Difference between FTP and TFTP:-

FTP:
FTP stands for File Transfer Protocol. This type of protocol is used to transfer or copies the file from one host to another host. But there may be some problems like different file name and different file directory while sending and receiving file in different hosts or systems. And in FTP, secure channel is not provided to transfer the files between the hosts or systems. FTP works on two ports: 20 and 21 One for data and another is for connection control.

TFTP:
TFTP stands for Trivial File Transfer Protocol. TFTP is used to transfer a file either from client to server or from server to client without the need of FTP feature. Software of TFTP is smaller than FTP. TFTP works on 69 Port number and its service is provided by UDP.
Now, we shall see the difference between FTP and TFTP:

S.NO	FTP	TFTP
1.	FTP stands for File Transfer Protocol.	TFTP stands for Trivial File Transfer Protocol.
2.	The software of FTP is larger than TFTP.	While software of TFTP is smaller than FTP.
3.	FTP works on two ports: 20 and 21.	While TFTP works on 69 Port number.
4.	FTP services are provided by TCP.	While TFTP services are provided by UDP.
5.	The complexity of FTP is higher than TFTP.	While the complexity of TFTP is less than FTP complexity.
6.	There are many commands or messages in FTP.	There are only 5 messages in TFTP.
7.	FTP need authentication for communication.	While TFTP does not need authentication for communication.

Following are the important difference between FTP and TFTP.

Sr. No.	Key	FTP	TFTP
1	Stands For	File Transfer Protocol	Trivial File Transfer Protocol
2	Software Size	FTP software is heavyier than TFTP.	TFTP is light.
3	Ports	FTP works on port 20, 21.	TFTP works on port 69.
4	Protocol used	FTP is based on TCP.	TFTP is based on UDP.

5	Complexity	FTP is more complex than TFTP.	TFTP is less complex than FTP.
6	Commands	FTP has lots of commands or messages.	TFTP has only five messages.
7	Authentication	Authentication is must for FTP.	Authentication is not requird in case of TFTP.

Difference between BOOTP and DHCP:-

BOOTP stands for **Bootstrap Protocol. DHCP** stands for **Dynamic host configuration protocol**. The operating of each protocol is totally different in some manner. Dynamic host configuration protocol is also the extended version of the Bootstrap Protocol. BOOTP, Bootstrap protocol, is used to configure host and get address of host along with bootstrap info. DHCP, Dynamic Host Configuration Protocol Server is an extended version of BOOTP and is used to configure hosts mechanically.

S.NO	BOOTP	DHCP
1.	BOOTP stands for Bootstrap Protocol.	While DHCP stands for Dynamic host configuration protocol.
2.	BOOTP does not provide temporary IP addressing.	While DHCP provides temporary IP addressing for only limited amount of time.
3.	BOOTP does not support DHCP clients.	While it support BOOTP clients.
4.	In BOOTP, manual-configuration takes place.	While in DHCP, auto-configuration takes place.
5.	BOOTP does not support mobile machines.	Whereas DHCP supports mobile machines.
6.	BOOTP can have errors due to manual-configuration.	Whereas in DHCP errors do not occure mostly due to auto-configuration.

Following are the important differences between BOOTP and DHCP.

Sr. No.	Key	BOOTP	DHCP
1	Definition	BOOTP stands for Bootstrap protocol.	DHCP stands for Dynamic Host Configuration Protocol.
2	Temporary IP Address	BOOTP has no support for temporary IP Addressing.	DHCP Server support for temporary IP Addressing but for limited period of time.
3	Client Support	BOOTP does not support DHCP Clients.	DHCP server supports BOOTP Clients.
4	Configuration Type	In BOOTP, configuration has to be done manually.	In DHCP, configuration is automatic.

5	Mobile Machine Support	Mobile machine are not supported.	Mobile machines are supported
6	Error Probability	Configuration being manual often leads to errors.	Automatic configuration prevents any error to occur.

BOOTP Vs DHCP

BASIS OF COMPARISON	**BOOTP**	**DHCP**
Acronym For	**BOOTP** stands for Bootstrap Protocol.	**DHCP** stands for Dynamic host configuration protocol.
IP	BOOTP can only provide an IP to a computer while it is booting.	DCHP can provide an IP when the OS is already loaded.
Client Configuration	BOOTP supports a limited number of client configuration parameters referred to as vendor extensions.	DHCP supports a larger and extensible set of client configuration parameters referred to as options.
System Restart	BOOTP client do not rebind or renew configuration with the BOOTP server except when the system restarts.	DHCP clients do not require a system restart to rebind or renew configuration with the DHCP server.
Configuration Process	BOOTP uses a two-phase bootstrap configuration process in which clients contact BOOTP servers to perform address determination and boot file name selection and clients contact Trivial File Transfer Protocol (TFTP) servers to perform file transfer of their boot image.	DHCP uses a single-phase boot configuration process whereby a DHCP client negotiates with a DHCP server to determine its IP address and obtain any other initial configuration details it needs for network operation.
Lease	BOOTP has a 30 day lease on the IP address as a default.	DHCP has eight-day lease duration for Microsoft and one day for Cisco routers.
Mobile Machine Support	BOOTP does not support mobile machines.	DHCP supports mobile machines.
Configuration Type	In BOOTP, manual-configuration takes place.	In DCHP, auto-configuration takes place.
IP Addressing	BOOTP does not provide temporary IP addressing.	DCHP provides temporary IP addressing for only limited amount of time.
Possibility Of Errors	Due to manual-configuration BOOTP is faced with errors.	Due to auto-configuration in DHCP, it is immune to errors.
Storage Disk	BOOTP provides the information to the diskless computer or workstation.	DCHP requires disks to store and forward the information.
Compatibility With Clients	BOOTP is not compatible with DHCP clients.	DHCP support BOOTP clients.

Difference between FQDN and PQDN:-

Fully Qualified Domain Name (FQDN) If a label is terminated by a null string, it is called a **fully qualified domain name (FQDN).** An FQDN is a domain name that contains the full name of a host. It contains all labels, from the most specific to the most general, that uniquely define the name of the host. For example, the domain name is the FQDN of a computer named *challenger* installed at the Advanced Technology Center (ATC) at De Anza College. A DNS server can only match an FQDN to an address. Note that the name must end with a null label, but because null means nothing, the label ends with a dot (.)

challenger.atc.fhda.edu.

Partially Qualified Domain Name (PQDN) If a label is not terminated by a null string, it is called a **partially qualified domain name (PQDN).** A PQDN starts from a node, but it does not reach the root. It is used when the name to be resolved belongs to the same site as the client. Here the resolver can supply the missing part, called the *suffix*, to create an FQDN. For example, if a user at the *fhda.edu.* site wants to get the IP address of the challenger computer, he or she can define the partial name

challenger

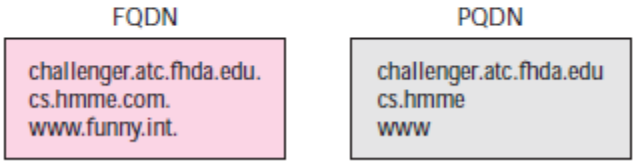
The DNS client adds the suffix *atc.fhda.edu*. before passing the address to the DNS server.

The DNS client normally holds a list of suffixes. The following can be the list of suffixes at De Anza College. The null suffix defines nothing. This suffix is added when the user defines an FQDN.

atc.fhda.edu fhda.edu null

Figure 19.4 shows some FQDNs and PQDNs.

Figure 19.4 *FQDN and PQDN*



BOOTP Packet Format:-

BOOTP requests and replies are encapsulated in UDP datagrams, as shown in Figure 16.1.

Figure 16.1. Encapsulation of BOOTP requests and replies within a UDP datagram.

