DNS Resolution:

If ROOT and
TLD name servers support Recursion

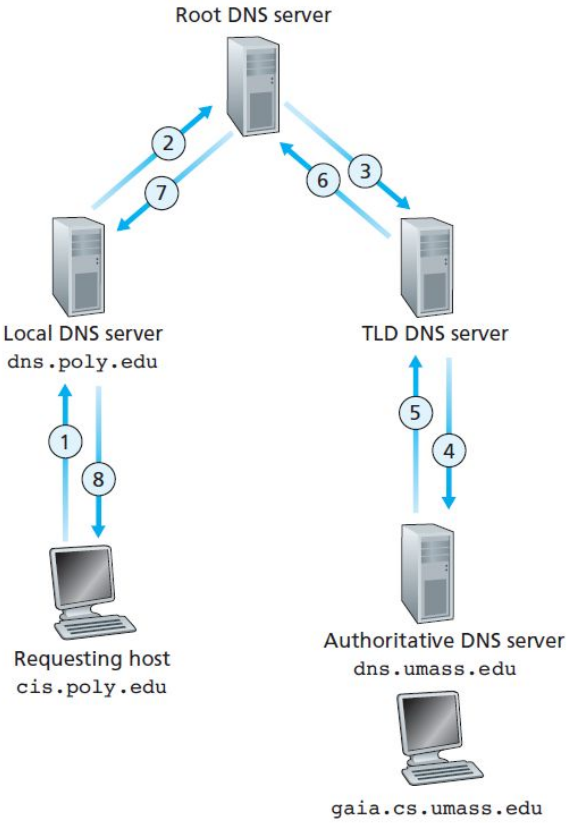
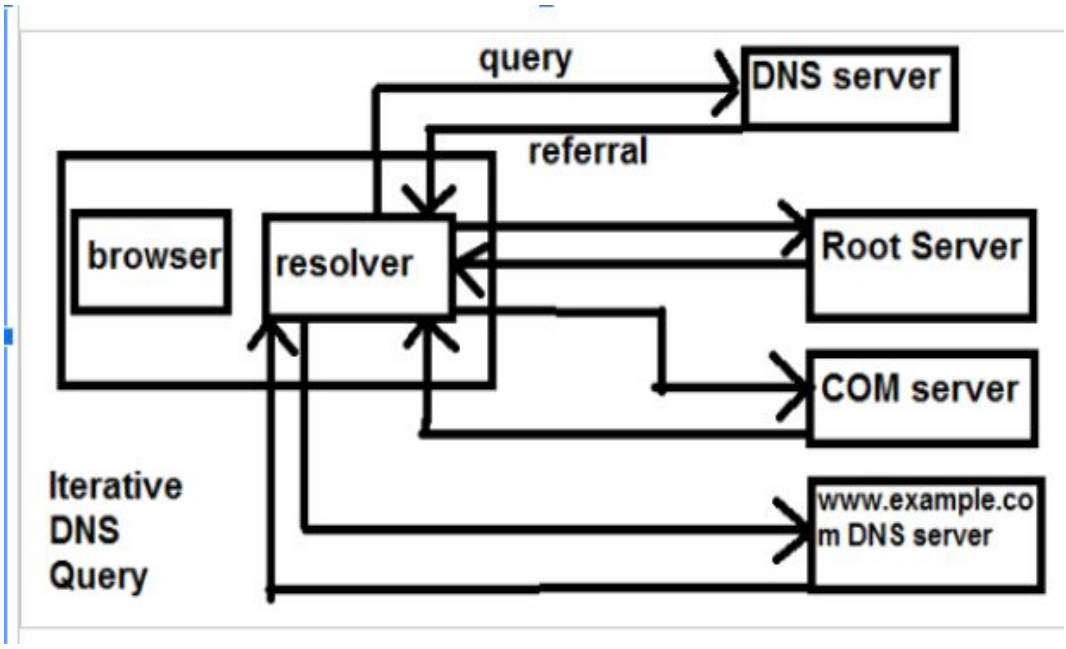


Figure 2.22 ♦ Recursive queries in DNS

DNS: Iterative Resolution

- STEP 1: You enter www.example.com in the browser. So the operating system's resolver will send a DNS query for the A record to the DNS server 172.16.200.30.
- The DNS server 172.16.200.30 on receiving the query, will look through its tables(cache) to find the IP address(A record) for the domain www.example.com. But it does not have the entry.
- STEP 3: Now instead of querying the root server's, our DNS server will reply us back with a referral to root servers. Now our operating system resolver, will query the root servers for the answer and so on. All the Name Servers will return a referral without accepting to perform recursion on behalf of the resolver.

DNS: Iterative Resolution



Dr. Gaurav Varshney IIT Jammu

DNS Message Format

A 1-bit query/reply flag indicates whether the message is a query (0) or a reply (1).

A1-bit authoritative flag is set in a reply message when a DNS server is an authoritative server for a queried name.

A 1-bit recursion-desired flag is set when a client (host or DNS server) desires that the DNS server perform recursion when it doesn't have the record.

A 1-bit recursion available field is set in a reply if the DNS server supports recursion.

