page content in the */var/www/html* directory.

Setting the Global environment

- The ServerTokens directive lets us prevent remove computers from finding out what subcomponens are running on our Apache Server.
  Setting ServerTokens to ProductOnly would minimize the connection information a potential cracker could see.
- The ServerRoot directive specifies that contains the configuration files, a link to the log file directory, and a link to the module directory.
- The Apache Web server keeps track of the PID for the running process. Apache uses the PidFile to store the process ID of the first master daemon process.
- We can set the several values that relate to timeout. The timeout directive determines the number of seconds that Apache will hold a connection open between the receipt of packets, between the receipt of acknowledgements on sent responses, ore while receiving an incoming request. The default of two minutes can be lowered if you find and excessive number of open, idle connections on our machine.

Settting the number of Server Processes

To operate effeciently, Web server has to able to handle lots of incoming requests for content simultaneously. To be ready for requests for web content, Apache has multiple server processes running and listening to service requests.
With the multi-processing Module (MPM) feature, support for therads was added to Apache. Here are some issues to consider if we want to change any of the MPM-related parameters.

- **Ram is Critical :** Every process and thread that is active consumes some amount of memory.
- **Configure for maximum load:** Apache is able to create new server processes and therads as they are needed. You don't need to have the maximum number of processes and threads available at all times.
- **Configure for Performance:** Performance degrades when Apache has to start a new thread, start a server process, or use swap space.

Choosing the Server's user and group

The httpd daemon doesn't have to run as the root user, the fact that it doesn't run as root by default makes our system more secure. By setting User and Group entries, we can have the httpd daemon run using the permissions associated with a different user and group. By default, Apache is defined as both the user and group for the server. If we change the User and Group directives, we should specify a non-privileged entity.

Setting the main server's configuration

- **Setting an email address:** we can identify an address where users can send email if there is a problem with our server. This is done with the ServerAdmin directive. The serverAdmin directive can be set to any valid e-mail address. The default is root@localhost.
- **Setting the server name:** if our server name is anything but our exact registered host or domain name,we should identify our server name here.
  ServerName jukebox.linuxtoys.net
- **Setting canonical names:** use the UseCanonicalName directive to create a self-referencing URL. The UseCanonicalName directive provides a form of naming consistency. When it is set to on, Apache uses the ServerName and port directives to create a URL that references a file on the same machine.
- **Identifying HTTP content directories:** There are several directives for determining the location of your server's Web content. The main location for your Web content is set to /var/www/html by the DocumentRoot directive.

Monitoring Server Activities

An Apache Web server is a tempting target for someone with a desire to hijack a computer for bad purposes. There are many techniques we can use to secure our server which range from SELinux to using certificates to controlling how scripts are run.
Watching your server carefully can often stop an attack before it gets anywhere. Apache provides two unique built-in methods to check the performance and status of your web server. The server-status handler can be configured to show information about server processes. The server-info handler can be configured to display a detailed summary of the Web Server's Configuration.

Displaying server Information

The Server Information page contains the server version information and various general configuration parameters and breaks up the rest of the data by module. Each loaded module is listed, with information about all directives supported by that modul, and the current value of any defined directives from that module.
The server information is usually quite verbose and contains much more information than anc be displayed, which shows only the links to each module's section and the general Server Setting section.

Displaying Server Status

The contents of the server-status page include version information for the server, the current time, a timestamp of when the server was last started, and the server's uptime. The page also details the status of each server process, choosing from several possible states

The bottom of the server-status page lists each server by process ID and indicates its state, using the same possible values.
The server-status page also performs automatic updates to provide even closer monitoring of the server. The server-status page displayed in our browser will be updated in every 40 seconds. This enables a browser window to be entirely devoted to continually monitoring the activities of the web server.

Further Security of server-info and server-status

Because both the server-info and server-status pages contain private information that should not be accessible to just anyone on the network, there are a few extra ways we can secure that information. We can restrict that information only to the local host; however, in some environments that may not be practical.
It may also be beneficial to change the URL used to reference both of the aforementioned pages. This is an example of "Security through obscurity," which should not be relied on but which can make it just a little more difficult for unauthorized individuals to obtain information about our Web server's configuration.

Logging Errors

The error log contains messages generated by the server that describe various error conditions. The ErrorLog and LogLevel directives in the httpd.conf file can modify the filename and the amount of information that is logged. The default file is /etc/httpd/logs/error_lo (which is a link to /var/log/httpd/error_log).

Analyzing Web-Server traffic

The webalizer package can take Apache log files and produce usage reports for our server. Those reports are created in HYML format so we can display the information graphically. Information is produced in both table and graph form.