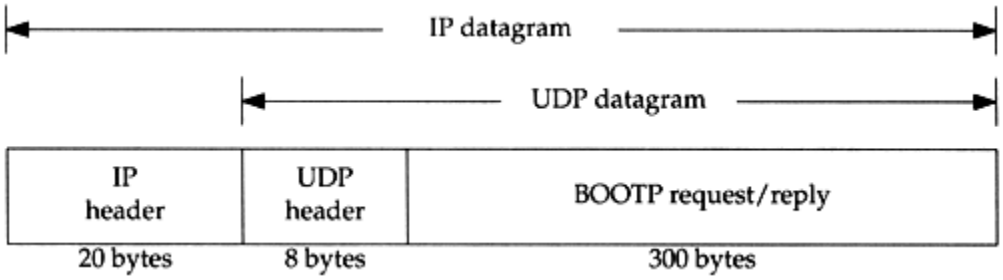
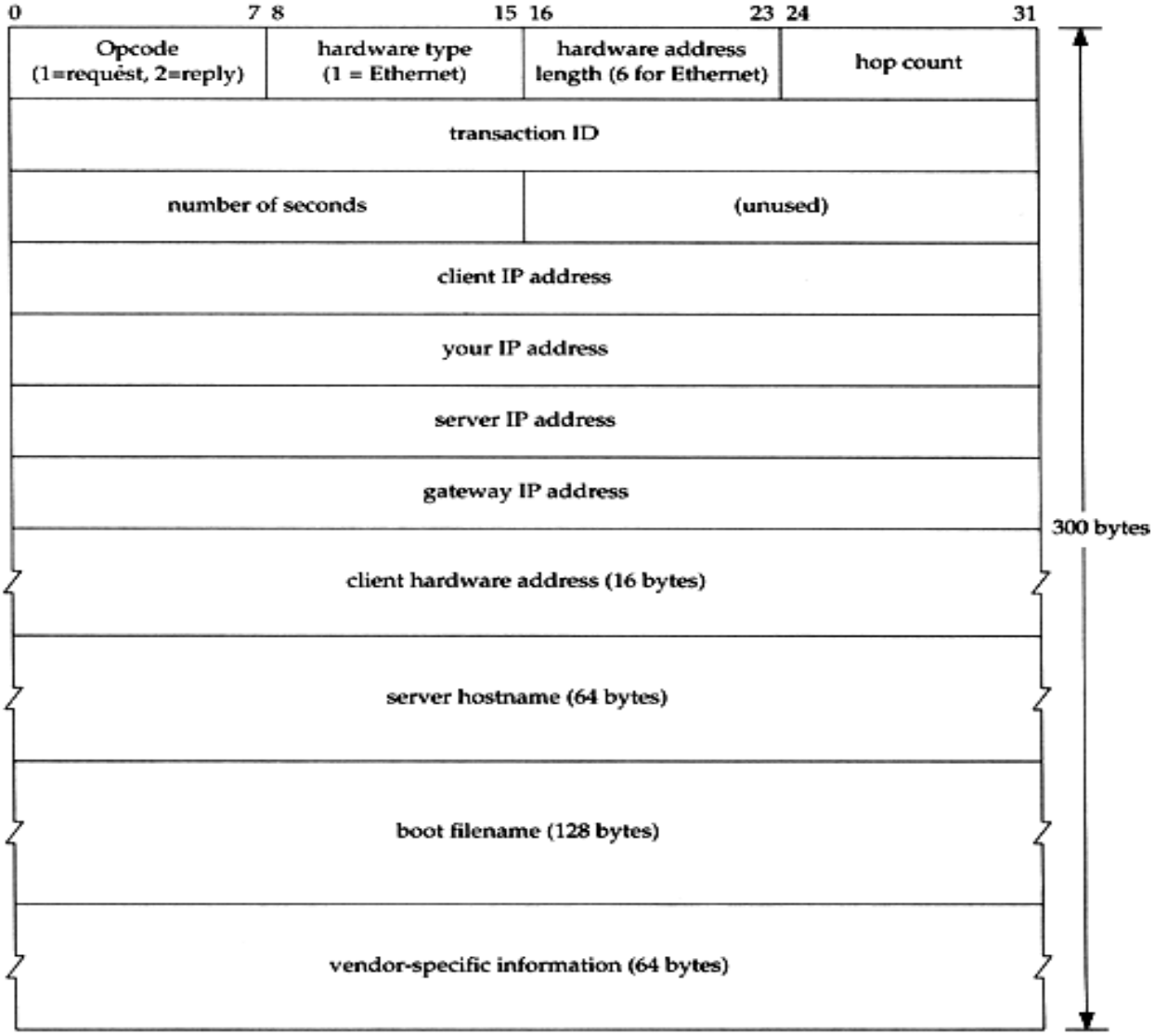


Figure 16.2 shows the format of the 300-byte BOOTP request and reply.

Figure 16.2. Format of BOOTP request and reply.



Opcode is 1 for a request and 2 for a reply. The *hardware type* field is 1 for a 10 Mbits/sec Ethernet, the same value that is in the field of the same name in an ARP request or reply (Figure 4.3). Similarly, the *hardware address length* is 6 bytes for an Ethernet.

The *hop count* is set to 0 by the client, but can be used by a proxy server (described in Section 16.5).

The *transaction ID* is a 32-bit integer set by the client and returned by the server. This lets the client match a response with a request. The client should set this to a random number for each request.

Number of seconds can be set by the client to the time since it started trying to bootstrap. The servers can look at this value, and perhaps a secondary server for a client won't respond until the number of seconds has exceeded some value, implying that the client's primary server is down.

If the client already knows its IP address, it fills in the *client IP address*. Otherwise, the client sets this to 0. In the latter case the server fills in *your IP address* with the client's IP address. The *server IP address* is filled in by the server. If a proxy server is used (Section 16.5), that proxy server fills in its *gateway IP address*.

The client must set its *client hardware address*. Although this is the same value as in the Ethernet header, by placing the field in the UDP datagram also, it is easily available to any user process (e.g., a BOOTP server) that receives the datagram. It is normally much harder (or impossible) for a process reading UDP datagrams to determine the fields in the Ethernet header that carried the UDP datagram.

The *server hostname* is a null terminated string that is optionally filled in by the server. The server can also fill in the *boot filename* with the fully qualified, null terminated pathname of a file to bootstrap from.

The *vendor-specific area* is used for various extensions to BOOTP. Section 16.6 describes some of these extensions.

When a client is bootstrapping using BOOTP (an opcode of 1) the request is normally a link-layer broadcast and the destination IP address in the IP header is normally 255.255.255.255 (the limited broadcast, Section 12.2). The source IP address is often 0.0.0.0 since the client does not know its own IP address yet. Recall from Figure 3.9 that 0.0.0.0 is a valid source IP address when a system is bootstrapping itself.

Port Numbers

There are two well-known ports for BOOTP: 67 for the server and 68 for the client. This means the client does not choose an unused ephemeral port, but uses 68 instead. The reason two port numbers were chosen , instead of just one for the server, is that a server's reply can be (but normally isn't) broadcast.

If the server's reply were broadcast, and if the client were to choose an ephemeral port number, these broadcasts would also be received by other applications on other hosts that happen to be using the same ephemeral port number. Hence, it is considered bad form to broadcast to a random (i.e., ephemeral) port number.

If the client also used the server's well-known port (67) as its port, then all servers on the network are awakened to look at each broadcast reply. (If all the servers were awakened, they would examine the opcode, see that it's a reply and not a request, and go back to sleep.) Therefore the choice was made to have all clients use a single well-known port that differs from the server's well-known port.

If multiple clients are bootstrapping at the same time, and if the server broadcasts the replies, each client sees the replies intended for the other clients. The clients can use the transaction ID field in the BOOTP header to match replies with requests, or the client can examine the returned client hardware address.

DHCP Packet Format:-

DHCP servers can interoperate with BOOTP clients.
DHCP options have the same format as BOOTP vendor extensions.
There are two primary differences between DHCP and BOOTP. First, DHCP defines mechanisms through which clients can be assigned a network address for a finite lease, allowing for serial reassignment of network addresses to different clients. Second, DHCP provides the mechanism for a client to acquire all of the IP configuration parameters that it needs in order to operate.

DHCP introduces a small change in terminology intended to clarify the meaning of one of the fields. What was the "vendor extensions" field in BOOTP has been re-named the "options" field in DHCP. Similarly, the tagged data items that were used inside the BOOTP "vendor extensions" field, which were formerly referred to as "vendor extensions," are now termed simply "options."

MAC header	IP header	UDP header	DHCP header	Data :::

DHCP header:

0 0	0 1	0 2	0 3	0 4	0 5	0 6	0 7	0 8	0 9	1 0	1 1	1 2	1 3	1 4	1 5	1 6	1 7	1 8	1 9	2 0	2 1	2 2	2 3	2 4	2 5	2 6	2 7	2 8	2 9	3 0	3 1
Opcode								Hardware type								Hardware address length								Hop count							
Transaction ID																															
Number of seconds																Flags															
Client IP address																															
Your IP address																															
Server IP address																															
Gateway IP address																															
Client hardware address :::																															
Server host name :::																															
Boot filename :::																															
Options :::																															

Opcode. 8 bits.

Value	Description
1	BOOTREQUEST, Boot request.
2	BOOTREPLY, Boot reply.