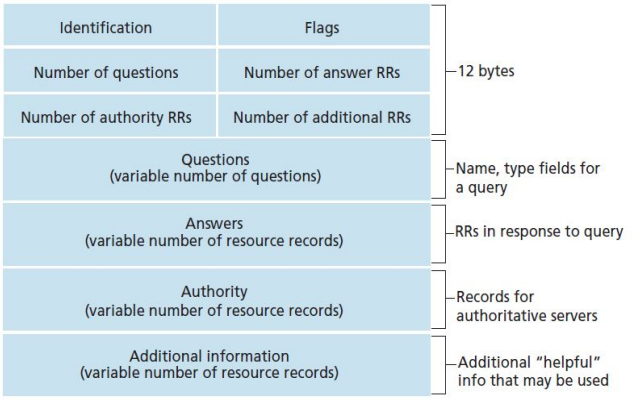


Figure 2.23 ♦ DNS message format

```
Domain Name System (response)
   Transaction ID: 0x6127
 ⌄ Flags: 0x8180 Standard query response, No error
      1... .... .... .... = Response: Message is a response
      .000 0... .... .... = Opcode: Standard query (0)
      .... .0.. .... .... = Authoritative: Server is not an authority for domain
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... 1... .... = Recursion available: Server can do recursive queries
      .... .... .0.. .... = Z: reserved (0)
      .... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by
      .... .... ...0 .... = Non-authenticated data: Unacceptable
      .... .... .... 0000 = Reply code: No error (0)
   Questions: 1
```

Dr. Gaurav Varshney IIT Jammu

DNS Message Format

```
Answer RRs: 0
Authority RRs: 0
Additional RRs: 0
Queries
  chenab.iitjammu.ac.in: type A, class IN
    Name: chenab.iitjammu.ac.in
    [Name Length: 21]
```

```
Authority RRs: 0
Additional RRs: 0
Queries
  chenab.iitjammu.ac.in: type A, class IN
Answers
  chenab.iitjammu.ac.in: type A, class IN, addr 10.10.28.5
```


Dr. Gaurav Varshney IIT Jammu

DNS: Records

A resource record is a four-tuple that contains the following fields:

(Name, Value, Type, TTL)

- A Record: An A record points a domain or a sub-domain to an IP address
- CNAME Record: A CNAME (Canonical Name) points one domain or sub domain to another domain name, allowing you to update one A Record each time you make a change, regardless of how many Host Records need to resolve to that IP address.
- MX Record: An MX Entry (Mail Exchanger) directs email to a particular mail server. Like a CNAME, MX entries must point to a domain and never point directly to an IP address.
- TXT Records: A TXT (Text) record was originally intended for human readable text. These records are dynamic and can be used for several purposes such as storing email security policies (SPF, DKIM, DMARC) and TLS security policies (CAA, DANE etc.)
- SOA Record: This records points to the start of authority information for the domain including policies for zone transfers. This includes primary and first name server in the list and the admin information responsible for this domain with other zone related information.
- NS Record: This record mentions the authoritative name servers responsible for this domain name. These name servers will be holding DNS resolution information for the resources hosted under this domain name.

Dr. Gaurav Varshney IIT Jammu

Dr. Gaurav Varshney IIT Jammu

DNS: Records

- **A Record**
  - An A record (Address Record) points a domain or subdomain to an IP address.
- **CNAME**
  - A CNAME (Canonical Name) points one domain or subdomain to another domain name, allowing you to update one A Record each time you make a change, regardless of how many Host Records need to resolve to that IP address.

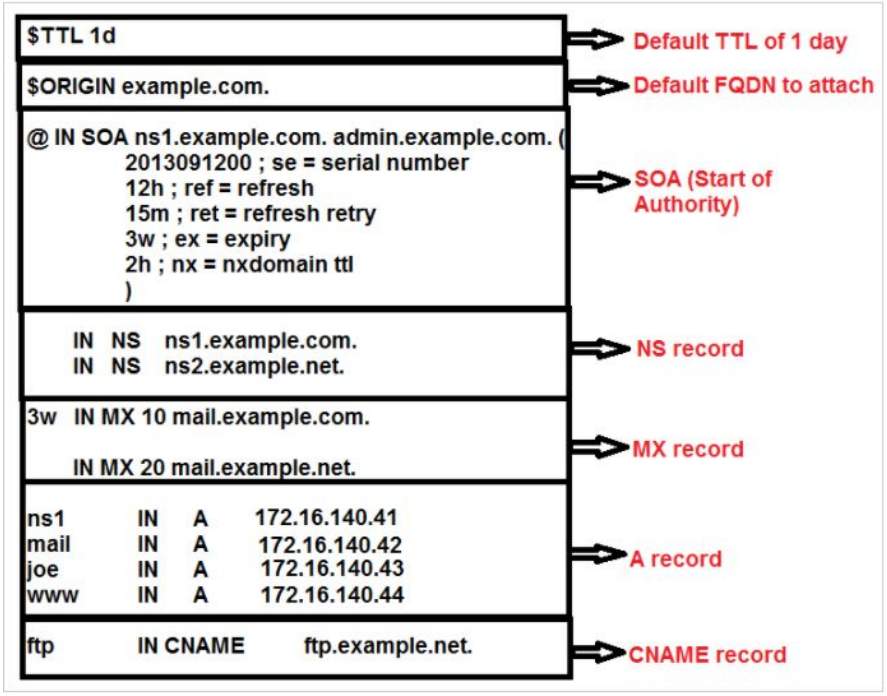
```
NAME                    TYPE    VALUE
-----------------------------------------------------
bar.example.com.        CNAME   foo.example.com.
foo.example.com.        A       192.0.2.23
```

- **MX Entry**
  - An MX Entry (Mail Exchanger) directs email to a particular mail server. Like a CNAME, MX Entries must point to a domain and never point directly to an IP address.

```
norbert.dept1.cornell.edu    86400   A     128.253.180.254
norbert.dept1.cornell.edu    86400   MX  10 mailhost.dept1.cornell.edu
norbert.dept1.cornell.edu    86400   MX  20 mailhost2.dept1.cornell.edu
norbert.dept1.cornell.edu    86400   MX  30 mailhost3.dept1.cornell.edu
```