To use the webalizer command, the webalizer package must be installed(yum install webalizer), we can run webalizer with no options and have it take ta values in /etc/webalizer.conf files to get information it needs. An alternative, we cand use command line options to override settings in the webalizer.conf file. To use the defaults simply run the following:

```
#webalizer
```

Introduction to DNS

Domain Name System (DNS) enables you to use hierarchical, friendly names to easily locate computers and other resources on an IP network.
The Domain Name System (DNS) is essentially a distributed database that translates host names into IP addresses (and IP addresses back to hostnames). That database also contains information related to each domain, such as how the domain is organized into zones, where to route mail for that domain, and whom to contact with questions associated with the domain.

By setting up a DNS server, you become part of a hierarchy of DNS servers that make up the Internet. At the top of this hierarchy is the root server, represented by a dot (.). Below the root server are the Top Level Domains, or TLDs (such as .com , .org , and so on). Domains that individual organizations own and maintain lie below the TLDs, branching in a way that looks like an upside-down tree structure

It is also a protocol for Transmission Control Protocol/Internet Protocol (TCP/IP) networks, defined by the Requests for Comments (RFCs) that pertain to DNS. DNS defines the following:

- Mechanism for querying and updating the database.
  1. Mechanism for replicating the information in the database among servers.
  2. Schema for the database.

Linux DNS Servers can operate in three modes and these modes are, caching only DNS server, Primary DNS server and Slave (secondary) DNS server.

- **Domain:** A Domain is any tree or sub-tree within the overall domain namespace.
- **Root Domain:** Root Domain is the root of the DNS tree. It is unnamed and is represented by a period (.).
- **Top-level Domain:** Usually top-level domain name is a two or three character name code that identifies the organizational or geographical status for the domain name.
  Example, .com, .biz, .net, .org, .gov, .in, .au etc.
- **Second-level Domain:** A second level domain is a unique name of variable length assigned to individuals or organizations that connect to the internet. Example: myuniversity.edu. Here second level name refers to ".myuniversity", which is assigned by InterNIC.
- **Sub Domains:** Large organizations can further subdivide its registered domain name by adding subdivisions that are represented by separate name portion.
  Example: mcse.omnisecu.com, rhce.omnisecu.com
- **Fully Qualified Domain Name (FQDN):** The entire hostname with its domain suffix such as sys-102.gas.chevron.com is called a Fully Qualified Domain Name (FQDN).
- **DNS Zone:** As shown in above figure, DNS namespace is hierarchical. Administratively, each level or node in the hierarchy represents a potential boundary of authority for management of the name space. A DNS zone is a portion of the global Domain Name System (DNS) namespace for which administrative responsibility has been delegated.

Why is DNS important?

DNS is like a phone book for the Internet. If you know a person's name but don't know their telephone number, you can simply look it up in a phone book. DNS provides this same service to the Internet.
When you visit *http://abc.com* in a browser, your computer uses DNS to retrieve the website's IP address of *204.13.248.115*. Without DNS, you would only be able to visit any website by visiting its IP address directly, such as http://204.13.248.115.

How DNS does works?

When you visit a domain such as *abc.com,* your computer follows a series of steps to turn the human-readable web address into a machine-readable IP address. This happens every time you use a domain name, whether you are viewing websites.

Step 1: Request information
The process begins when you ask your computer to resolve a hostname, such as visiting *http://abc.com*. The first place your computer looks is its local DNS cache, which stores information that your computer has recently retrieved.
If your computer doesn't already know the answer, it needs to perform a **DNS query** to find out.

Step 2: Ask the recursive DNS servers
If the information is not stored locally, your computer queries (contacts) your ISP's recursive DNS servers. These specialized computers perform the legwork of a DNS query on your behalf. Recursive servers have their own caches, so the process usually ends here and the information is returned to the user

Step 3: Ask the root nameservers
If the recursive servers don't have the answer, they query the root nameservers**.** A nameserver is a computer that answers questions about domain names, such as IP addresses. The thirteen root nameservers act as a kind of telephone switchboard for DNS. They don't know the answer, but they can direct our query to someone that knows where to find it.

Step 4: Ask the TLD nameservers
The root nameservers will look at the first part of our request, reading from right to left — *www.abc.**com*** — and direct our query to the Top-Level Domain (TLD) nameservers for *.com*. Each TLD, such as *.com*, *.org*, and *.us*, have their own set of nameservers, which act like a receptionist for each TLD. These servers don't have the information we need, but they can refer us directly to the servers that *do* have the information.