Hardware type. 8 bits.
Hardware address length. 8 bits.
Hop count. 8 bits.
This field is used by relay agents.
Transaction ID. 32 bits.
A random number chosen by the client, used by the client and server to associate messages and responses between a client and a server.
Number of seconds. 16 bits.
The elapsed time in seconds since the client began an address acquisition or renewal process.

Flags. 16 bits.

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
B	0														

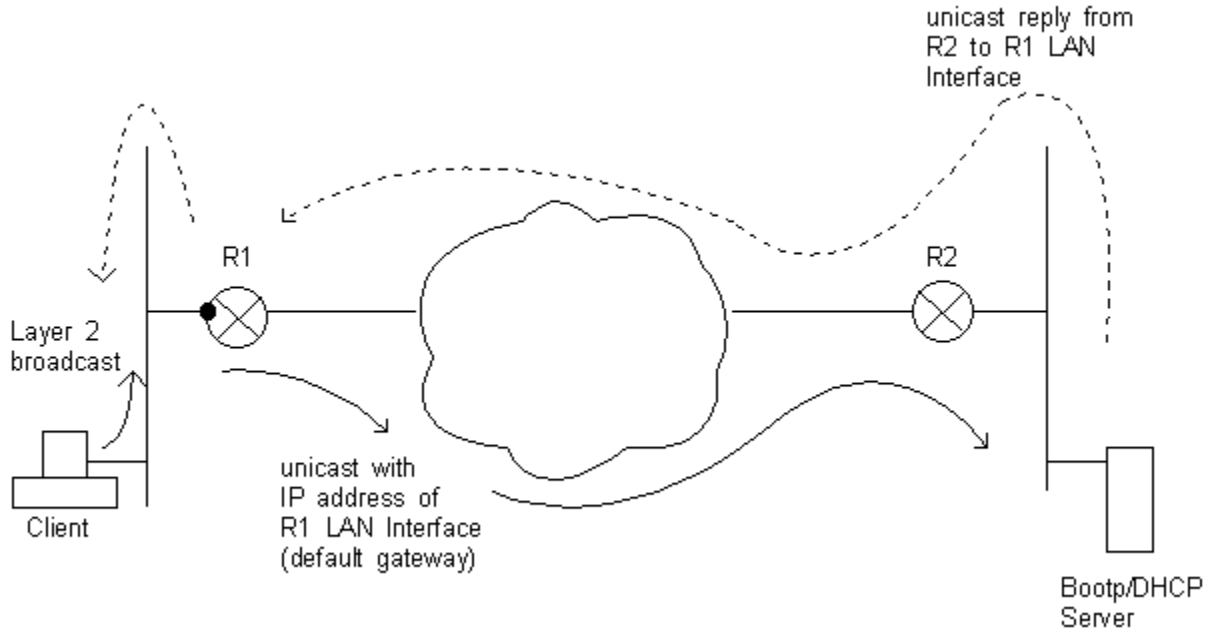
B, Broadcast. 1 bit.
Client IP address. 32 bits.
Your IP address. 32 bits.````
Server IP address. 32 bits.
Gateway IP address. 32 bits.
Client hardware address. 16 bytes.
Server host name. 64 bytes.
Boot filename. 128 bytes.
BOOTP/DHCP options. Variable length.

The first four bytes contain the (decimal) values 99, 130, 83 and 99. This is the same magic cookie as has been defined for BOOTP. The remainder of the field consists of a list of tagged parameters that are called options. All of the vendor extensions used by BOOTP are also DHCP options.

Bootstrap Protocol (BootP) and DHCP:-

BootP:-

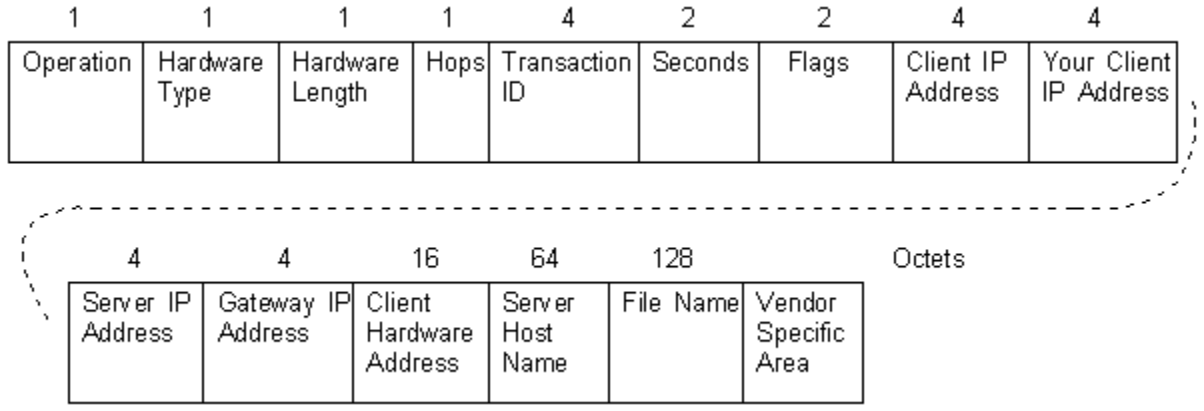
The problem with using **Reverse Address Resolution Protocol (RARP)** (**RFC 903**) for assigning IP addresses to clients is that it operates at the data link layer and is therefore limited to the local LAN. Bootp however uses IP/UDP (port 67 for the server destination port, and port 68 for the client source port) and can cross routers. If a client does not know its own IP address when it boots up then it can utilise a **Bootp** server to obtain its IP address. It will use a IP broadcast address of 255.255.255.255. This IP broadcast is transmitted in a link-layer broadcast of FFFF.FFFF.FFFF. If the client is on the same LAN as the Bootp server, then the Bootp server will respond to the broadcast, by using the MAC address of the client, and the IP address will be given to the client. If the Bootp server is on another LAN i.e. on the other side of a router then we are left with the situation whereby broadcasts are being sent by the client and the router is designed not to pass broadcasts. Consider the following scenaro:



1. The client requires an IP address and a default gateway address so it sends out an IP broadcast BOOTREQUEST.
2. No Bootp server exists locally so the router R1, which is configured to forward Bootp/DHCP requests, sends a unicast with the source IP address of it's LAN interface on which it received the initial broadcast. Note that this is different from TCP connections which use the WAN IP address on the router which owns the LAN interface from whence the packet came. The unicast contains the IP address of the Bootp server which R1 knows since this is part of the Bootp forwarding configuration.
3. R2 forwards the unicast to the Bootp server.
4. The server examines the packet and checks for the client's hardware address in its database. If the MAC address has been entered then the server can map the client MAC address to the IP address configured in the database.
5. The server then looks to see if the client is requesting a configuration file (boot file).
6. The server replies with a unicast BOOTREPLY containing the client's IP address and details of the configuration file.
7. R1 recognises the unicast reply and forwards the packet to the clients MAC address.
8. The client's TCP/IP stack then pings the IP address to check that it is not being used before taking it for itself.
9. The client then can use TFTP etc. to download the configuration file including information such as the default gateway to the LAN interface of R1.

Note that the TCP/IP stack needs to be in place on the client, since the client needs to be able to send an IP broadcast (not a layer 2 broadcast!).

Below is **the frame format which is used for both Bootp and DHCP:**



- **Operation** - indicates a request **1** or a reply **2**.
- **Hardware Type** - Ethernet is **1**.
- **Hardware Length** - length of the address
- **Hops** - the client starts this off with **0** and then this increment by each Bootp server if the packet is passed on.
- **Transaction ID** - diskless nodes use this number to match responses with the requests.
- **Seconds** - this gives the elapsed time since the client started the boot process
- **Client IP Address** - if a client knows its IP address it puts it in here, otherwise it is a **0**.
- **Your IP Address** - the server puts an IP address here if the Client IP address field is **0**.
- **Server IP Address** - if the client knows this then the client can place the server address in here, otherwise the server does.
- **Server Host Name** - same as previously.

- **Gateway IP Address** - the client initially sets this to **0**. The router receiving the packet will put the IP address of the interface on which the packet was received (i.e. the LAN interface).
- **Client Hardware Address** - so that the servers/routers know where to send responses back to.
- **Server Host Name** - if the client puts a specific server name in here then it restricts itself to booting from that server alone. If no name is in here then the client can boot from any server that responds to a bootp request.
- **Boot File Name** - the client can request a specific boot filename.
- **Vendor Specific Area** - up to 64 octets long for Bootp. This is called the **Options** field in a DHCP packet and is up to 312 octets long. In DHCP this field contains optional information that the server may wish to give the client e.g. **client identifier** and **server identifier**.