Dr. Gaurav Varshney IIT Jammu

DNS Zone File



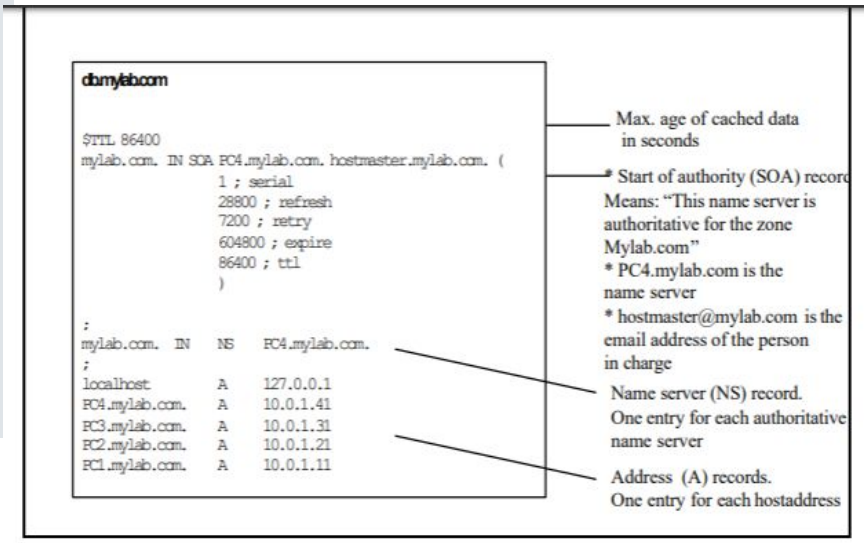
Dr. Gaurav Varshney IIT Jammu

https://www.slashroot.in/what-dns-zone-file-complete-tutorial-zone-file-and-its-contents

Dr. Gaurav Varshney IIT Jammu

DNS Zone File

```
$TTL 86400      ; 1 day
    example.com IN SOA dns.example.com. root@example.com. (
                        57         ; serial
                        28800      ; refresh (8 hours)
                        7200       ; retry (2 hours)
                        2419200    ; expire (4 weeks)
                        86400      ; minimum (1 day)
                        )
                IN NS dns.example.com.
    dns           IN         A        192.0.2.1
        example.com   IN         A        192.0.2.1
        host01        IN         A        192.0.2.101
        host02        IN         A        192.0.2.102
        host03        IN         A        192.0.2.103
```

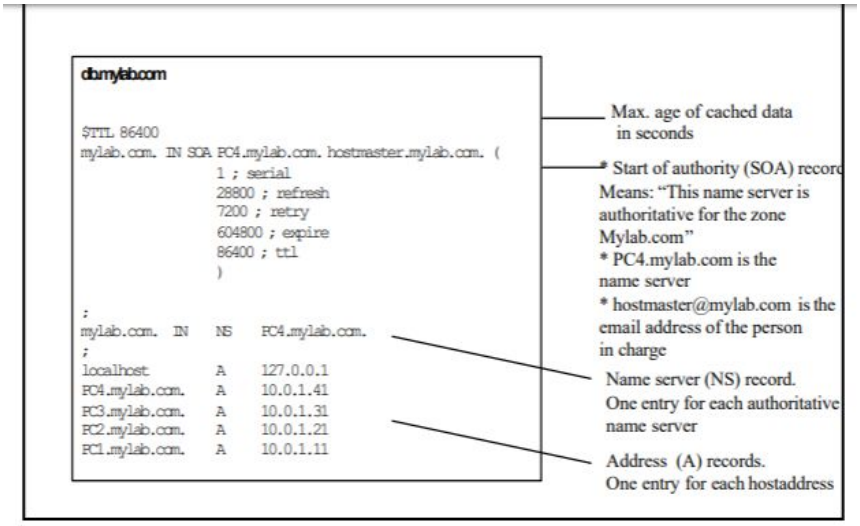


https://www.slashroot.in/what-dns-zone-file-complete-tutorial-zone-file-and-its-contents

Dr. Gaurav Varshney IIT Jammu

DNS Zone File

https://www.slashroot.in/what-dns-zone-file-complete-tutorial-zone-file-and-its-contents



Dr. Gaurav Varshney IIT Jammu

- REFRESH: The length of time (in seconds) secondary servers should wait before asking primary servers for the SOA record to see if it has been updated.
- RETRY: The length of time a server should wait for asking an unresponsive primary nameserver for an update again.
- EXPIRE: If a secondary server does not get a response from the primary server for this amount of time, it should stop responding to queries for the zone.