Step 5: Ask the authoritative DNS servers
The TLD nameservers review the next part of our request — *www.*abc.com — and direct our query to the nameservers responsible for this *specific* domain. These authoritative nameservers are responsible for knowing all the information about a specific domain, which are stored in DNS records.

Step 6: Retrieve the record
The recursive server retrieves the A record for *abc.com* from the authoritative nameservers and stores the record in its local cache. If anyone else requests the host record for *abc.com*, the recursive servers will already have the answer and will not need to go through the lookup process again. All records have a time-to-live value, which is like an expiration date. After a while, the recursive server will need to ask for a new copy of the record to make sure the information doesn't become out-of-date.

Step 7: Receive the answer
Armed with the answer, recursive server returns the A record back to your computer. Your computer stores the record in its cache, reads the IP address from the record, then passes this information to your browser. The browser then opens a connection to the webserver and receives the website.
This entire process, from start to finish, takes only milliseconds to complete.

What is reverse DNS?

Reverse DNS (rDNS) is a method of resolving an IP address into a domain name, just as the domain name system (DNS) resolves domain names into associated IP addresses. One of the applications of reverse DNS is as a spam filter. Here's how it works: Typically, a spammer uses an invalid IP address, one that doesn't match the domain name. A reverse DNS lookup program inputs IP addresses of incoming messages to a DNS database. If no valid name is found to match the IP address, the server blocks that message.

What are top level domains?

A top-level domain, or TLD, is the most general part of the domain. A top-level domain (TLD) is the last segment of the domain name. The TLD is the letters immediately following the final dot in an Internet address. The top-level domain is the furthest portion to the right (as separated by a dot). Common top-level domains are "com", "net", "org", "gov", "edu", and "io". Top-level domains are at the top of the hierarchy in terms of domain names. Certain parties are given management control over top-level domains by ICANN (Internet Corporation for Assigned Names and Numbers). These parties can then distribute domain names under the TLD, usually through a domain registrar.

What is BIND software?

BIND is open source software that enables you to publish your Domain Name System (DNS) information on the Internet, and to resolve DNS queries for your users. The name BIND stands for “Berkeley Internet Name Domain”, because the software originated in the early 1980s at the University of California at Berkeley.
The BIND software distribution has three parts:

1. Domain name resolver
2. Domain name authority server
3. DNS look up Tools such as DIG

BIND is transparent open source. If your organization needs some functionality that is not in BIND, you can modify it, and contribute the new feature back to the community by sending your source. BIND has evolved to be a very flexible, full-featured DNS system. BIND runs and is supported on a very wide variety of new and old operating systems, including most UNIX and LINUX variants, and some Windows platforms.

Name Server

Name servers are a fundamental part of the Domain Name System (DNS).The basic function of a name server is to answer queries by providing the information that those queries request. A DNS name server primarily translates domain and hostnames into IP addresses. Each domain is typically represented by at least two DNS servers.

- **Primary (master) name server** — this name server contains the master copy of authoritative information about the domains that it serves. In response to queries for information about its domains, this server provides that information marked as being authoritative. The primary is the ultimate source for data about the domain.
- **Secondary (slave) name server** — this name server gets all information for the domain from the primary name server. As is the case for the primary server, DNS considers the secondary’s information about the domain that it serves authoritative, although the domain records can be overwritten from the primary

Setting up DNS and Configuration

The DNS server software that comes with the current Fedora and RHEL versions is Berkeley Internet Name Domain (BIND) version 9.5. To configure BIND 9.5, you work with the following components:

- **Configuration file** (/var/named/chroot/etc/named.conf) – The main DNS server configuration file.
- **Zone directory** (/var/named/chroot/var/named) – The directory containing files that keep information about Internet root DNS servers (named.ca file) and information about the zones that you create for your DNS server.

- **Daemon process** (/usr/sbin/named) – The daemon process that listens for DNS requests and responds with information that the named.conf file presents.
- **Debugging tools** (named-checkconf , named-checkzone , and named- compilezone) – What you use to determine whether you created your DNS configuration correctly.

This will help you to step by step setup DNS server on CentOS and RedHat systems.

