

Chapter 4

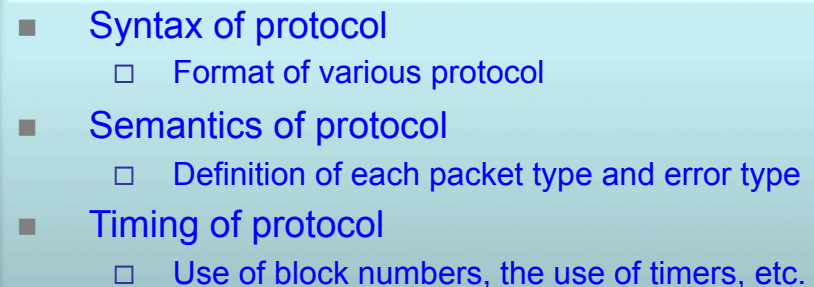
UPDATED: 9/8/16

Traditional Internet Applications



Application-Layer Protocols

- Whenever a programmer creates two network applications, the programmer specifies some details, such as:
 - The **syntax** and **semantics** of messages that can be exchanged
 - Whether the client or server **initiates** interaction
 - Actions to be taken if an **error** arises
 - How the two sides know when to **terminate** communication
- There are two broad types of application-layer protocols that depend on the intended use:
 - **Private** communication
 - **Standardized** service
 - **Requires standardization**

- 
- Syntax of protocol
 - Format of various protocol
 - Semantics of protocol
 - Definition of each packet type and error type
 - Timing of protocol
 - Use of block numbers, the use of timers, etc.

Application-Layer Protocols

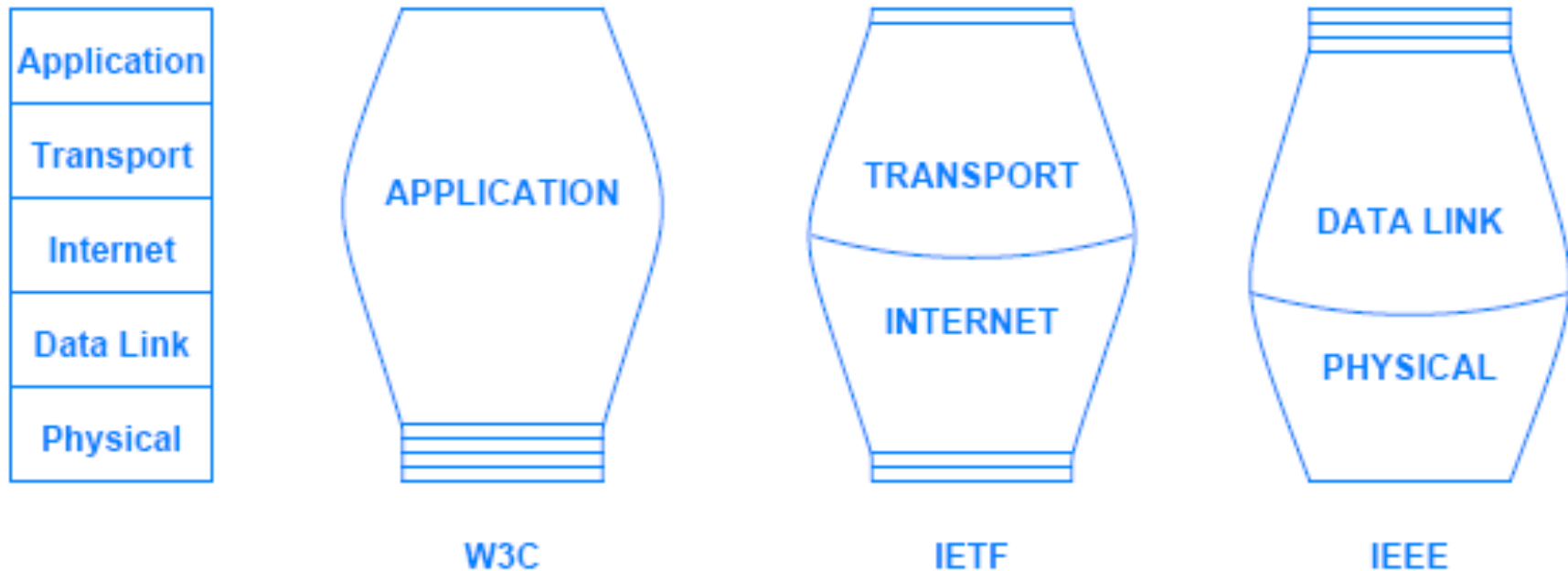
- Private communication
 - A programmer creates a **pair** of applications that communicate over the Internet with the intention that the pair is for private use
 - Interaction between the two applications is straightforward
 - code can be written without writing a formal protocol specification
- Standardized service
 - Expectation is that many programmers will create server software to offer the service or client software to access the service, in this case
 - Application protocol must be documented **independent** of implementation
 - The specification must be precise and unambiguous

So, we need standardization!

WHO IS WHO on the Internet

- Internet Corporation for Assigned Names and Numbers (ICANN)
 - It is contracted by the U.S. government to supply IANA (Internet Assigned Number Authority) – responsible for all IP addresses!
- Institute of Electrical and Electronics Engineers (IEEE)
- The European Computer Manufacturers Association (ECMA)
- The International Electro-technical Commission (IEC)
- The International Organization for Standardization (ISO)
- World Wide Web Consortium (W3C)
 - Develops technologies for www, including specifications, guidelines, and tools (HTML, DHTML, XML were all developed by W3C)
- The *Internet Engineering Task Force (IETF)*
 - Protocol engineering and development arm of the Internet
 - IETF's technical management is handled by IESG (Internet Engineering Steering Group)
 - the **RFC repository maintained by the IETF**
 - *RFC → IETF → Review →*
 - *If not accepted goes to the Repository "historical"*
 - *If accepted it become an standard*

Various Standard Emphasis



Institute of Electrical and Electronics Engineers

Representation and Transfer

- Application-layer protocols specify two aspects of interaction
 - Representation
 - Transfer

Representation:
Data syntax /
e.g., ASCII

Transfer:
Interaction
between C/S

Aspect	Description
Data Representation	Syntax of data items that are exchanged, specific form used during transfer, translation of integers, characters, and files between computers
Data Transfer	Interaction between client and server, message syntax and semantics, valid and invalid exchange error handling, termination of interaction

Review

OSI & TCP/IP Protocol Architectures

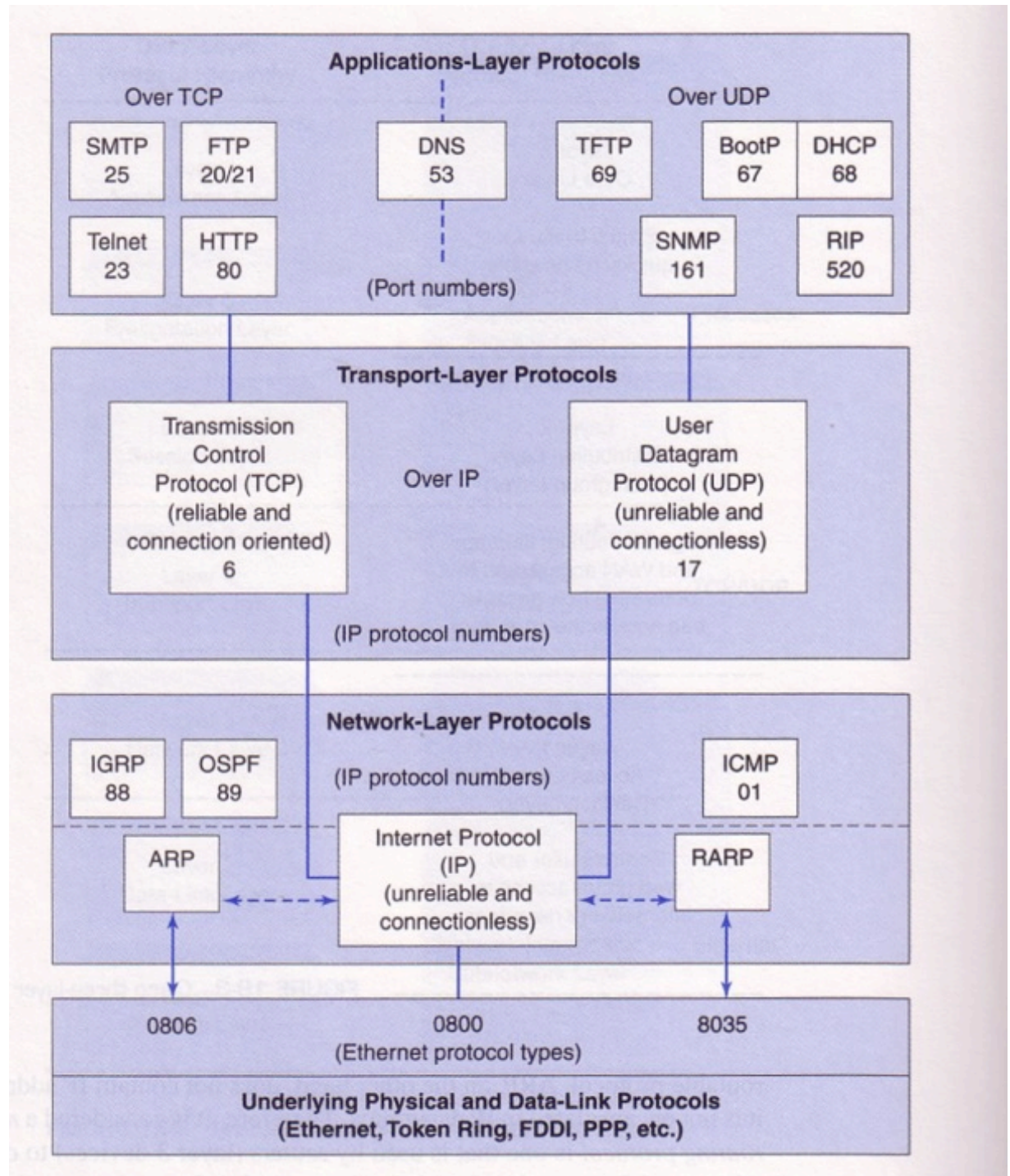
OSI Model			
	Data unit	Layer	Function
Host layers	Data	7. Application	Network process to application
		6. Presentation	Data representation and encryption
		5. Session	Interhost communication
	Segment	4. Transport	End-to-end connections and reliability (TCP)
Media layers	Packet/Datagram	3. Network	Path determination and logical addressing (IP)
	Frame	2. Data link	Physical addressing (MAC & LLC)
	Bit	1. Physical	Media, signal and binary transmission