http://www.tcpipguide.com/free/t_DNSReverseNameResolutionUsingtheINADDRARPADomain-2.htm

https://www.cloudflare.com/learning/dns/dns-records/dns-soa-record/

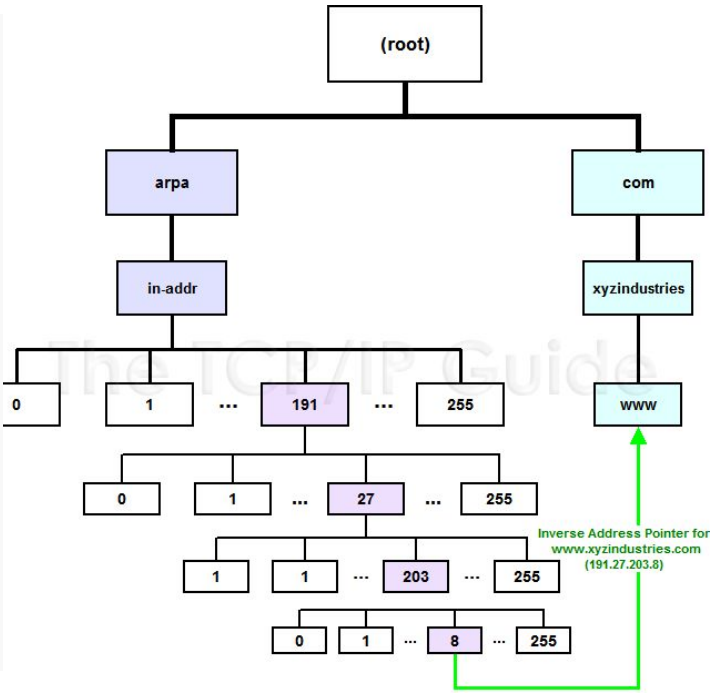
RDNS

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/4/html/reference_guide/s2-bind-configuration-zone-reverse
http://www.tcpipguide.com/free/t_DNSReverseNameResolutionUsingtheINADDRARPADomain-2.htm

```
$ORIGIN 1.0.10.in-addr.arpa.
$TTL 86400
@     IN     SOA    dns1.example.com.     hostmaster.example.com. (
                    2001062501 ; serial
                    21600      ; refresh after 6 hours
                    3600       ; retry after 1 hour
                    604800     ; expire after 1 week
                    86400 )    ; minimum TTL of 1 day

      IN     NS     dns1.example.com.
      IN     NS     dns2.example.com.

20    IN     PTR    alice.example.com.
21    IN     PTR    betty.example.com.
22    IN     PTR    charlie.example.com.
23    IN     PTR    doug.example.com.
24    IN     PTR    ernest.example.com.
25    IN     PTR    fanny.example.com.
```



Jon Postel proposed a Mail Transfer Protocol in 1980

Electronic mail service: Email

As with ordinary postal mail, e-mail is an asynchronous communication medium—people send and read messages when it is convenient for them, without having to coordinate with other people's schedules.

SMTP is the principal application-layer protocol for Internet electronic mail. It uses the reliable data transfer service of TCP to transfer mail from the sender's mail server to the recipient's mail server.

As with most application-layer protocols, SMTP has two sides: a client side, which executes on the sender's mail server, and a server side, which executes on the recipient's mail server. Both the client and server sides of SMTP run on every mail server.

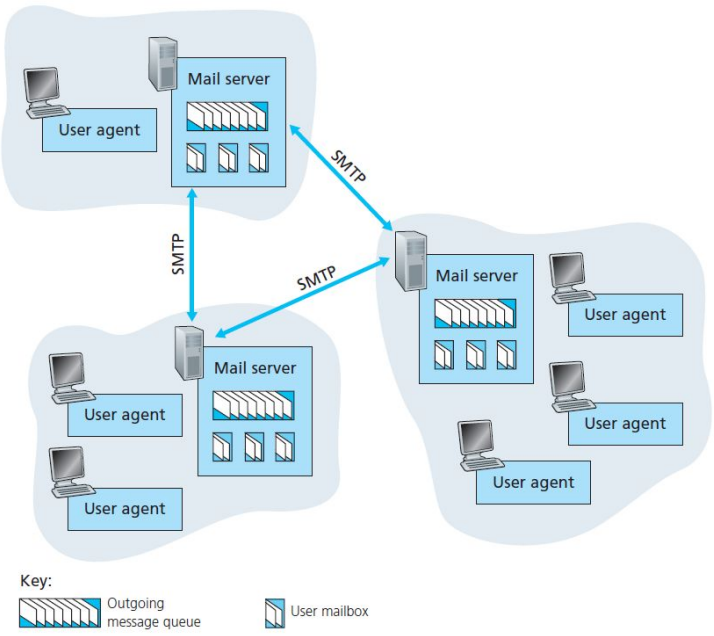


Figure 2.16 ♦ A high-level view of the Internet e-mail system

Dr. Gaurav Varshney IIT Jammu

SMTP: RFC 5321

SMTP is much older than HTTP. (The original SMTP RFC
dates back to 1982,

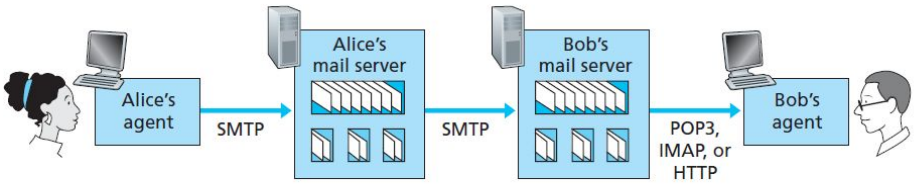


Figure 2.18 ♦ E-mail protocols and their communicating entities

```
telnet serverName
25/465/587
```

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr ... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

The first Web Mail implementation was developed at CERN in 1993 by Phillip Hallam-Baker

Dr. Gaurav Varshney IIT Jammu

Mail Access Protocols [POP3 : Post Office Protocol]

POP3 is an extremely simple mail access protocol. It is defined in [RFC 1939], which is short and quite readable.

```
telnet mailServer 110
+OK POP3 server ready
user bob
+OK
pass hungry
+OK user successfully logged on
```

POP3 begins when the user agent (the client) opens a TCP connection to the mail server (the server) on port 110.