Network Scenario:

DNS Server IP: 192.168.1.254
DNS Server Name: ns1.tecadmin.net, ns2.tecadmin.net
Domain Name: demotecadmin.net
Domain IP to point: 192.168.1.100

Step 1: Install Bind package

Bind packages are available under default yum repositories. To install packages simple execute below command.

```
#yum install bind bind-chroot
```

Step 2: Edit main configuration file

Using chroot environment this file is located at /var/named/chroot/etc directory. Now edit main configuration file and update content.

```
#vim /var/named/chroot/etc/named.conf
```

Step 3: Create Zone file for your Domain

After creating bind main configuration file, create a zone file for you domain as per configuration, for example **demotecadmin.net.db** here.

```
#vim /var/named/chroot/var/named/demotecadmin.net.db
```

Step 4: Start bind service

Start named (bind) service using following command.

```
#service named restart
```

Enable auto start on system boot.

```
#chkconfig named on
```

Step 5: Test your DNS setup

Send query to your dns server directly using below command.

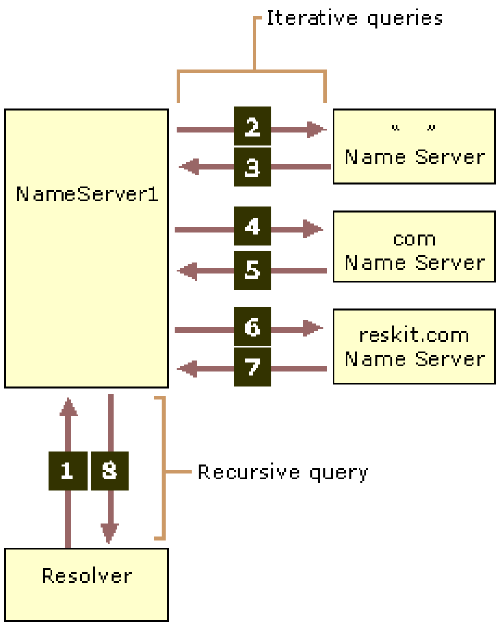
Syntax: nslookup <domainname> <dns server name/ip>

```
#nslookup demotecadmin.net 192.168.1.254
```

DNS Queries

The major task carried out by a DNS server is to respond to queries (questions) from a local or remote resolver or other DNS acting on behalf of a resolver. A query would be something like 'what is the IP address of fred.example.com'.

Most of the queries that a DNS server will receive will be for domains for which it has no knowledge, that is, for which it has no local zone files. DNS software typically allows the name server to respond in different ways to queries about which it has no knowledge.



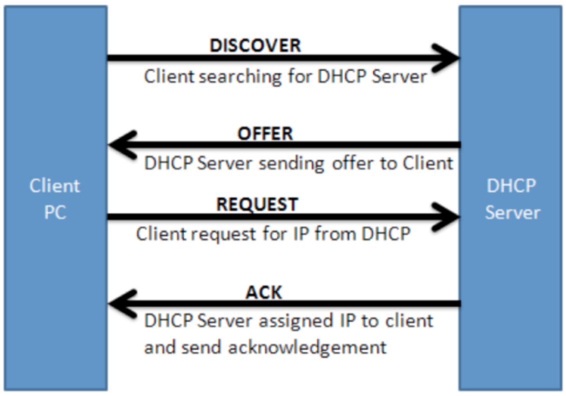
There are three types of queries defined for DNS:

1. A recursive query - the complete answer to the question is always returned. DNS servers are not required to support recursive queries.
2. An Iterative (or non-recursive) query - where the complete answer MAY be returned or a referral provided to another DNS. All DNS servers must support Iterative queries.
3. An Inverse query - where the user wants to know the domain name given a resource record. Reverse queries were poorly supported, very infrequent and are now obsolete (RFC 3425).

Introduction to DHCP

Dynamic Host Configuration Protocol (DHCP) is a client/server protocol that automatically provides an Internet Protocol (IP) host with its IP address and other related configuration information such as the subnet mask and default gateway.
Every device on a TCP/IP-based network must have a unique unicast IP address to access the network and its resources. Without DHCP, IP addresses must be configured manually for new computers or computers that are moved from one subnet to another, and manually reclaimed for computers that are removed from the network.
DHCP enables this entire process to be automated and managed centrally. The DHCP server maintains a pool of IP addresses and leases an address to any DHCP-enabled client when it starts up on the network. Because the IP addresses are dynamic (leased) rather than static (permanently assigned), addresses no longer in use are automatically returned to the pool for reallocation.

How DHCP assigns IP addresses or how does DHCP works?