Remember

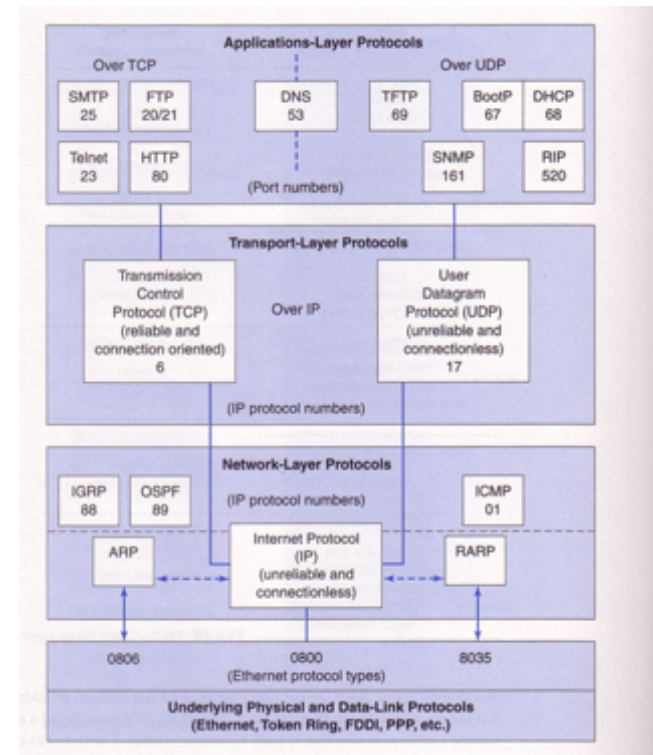
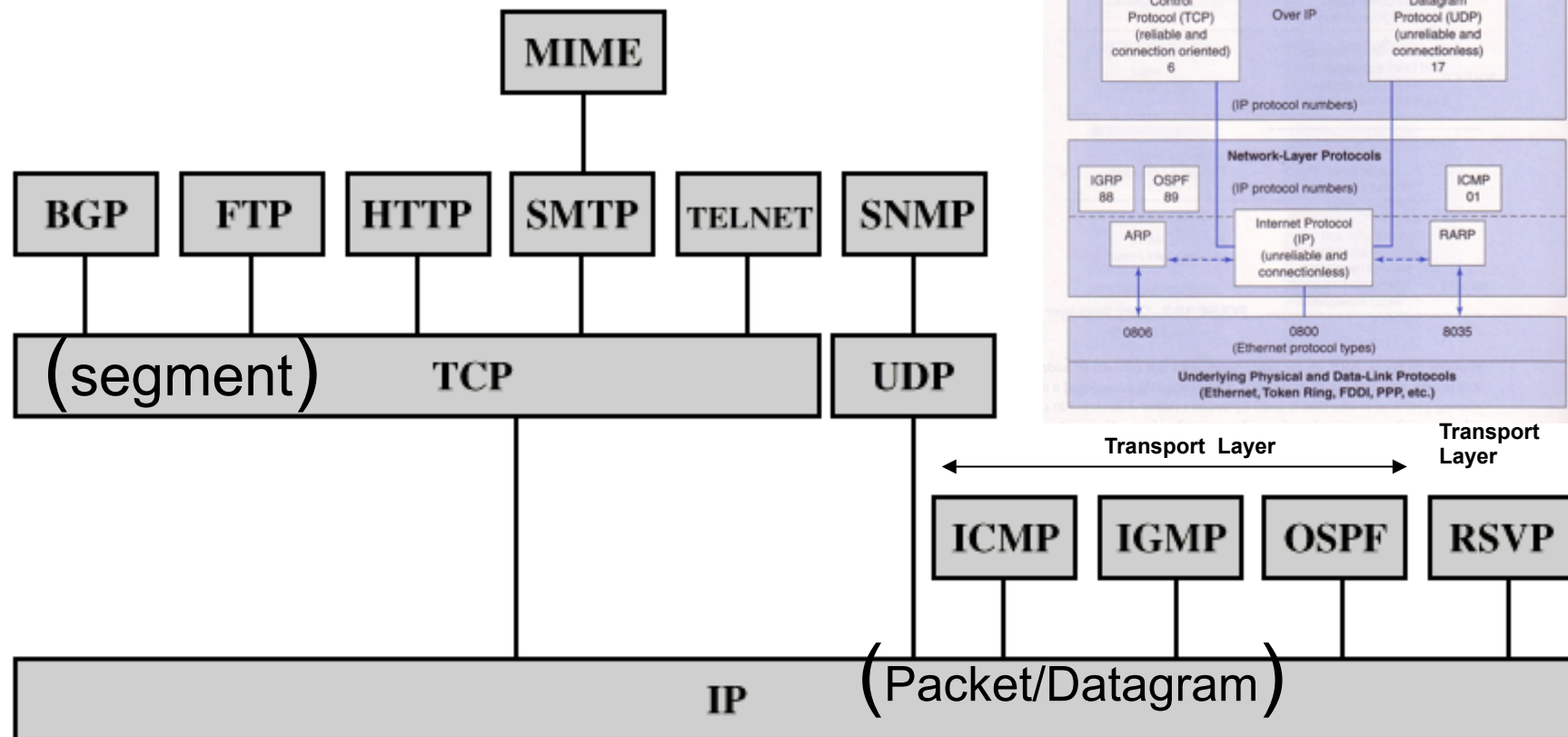
- The five-layer TCP/IP model
- 5. Application layer**
- DHCP · DNS · FTP · [Gopher](#) · HTTP · IMAP4 · IRC · NNTP · XMPP · POP3 · SIP · SMTP · SNMP · SSH · TELNET · RPC · RTCP · RTSP · TLS · SDP · SOAP · GTP · STUN · NTP · (more)
- 4. Transport layer**
- TCP · UDP · DCCP · SCTP · RTP · RSVP · (more)
- 3. Network/Internet layer**
- IP (IPv4 · IPv6) · OSPF · IS-IS · BGP · IPsec · ARP · RARP · RIP · ICMP · ICMPv6 · IGMP · (more)
- 2. Data link layer**
- 802.11 (WLAN) · 802.16 · Wi-Fi · WiMAX · ATM · DTM · Token ring · Ethernet · FDDI · Frame Relay · GPRS · EVDO · HSPA · HDLC · PPP · PPTP · L2TP · ISDN · ARCnet · (more)
- 1. Physical layer**
- Ethernet physical layer · Modems · PLC · SONET/SDH · G.709 · Optical fiber · Coaxial cable · Twisted pair · (more)

Applications layers and their ports

- Examples of physical layers: RS-232, V.35, RJ-48, DS3, OC-n, High Speed Serial Interface



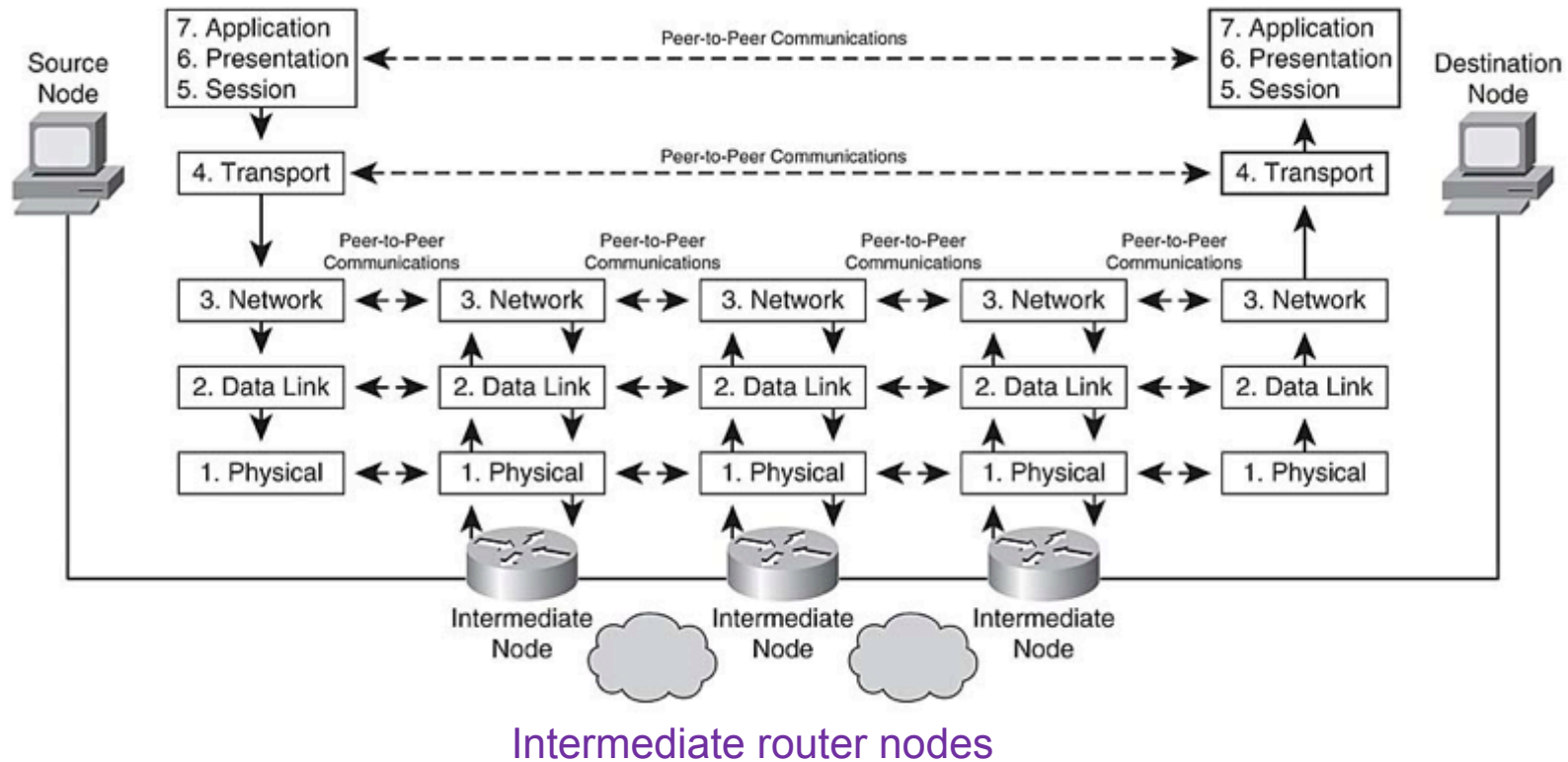
TCP/IP Applications



BGP = Border Gateway Protocol
FTP = File Transfer Protocol
HTTP = Hypertext Transfer Protocol
ICMP = Internet Control Message Protocol
IGMP = Internet Group Management Protocol
IP = Internet Protocol
MIME = Multi-Purpose Internet Mail Extension

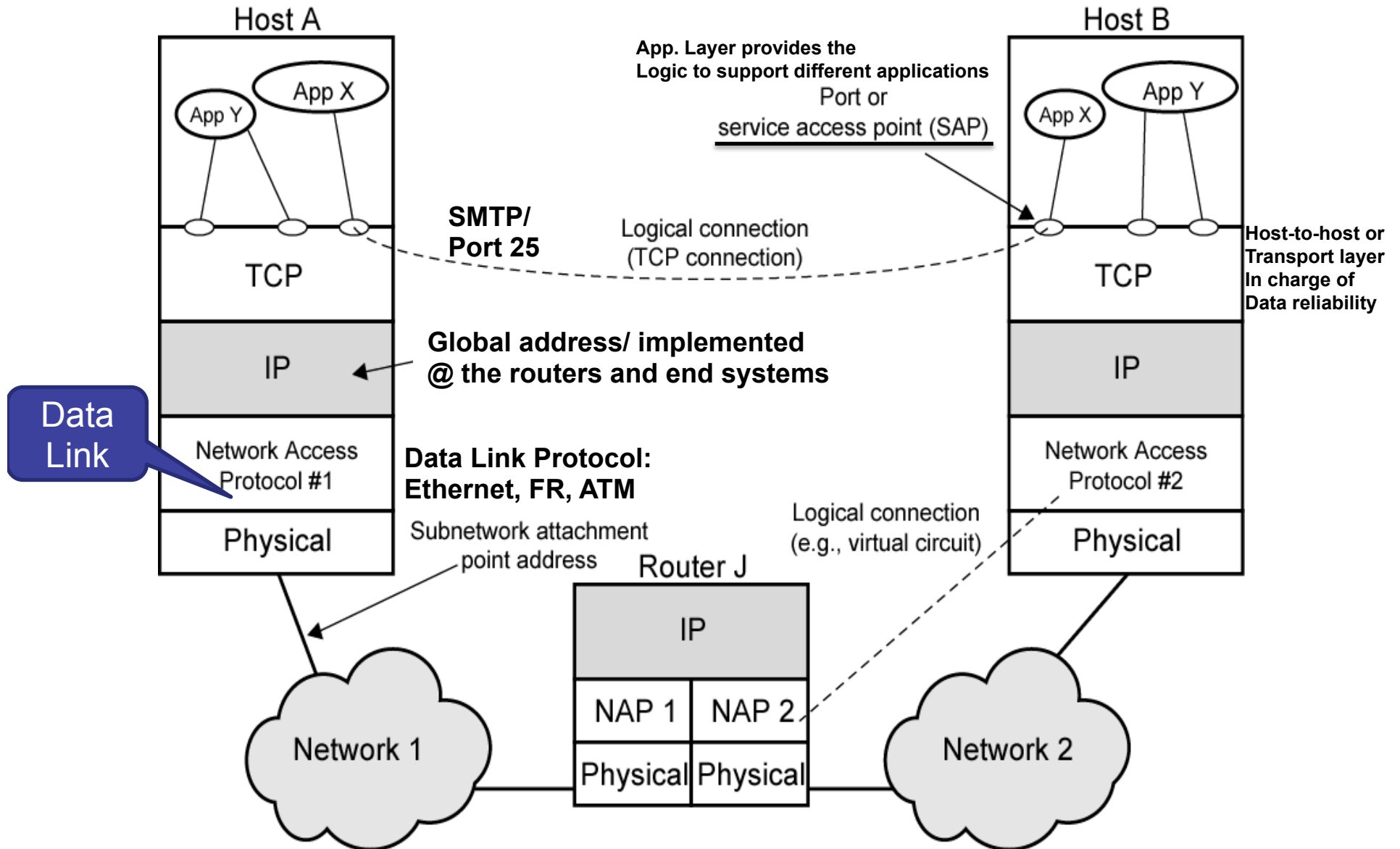
OSPF = Open Shortest Path First
RSVP = Resource ReSerVation Protocol
SMTP = Simple Mail Transfer Protocol
SNMP = Simple Network Management Protocol
TCP = Transmission Control Protocol
UDP = User Datagram Protocol

Cross-Layer Model

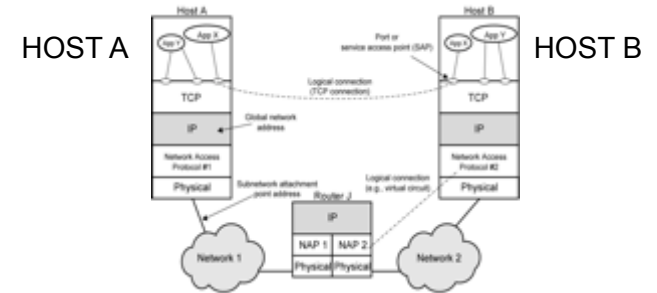


TCP is only implemented at the **end** system

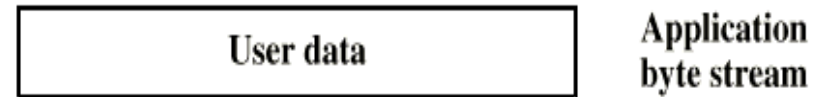
Operation of TCP and IP



Operation of TCP/IP



Process at **host A** hands the message to TCP layer:
Send the message to **host B** port 2