POP3 progresses through three phases: authorization, transaction, and update. During the first phase, authorization, the user agent sends a username and a password (in the clear) to authenticate the user. During the second phase, transaction, the user agent retrieves messages; also during this phase, the user agent can mark messages for deletion, remove deletion marks, and obtain mail statistics.

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: (blah blah ...
S: .................
S: ..........blah)
S: .
C: dele 1
C: retr 2
S: (blah blah ...
S: .................
S: ..........blah)
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

Dr. Gaurav Varshney IIT Jammu

Mail Access Protocols [POP3 : Post Office Protocol]

```
STAT
+OK 3 5467
```

```
LIST
+OK Mailbox scan listing follows
1 1823
2 1825
3 1819
.
```

```
RETR 1
+OK 1823 octets
--- all message headers and message ---
.
```

```
DELE 1
+OK Message deleted
```

```
RSET
+OK Reset state
```

```
TOP 1 0
+OK Top of message follows
--- all message headers ---
.
```

```
TOP 1 10
+OK Top of message follows
--- all message headers ---
--- first 10 lines of body ---
.
```

```
QUIT
+OK Sayonara
```


Dr. Gaurav Varshney IIT Jammu

IMAP- Internet Message Access Protocol]

IMAP was designed by Mark Crispin in 1986 as a remote access mailbox protocol,

IMAP allow the user to do operations remotely on the mailbox. This was not possible in POP3 as the mails are downloaded to the client machine and doing all remote operations such as moving mail from one to other folder, creating remote folder, retrieving components of a message etc. are not possible.

IMAP provides additional functionalities to the email access clients

https://www.atmail.com/blog/imap-commands/

Logging in to Remote IMAP Server

telnet imap.servername.com 143

Login

A1 LOGIN username password

Displaying remote folder list

A2 LIST “” *

Examine a specific folder contents

A3 EXAMINE INBOX

Select a specific message from INBOX

A FETCH 1 BODY[]

Check out the sent box

A EXAMINE INBOX.Sent

Create a New Folder in Inbox

A CREATE INBOX.NEW

Check Status of an INBOX

A STATUS INBOX (MESSAGES)

Check Flags of Specific Message

A FETCH 1 FLAGS

Copy and email from one to another

A COPY 1 INBOX.Trash

Dr. Gaurav Varshney IIT Jammu

FTP File Transfer Protocol

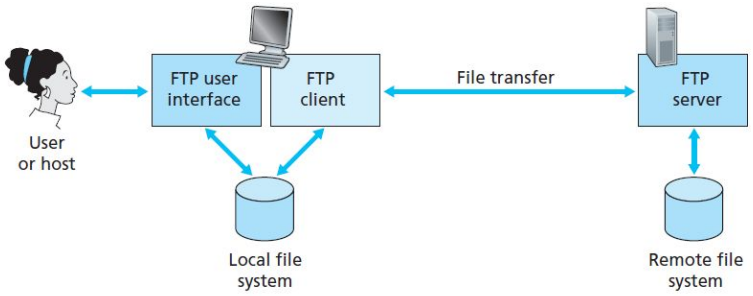


Figure 2.14 ♦ FTP moves files between local and remote file systems

The most striking difference is that FTP uses two parallel TCP connections to transfer a file, a **control connection** and a **data connection**.

The control connection is used for sending control information between the two hosts—information such as user identification, password, commands to change remote directory, and commands to "put" and "get" files.

The data connection is used to actually send a file.

When the server side receives a command for a file transfer over the control connection (either to, or from, the remote host), the server side initiates a TCP data connection to the client side.

FTP sends exactly one file over the data connection and then closes the data connection. If, during the same session, the user wants to transfer another file, FTP opens another data connection.

- `USER username:` Used to send the user identification to the server.
- `PASS password:` Used to send the user password to the server.
- `LIST:` Used to ask the server to send back a list of all the files in the current remote directory. The list of files is sent over a (new and non-persistent) data connection rather than the control TCP connection.
- `RETR filename:` Used to retrieve (that is, get) a file from the current directory of the remote host. This command causes the remote host to initiate a data connection and to send the requested file over the data connection.
- `STOR filename:` Used to store (that is, put) a file into the current directory of the remote host.


```
2160 54.986371364  192.168.193.217  45.87.81.141      FTP      62 Request: SYST
2195 55.179968897  45.87.81.141      192.168.193.217   TCP      62 21 → 51564 [ACK] Seq=129 Ack=57 Win=29312 Len=0
2196 55.179969207  45.87.81.141      192.168.193.217   FTP      75 Response: 215 UNIX Type: L8
2197 55.180015297  192.168.193.217   45.87.81.141      TCP      56 51564 → 21 [ACK] Seq=57 Ack=148 Win=64256 Len=0
2854 74.673121594  192.168.193.217   45.87.81.141      FTP      86 Request: PORT 192,168,193,217,151,109
2857 75.008820541  45.87.81.141      192.168.193.217   FTP      85 Response: 200 PORT command successful
2858 75.008838297  192.168.193.217   45.87.81.141      TCP      56 51564 → 21 [ACK] Seq=87 Ack=177 Win=64256 Len=0
2859 75.008903010  192.168.193.217   45.87.81.141      FTP      62 Request: LIST
2869 75.209715120  45.87.81.141      192.168.193.217   TCP      76 20 → 38765 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 W…
2870 75.209732885  192.168.193.217   45.87.81.141      TCP      68 38765 → 20 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SA…
2871 75.244318914  45.87.81.141      192.168.193.217   TCP      62 21 → 51564 [ACK] Seq=177 Ack=93 Win=29312 Len=0
2877 75.470253246  45.87.81.141      192.168.193.217   FTP     110 Response: 150 Opening ASCII mode data connection for file list
2878 75.470264318  192.168.193.217   45.87.81.141      TCP      56 51564 → 21 [ACK] Seq=93 Ack=231 Win=64256 Len=0
2879 75.470253287  45.87.81.141      192.168.193.217   TCP
2880 75.471177538  45.87.81.141      192.168.193.217   FTP-DA…
2881 75.471207479  192.168.193.217   45.87.81.141      TCP
2882 75.471365833  192.168.193.217   45.87.81.141      TCP
2894 75.735778089  45.87.81.141      192.168.193.217   FTP
2895 75.735795584  192.168.193.217   45.87.81.141      TCP
2896 75.735778113  45.87.81.141      192.168.193.217   TCP
5587 136.068728307 45.87.81.141      192.168.193.217   FTP
5588 136.068761132 192.168.193.217   45.87.81.141      TCP
5589 136.068728680 45.87.81.141      192.168.193.217   TCP
```