DHCP (Dynamic Host Configuration Protocol)

DHCP works slightly differently from Bootp. The following steps occur when a client retrieves IP configuration details from a DHCP server:

- The client sends a **DHCPDISCOVER** IP broadcast to find available DHCP servers on its subnet. This broadcast message could give indications of the address required and the length of the lease.
- A router configured with Bootp Relay agent forwards the request on to other subnets specified in the router configuration (e.g. Cisco's **ip helper**).
- The DHCP server responds with a unicast **DHCPOFFER** containing an available IP address in the 'Your IP Address' field and other configuration parameters such as default gateway, subnet mask and lease time. This occurs when the DHCPDISCOVER has a **0** in the Client IP address field. The server should check that the IP address has not already been allocated. Optionally, the server could be configured to reserve previously allocated IP addresses. In the case of the server being on a different subnet to the client, this DHCPOFFER is sent to the client's router's LAN IP address which is the source of the original DHCPDISCOVER broadcast. Where the server and the client sit on the same LAN, the MAC address of the client is sufficient, being as it is the source of the broadcast.
- The client examines the various offers it receives, picks one and sends a **DHCPREQUEST** broadcast including the server identifier of that particular server to obtain the offered configuration parameters. The DHCPREQUEST broadcast is also used to renew a lease or to decline an offer.
- All servers receive this broadcast if they are on the LAN. If they are on another subnet and a relay agent is configured on the router, then this broadcast is unicast to the server identified by the server identifier. The router is configured with the IP address of the desired server.
- The chosen server replies with a unicast **DHCPACK** containing IP address, duration for the lease of that IP address and the other configuration parameters. The server commits to this IP address and it is passed to persistent memory. A **DHCPNACK** is sent if the IP address happens to have been allocated in the meantime.
- The client receives the DHCPACK and checks the IP address is not being used locally e.g. with an ARP request. If the IP address IS in use then the client sends a **DHCPDECLINE**, waits at least 10 seconds and then restarts the process.

The DHCP server may have a number of scopes configured for different subnets, it knows from which subnet a request comes from because the unicast sent from the local router contains the source IP address of the LAN interface on which the client's broadcast request was received.

What is DNS?

The Domain Name System (DNS) is the phonebook of the Internet. Humans access information online through domain names, like nytimes.com or espn.com. Web browsers interact through Internet Protocol (IP) addresses. DNS translates domain names to IP addresses so browsers can load Internet resources.

Each device connected to the Internet has a unique IP address which other machines use to find the device. DNS servers eliminate the need for humans to memorize IP addresses such as 192.168.1.1 (in IPv4), or more complex newer alphanumeric IP addresses such as 2400:cb00:2048:1::c629:d7a2 (in IPv6).

How does DNS work?

The process of DNS resolution involves converting a hostname (such as www.example.com) into a computer-friendly IP address (such as 192.168.1.1). An IP address is given to each device on the Internet, and that address is necessary to find the appropriate Internet device - like a street address is used to find a particular home. When a user wants to load a webpage, a translation must occur between what a user types into their web browser (example.com) and the machine-friendly address necessary to locate the example.com webpage.

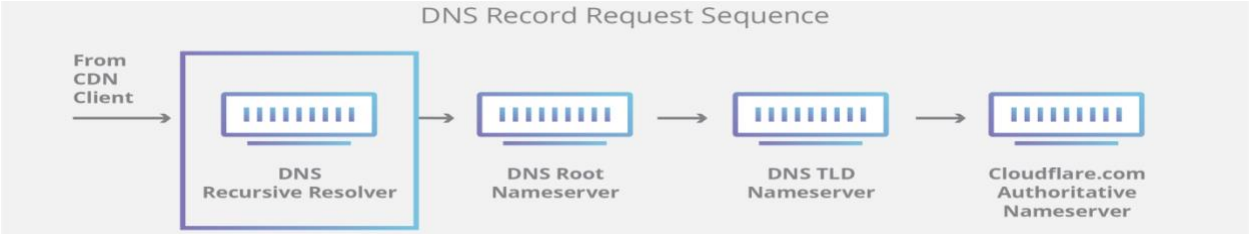
In order to understand the process behind the DNS resolution, it's important to learn about the different hardware components a DNS query must pass between. For the web browser, the DNS lookup occurs " behind the scenes" and requires no interaction from the user's computer apart from the initial request.

What's the difference between an authoritative DNS server and a recursive DNS resolver?

Both concepts refer to servers (groups of servers) that are integral to the DNS infrastructure, but each performs a different role and lives in different locations inside the pipeline of a DNS query. One way to think about the difference is the recursive resolver is at the beginning of the DNS query and the authoritative nameserver is at the end.

Recursive DNS resolver

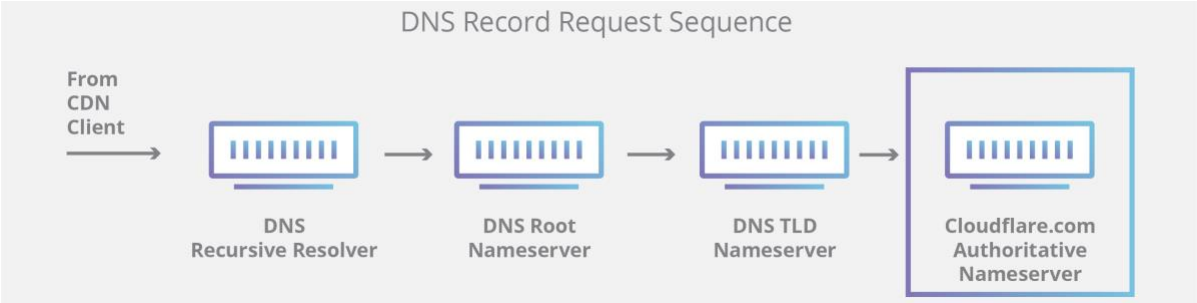
The recursive resolver is the computer that responds to a recursive request from a client and takes the time to track down the DNS record. It does this by making a series of requests until it reaches the authoritative DNS nameserver for the requested record (or times out or returns an error if no record is found). Luckily, recursive DNS resolvers do not always need to make multiple requests in order to track down the records needed to respond to a client; caching is a data persistence process that helps short-circuit the necessary requests by serving the requested resource record earlier in the DNS lookup.



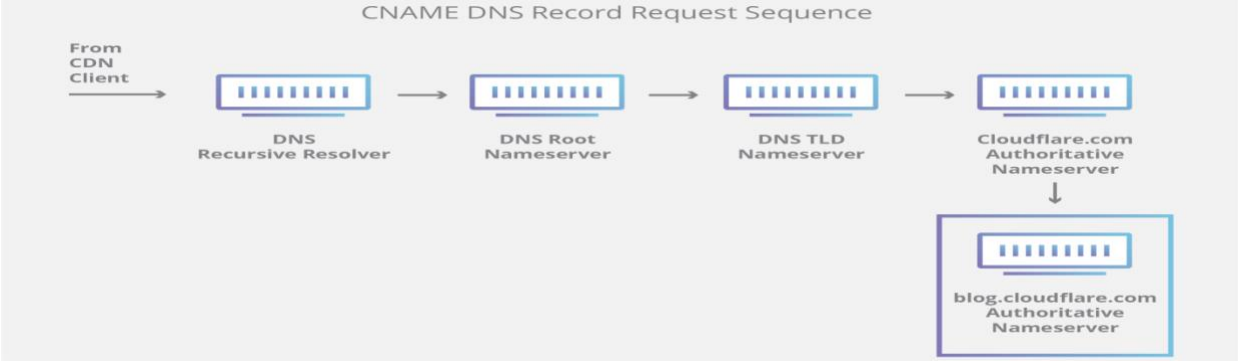
Authoritative DNS server

Put simply, an authoritative DNS server is a server that actually holds, and is responsible for, DNS resource records. This is the server at the bottom of the DNS lookup chain that will respond with the

queried resource record, ultimately allowing the web browser making the request to reach the IP address needed to access a website or other web resources. An authoritative nameserver can satisfy queries from its own data without needing to query another source, as it is the final source of truth for certain DNS records.



It's worth mentioning that in instances where the query is for a subdomain such as foo.example.com or blog.cloudflare.com, an additional nameserver will be added to the sequence after the authoritative nameserver, which is responsible for storing the subdomain's CNAME record.