TCP hands the message to IP – destination will be **Host B**



IP hands it to network Layer -> next hop is an intermediate **router J**



Conditions
The signal format
For the physical path



Checking the Physical and IP Address

- Available tools
 - ifconfig
 - hostname
 - `$hostname`
 - `$sudo hostname farid_machine`
 - nslookup
 - `$nslookup www.sonoma.edu`
- Installing new tools:
 - `$sudo apt-get install nmap`

Other Tools:

https://docs.google.com/document/d/1HGdP1xDdnA5BG6Cglh_5hcK467nF4RZJsivqjJ3e4l0/edit?usp=sharing

IFCONFIG COMMAND

```
47229q-shz2010a:~ farid11$ ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
    inet6 ::1 prefixlen 128
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
    inet 127.0.0.1 netmask 0xff000000
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    ether 10:9a:dd:53:40:c1
    inet6 fe80::129a:ddff:fe53:40c1%en0 prefixlen 64 scopeid 0x4
    inet 10.10.18.128 netmask 0xfffffe00 broadcast 10.10.19.255
    media: autoselect (100baseTX <full-duplex,flow-control>)
    status: active
en1: flags=8923<UP,BROADCAST,SMART,PROMISC,SIMPLEX,MULTICAST> mtu 1500
    ether f0:b4:79:20:df:a6
    media: autoselect (<unknown type>)
    status: inactive
fw0: flags=8822<BROADCAST,SMART,SIMPLEX,MULTICAST> mtu 4078
    lladdr 70:cd:60:ff:fe:10:bb:0e
    media: autoselect <full-duplex>
    status: inactive
```

IFCONFIG COMMAND

```
47229q-shz2010a:~ farid11$ ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
    inet6 ::1 prefixlen 128
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
    inet 127.0.0.1 netmask 0xff000000
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    ether 10:9a:dd:53:40:c1
    inet6 fe80::129a:ddff:fe53:40c1%en0 prefixlen 64 scopeid 0x4
    inet 10.10.18.128 netmask 0xfffffe00 broadcast 10.10.19.255
    media: autoselect (100baseTX <full-duplex,flow-control>)
    status: active
en1: flags=8923<UP,BROADCAST,SMART,PROMISC,SIMPLEX,MULTICAST> mtu 1500
    ether f0:b4:79:20:df:a6
    media: autoselect (<unknown type>)
    status: inactive
fw0: flags=8822<BROADCAST,SMART,SIMPLEX,MULTICAST> mtu 4078
    lladdr 70:cd:60:ff:fe:10:bb:0e
    media: autoselect <full-duplex>
    status: inactive
```


Application Services and Examples

- WWW
- FTP
- TFTP
- MAIL
- DNS

Web Protocols

- The World Wide Web (WWW) is one of the most widely used services in the Internet
- Web is complex
 - many protocol standards have been devised to specify various aspects and details

Standard	Purpose
HyperText Markup Language (HTML)	A representation standard used to specify the contents and layout of a web page
Uniform Resource Locator (URL)	A representation standard that specifies the format and meaning of web page identifiers
HyperText Transfer Protocol (HTTP)	A transfer protocol that specifies how a browser interacts with a web server to transfer data

Document Representation with HTML

- **HyperText Markup Language** (HTML) is a representation standard that specifies the **syntax** of a web page
 - A Markup Language
- HTML has the following general characteristics:
 - Uses a textual representation
 - Uses text file with html extension
 - Describes pages that contain **multimedia**
 - Follows a **declarative** rather than **procedural** paradigm
 - Indicated WHAT to represent not HOW
 - Provides **markup** specifications instead of formatting
 - Uses Tags: HTML, IMG, <A HRER...>
 - The displayed format depends on the browser
 - Permits a **hyperlink** to be embedded in an arbitrary object
 - Using Tags
 - Allows a document to include **metadata**

Document Representation with HTML

```
<HTML>  
  <HEAD>  
    <TITLE>  
      text that forms the document title  
    </TITLE>  
  </HEAD>  
  <BODY>  
    body of the document appears here  
  </BODY>  
</HTML>
```

PHP

- PHP is a general-purpose server-side **scripting language**
 - originally designed for Web development to produce dynamic Web pages
- It is one of the first developed server-side scripting languages to be **embedded into an HTML** source document rather than calling an external file to process data
- Advantages of PHP over HTML
 - Database interaction (add, modify, delete data. Alter database structures and more)
 - Output dynamic contents (does different things according to the time of day, number of time the user has logged in, number of files in a directory, entries in database, etc.)
 - String/text/date manipulation
 - Error checks
 - Sessions and cookies (where website remembers you for a period of time)
 - Compressions and archives
 - Cryptography extensions
 - Math functions

PHP Code Example

```
1 <?php
2 /*
3  Parameter Reader
4  Language: PHP
5  To run type:  http://www.sonomaesdep.host-ed.me/parameter_reader.php?name=farid&age=23
6                or:      http://www.sonomaesdep.host-ed.me/parameter_reader.php?name=farid&age=23&Last=Farah
7 */
8 // print beginig of an HTML page:
9
10 echo "<html><head></head><body>\n";
11
12 echo "<p> Hello \n </p>"; // this is how you add text
13
14 // Print out all the variables
15 foreach ($_REQUEST as $key => $value)
16 {
17     echo "$key: $value<br>\n";
18 }
19
20 // finish the HTML
21 echo "</body></html>\n";
22
23 ?>
```

TRY THIS:

http://www.sonomaesdep.host-ed.me/parameter_reader.php?name=farid&age=23&Last=Farah

Uniform Resource Locators and Hyperlinks

- The Web uses a **syntactic** form known as a **Uniform Resource Locator (URL)** to specify a web page
- The general form of a URL is:

```
protocol://computer_name:port/document_name%parameters
```

- where
 - **protocol** is the name of the protocol used to access the document
 - ftp, http, etc.
 - **computer_name** is the domain name of the computer on which the document resides
 - **port** (optional) port number at which the server is listening
 - **document_name** (optional) name of the document
 - **%** (optional) parameters for the page: #title
 - Example:

```
http://www.sonoma.edu/users/f/farahman/sonoma/courses/es110/index.html#Tentative\_Weekly\_Schedule
```

Uniform Resource Locators and Hyperlinks

- In a typical URL, a user can omit many of the parts

www.netbook.cs.purdue.edu

- Which omits
 - the protocol (http is assumed)
 - **the port (80 is assumed)**
 - the document name (index.html is assumed)
 - and parameters (none are assumed)

Try these in your URL:

- <http://aitislab.com/phpf/RxNewData.php>
- <http://192.168.1.73:8000/>
- <http://130.157.3.70/>

What is happening?

Alternatively:

<http://www.netbook.cs.purdue.edu/toc/toc01.htm>

Google Autocomplete!

Web Document Transfer with HTTP

- **HyperText Transfer Protocol (HTTP)** is the primary **transfer** protocol that a browser uses to interact with a web server
- The current version of HTTP is 1.1
- HTTP is a **stateless** protocol, meaning that Web pages are sent independent of each other
 - This makes it more challenging to create a shopping cart application
- HTTP 1.1 supports **persistent connections**
 - This allows the browser to receive multiple files in one TCP connection (without generating multiple connections)
 - This can speed up communication
 - Although you see a single page in your browser, it can be composed of many text and image files
- HTTP can be characterized as follows:
 - Uses textual **control messages**
 - Requests: GET, PUT, HEAD (status information) , POST (replace with new data)
 - Transfers **binary data** files
 - Can download or upload data
 - Incorporates **caching**

Web Document Transfer with HTTP

- The most common form of **interaction** begins with the browser **requesting** a page from the server
- The browser (client) sends a **GET** request over
- The server responds by sending a header, a blank line, and the requested document
- A **GET** request has the following form:

GET /item version CRLF

Status Code	Corresponding Status String
200	OK
400	Bad Request
404	Not Found

```
GET /hello.htm HTTP/1.1
Host: www.technowidgets.com
```

```
HTTP/1.1 200 OK
Date: Sat, 15 Mar 2008 07:35:25 GMT
Server: Apache/1.3.37 (Unix)
Last-Modified: Tue, 1 Jan 2008 12:03:37 GMT
ETag: "78595-81-3883bbe9"
Accept-Ranges: bytes
Content-Length: 16
Connection: close
Content-Type: text/plain
```

This is a test.

sample output from an **Apache web server**

How a Web Server Works

- HTTP (Hypertext Transfer Protocol) defines how information is **passed** between a **browser and a Web server**
- The two most popular Web servers are
 - **Apache** from Apache Software Foundation
 - Internet Information Services (**IIS**) from Microsoft
- Almost two-thirds of all Web servers use Apache

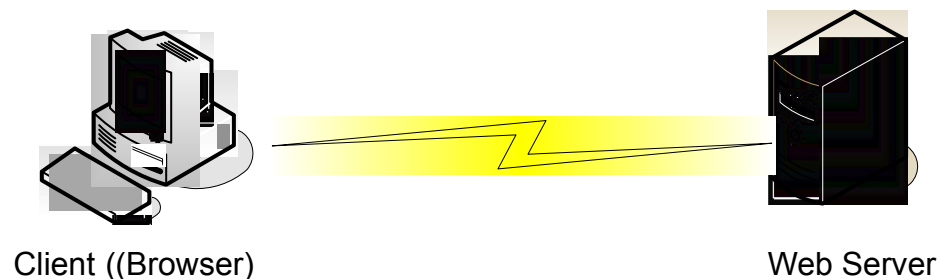
Understanding HTTP

Establishing a connection

- The user types <http://www.technowidgets.com/hello.htm> in the browser
- The Web browser contact the DNS to **resolve** the address
- When the browser sends a request to a Web server, it looks like:

```
GET /hello.htm HTTP/1.1  
Host: www.technowidgets.com
```

- The CLIENT requests the **hello.htm file** from the root of the Web server
 - Using the GET command
- NOTE: There could be multiple hosts at the same IP address: e.g., ftp.technowidgets.com



We will talk
about DNS in
the future slides

Server Response:

```
HTTP/1.1 200 OK
Date: Sat, 15 Mar 2008 07:35:25 GMT
Server: Apache/1.3.37 (Unix)
Last-Modified: Tue, 1 Jan 2008 12:03:37 GMT
ETag: "78595-81-3883bbe9"
Accept-Ranges: bytes
Content-Length: 16
Connection: close
Content-Type: text/plain
```

This is a test.

Status Code	Corresponding Status String
200	OK
400	Bad Request
404	Not Found

sample output from an [Apache web server](#)

Server Response

- The following shows some of the headers along with the HTML that the Web server would send:

```
HTTP/1.1 200 OK
```

```
Server: Microsoft-IIS/5.0
```

```
Content-Type: text/html
```

```
Last-Modified: Fri, 17 May 2005 18:21:25 GMT
```

```
Content-Length: 43
```

```
<html><body>
```

```
Hello, World
```

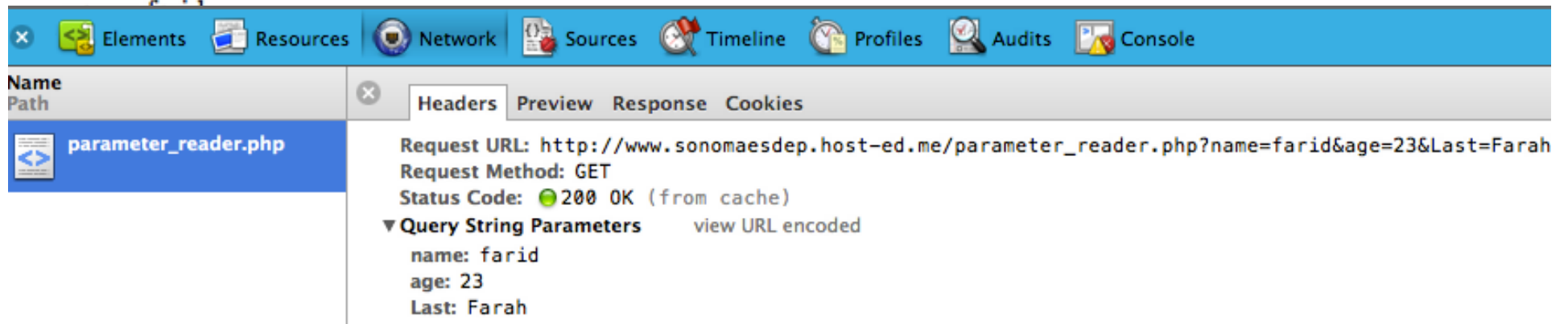
```
</body></html>
```

- The headers contain information about the page

Viewing the Responses!