The following is a summary of the commonly used FTP Commands.

Command	Description
!	Preceding a command with the exclamation point will cause the command to execute on the local system instead of the remote system.
?	Request assistance or information about the FTP commands. This command does not require a connection to a remote system.
ascii	Set the file transfer mode to ASCII *(Note: this is the default mode for most FTP programs).*
bell	Turns bell mode on / off. This command does not require a connection to a remote system.
binary	Set the file transfer mode to binary (Note: the binary mode transfers all eight bits per byte and must be used to transfer non-ASCII files).
bye	Exit the FTP environment (same as quit). This command does not require a connection to a remote system.
cd	Change directory on the remote system.
close	Terminate a session with another system.
debug	Sets debugging on/off. This command does not require a connection to a remote system.
delete	Delete (remove) a file in the current remote directory (same as rm in UNIX).
dir	Lists the contents of the remote directory. The asterisk (*) and the question mark (?) may be used as wild cards.
get	*RemoteName LocalName*
help	Request a list of all available FTP commands. This command does not require a connection to a remote system.
lcd	Change directory on your local system (same as CD in UNIX).
ls	List the names of the files in the current remote directory.
mget	Copy multiple files from the remote system to the local system. Note: You will be prompted for a "y/n" response before copying each file.
mkdir	Make a new directory within the current remote directory.
mput	Copy multiple files from the local system to the remote system. (Note: You will be prompted for a "y/n" response before copying each file).
open	Open a connection with another system.
put	Copy a file from the local system to the remote system.
pwd	Find out the pathname of the current directory on the remote system.
quit	Exit the FTP environment (same as "bye"). This command does not require a connection to a remote system.
rmdir	Remove (delete) a directory in the current remote directory.
trace	Toggles packet tracing. This command does not require a connection to a remote system.

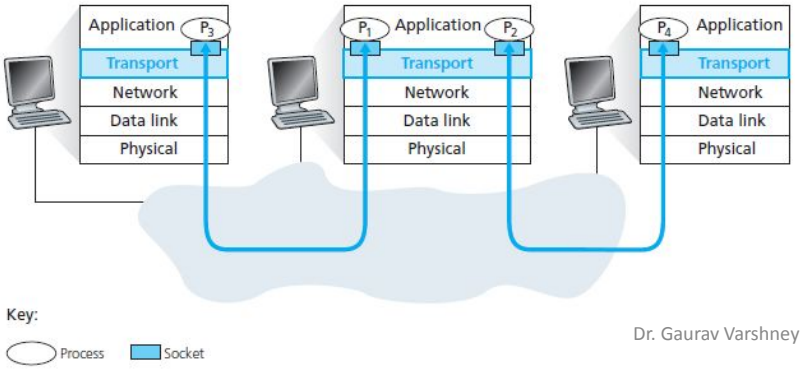
`PORT (h1,h2,h3,h4,p1,p2).`

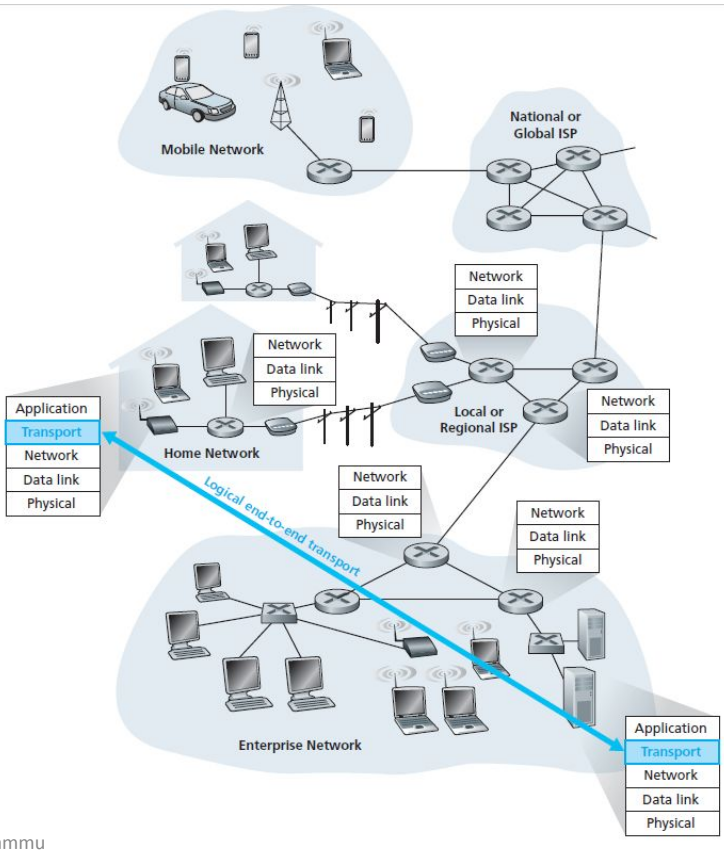
`(p1 * 256) + p2 = data port`

http://www.simotime.com/ftp4cmd1.htm

Transport Layer Services

- Transport Layer provides process to process delivery by multiplexing and demultiplexing.
- It provides a logical connection between processes running on two end systems