There is a key difference between many DNS services and the one that Cloudflare provides. Different DNS recursive resolvers such as Google DNS, OpenDNS, and providers like Comcast all maintain data center installations of DNS recursive resolvers. These resolvers allow for quick and easy queries through optimized clusters of DNS-optimized computer systems, but they are fundamentally different than the nameservers hosted by Cloudflare.

Cloudflare maintains infrastructure-level nameservers that are integral to the functioning of the Internet. One key example is the f-root server network which Cloudflare is partially responsible for hosting. The F-root is one of the root level DNS nameserver infrastructure components responsible for the billions of Internet requests per day. Our Anycast network puts us in a unique position to handle large volumes of DNS traffic without service interruption

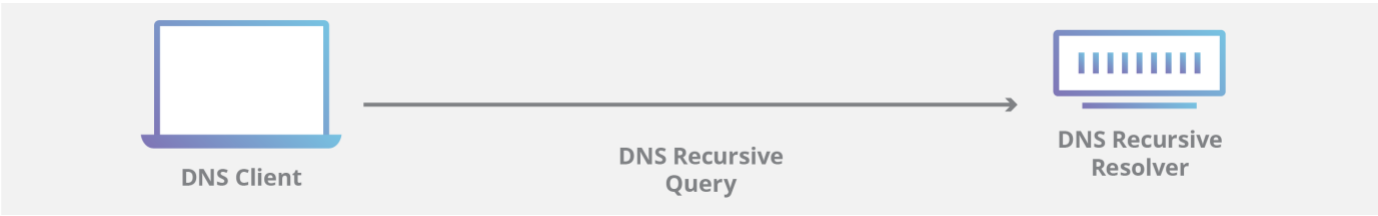
What is a DNS resolver?

The DNS resolver is the first stop in the DNS lookup, and it is responsible for dealing with the client that made the initial request. The resolver starts the sequence of queries that ultimately leads to a URL being translated into the necessary IP address.

Note: A typical uncached DNS lookup will involve both recursive and iterative queries.

It's important to differentiate between a recursive DNS query and a recursive DNS resolver. The query refers to the request made to a DNS resolver requiring the resolution of the query. A DNS recursive

resolver is the computer that accepts a recursive query and processes the response by making the necessary requests.



What are the types of DNS Queries?

In a typical DNS lookup three types of queries occur. By using a combination of these queries, an optimized process for DNS resolution can result in a reduction of distance traveled. In an ideal situation cached record data will be available, allowing a DNS name server to return a non-recursive query.

3 types of DNS queries:

1. **Recursive query** - In a recursive query, a DNS client requires that a DNS server (typically a DNS recursive resolver) will respond to the client with either the requested resource record or an error message if the resolver can't find the record.
2. **Iterative query** - in this situation the DNS client will allow a DNS server to return the best answer it can. If the queried DNS server does not have a match for the query name, it will return a referral to a DNS server authoritative for a lower level of the domain namespace. The DNS client will then make a query to the referral address. This process continues with additional DNS servers down the query chain until either an error or timeout occurs.
3. **Non-recursive query** - typically this will occur when a DNS resolver client queries a DNS server for a record that it has access to either because it's authoritative for the record or the record exists inside of its cache. Typically, a DNS server will cache DNS records to prevent additional bandwidth consumption and load on upstream servers.

Overview

When **DNS** was not into existence, one had to download a **Host file** containing host names and their corresponding IP address. But with increase in number of hosts of internet, the size of host file also increased. This resulted in increased traffic on downloading this file. To solve this problem the DNS system was introduced.

Domain Name System helps to resolve the host name to an address. It uses a hierarchical naming scheme and distributed database of IP addresses and associated names

IP Address

IP address is a unique logical address assigned to a machine over the network. An IP address exhibits the following properties:

- IP address is the unique address assigned to each host present on Internet.
- IP address is 32 bits (4 bytes) long.
- IP address consists of two components: **network component** and **host component**.
- Each of the 4 bytes is represented by a number from 0 to 255, separated with dots. For example 137.170.4.124

IP address is 32-bit number while on the other hand domain names are easy to remember names. For example, when we enter an email address we always enter a symbolic string such as webmaster@tutorialspoint.com.

Uniform Resource Locator (URL)

Uniform Resource Locator (URL) refers to a web address which uniquely identifies a document over the internet.

This document can be a web page, image, audio, video or anything else present on the web.

For example, **www.softpoint.com/internet_technology/index.html** is an URL to the index.html which is stored on tutorialspoint web server under internet_technology directory.

URL Types

There are two forms of URL as listed below:

- Absolute URL
- Relative URL

Absolute URL

Absolute URL is a complete address of a resource on the web. This completed address comprises of protocol used, server name, path name and file name.

For example http:// www.tutorialspoint.com / internet_technology /index.htm. where:

- **http** is the protocol.
- **softpoint.com** is the server name.
- **index.htm** is the file name.

The protocol part tells the web browser how to handle the file. Similarly we have some other protocols also that can be used to create URL are:

- FTP
- https
- Gopher
- mailto
- news

Relative URL

Relative URL is a partial address of a webpage. Unlike absolute URL, the protocol and server part are omitted from relative URL.

Relative URLs are used for internal links i.e. to create links to file that are part of same website as the WebPages on which you are placing the link.

For example, to link an image on softpoint.com/internet_technology/internet_referemce_models, we can use the relative URL which can take the form like **/internet_technologies/internet-osi_model.jpg.**

Difference between Absolute and Relative URL

Absolute URL	Relative URL
Used to link web pages on different websites	Used to link web pages within the same website.
Difficult to manage.	Easy to Manage
Changes when the server name or directory name changes	Remains same even of we change the server name or directory name.
Take time to access	Comparatively faster to access.

Domain Name System Architecture

The Domain name system comprises of **Domain Names, Domain Name Space, Name Server** that have been described below:

Domain Names

Domain Name is a symbolic string associated with an IP address. There are several domain names available; some of them are generic such as **com, edu, gov, net** etc, while some country level domain names such as **au, in, za, us** etc.

The following table shows the **Generic** Top-Level Domain names:

Domain Name	Meaning
Com	Commercial business
Edu	Education
Gov	U.S. government agency
Int	International entity
Mil	U.S. military
Net	Networking organization
Org	Non profit organization

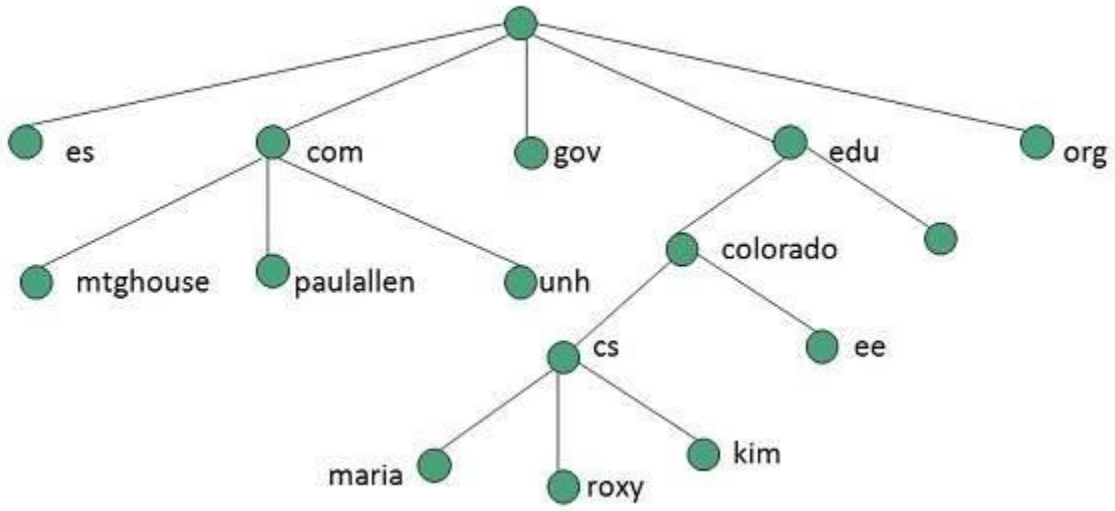
The following table shows the **Country top-level** domain names:

Domain Name	Meaning
au	Australia

in	India
cl	Chile
fr	France
us	United States
za	South Africa
uk	United Kingdom
jp	Japan
es	Spain
de	Germany
ca	Canada
ee	Estonia
hk	Hong Kong

Domain Name Space

The domain name space refers a hierarchy in the internet naming structure. This hierarchy has multiple levels (from 0 to 127), with a root at the top. The following diagram shows the domain name space hierarchy:



In the above diagram each subtree represents a domain. Each domain can be partitioned into sub domains and these can be further partitioned and so on.