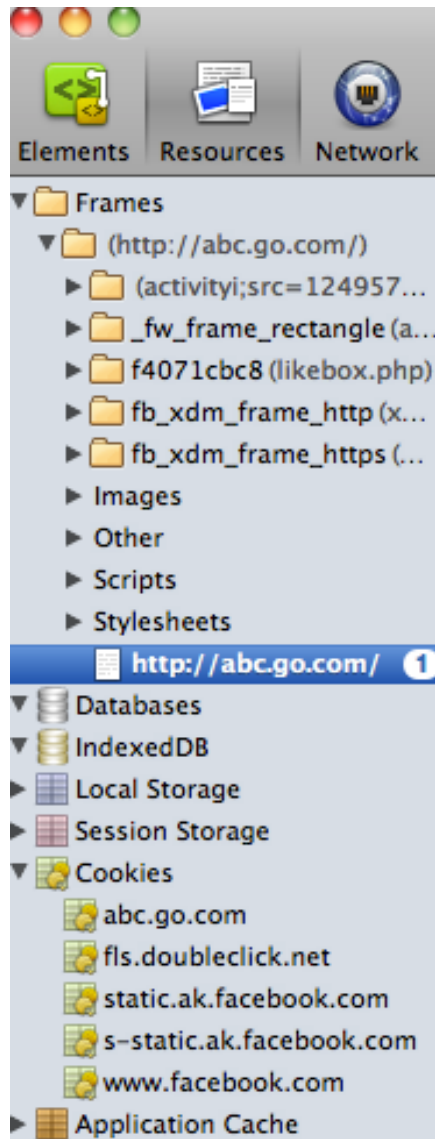
Hello



Using CHROME's DEVELOPMENT TOOL
You can see the responses!

<http://www.sonomaesdep.host-ed.me/ES465/Farid/hello.php>

What Happens When You Go to abc.com?



Cookies & Web Sites!

Name	Method	Status	Type	Initiator	Size	Time	Latency
1x1_FY11Lexu: s0.2mdn.net/vi	GET	200 OK	image	http://ad	403B	403ms	489ms
xaxis_sync.xgi e.nexac.com/e	GET	302 Found	text	https://t	555B	606ms	733ms
xrefid.xgi e.nexac.com/e	GET	200 OK	image	https://e	524B	977ms	977ms
B6251679.35;s ad.doubleclick.i	GET	200 OK	text	1:1 Script	272B	444ms	
B6251679.32;s ad.doubleclick.i	GET	302 Found	text	all.js:87 Script	598B	690ms	
www.facebook.com	GET	302 Found	text	activity:s	1.22kB	446ms	
Retarget_ABCH oasn04.247real	GET	302 Found	text	http://oa	684B	842ms	
rsp t.mookie1.com	GET	302 Found	text	Other	789B	440ms	
s8828590004 w88.go.com/b/	GET	302 Found	text	http://w8	309B	222ms	
r b.scorecardrese	GET	200 OK	image	Other	428B	26ms	
abc.com	GET	301 Moved	text	Other	428B	25ms	
1x1_FY11Lexu: s0.2mdn.net/vi	GET	200 OK	image	http://ad	359B	810ms	

Which WEBS SITES you communicated with!

Features in Apache

- Apache 1.3 was used for many years but version 2.0 was released in 2001
- Apache can also be used as a proxy server
 - A [proxy](#) server isolates your real Web server from the Internet
 - The request is taken from the Internet and it is transferred to the Web server
- Apache 2.0 has
 - Better support for Windows
 - Support for IPv6
 - Simplified configuration
 - Unicode support in Windows
 - Multilanguage error responses
- Apache supports many programming languages such as Perl and PHP

FTP Protocol

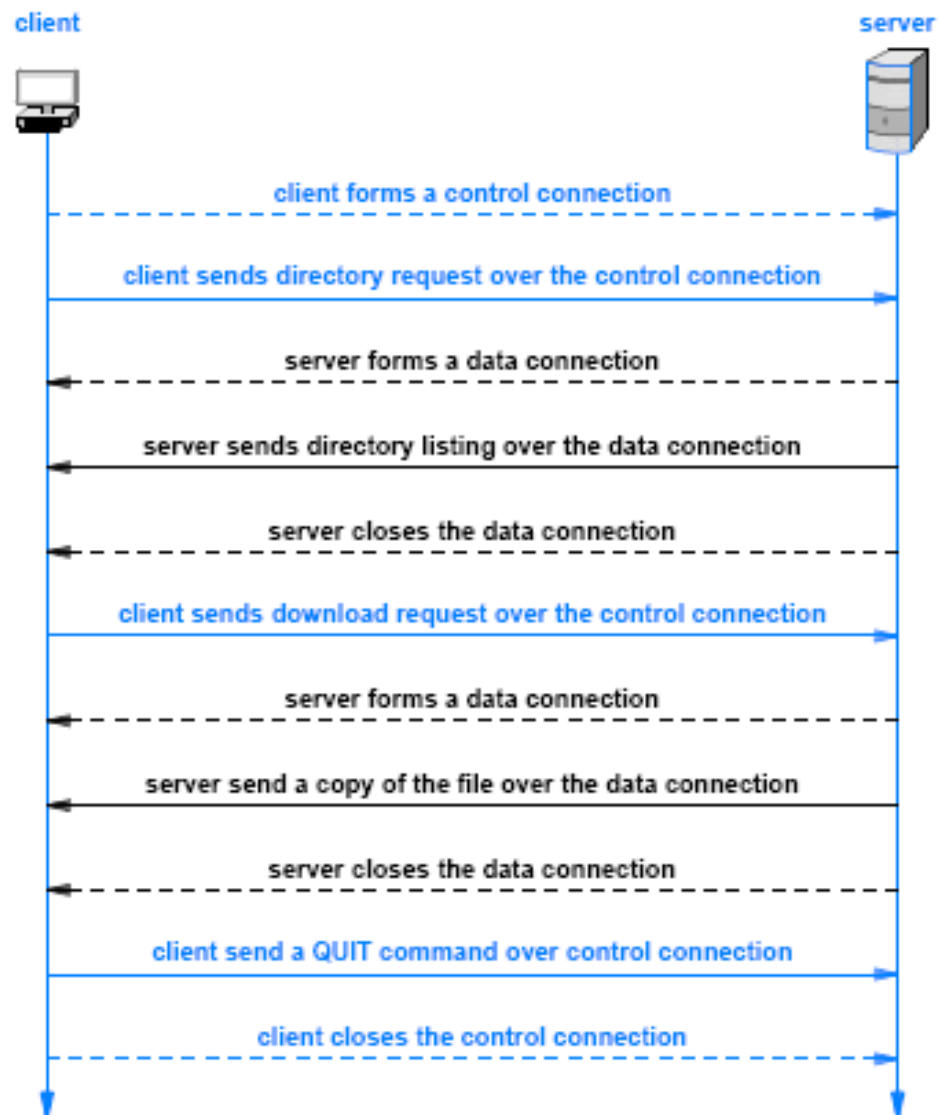
File Transfer Protocol (FTP)

- A file is the fundamental storage **abstraction**
- A file can hold an arbitrary **object** (e.g., a document, spreadsheet, computer program, graphic image, or data)
- FTP can send a copy of a file from one computer to another
 - provides a powerful mechanism for **the exchange of data**
- File transfer across the Internet is complicated because computers are **heterogeneous**.
- Each computer system may have a different:
 - file representations
 - type information
 - Naming (jpg vs. jpeg)
 - file access mechanisms

FTP Communication Paradigm

- A client allocates a protocol port on its local OS and sends the port number to the server
- FTP employs the way a client and server interact
 - a client establishes a connection to an FTP server and sends a series of requests to which the server responds
 - an FTP server **does not send responses** over the same connection on which the client sends requests
 - Instead, the original connection the client creates, called a **control connection**, is reserved for commands
 - Each time the server needs to download or upload a file, the server opens a new connection
 - To distinguish them from the control connection, the connections used to transfer files are called **data connections**

Control vs. Data Connection



Control vs. Data Connection

FTP to 176.9.105.210

Filter: `ip.addr == 176.9.105.210` Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
66	7.613674000	192.168.5.27	176.9.105.210	TCP	78	59679 > ftp [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 TSval=
75	8.360942000	176.9.105.210	192.168.5.27	TCP	74	ftp > 59679 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1356
76	8.361017000	192.168.5.27	176.9.105.210	TCP	66	59679 > ftp [ACK] Seq=1 Ack=1 Win=262140 Len=0 TSval=91873:

Check the packets!

▶ Frame 149: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
▶ Ethernet II, Src: Apple_20:df:a6 (f0:b4:79:20:df:a6), Dst: Portwell_1a:40:1a (00:90:fb:1a:40:1a)
▶ Internet Protocol Version 4, Src: 192.168.5.27 (192.168.5.27), Dst: 176.9.105.210 (176.9.105.210)
▼ Transmission Control Protocol, Src Port: 59679 (59679), Dst Port: ftp (21), Seq: 1, Ack: 60, Len: 15

Source port: 59679 (59679)

Destination port: ftp (21)

[Stream index: 9]

Sequence number: 1 (relative sequence number)

[Next sequence number: 16 (relative sequence number)]

Acknowledgment number: 60 (relative ack number)

Header length: 32 bytes

▶ Flags: 0x018 (PSH, ACK)

Window size value: 65535

[Calculated window size: 262140]

[Window size scaling factor: 4]

▶ Checksum: 0x9b7a [validation disabled]

▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps

▶ [SEQ/ACK analysis]

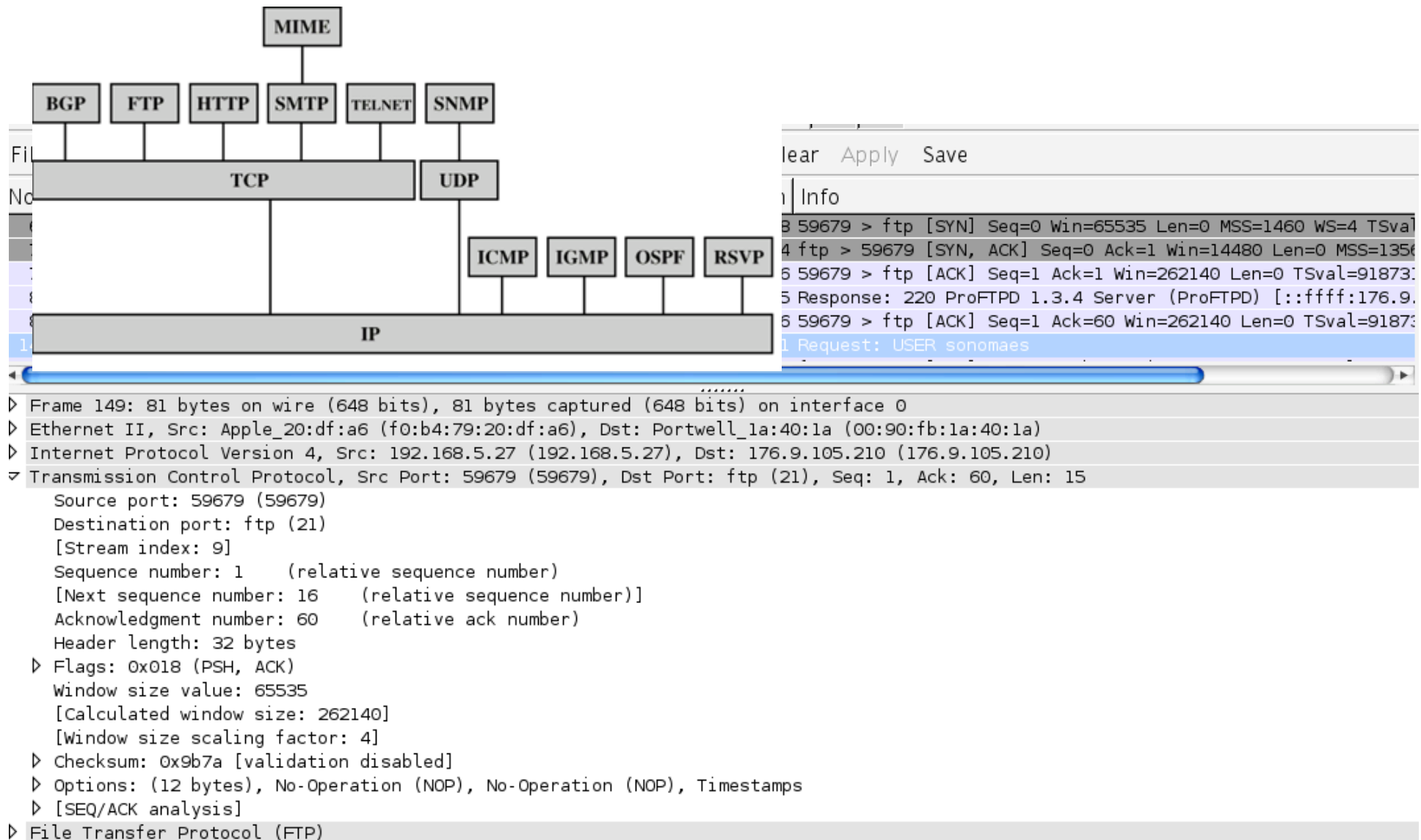
▶ File Transfer Protocol (FTP)

It is clear text and all
the information can be
viewed!