Client PC
DISCOVER
Client searching for DHCP Server
OFFER
DHCP Server sending offer to Client
REQUEST
Client request for IP from DHCP
ACK
DHCP Server assigned IP to client and send acknowledgement
DHCP Server

DHCP assigns an IP address when a system is started, for example:

1. A user turns on a computer with a DHCP client.
2. The client computer sends a broadcast request (called a DISCOVER or DHCPDISCOVER), looking for a DHCP server to answer.
3. The router directs the DISCOVER packet to the correct DHCP server.
4. The server receives the DISCOVER packet. Based on availability and usage policies set on the server, the server determines an appropriate address (if any) to give to the client. The server then temporarily reserves that address for the client and sends back to the client an OFFER (or DHCPOFFER) packet, with that address information. The server also configures the client's DNS servers, WINS servers, NTP servers, and sometimes other services as well.
5. The client sends a REQUEST (or DHCPREQUEST) packet, letting the server know that it intends to use the address.
6. The server sends an ACK (or DHCPACK) packet, confirming that the client has a been given a lease on the address for a server-specified period of time.

A computer running the DHCP Server service that holds information about available IP addresses and related configuration information as defined by the DHCP administrator and responds to requests from DHCP clients is called **DHCP Server.** A computer that gets its IP configuration information by using DHCP is called **DHCP client**.

Setting up DHCP server

To configure a DHCP server you need to install the dhcp package. Assuming you have already set up the physical connections between your DHCP server and the client computers on your network (presumably an Ethernet LAN), the minimum tools you need to get the DHCP servers working are:

1. A firewall that allows requests for DHCP service
2. A configured /etc/dhcpd.conf file
3. A running dhcpd server daemon (which can be started at boot time)

After the DHCP server is running, it listens to UDP port 67 for requests from DHCP clients on the LAN. A client simply boots up (with an Ethernet network interface turned on and DHCP identified as its method of getting network addresses). This causes the client to send out a DHCP discovery request to 255.255.255.255 (the global broadcast address). The DHCP server picks up that request and feeds the client the information it needs to get up and running on the network.

Configuration of DHCP server

Step 1: Prepare your server before DHCP server configuration.
Before we start configuration of DHCP Server first we need to prepare our server for that. First Assign a Static IP Address to your Server. Give a proper Hostname to your Server.

Step 2: Install DHCP package.
Install required packages and dependencies for Linux DHCP Server

```
#yum install dhcp
```

Step 3: Edit the `/etc/dhcp/dhcpd.conf` file to store the settings.
At this point we can add few additional settings to our DHCP configuration, namely the default and max lease time expiry.

Step 4: Edit `/etc/sysconfig/dhcpd` to configure settings for the DHCP server itself.

Step 5: Touch the `/var/lib/dhcpd/dhcpd.leases` file, which stores information about client leases.

```
#touch /var/lib/dhcpd/dhcpd.leases
```

Step 6: Enter the following commands to start the DHCP service and ensure that it starts after a reboot.

```
# service dhcpd start
# chkconfig dhcpd on
```

Setting up DHCP client

Most client computers (Window, Linux, or Mac) are configured by default to boot up using DCHP to connect to the network. However, when that is not the case, you may need to indicate manually that DHCP be used.
Configuring a network client to get addresses from your DHCP server is fairly easy. Different types of operating systems, however, have different ways of using DHCP. Here is example for setting up Linux DHCP clients

1. While you are initially installing Linux, click Configure using DHCP on the Network Configuration screen. Your network client should automatically pick up its IP address from your DHCP server when it starts up.
2. To set up DHCP after installation, open the Network Configuration window (select System → Administration→ Network or run the system-config-network command).
3. From the Network Configuration window
   a. Click the Devices tab (on by default).
   b. Click your Ethernet device (probably eth0).

   c. Click Edit.
   d. Click the General tab.
   e. Click "Automatically obtain IP address settings with" and select dhcp.
   f. Click OK.
   g. Click Activate.
4. Then, from a Terminal window, type:
```
# service network restart
```

Network Information Service (NIS)

Network Information Service (NIS) is the traditional directory service on Unix/Linux platforms. NIS enables you to create user accounts that can be shared across all systems on your network. The user account is created only on the NIS server. NIS clients download the necessary user name and password data from the NIS server to verify each user login.
NIS was originally known as Yellow Pages but the name was changed due to trademark issues. This is the reason why NIS commands begin with `yp`. NIS is a Remote Procedure Call (RPC)-based client/server system that allows a group of machines within an NIS domain to share a common set of configuration files.