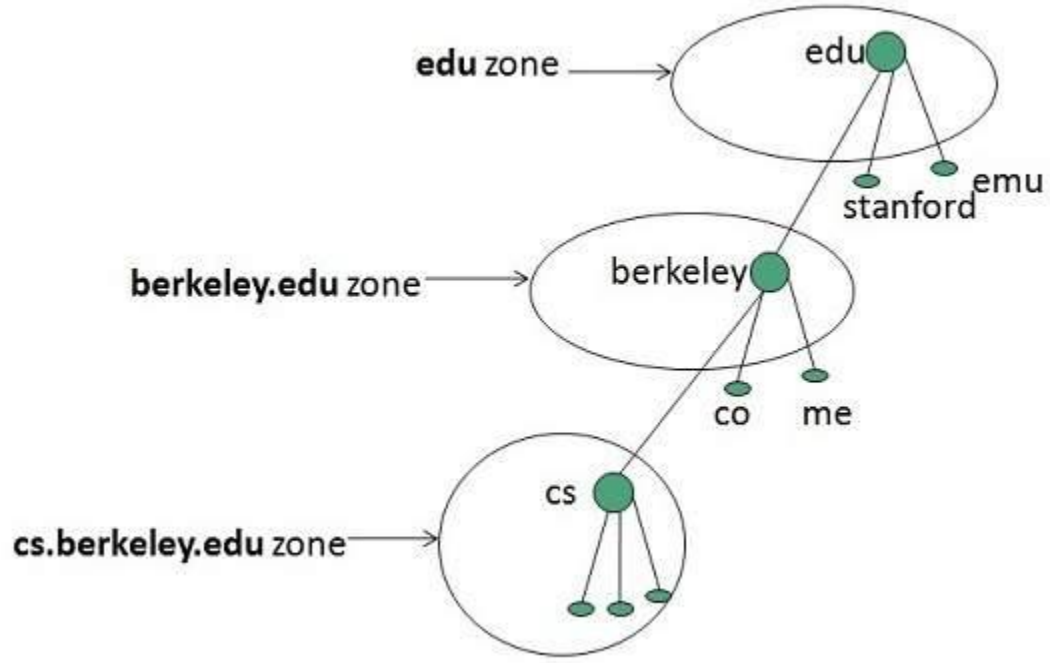
Name Server

Name server contains the DNS database. This database comprises of various names and their corresponding IP addresses. Since it is not possible for a single server to maintain entire DNS database, therefore, the information is distributed among many DNS servers.

- Hierarchy of server is same as hierarchy of names.
- The entire name space is divided into the zones

Zones

Zone is collection of nodes (sub domains) under the main domain. The server maintains a database called zone file for every zone.



If the domain is not further divided into sub domains then domain and zone refers to the same thing.

The information about the nodes in the sub domain is stored in the servers at the lower levels however; the original server keeps reference to these lower levels of servers.

Types of Name Servers

Following are the three categories of Name Servers that manages the entire Domain Name System:

- Root Server
- Primary Server
- Secondary Server

Root Server

Root Server is the top level server which consists of the entire DNS tree. It does not contain the information about domains but delegates the authority to the other server

Primary Servers

Primary Server stores a file about its zone. It has authority to create, maintain, and update the zone file.

Secondary Server

Secondary Server transfers complete information about a zone from another server which may be primary or secondary server. The secondary server does not have authority to create or update a zone file.

DNS Working

DNS translates the domain name into IP address automatically. Following steps will take you through the steps included in domain resolution process:

- When we type **www.softpoint.com** into the browser, it asks the local DNS Server for its IP address.

  Here the local DNS is at ISP end.

- When the local DNS does not find the IP address of requested domain name, it forwards the request to the root DNS server and again enquires about IP address of it.
- The root DNS server replies with delegation that **I do not know the IP address of www.softpoint.com but know the IP address of DNS Server.**
- The local DNS server then asks the com DNS Server the same question.
- The **com** DNS Server replies the same that it does not know the IP address of www.softpont.com but knows the address of tutorialspoint.com.
- Then the local DNS asks the tutorialspoint.com DNS server the same question.
- Then tutorialspoint.com DNS server replies with IP address of www.softpoint.com.
- Now, the local DNS sends the IP address of www.tutorialspoint.com to the computer that sends the request.

TRANSITION STATES:-

To provide dynamic address allocation, the DHCP client acts as a state machine that Performs transitions from one state to another depending on the messages it receives or sends. The type of the message in this case is defined by the option with tag 53 that is included in the DHCP packet. In other words, instead of adding one extra field to the BOOTP protocol to define DHCP type, the designer decided to add an extra option for this purpose. Figure 18.7 shows the type option and the interpretation of its value to define the type of the DHCP packet.
Figure 18.8 shows the transition diagram with main states. The RFC and some implementations offer some more states that we leave the investigation as exercises.

Figure 18.7 *Option with tag 53*

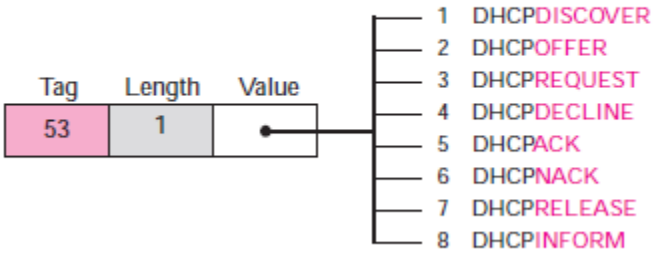
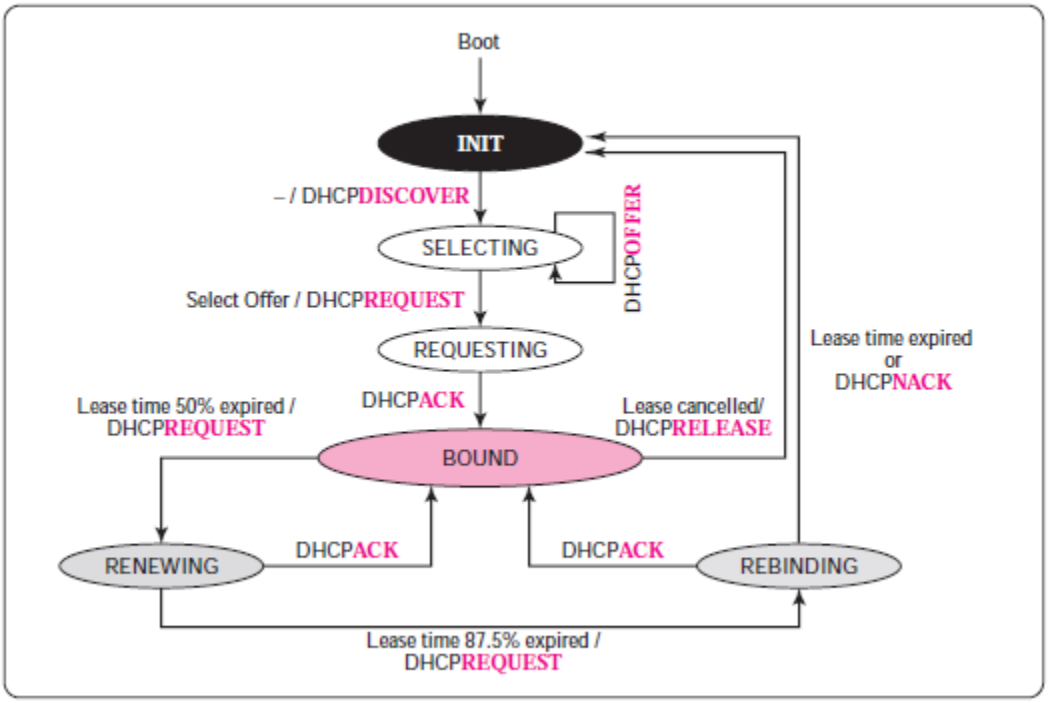