From a terminal type:

```
village-158-231:~ farid11$ ftp 176.9.105.210
Connected to 176.9.105.210.
220 ProFTPD 1.3.4 Server (ProFTPD) [::ffff:176.9.105.210]
Name (176.9.105.210:farid11):
```

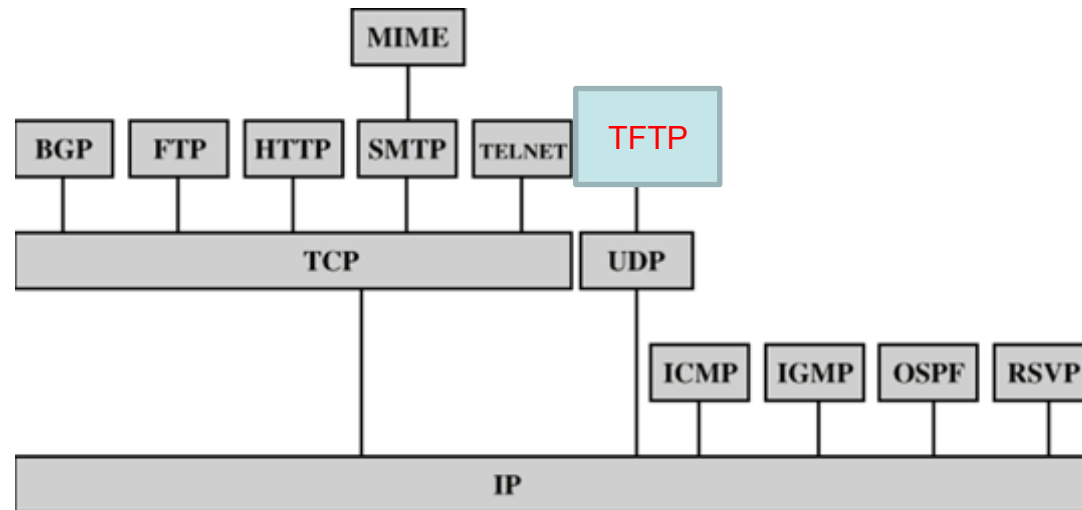
FTP Protocol Stack



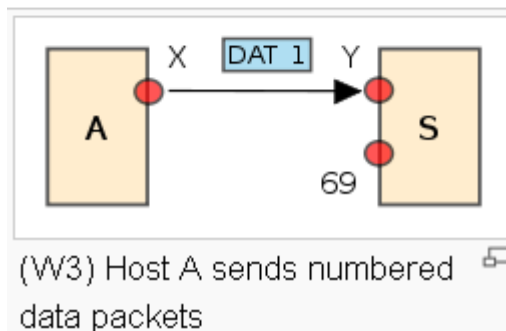
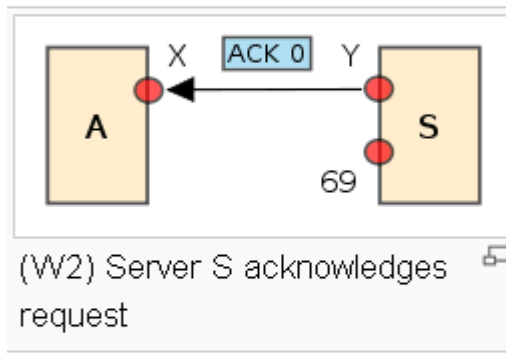
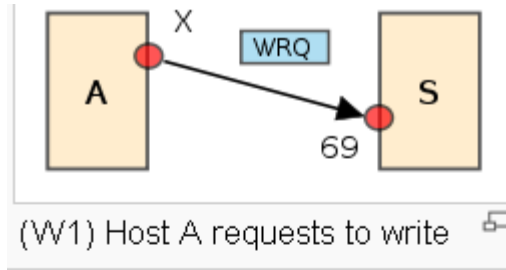
TFTP

Trivial FTP (RFC 1350)

- Much simpler than FTP (RFC 959)
- Has no access control or user ID
- Uses port 69 (FTP uses port 21)
- Encapsulated in UDP (not TCP)
- Can use raw 8-bit or ASCII (Mode of operation)
- Often uses 512-byte blocksize
 - <http://www.rfc-archive.org/getrfc.php?rfc=1783>



Protocol Example: Trivial FTP

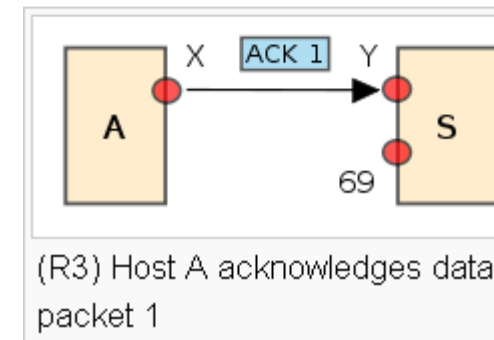
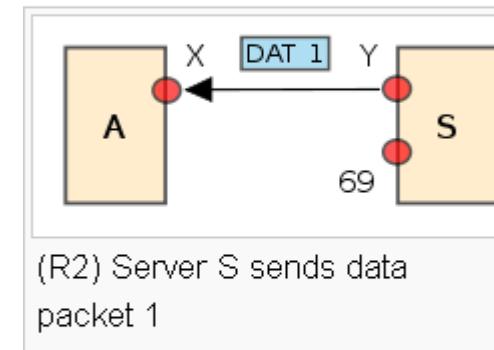
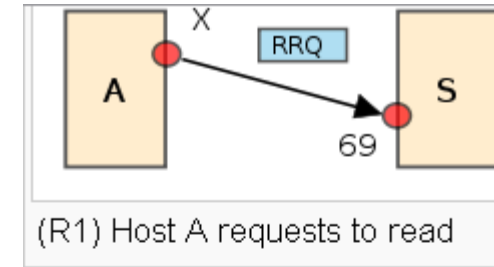


Trivial FTP uses **UDP** port **69** as its transport protocol (unlike FTP which uses TCP port 21).

Each file transferred via TFTP constitutes an independent exchange. That transfer is performed in block-step, with only one packet (either a block of data, or an acknowledgement) - **one block followed by an ACK**

→TFTP has no authentication or encryption mechanisms.

→Packet types: RRQ, WRA, ACT. ERR, DAT



Trivial FTP – Frame Format (Syntax)

TFTP supports five types of packets, all of which have been mentioned above:

opcode	operation
1	Read request (RRQ)
2	Write request (WRQ)
3	Data (DATA)
4	Acknowledgment (ACK)
5	Error (ERROR)

The TFTP header of a packet contains the opcode associated with that packet.

2 bytes	string	1 byte	string	1 byte	

Opcode	Filename	0	Mode	0	RRQ/WRQ PACKET

2 bytes	2 bytes	n bytes			

Opcode	Block #		Data		DATA PACKET

2 bytes	2 bytes				

Opcode	Block #				ACK PACKET

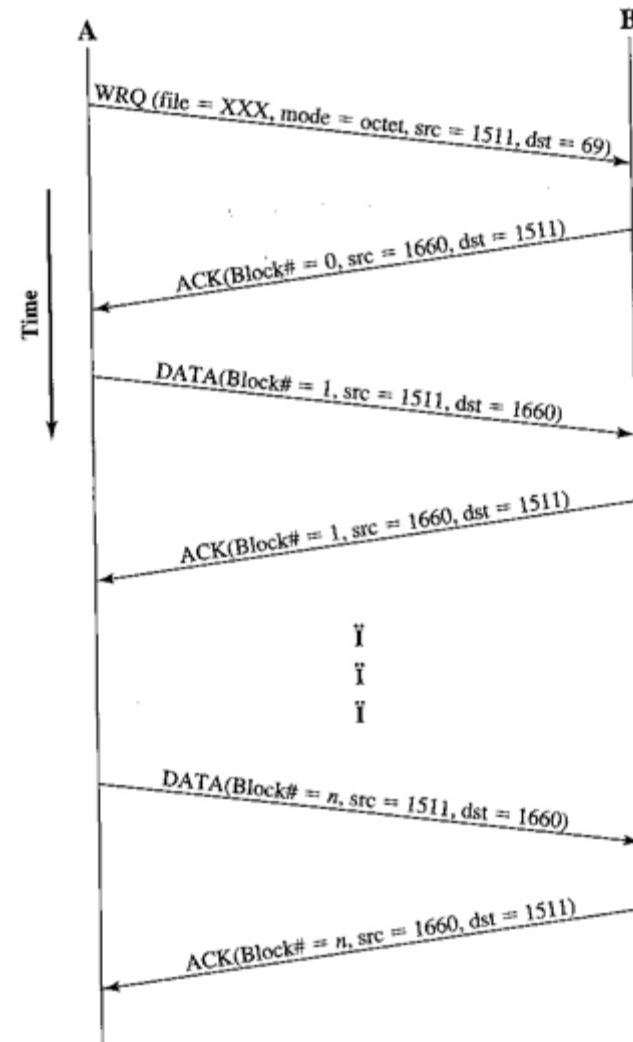
2 bytes	2 bytes	string	1 byte		

Opcode	ErrorCode	ErrMsg	0		ERROR PACKET

TFTP Timing Diagram

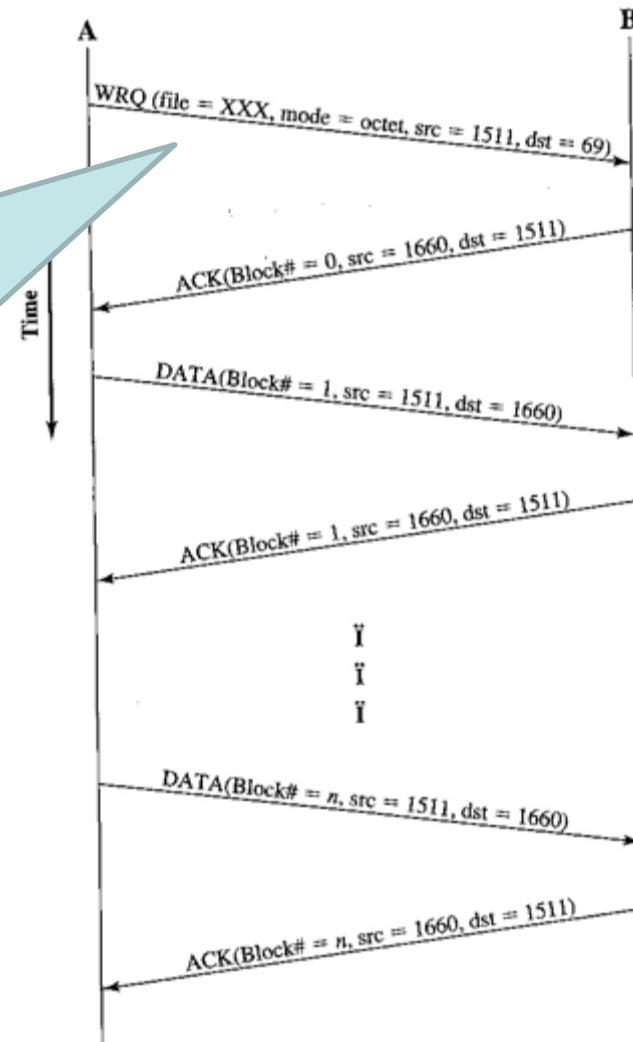
- Timeout mechanism – retransmit the DATA if no ACK is received
 - Retransmitted ACK and DATA have the same block number
 - No further DATA is transmitted unless the previous one is acknowledged.
- The **final** DATA packet must contain less than a full-sized block of data to signal that it is the last.
- Using UDP, thus provides its own transport and session support through the **ACK**
- Mode of operation "netascii", "octet", or "mail"

- Syntax of TFTP
 - Format of various TFTP
- Semantics of TFTP
 - Definition of each packet type and error type
- Timing of TFTP
 - Use of block numbers, the use of timers, etc.



TFTP Timing Diagram

- Initial dst = 69
- B changes port
- A and B agree on port number
- For each DATA block the server sends an ACK
- Total of n blocks of data was transferred



SFTP

- Secure FTP

Filter: `ip.addr == 176.9.105.210` Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
49	8.348619000	130.157.158.231	176.9.105.210	SSH	210	Encrypted request packet len=144
51	8.521433000	176.9.105.210	130.157.158.231	SSH	146	Encrypted response packet len=80
52	8.521476000	130.157.158.231	176.9.105.210	TCP	66	59143 > ssh [ACK] Seq=145 Ack=81 Win=65535 Len=0 TSval=309
53	8.522225000	130.157.158.231	176.9.105.210	TCP	66	59143 > ssh [FIN, ACK] Seq=145 Ack=81 Win=65535 Len=0 TSva
54	8.694416000	176.9.105.210	130.157.158.231	TCP	66	ssh > 59143 [FIN, ACK] Seq=81 Ack=146 Win=175 Len=0 TSval=
55	8.694459000	130.157.158.231	176.9.105.210	TCP	66	59143 > ssh [ACK] Seq=146 Ack=82 Win=65535 Len=0 TSval=309
64	11.398653000	130.157.158.231	176.9.105.210	TCP	78	59151 > ssh [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=8 TSva
66	11.570921000	176.9.105.210	130.157.158.231	TCP	74	ssh > 59151 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=146
67	11.570967000	130.157.158.231	176.9.105.210	TCP	66	59151 > ssh [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=30960
70	11.746052000	176.9.105.210	130.157.158.231	SSHv2	87	Server Protocol: SSH-2.0-OpenSSH 5.3\r

It is using SSH and encrypted!