Configuring a NIS server

Step 1: Make sure the following packages are installed in your machine.
ypserv - This package contains the NIS server daemon - `ypserv`, and the NIS password daemon- `yppasswdd`
portmap - This is an RPC daemon upon which NIS runs. This package is mandatory.
Step 2: Insert the following line in the /etc/sysconfig/network file.
```
NISDOMAIN=mynisdomain
```
Step 3: Specify the networks you wish NIS to recognize in /var/yp/securenets.
```
#Permit access to xyz.com network:
```
Step 4: If you have slave NIS servers then enter their names in /var/yp/ypservers
Step 6: Finally run the following command
```
# /usr/lib/yp/ypinit -m
```

Introduction to Database Server

A database is a collection of information that is organized so that it can easily be accessed, managed, and updated. A database server is a computer program that provides database services to other computer programs or computers using a client-server model. The term may also refer to a computer dedicated to running such a program.

Database management systems frequently provide database-server functionality, and some database management systems (DBMSs) (such as MySQL) rely exclusively on the client–server model for database access.

Most database servers respond to a query language. Each database understands its query language and converts each submitted query to server-readable form and executes it to retrieve results.

MySQL Database Server

MySQL is a popular structured query language (SQL) database server. Like other SQL servers, MySQL provides the means of accessing and managing SQL databases. However, MySQL also provides tools for creating database structures, as well as for adding, modifying, and removing data from those structures. Because MySQL is a relational database, data can be stored and controlled in small, manageable tables. Those tables can be used in combination to create flexible yet complex data structures.

MySQL has been ported to several different operating systems (primarily UNIX and Linux systems, although there are Windows versions and now even a Mac OS X version as well).
MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company. MySQL is **becoming so popular** because of many good reasons:

1. MySQL is released under an open-source license. So you have nothing to pay to use it.
2. MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
3. MySQL uses a standard form of the well-known SQL data language.
4. MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
5. MySQL works very quickly and works well even with large data sets.
6. MySQL is very friendly to PHP, the most appreciated language for web development.
7. MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
8. MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

You need at least the mysql and mysql-server packages installed to set up MySQL using the procedures described here.

- **mysql** — This software package contains a lot of MySQL client programs (in /usr/bin), several client shared libraries, the default MySQL configuration file (/etc/my.cnf), a few sample configuration files, files to support different languages, and documentation.
- **Mysql -server** — This software package contains the MySQL server daemon (mysqld) and the mysqld startup script (/etc/init.d/mysqld). The package also creates various administrative files and directories needed to set up the MySQL databases.
- **mysql-devel** — This software package contains libraries and header files required for developing MySQL applications.

If MySQL isn't installed yet, type the following:

```
# yum install mysql-server
```

Configuring MySQL Server

Like most server software in Fedora and RHEL, the MySQL server relies on a startup script and a configuration file to provide the service. Server activities are logged to the mysqld.log file in the /var/log directory. There are also mysql user and group accounts for managing MySQL activities.

When the MySQL software is installed, it automatically creates a mysql user account and a mysql group account. These user and group accounts are assigned to MySQL files and activities. In this way, someone can manage the MySQL server without needing to have root permission.
The mysql user entry appears in the /etc/password file as follows:

```
mysql:x:27:27:MySQL Server:/var/lib/mysql:/bin/bash
```

To administer MySQL, you need to have at least one administrative account. By default, the root user has full access to your MySQL server database and no password assigned. You can assign a password to the root user using the mysqladmin command, once the mysqld server is running. To add the root user as a MySQL administrator, log in as the root user and type the following from a Terminal window:

```
# mysqladmin -u root password myownpasswd
```

MySQL maintains a list of users and passwords that is separate from the list maintained by Linux.

Installing and Configuring MySQL database server

The following are example instructions to install and configure MySQL database for the Oracle Linux distribution of Linux operating system:

Step 1: Install the MySQL database server package.
You can use the Yum tool to install MySQL on Oracle Linux:
#yum install mysql-community-server.

Step 2: To start the MySQL server immediately, type the following from a Terminal window as root:
# service mysqld start
To set the MySQL server to start each time the computer reboots, type the following
# chkconfig mysqld on

Step 3: Launch the MySQL Command-Line Client:
#mysql -u root -p
The -p option is needed only if a root password is defined for MySQL. Enter the password when prompted.

Step 4: Create a user (for example, `abc`) and a strong password:
#mysql> create user 'abc' identified by 'abc';

Step 5: Create the database (for example, `abc`) and grant all access to the user `abc` as follows:
#mysql> create database abc;
#mysql> grant all on abc.* to 'abc';

Step 6: Configure your MySQL installation to handle large entries;
#vi /etc/mysql/my.cnf

Step 7: Save and close the file. You can reload or restart the changes as follows:
`#/sbin/service mysqld restart`

Working with MySQL database

The first time you start the MySQL server (using the startup script described previously), the system creates the initial grant tables for the MySQL database. It does this by running the mysql_install_db command.