Figure 18.8 *DHCP client transition diagram*



- ***INIT State***
  When the DHCP client first starts, it is in the INIT state (initializing state). The client broadcasts a DHCPDISCOVER message (a request message with the DHCPDISCOVER option), using port 67.
- ***SELECTING State***
  After sending the DHCPDISCOVER message, the client goes to the selecting state. Those servers that can provide this type of service respond with a DHCPOFFER message. In these messages, the servers offer an IP address. They can also offer the lease duration. The default is 1 hour. The server that sends a DHCPOFFER locks the offered IP address so that it is not available to any other clients. The client chooses one of the requesting state. However, if the client receives no DHCPOFFER message, it tries four more times, each with a span of 2 seconds. If there is no reply to any of these DHCPDISCOVERs, the client sleeps for 5 minutes before trying again.
- ***REQUESTING State***
  The client remains in the requesting state until it receives a DHCPACK message from the server that creates the binding between the client physical address and its IP address. After receipt of the DHCPACK, the client goes to the bound state.
- ***BOUND State***
  In this state, the client can use the IP address until the lease expires. When 50 percent of the lease period is reached, the client sends another DHCPREQUEST to ask for renewal. It then goes to the renewing state. When in the bound state, the client can also cancel the lease and go to the initializing state.
- ***RENEWING State***
  The client remains in the renewing state until one of two events happens. It can receive a DHCPACK, which renews the lease agreement. In this case, the client resets its timer and goes back to the bound state. Or, if a DHCPACK is not received, and 87.5 percent of the lease time expires, the client goes to the rebinding state.
- ***REBINDING State***

The client remains in the rebinding state until one of three events happens. If the client receives a DHCPNACK or the lease expires, it goes back to the initializing state and tries to get another IP address. If the client receives a DHCPACK, it goes to the bound state and resets the timer.

DNS IN THE INTERNET:-

DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) is divided into three different sections: generic domains, country domains, and the inverse domain (see Figure 19.8).

Generic Domains

The generic domains define registered hosts according to their generic behavior. Each node in the tree defines a domain, which is an index to the domain name space database (see Figure 19.9)

Figure 19.8 *DNS used in the Internet*

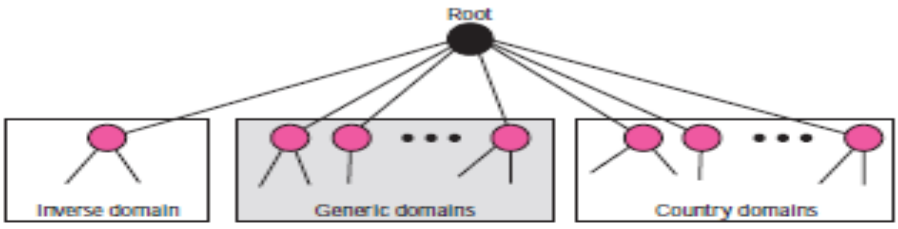
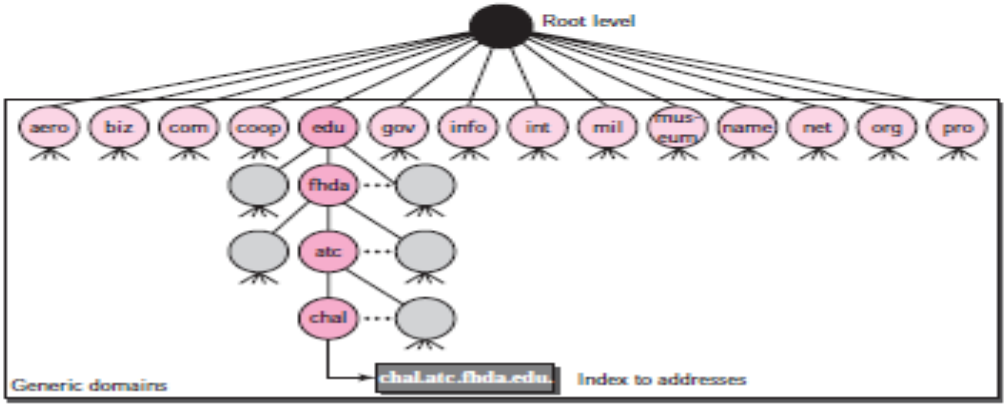


Figure 19.9 *Generic domains*



Looking at the tree, we see that the first level in the generic domains section allows 14 possible labels. These labels describe the organization types as listed in Table 19.1.

Country Domains

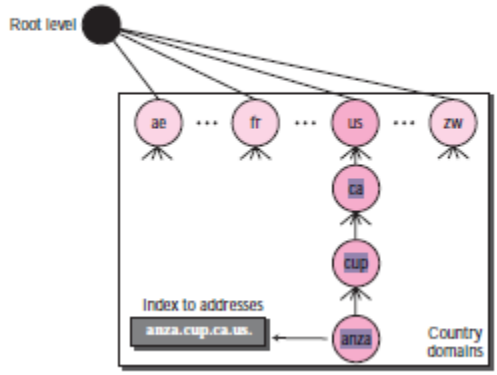
The country domains section uses two-character country abbreviations (e.g., us for United States). Second labels can be organizational, or they can be more specific, national designations. The United States, for example, uses state abbreviations as a subdivision of us (e.g., ca.us.).

Figure 19.10 shows the country domains section. The address anza.cup.ca.us can be translated to De Anza College in Cupertino in California in the United States.

Table 19.1 *Generic domain labels*

Label	*Description*
aero	Airlines and aerospace companies
biz	Businesses or firms (similar to "com")
com	Commercial organizations
coop	Cooperative business organizations
edu	Educational institutions
gov	Government institutions
info	Information service providers
int	International organizations
mil	Military groups
museum	Museums and other non-profit organizations
name	Personal names (individuals)
net	Network support centers
org	Nonprofit organizations
pro	Professional individual organizations

Figure 19.10 *Country domains*



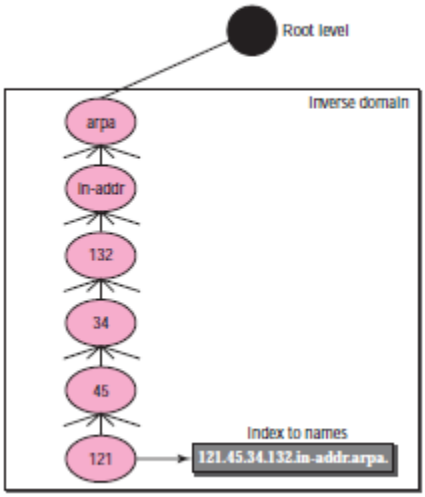
Inverse Domain

The inverse domain is used to map an address to a name. This may happen, for example, when a server has received a request from a client to do a task. Although the server has a file that contains a list of authorized clients, only the IP address of the client (extracted from the received IP packet) is listed. The server asks its resolver to send a query to the DNS server to map an address to a name to determine if the client is on the authorized list.

This type of query is called an inverse or pointer (PTR) query. To handle a pointer query, the inverse domain is added to the domain name space with the first-level node called arpa (for historical reasons). The second level is also one single node named in-addr (for inverse address). The rest of the domain defines IP addresses.

The servers that handle the inverse domain are also hierarchical. This means the netid part of the address should be at a higher level than the subnetid part, and the subnetid part higher than the hostid part. In this way, a server serving the whole site is at a higher level than the servers serving each subnet. This configuration makes the domain look inverted when compared to a generic or country domain. To follow the convention of reading the domain labels from the bottom to the top, an IP address such as 132.34.45.121 (a class B address with netid 132.34) is read as 121.45.34.132.in-addr. arpa. See Figure 19.11 for an illustration of the inverse domain configuration.

Figure 19.11 *Inverse domain*



RESOLUTION:-

Mapping a name to an address or an address to a name is called name-address resolution.

Resolver

DNS is designed as a client-server application. A host that needs to map an address to a name or a name to an address calls a DNS client called a resolver. The resolver accesses the closest DNS server with a mapping request. If the server has the information, it satisfies the resolver; otherwise, it either refers the resolver to other servers or asks other servers to provide the information.

After the resolver receives the mapping, it interprets the response to see if it is a real resolution or an error, and finally delivers the result to the process that requested it.

Mapping Names to Addresses

Most of the time, the resolver gives a domain name to the server and asks for the corresponding address. In this case, the server checks the generic domains or the country domains to find the mapping.

If the domain name is from the generic domains section, the resolver receives a domain name such as "chal.atc.fhda.edu.". The query is sent by the resolver to the local DNS server for resolution. If the local server cannot resolve the query, it either refers the resolver to other servers or asks other servers directly.

If the domain name is from the country domains section, the resolver receives a domain name such as "ch.fhda.cu.ca.us.". The procedure is the same.