Tools

- A Web-Based FTP Client Tool: <http://net2ftp.com/>

MAIL and DNS
Client-Server
Application Examples

Electronic Mail

- One of the most widely used Internet applications
- Email software is divided into two conceptually pieces:
 - An email **interface application**
 - A mechanism for a user to **compose and edit** outgoing messages as well as **read and process** incoming email
 - A mail **transfer program** – handling the mail transfer
 - acts as a client to send a message to the mail server on the destination computer;
 - the mail server accepts incoming messages and **deposits** each in the appropriate user's **mailbox**



Electronic Mail

- The protocols used for Internet email can be divided into three broad categories

transfer
program

interface
application

Type	Description
Transfer	A protocol used to move a copy of an email message from one computer to another
Access	A protocol that allows a user to access their mailbox and to view or send email messages
Representation	A protocol that specifies the format of an email message when stored on disk

The Simple Mail Transfer Protocol (SMTP)

Type	Description
Transfer	A protocol used to move a copy of an email message from one computer to another
Access	A protocol that allows a user to access their mailbox and to view or send email messages
Representation	A protocol that specifies the format of an email message when stored on disk

- The **Simple Mail Transfer Protocol (SMTP)** is the standard protocol that a mail transfer program uses
- SMTP can be characterized as:
 - Follows a stream paradigm
 - Uses textual control messages
 - Only transfers text messages
 - Allows a sender to specify recipients' names and check each name
- SMTP can send a single message to multiple recipients
 - The protocol allows a client to **list** users and then send a single copy of a message for all users on the list
- SMTP has a restriction to send only textual content
 - **MIME** standard that allows email to include **attachments** such as graphic images or binary files
 - MIME: Multipurpose Internet Mail Extension

```
Server: 220 somewhere.com Simple Mail Transfer Service Ready
John→ Client: HELO example.edu
Server: 250 OK
Client: MAIL FROM:<John_Q_Smith@example.edu>
Server: 250 OK
Client: RCPT TO:<Mathew_Doe@somewhere.com>
Server: 550 No such user here
Client: RCPT TO:<Paul_Jones@somewhere.com>
Server: 250 OK
Client: DATA
Server: 354 Start mail input; end with <CR><LF>.<CR><LF>
Client: ...sends body of mail message, which can contain
Client: ...arbitrarily many lines of text
Client: <CR><LF>.<CR><LF>
Server: 250 OK
Client: QUIT
Server: 221 somewhere.com closing transmission channel
```

Response of the server with a code

Establishing a session!

linefeed and Carriage Return

John (on example.edu) is sending an email to Mathew and Paul on somewhere.com

No.	Time	Source	Destination	Protocol	Length	Info
7	0.732749	10.10.1.4	74.53.140.153	SMTP	63	C: EHLO GP
9	1.074123	74.53.140.153	10.10.1.4	SMTP	191	S: 250-xc90.websitewelcome.com Hello GP [122.162.143.157]
10	1.076669	10.10.1.4	74.53.140.153	SMTP	66	C: AUTH LOGIN
11	1.419021	74.53.140.153	10.10.1.4	SMTP	72	S: 334 VXNlcm5hbWU6
12	1.419595	10.10.1.4	74.53.140.153	SMTP	84	C: Z3VycGFydGFwQHhhdHJpb3RzLmlu
13	1.761484	74.53.140.153	10.10.1.4	SMTP	72	S: 334 UGFzc3dvcmQ6
14	1.762058	10.10.1.4	74.53.140.153	SMTP	72	C: cHVuamFiQDEyMw==
15	2.121738	74.53.140.153	10.10.1.4	SMTP	84	S: 235 Authentication succeeded

Server: 220 somewhere.com Simple Mail Transfer Service Ready
Client: HELO example.edu
Server: 250 OK
Client: MAIL FROM:<John_Q_Smith@example.edu>
Server: 250 OK
Client: RCPT TO:<Mathew_Doe@somewhere.com>
Server: 550 No such user here
Client: RCPT TO:<Paul_Jones@somewhere.com>
Server: 250 OK
Client: DATA
Server: 354 Start mail input; end with <CR><LF>.<CR><LF>
Client: ...sends body of mail message, which can contain
Client: ...arbitrarily many lines of text
Client: <CR><LF>.<CR><LF>
Server: 250 OK
Client: QUIT
Server: 221 somewhere.com closing transmission channel

From Command Line:
 >>>> mail -s "Hello world" you@youremailid.com

John (on example.edu) is sending an email to Math and Paul on somewhere.com

Using Sample Capture:
<http://wiki.wireshark.org/SampleCaptures?action=AttachFile&do=view&target=sntp.pcap>

Server: 220 somewhere.com Simple Mail Transfer Service Ready

Client: HELO example.edu

Server: 250 OK

Client: MAIL FROM:<John_Q_Smith@example.edu>

Server: 250 OK

Client: RCPT TO:<Mathew_Doe@somewhere.com>

Server: 550 No such user here

Client: RCPT TO:<Paul_Jones@somewhere.com>

Server: 250 OK

Client: DATA

Server: 354 S

Client: ...se

Client: ...ar

Client: <CR><

Server: 250 O

Client: QUIT

Server: 221 S

- **HELO** - introduce yourself
- **EHLO** - introduce yourself and request extended mode
- **MAIL FROM:** - specify the sender
- **RCPT TO:** - specify the recipient
- **DATA** - specify the body of the message (To, From and Subject should be the first three lines.)
- **RSET** - reset
- **QUIT** - quit the session
- **HELP** - get help on commands
- **VERFY** - verify an address
- **EXPN** - expand an address
- **VERB** - verbose

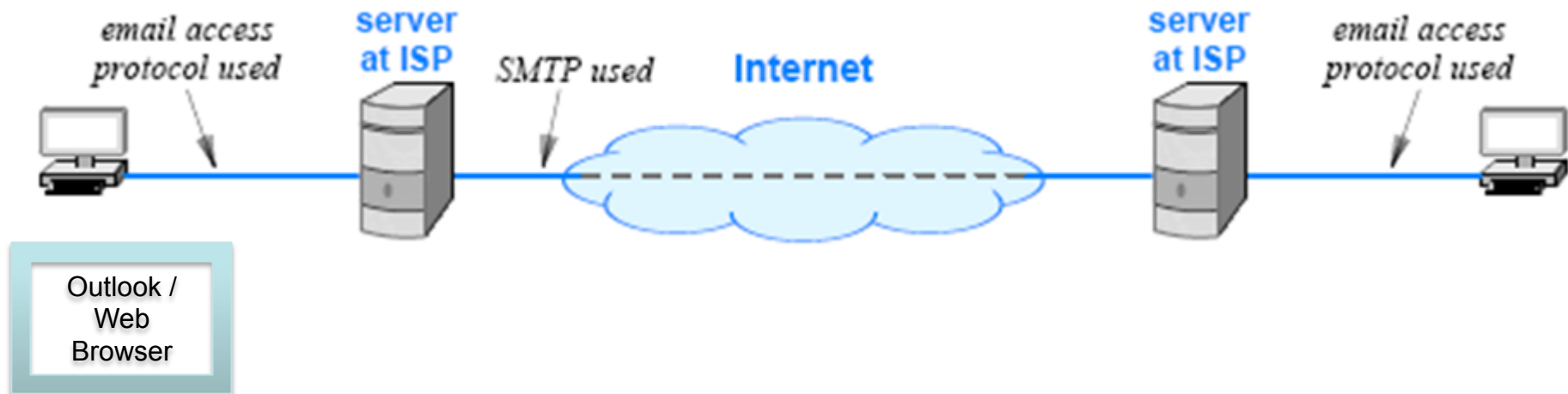
Other commands

Text commands used by SMTP application

ISPs, Mail Servers, and Mail Access

Type	Description
Transfer	A protocol used to move a copy of an email message from one computer to another
Access	A protocol that allows a user to access their mailbox and to view or send email messages
Representation	A protocol that specifies the format of an email message when stored on disk

- ISPs can offer email services
 - An ISP runs an email server and provides a mailbox for each user
 - each ISP provides interface that allows a user to access their mailbox
- Email access follows one of two forms:
 - A **special-purpose email interface** application (OUTLOOK)
 - A **web browser** that accesses an email web page

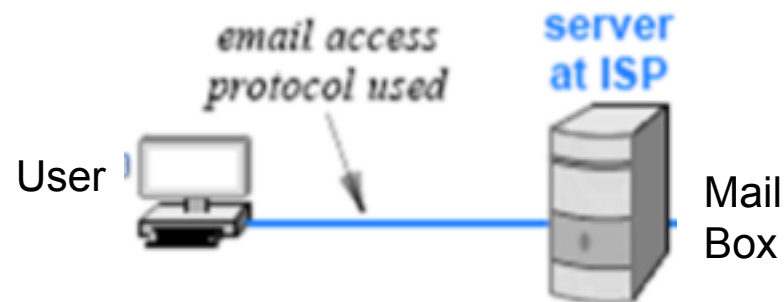


ISPs, Mail Servers, and Mail Access

- The **web browser** approach is straightforward:
 - an ISP provides a special web page that displays messages from a user's mailbox
- In case of **special purpose mail** interface
 - Using a special mail application can download an entire mailbox onto a local computer, such as a laptop

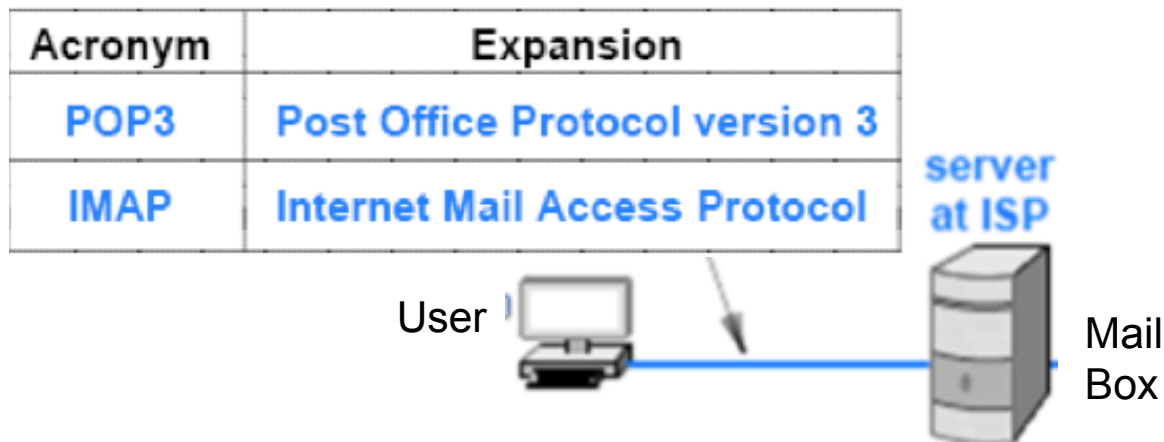
Mail Access Protocols (POP, IMAP)

- Protocols have been created that provide email **access**
- An **access protocol** is distinct from a **transfer protocol**
 - **access** only involves a single user interacting with a single mailbox
 - **transfer** protocols allow a user to send mail to other users
- Viewing a list of messages without downloading the message contents is useful
 - Especially in cases where the link between two parties is slow
 - For example, a user browsing on a cell phone may look at **headers** and delete **spam** without waiting to download the message **contents**



Mail Access Protocols (POP, IMAP)

- A variety of mechanisms available for email access
 - Some ISPs provide free email access software to their subscribers
 - In addition, two standard email access protocols have been created
- Two access protocols differ in many details
 - In particular, each provides its own **authentication** mechanism that a user follows to **identify** themselves



POP3

- Post Office Protocol version 3
- When you **check your e-mail**, your e-mail client connects to the POP3 server using **port 110**.
 - It allows you to have a collection of messages stored in a text file on the server.
- The POP3 server understands a very simple set of **text commands**. Here are the most common commands:
 - **USER** - enter your user ID
 - **PASS** - enter your password
 - **QUIT** - quit the POP3 server
 - **LIST** - list the messages and their size
 - **RETR** - retrieve a message, pass it a message number
 - **DELE** - delete a message, pass it a message number
 - **TOP** - show the top x lines of a message, pass it a message number and the number of lines

IMAP

- Many users want to do more with their e-mail, and they want their e-mail to **remain** on the server.
 - The POP3 protocol assumes that there is only one client connected to the mailbox.
- IMAP (Internet Mail Access Protocol) is a **more advanced** protocol that solves these problems.
 - the IMAP protocol allows simultaneous access by **multiple** clients.
 - IMAP is suitable for you if your mailbox is about to be managed by multiple users.
 - e-mail client connects to the IMAP server using **port 143**
 - With IMAP, your mail **stays** on the e-mail server.
 - You can organize your mail into folders, and **all the folders** live on the server as well.