Dr. Gaurav Varshney IIT Jammu

Transport Layer Services

application messages = letters in envelopes
processes = cousins
hosts (also called end systems) = houses
transport-layer protocol = Ann and Bill
network-layer protocol = postal service (including mail carriers)

- Transport Layer Protocol run in end systems and is responsible for moving messages from applications to network layer.
- Different Transport Layer Protocols provide different set of services to the applications.
- Transport layer services are constrained by services provided by network layer. Like Transport layer has not much to commit on delay and bandwidth if lower layers do not provide any commitments.
- Though it can add some services: such as reliability

Dr. Gaurav Varshney IIT Jammu

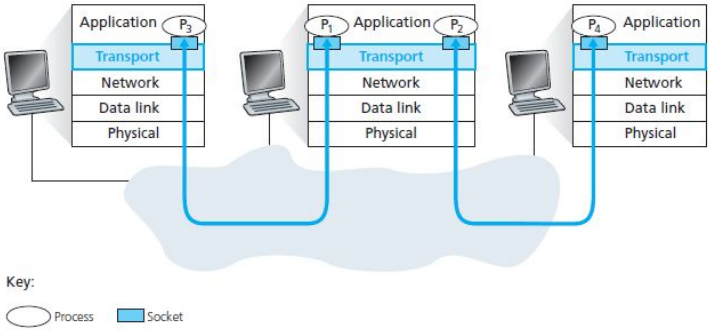
Dr. Gaurav Varshney IIT Jammu

Transport Layer Protocols: UDP and TCP

- Always remember that protocol used at network layer or IP is unreliable and best effort delivery service. [we will come back to it in the Network Layer]
- User Datagram Protocol provides the minimal services expected from a transport layer protocol that includes:
  - Process to process data delivery
  - Error checking
- Transmission Control Protocol provides additional services that includes:
  - Reliable data transfer between two processes
  - Congestion control: Fair share of network resources to communicating applications.

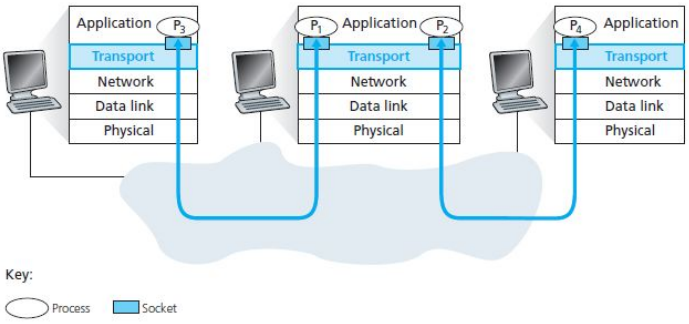
Transport Layer Services

- The job of gathering data chunks at the source host from different sockets, encapsulating each data chunk with header information (that will later be used in demultiplexing) to create segments, and passing the segments to the network layer is called **multiplexing**.
- Job of delivering the data in a transport-layer segment to the correct socket is called **demultiplexing.**
- A process (as part of a network application) can have one or more **sockets**, doors through which data passes from the network to the process and through which data passes from the process to the network.



Transport Layer Services

- **Source port number field** and the **Destination port number field** help in process to process delivery between two hosts.
- Applications open socket which are uniquely identified by transport layer protocol, port numbers and IP addresses.
- Transport layer while reading the segment headers [having destination port number field] passes the segment data to the appropriate socket attached to the given port number in the end system.
- Each port number is a 16-bit number, ranging from 0 to 65535. The port numbers ranging from 0 to 1023 are called **well-known port numbers** and are restricted



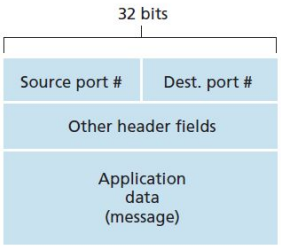


Figure 3.3 ♦ Source and destination port-number fields in a transport-layer segment

Dr. Gaurav Varshney IIT Jammu

Transport Layer: Connectionless UDP

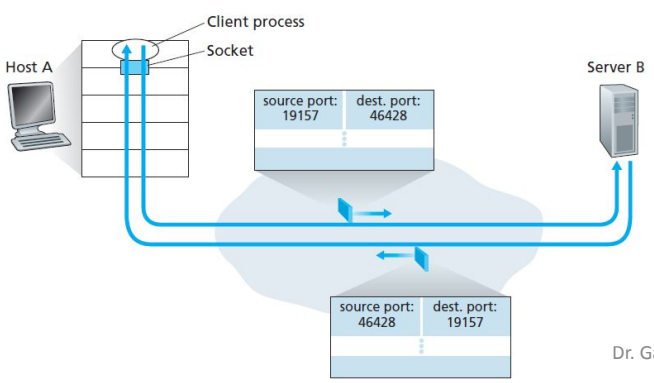
```
clientSocket = socket(socket.AF_INET, socket.SOCK_DGRAM)
clientSocket.bind(('', 19157))
```