Mapping Addresses to Names

A client can send an IP address to a server to be mapped to a domain name. As mentioned before, this is called a PTR query. To answer queries of this kind, DNS uses the inverse domain. However, in the request, the IP address is reversed and two labels, in-addr and arpa, are appended to create a domain acceptable by the inverse domain section. For example, if the resolver receives the IP address 132.34.45.121, the resolver first inverts the address and then adds the two labels before sending. The domain name sent is "121.45.34.132.in-addr.arpa.", which is received by the local DNS and resolved.

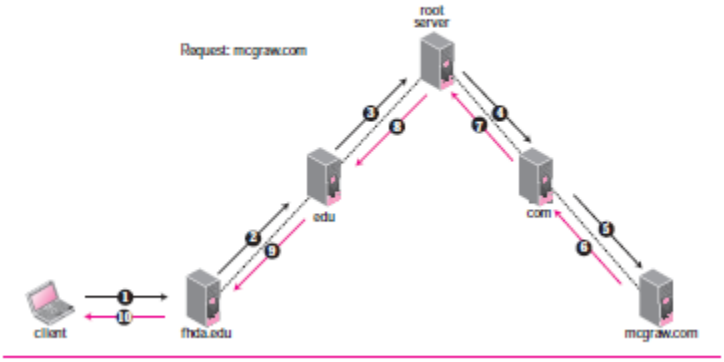
Recursive Resolution

Figure 19.12 shows the recursive resolution.

The client (resolver) can ask for a recursive answer from a name server. This means that the resolver expects the server to supply the final answer. If the server is the authority for the domain name, it checks its database and responds. If the server is not

the authority, it sends the request to another server (the parent usually) and waits for the response. If the parent is the authority, it responds; otherwise, it sends the query to yet another server. When the query is finally resolved, the response travels back until it finally reaches the requesting client.

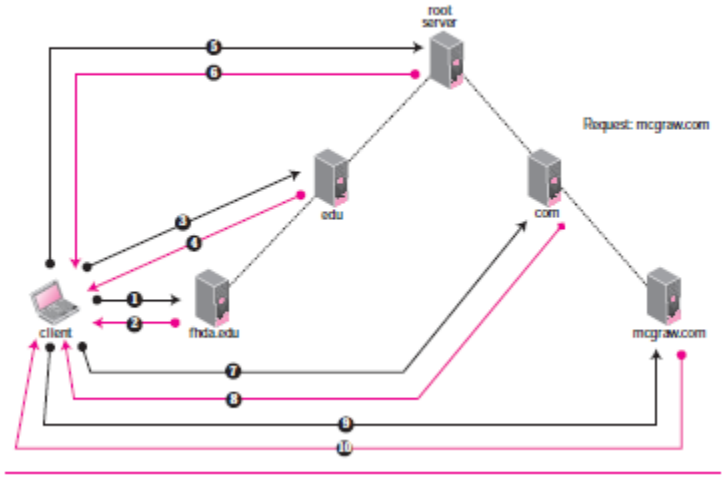
Figure 19.12 *Recursive resolution*



Iterative Resolution

If the client does not ask for a recursive answer, the mapping can be done iteratively. If the server is an authority for the name, it sends the answer. If it is not, it returns (to the client) the IP address of the server that it thinks can resolve the query. The client is responsible for repeating the query to this second server. If the newly addressed server can resolve the problem, it answers the query with the IP address; otherwise, it returns the IP address of a new server to the client. Now the client must repeat the query to the third server. This process is called iterative because the client repeats the same query to multiple servers. In Figure 19.13 the client queries five servers before it gets an answer from the mcgraw.com server.

Figure 19.13 *Iterative resolution*

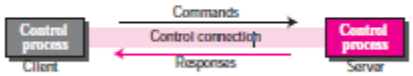


that each server keep a TTL counter for each mapping it caches. The cache memory must be searched periodically and those mappings with an expired TTL must be purged

Command Processing

FTP uses the control connection to establish a communication between the client control process and the server control process. During this communication, the commands are sent from the client to the server and the responses are sent from the server to the client (see Figure 21.6)

Figure 21.6 *Command processing*



Commands

Commands, which are sent from the FTP client control process, are in the form of ASCII uppercase, which may or may not be followed by an argument. We can roughly divide the commands into six groups: access commands, file management commands, data formatting commands, port defining commands, file transferring commands, and miscellaneous commands.

Access commands. These commands let the user access the remote system. Table 21.1 lists common commands in this group.

Table 21.1 *Access commands*

Command	*Argument(s)*	*Description*
USER	User id	User information
PASS	User password	Password
ACCT	Account to be charged	Account information
REIN		Reinitialize
QUIT		Log out of the system
ABOR		Abort the previous command

File management commands. These commands let the user access the file system on the remote computer. They allow the user to navigate through the directory structure, create new directories, delete files, and so on. Table 21.2 gives common commands in this group.

Table 21.2 *File management commands*

Command	*Argument(s)*	*Description*
CWD	Directory name	Change to another directory
CDUP		Change to parent directory
DELE	File name	Delete a file
LIST	Directory name	List subdirectories or files
NLIST	Directory name	List subdirectories or files without attributes
MKD	Directory name	Create a new directory
PWD		Display name of current directory
RMD	Directory name	Delete a directory
RNFR	File name (old)	Identify a file to be renamed
RNTO	File name (new)	Rename the file
SMNT	File system name	Mount a file system

Data formatting commands. These commands let the user define the data structure, file type, and transmission mode. The defined format is then used by the file transfer commands. Table 21.3 shows common commands in this group.

Table 21.3 *Data formatting commands*

Command	*Argument(s)*	*Description*
TYPE	**A** (ASCII), **E** (EBCDIC), **I** (Image), **N** (Nonprint), or **T** (TELNET)	Define file type
STRU	**F** (File), **R** (Record), or **P** (Page)	Define organization of data
MODE	**S** (Stream), **B** (Block), or **C** (Compressed)	Define transmission mode

Port defining commands. These commands define the port number for the data connection on the client site. There are two methods to do this. In the first method, using the PORT command, the client can choose an ephemeral port number and send it to the server using a passive open. The server uses that port number and creates an active open. In the second method, using the PASV command, the client just asks the server to first choose a port number. The server does a passive open on that port and sends the port number in the response (see response numbered 227 in Table 21.7). The client issues an active open using that port number. Table 21.4 shows the port defining commands.

Table 21.4 *Port defining commands*

Command	*Argument(s)*	*Description*
PORT	6-digit identifier	Client chooses a port
PASV		Server chooses a port

File transfer commands. These commands actually let the user transfer files. Table 21.5 lists common commands in this group.

Table 21.5 *File transfer commands*

Command	*Argument(s)*	*Description*
RETR	File name(s)	Retrieve files; file(s) are transferred from server to client
STOR	File name(s)	Store files; file(s) are transferred from client to server
APPE	File name(s)	Similar to STOR, but if file exists, data must be appended to it
STOU	File name(s)	Same as STOR, but file name will be unique in the directory
ALLO	File name(s)	Allocate storage space for files at the server
REST	File name(s)	Position file marker at a specified data point
STAT	File name(s)	Return status of files

Miscellaneous commands. These commands deliver information to the FTP user at the client site. Table 21.6 shows common commands in this group.

Table 21.6 *Miscellaneous commands*

Command	*Argument(s)*	*Description*
HELP		Ask information about the server
NOOP		Check if server is alive
SITE	Commands	Specify the site-specific commands
SYST		Ask about operating system used by the server

ERROR CONTROL:-

The TFTP error-control mechanism is different from those of other protocols. It is symmetric, which means that the sender and the receiver both use time-outs. The sender uses a time-out for data messages; the receiver uses a time-out for acknowledgment messages. If a data message is lost, the sender retransmits it after time-out expiration. If an acknowledgment is lost, the receiver retransmits it after time-out expiration. This guarantees a smooth operation.

Error control is needed in four situations: a damaged message, a lost message, a lost acknowledgment, or a duplicated message.

Damaged Message There is no negative acknowledgment. If a block of data is damaged, it is detected by the receiver and the block is discarded. The sender waits for the acknowledgment and does not receive it within the time-out period. The block is then sent again. Note that there is no checksum field in the DATA message of TFTP. The only way the receiver can detect data corruption is through the checksum field of the UDP user datagram.

Lost Message If a block is lost, it never reaches the receiver and no acknowledgment is sent. The sender resends the block after the time-out.

Lost Acknowledgment If an acknowledgment is lost, we can have two situations. If the timer of the receiver matures before the timer of the sender, the receiver retransmits the acknowledgment; otherwise, the sender retransmits the data.

Duplicate Message Duplication of blocks can be detected by the receiver through block number. If a block is duplicated, it is simply discarded by the receiver.