Example: MS Exchange

- Works with IMAP and POP
 - With POP, you can only access your Inbox
 - IMAP allows you to access all of your folders
- Note:
 - POP is a protocol for **receiving** messages only,
 - SMTP is the protocol used for **sending** them

Email Representation Standards (RFC2822, MIME) – RFC3822

- Two important email representation standards exist:
 - RFC (Request For Comments) 2822 Mail Message Format
 - Multi-purpose Internet Mail Extensions (MIME)
- RFC 2822 Mail Message Format:
 - a mail message is represented as a text file and consists of
 - a header section
 - a blank line
 - and a body
 - Header lines each have the form:

Keyword: information

 - where the set of **keywords** is defined to include From:, To:, Subject:, Cc:



To:, Subject:, Cc:

Email Representation Standards - MIME

- Multi-purpose Internet Mail Extensions (MIME)
- The MIME standard **extends** the functionality of email to allow the transfer of **non-text** data in a message
- The **Base64 encoding**(^{*}) standard is most popular, but MIME does not restrict encoding to a specific form
 - MIME permits a sender/receiver to **choose** a convenient **encoding**
 - the sender includes additional lines in the header to specify encoding used
- Encoding different message parts differently:
 - A user can send a plain text message and attach a graphic image, a spreadsheet, and an audio clip, each with their own encoding
 - MIME allows a sender to divide a message into **several parts** and to specify an encoding for each part independently

(*) Check this web site for more information on Base64 coding: <http://www.motobit.com/util/base64-decoder-encoder.asp>

Email Representation Standards - MIME

- MIME adds **two** lines to an email header
 - one to declare that MIME has been used to create the message
 - another to specify how MIME information is included in the body
 - For example, the header lines:

MIME-Version: 1.0

*Content-Type: Multipart/Mixed; Boundary=**MIME_separator***

- When MIME is used to send a standard text message

Content-Type: text/plain

- MIME is **backward compatible** with email systems that do not understand the MIME standard or encoding
 - such systems have no way of extracting **non-text** attachments
 - they treat the body as a single block of text

Review

Algorithm 4.3

Given:

Email communication from one user to another.

Provide:

Transmission of a message to the intended recipient.

Method:

User invokes interface application and generates an email message for user *x@destination.com*;

User's email interface program queues message for transfer;

Mail transfer program on user's computer examines the outgoing mail queue, and finds message;

Mail transfer program opens connection to *destination.com*;

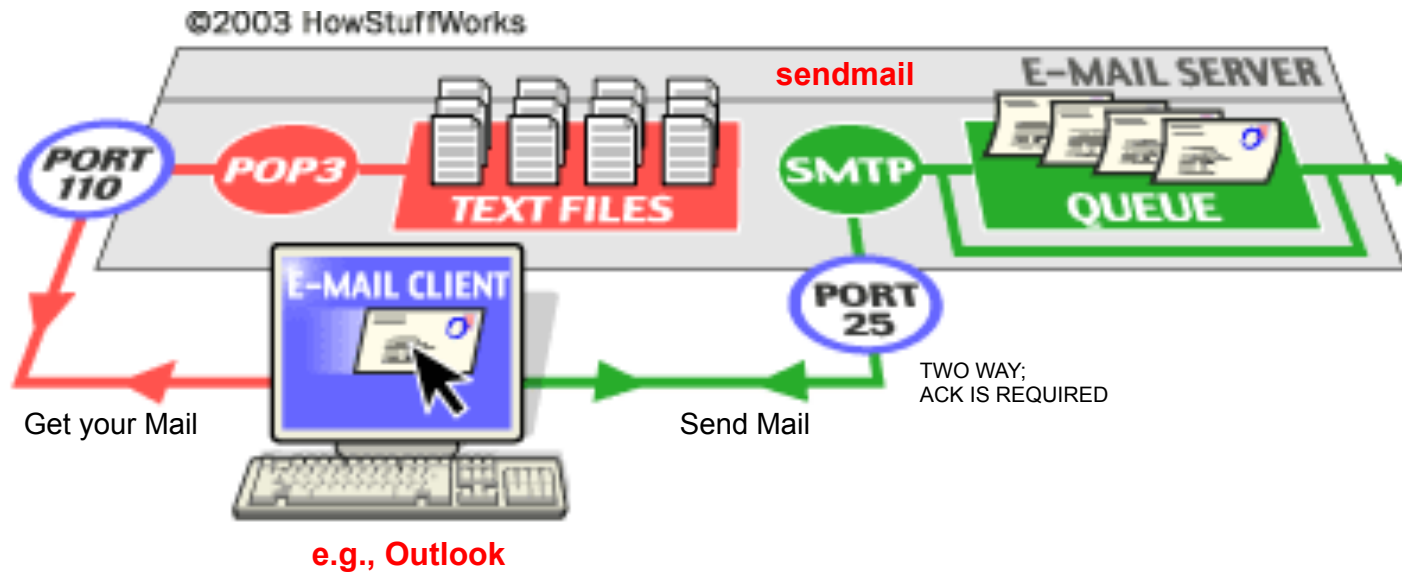
Mail transfer program uses SMTP to transfer the message;

Mail transfer program closes connection;

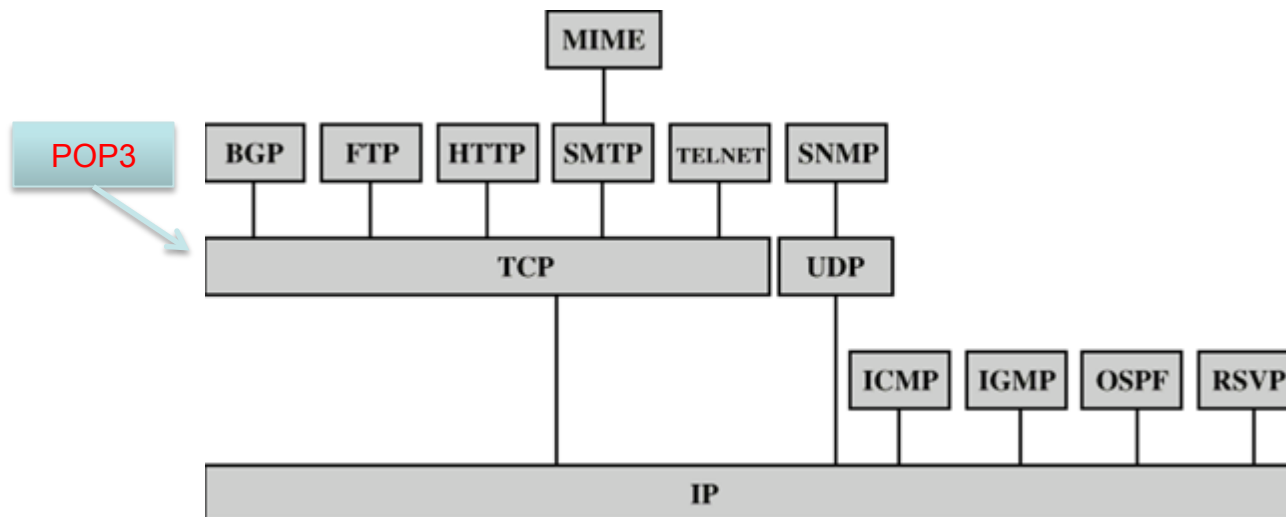
Mail server on *destination.com* receives message and places a copy in user *x*'s mailbox;

User *x* on *destination.com* runs mail interface program, which displays the user's mailbox, including the new message;





SMTP server to handle the sending
 The SMTP server on most machines uses a program called **sendmail** to do the actual sending



Simple Test

SuperTool ^{Beta}

Command: Or try: mail.sonomaesdep.host-ed.me

a:mail.sonoma.edu a

Type	Domain Name	IP Address	TTL
A	mail.sonoma.edu	130.157.18.46	5 min
reverse lookup	smtp diag	port scan	blacklist

Reported by **ns.sonoma.edu** on Thursday, September 09, 2010 at **12:08:47 PM** (GMT-5)

a:mail.sonoma.edu a

Type	Domain Name	IP Address	TTL
A	mail.sonoma.edu	130.157.18.46	5 min
reverse lookup	smtp diag	port scan	blacklist

Reported by **ns.sonoma.edu** on Thursday, September 09, 2010 at **12:04:29 PM** (GMT-5)

← Click on these!

<http://www.mxtoolbox.com/SuperTool.aspx?action=smtp%3asonoma.edu>

Telnet to a mail server: <http://www.activexperts.com/activemail/telnet/>

Setting TELNET: <http://www.tech-recipes.com/rx/4230/windows-7-install-the-telnet-client/>

Sending MAIL using TELNET

Telnet to SMTP Server

telnet mail.monitortools.com 25

helo dell01

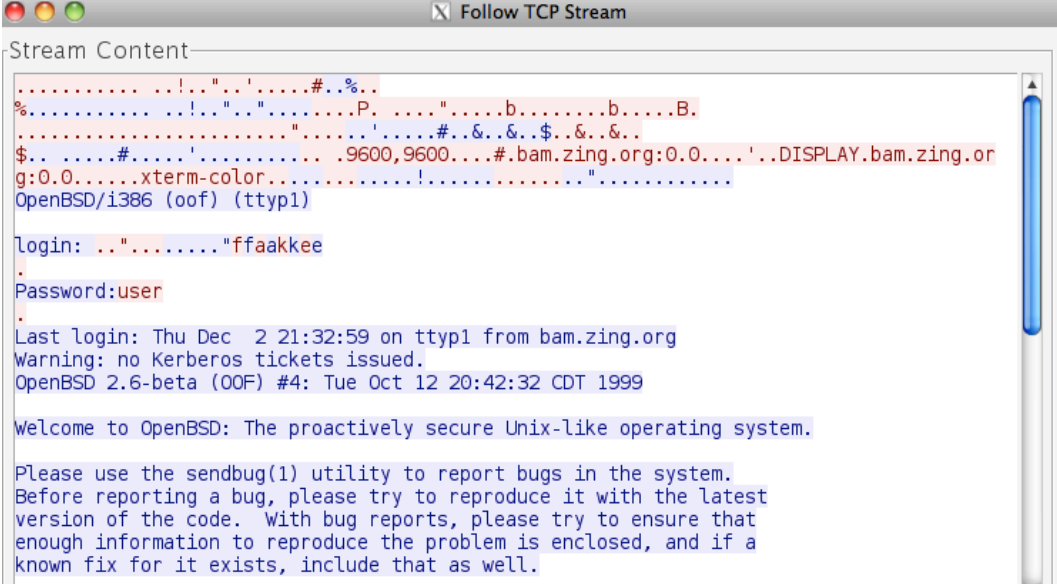
250 HELO 217.120.215.201, How can I help you?