The mysql_install_db command starts you off with two databases: mysql and test. As you create data for these databases, that information is stored in the /var/lib/mysql/mysql and /var/lib/mysql/test directories, respectively.

To get started creating databases and tables, you can use the mysql command. From any Terminal window, open the mysql database on your computer by typing the following:

```
$ mysql -u root -p mysql
Enter password: *********
```

1. To create a new database name, use the CREATE DATABASE command at the mysql> prompt. For example, to create a database named allusers, type the following: mysql> CREATE DATABASE allusers;

   This action creates a database called allusers in the /var/lib/mysql directory.
2. To see what databases are available for your mysql server, type the following at the mysql> command prompt.
   ```
   #mysql> SHOW DATABASES
   ```
3. To work with the database you just created (allusers), you need to make allusers your current database.
   ```
   mysql> USE allusers;
   Database changed
   ```
4. Creating a table for your database requires some planning and understanding of table syntax.
   ```
   mysql> CREATE TABLE name (
                     -> firstname varchar(20) not null,
                     -> lastname varchar(20) not null,
                     -> streetaddr varchar(30) not null,
                     -> city varchar(20) not null,
                     -> state varchar(20) not null,
                     -> zipcode varchar(10) not null
                     -> );
     Query OK, 0 rows affected (0.00 sec)
   ```

Loading data from a file

Using the LOAD DATA command during a MySQL session, you can load a file containing database records into your MySQL database. Here are a few things you need to know about creating a data file to be loaded into MySQL:

- You can create the file using any Linux text editor.
- Each record, consisting of all the columns in the table, must be on its own line. (A line feed indicates the start of the next record.)
- Separate each column by a Tab character.
- You can leave a column blank for a particular record by placing a \N in that column.
- Any blank lines you leave in the file result in blank lines in the database table.

Step 1: $ mysql -u root -p
```
Enter password: *******
Mysql>
```

Step 2: mysql> USE allusers;
```
Database changed
```

Step 3: mysql> LOAD DATA INFILE "/tmp/name.txt" INTO TABLE name;
```
Query OK, 3 rows affected (0.02 sec)
Records: 3 Deleted: 0 Skipped: 0 Warnings: 0
```

Step 4: Type the following to make sure that the records have been added correctly:
```
mysql> SELECT * FROM name;
```

Some of Example Queries

1. Type the following command to choose (SELECT) all records (*) from the name table and display them in the order in which they were entered into the database.
   ```
   mysql> SELECT * FROM name;
   ```

2. The following command displays all records from the name table that have the lastname column set to Jones.
   ```
   mysql> SELECT * FROM name WHERE lastname = "Jones";
   ```

3. In the following command, records that have either Chris or Howard as the first name are matched and displayed.
   ```
   mysql> SELECT * FROM name WHERE firstname = "Chris" OR firstname = "Howard";
   ```

4. To match and display a record based on the value of two columns in a record, you can use the AND operator.
   ```
   mysql> SELECT * FROM name WHERE firstname = "Chris" AND lastname = "Smith";
   ```

5. The following command displays the firstname , lastname , and zipcode records for all of the records in the name table:
   ```
   mysql> SELECT firstname, lastname, zipcode FROM name;
   ```

6. You can also mix column selection with record selection as shown in the following example:
   ```
   mysql> SELECT firstname,lastname,city FROM name WHERE firstname = "Chris";
   ```

7. You can sort records based on the values in any column you choose. For example, using the ORDER BY operator, you can display the records based on the lastname column:
   ```
   mysql> SELECT * FROM name ORDER BY lastname
   mysql> SELECT * FROM name ORDER BY city;
   ```

8. To add a column to the end of your table that displays the current date, type the following:
   ```
   mysql> ALTER TABLE name ADD curdate TIMESTAMP;
   ```
9. If you decide later that you want to remove that column, you can remove it by typing the following:
   ```
   mysql> ALTER TABLE name DROP COLUMN curdate;
   ```
10. If you want to change the name of an existing column, you can do so using the CHANGE option to ALTER .
    ```
    mysql> ALTER TABLE name CHANGE city town varchar(20);
    ```

11. You can use UPDATE to change the values in a selected table.
    ```
    mysql> UPDATE name SET streetaddr = "933 3rd Avenue" WHERE firstname = "Chris";
    ```
12. To remove an entire row (that is, one record), you can use the DELETE command.
    ```
    mysql> DELETE FROM name WHERE firstname = "Chris";
    ```

13. You can use the mysqldump command to back up your MySQL databases.
    ```
    # mysqldump -u root -p --opt --all-databases > /home/chris/all-databases
    ```

14. Stop MySQL temporarily by typing the following from a Terminal window as root user:
    ```
    # /etc/init.d/mysqld stop
    ```

What are modules on Linux system? How do you use them?