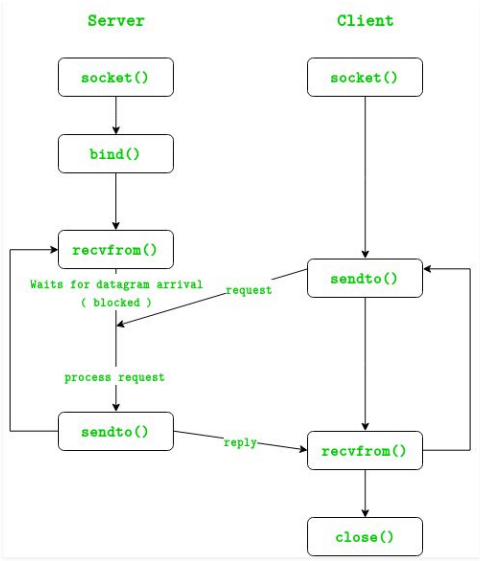
You may be wondering now, what is the purpose of the source port number in the segment header ?

the source port number serves as part of a "return address"—when B wants to send a segment back to A, the destination port in the B-to-A segment will take its value from the source port value of the A-to-B segment.

Dr. Gaurav Varshney IIT Jammu

It is important to note that a UDP socket is fully identified by a two-tuple consisting of a destination IP address and a destination port number.





Dr. Gaurav Varshney IIT Jammu

Transport Layer: Connection Oriented TCP

The TCP server application has a "welcoming socket," that waits for connection establishment requests from TCP clients

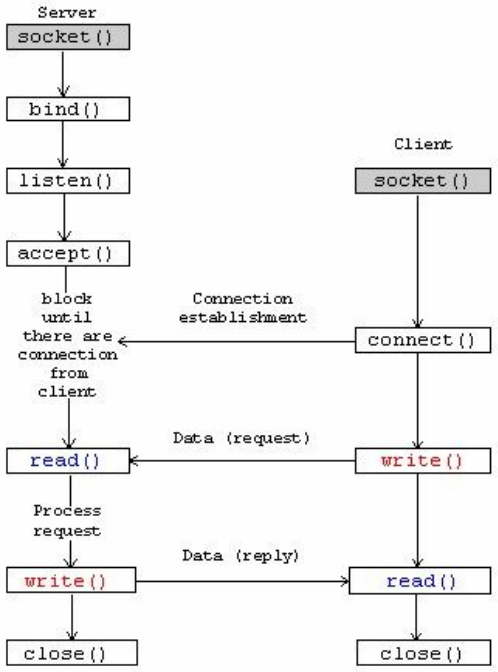
The TCP client creates a socket and sends a connection establishment request segment with the lines:

```
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,12000))
```

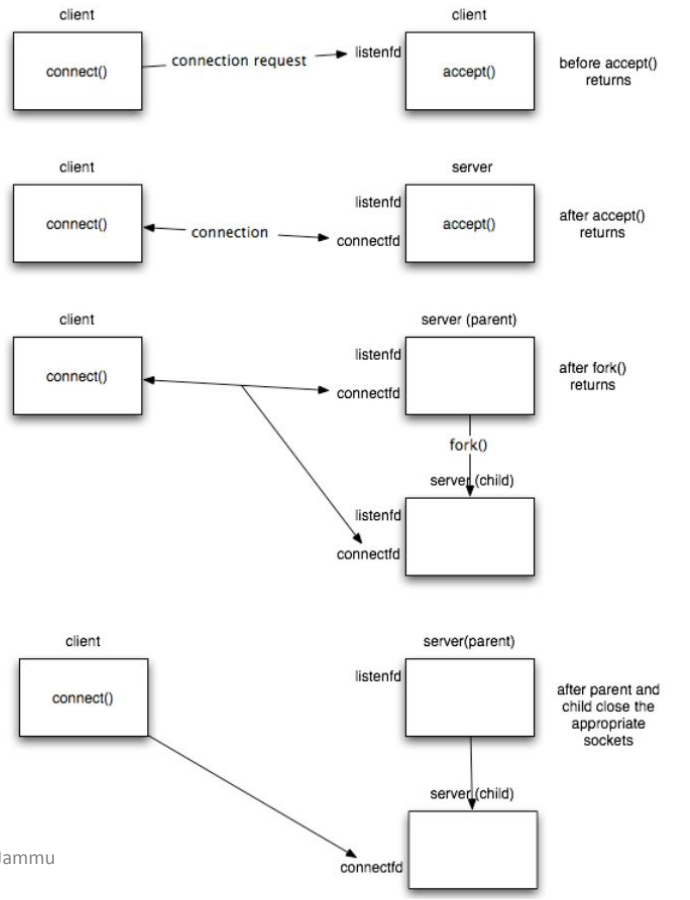
When the host operating system of the computer running the server process receives the incoming connection-request segment with destination port 12000, it locates the server process that is waiting to accept a connection on port number 12000. The server process then creates a new socket:

```
connectionSocket, addr = serverSocket.accept()
```

Also, the transport layer at the server notes the following four values in the connection-request segment: (1) the source port number in the segment, (2) the IP address of the source host, (3) the destination port number in the segment, and (4) its own IP address. The newly created connection socket is identified by these four values; all subsequently arriving segments whose source port, source IP address, destination port, and destination IP address match these four values will be demultiplexed to this socket. With the TCP connection now in place, the client and server can now send data to each other.



TCP: Concurrent



https://www.cs.dartmouth.edu/~campbell/cs50/socketprogramming.html

Dr. Gaurav Varshney IIT Jammu