Modules are pieces of code that can be loaded and unloaded into the kernel upon demand. They extend the functionality of the kernel without the need to reboot the system. For example, one type of module is the device driver, which allows the kernel to access hardware connected to the system. Most current Unix-like systems and Microsoft Windows support loadable kernel modules, although they might use a different name for them.

They are also known as **Kernel Loadable Modules** (or **KLM**), and simply as **Kernel Modules** (**KMOD**). Loadable kernel modules in Linux are loaded (and unloaded) by the **modprobe** command. They are located in `/lib/modules` and have had the extension `.ko` ("kernel object").

The intention is to have only the most critical drivers your system needs built into the kernel; these are called resident drivers. Other drivers that are added dynamically as needed are referred to as loadable modules. The trend is to keep the basic kernel as lean as possible, so that each running system has only a few resident drivers and it can dynamically add what it needs.

To see which modules are currently loaded into the running kernel on your computer, you can use the lsmod command. The *lsmod* command lists the loaded kernel modules. In emergency cases, when the system fails to boot due to e.g. broken modules, specific modules can be enabled or disabled by modifying the kernel boot parameters list (for example, if using GRUB, by pressing 'e' in the GRUB start menu, then editing the kernel parameter line).

You can load any module that has been compiled and installed (to the /lib/modules directory) into your running kernel using the modprobe command. The most common reasons for loading a module are that you want to use a feature temporarily (such as loading a module to support a special file system on a floppy you want to access) or to identify a module that will be used by a particular piece of hardware that could not be autodetected.

```
# modprobe parport
```

You can remove a module from a running kernel using the rmmod command. For example, to remove the module parport_pc from the current kernel, type the following:

```
# rmmod parport_pc
```

If the module is not currently busy, the parport_pc module is removed from the running kernel. (Instead of rmmod , you can use modprobe -r to remove the module, plus related modules.)

ISP simulation

Open-source switch and router software, combined with open-source server software and other open-source tools provides a set of resources that will allow a learner to simulate the functionality of complex enterprise or service-provider networks on a single personal computer.An integration server is primarily implemented in IT environments that are composed of IT products and solutions from different platforms and/or architectures. It also serves as a middleware server, serving as an intermediate server between different layers.

An ISP (Internet service provider) is a company that provides individuals and other company's access to the Internet and other related services such as Web site building and virtual hosting. ISPs provide Internet access, employing a range of technologies to connect users to their network. Internet hosting services provide email, web -hosting, or online storage services. Other services include virtual server, cloud services, or physical server operation

Integration of different servers such as DNS server, mail server, web server, DHCP server is also important for our practical life. We can set up and configure different servers in a single system which can then provide many facilities. Somehow it is quite complex to integrate such services on one system than setting up a single server. Your ISP should be able to maintain the Domain Name Server (DNS) entry for your machine. The DNS entry allows Internet surfers to access your IP address using your domain name.

Set the run-level to 3 such that it becomes the default run level. Explain different types of run-level with their functions.

Run-level script is a software package that has a service to start at boot time (or when the system changes run levels) can add a script to the /etc/init.d directory. That script can then is linked to an appropriate run-level directory and run with either the start or stop option. To use chkconfig , ensure that the following lines are included in the /etc/init.d/my_daemon script:

```
# chkconfig: 3 82 28
# description: Does something pretty cool - you really \
# have to see it to believe it!
# processname: my_daemon
```

The line chkconfig: 345 82 28 sets the script to start in run levels 3. It sets the scripts to be set to 82 for those run levels. It sets stop scripts to be set to 28 in all other levels. Here are some run level scripts examples:

acpid - Controls the Advanced Configuration and Power Interface daemon, which monitors events in the kernel and reports them to user level.

anacron - Runs cron jobs that were not run at their intended times due to the system being down.

atd - Starts or stops the at daemon to receive, queue, and run jobs submitted via the at or batch commands. (The anacron run-level script runs at and batch jobs that were not run because the Computer was down.)

Bluetooth - Starts services such as authentication, discovery, and human interface devices for communicating with Bluetooth devices. ConsoleKit maintains a list of user sessions. crond Starts or stops the cron daemon to periodically run routine commands.

cups - Controls the printer daemon (cupsd) that handles spooling printing requests.

"Why let Microsoft give you Windows while Linux can give you a house?"