For more information go to:

<http://www.activexperts.com/activemail/telnet/>

Not secure! Clear text!
Sample Captured:

<http://wiki.wireshark.org/SampleCaptures?action=AttachFile&do=view&target=telnet-raw.pcap>



```
Stream Content
.....!."'.#.%..
%......!."".P:.....b.....b.....B.
....."'.#.&.&$.&.&.
$.#.....'.....9600,9600...#.bam.zing.org:0.0....'.DISPLAY.bam.zing.or
g:0.0.....xterm-color.....!.....".....
OpenBSD/i386 (oof) (tty1)

Login: .."....."ffaakkee
.
Password:user
.
Last login: Thu Dec  2 21:32:59 on tty1 from bam.zing.org
Warning: no Kerberos tickets issued.
OpenBSD 2.6-beta (00F) #4: Tue Oct 12 20:42:32 CDT 1999

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
version of the code.  With bug reports, please try to ensure that
enough information to reproduce the problem is enclosed, and if a
known fix for it exists, include that as well.
```

Testing POP3

Test your SSL POP3 Mail Server

Mail Server Test

POP3 Server
Enter an IP address or a fully qualified hostname

Use SSL? Yes No

Username
Optional - will attempt to login with this username

Password
Optional. We strongly recommend that you do not use a real email account when testing password authentication.

```
Resolving hostname...
Connecting...
S:+OK Hello there.
C: QUIT
S:+OK Better luck next time.
POP3 test completed successfully.
```

http://www.wormly.com/test_pop3_mail_server

DNS Server

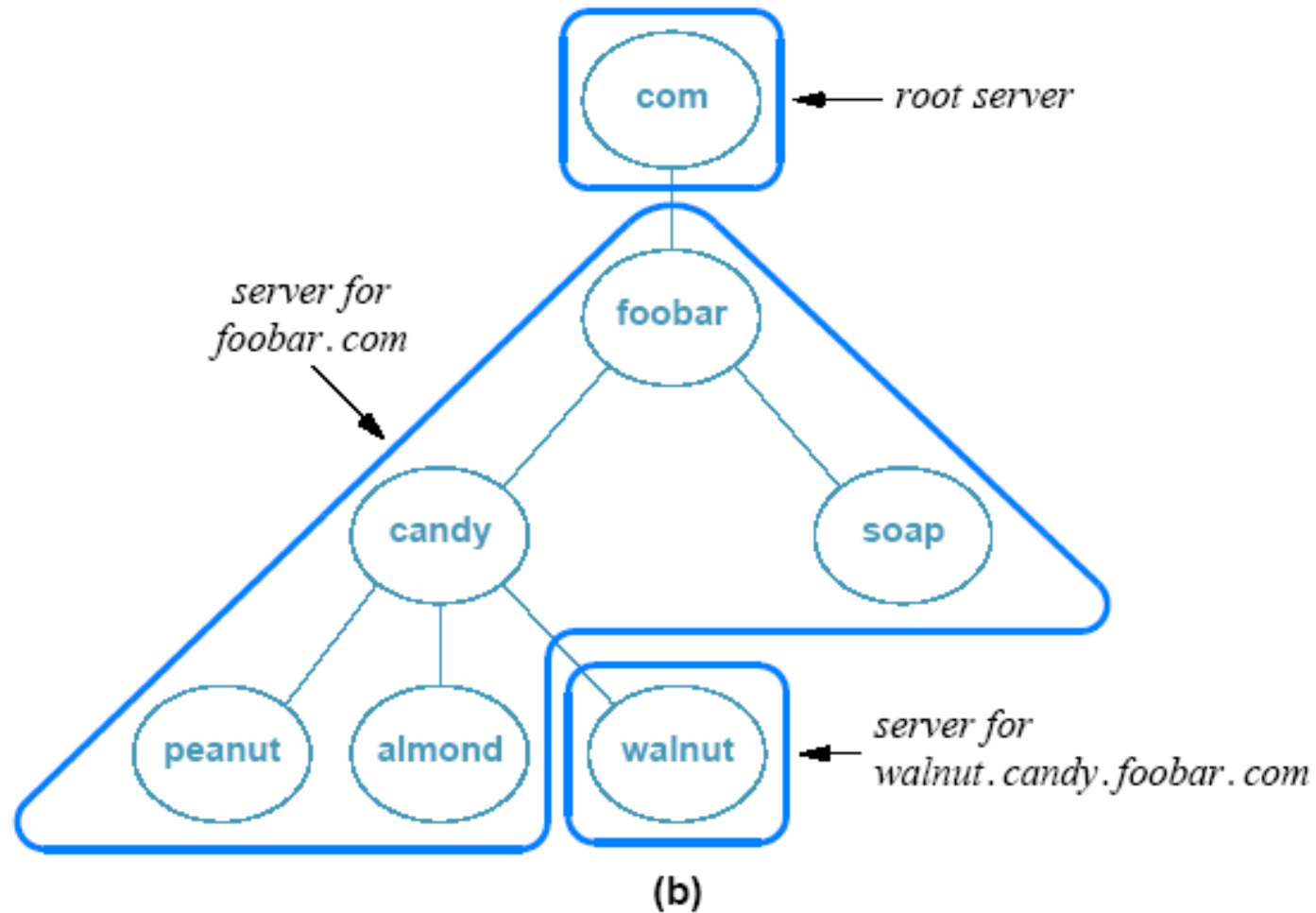
Domain Name System (DNS)

- **DNS** provides a service that maps **human-readable** symbolic names to computer addresses
 - Browsers, mail software, and most other Internet applications use the DNS
 - an example of client-server interaction

Domain Name System (DNS)

- Syntactically, each name consists of a sequence of **alpha-numeric segments** separated by **periods**
 - For example, a computer can have the following name:
mordred.es.sonoma.edu
 - A computer at Cisco, Inc. has the domain name:
anakin.cisco.com
- Domain names are **hierarchical**, with the most **significant part** of the name on the right

The DNS Hierarchy and Server Model



A hypothetical DNS hierarchy and two possible assignments of names to servers.

Domain Namespaces

- The **root level** domain is "."
 - Significant in creating DNS files
- **Top-level domains** include com, org, fr
- **Second-level** domains are often owned by companies and individuals
 - microsoft.com, ssu.edu
- A **subdomain** is a further division of a second-level domain
 - For ssu.edu, there is ssu.edu.gh/
- DNS does specify values for the most significant segment, which is called a **top-level domain** (TLD)
 - Controlled by the **Internet Corporation for Assigned Names and Numbers** (ICANN)
 - ICANN designates one or more **domain registrars** to administer a given top-level domain and approve specific names

Example top-level domains and the group to which each is assigned

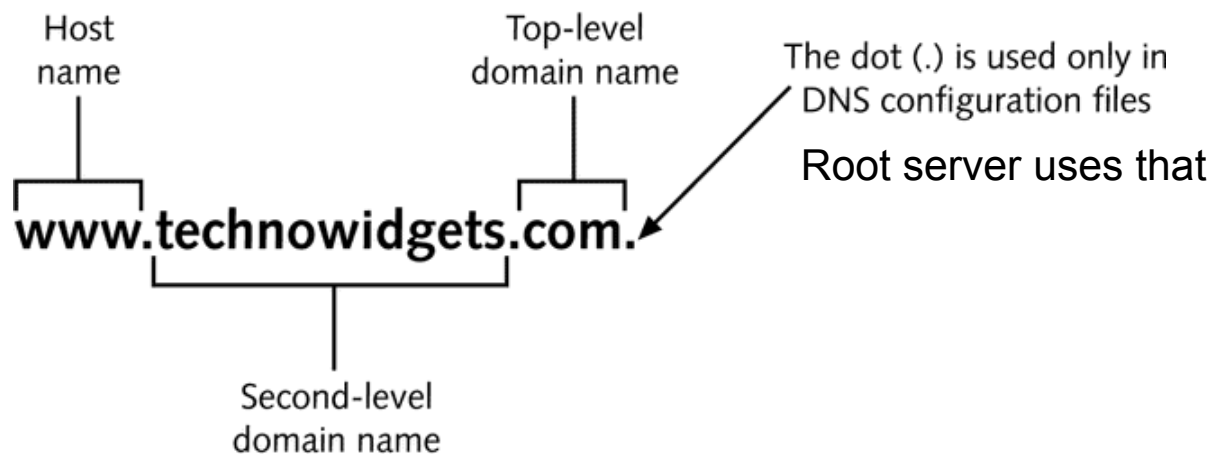
Domain Name	Assigned To
aero	Air transport industry
arpa	Infrastructure domain
asia	For or about Asia
biz	Businesses
com	Commercial organizations
coop	Cooperative associations
edu	Educational institutions
gov	United States Government
info	Information
int	International treaty organizations
jobs	Human resource managers
mil	United States military
mobi	Mobile content providers
museum	Museums
name	Individuals
net	Major network support centers
org	Non-commercial organizations
pro	Credentialed professionals
travel	Travel and tourism
<i>country code</i>	A sovereign nation

Domain Namespaces

- Second-level domains, such as [ssu.edu](#) have control over naming within their domain
 - Create hosts such as [www](#), [ftp](#)
 - A name such as [www.ssu.edu](#) is a fully qualified domain name (FQDN)
- We could create [subdomains](#) such as [phx](#)
 - [www.phx.ssu.edu](#)

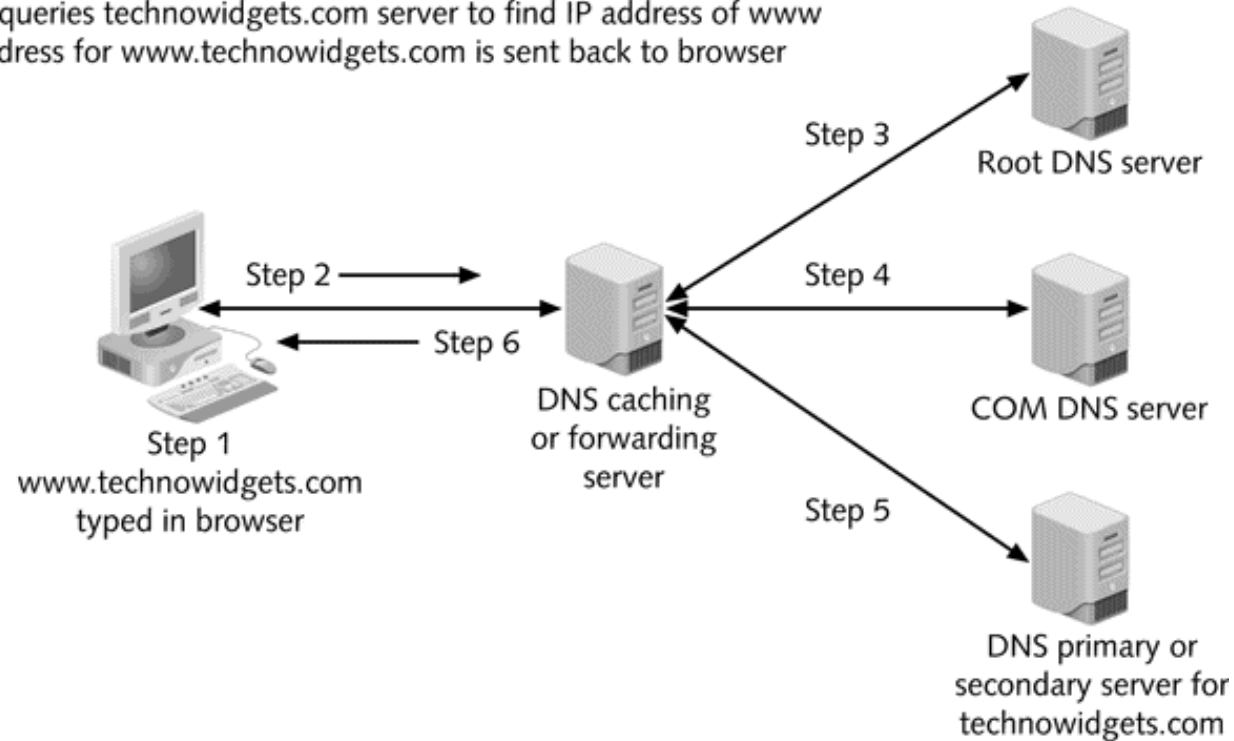
Dissecting URLs

- The first portion of a URL is typically a host name
- Typically different from the name of the computer
- Many hosts can be associated with the same Web server



How DNS Works

1. User types www.technowidgets.com in browser
2. Browser queries DNS server to get IP address
3. DNS server queries root server to find IP address of COM server
4. DNS server queries COM server to find IP address of technowidgets.com server
5. DNS queries technowidgets.com server to find IP address of www
6. IP address for www.technowidgets.com is sent back to browser



Primary and **secondary** servers store the host names used on the Internet
Caching and **forwarding** servers search the Internet for host names

Algorithm 4.4

Given:

A request message from a DNS name resolver

Provide:

A response message that contains the address

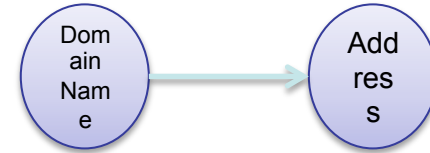
Method:

```
Extract the name,  $N$ , from the request
if ( server is an authority for  $N$  ) {
    Form and send a response to the requester;
else if ( answer for  $N$  is in the cache ) {
    Form and send a response to the requester;
else { /* Need to look up an answer */
    if ( authority server for  $N$  is known ) {
        Send request to authority server;
    } else {
        Send request to root server;
    }
    Receive response and place in cache;
    Form and send a response to the requester;
}
```

Steps a DNS server takes to resolve a name.

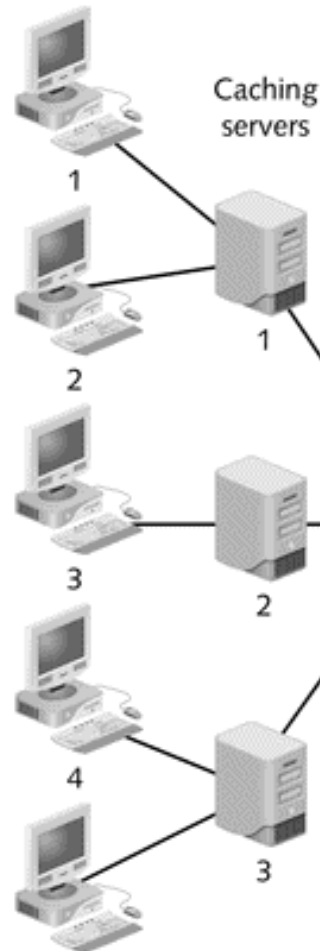
DNS Components

- The translation of a domain name into an address is called **name resolution**
 - and the name is said to be **resolved** to an address
- Name server – also known as **DNS server**
 - supports **name-to-address** and **address-to-name** resolution
- Name resolver – also called **DNS client**
 - Can **contact DNS server** to lookup name
 - Used by browsers, e-mail clients, and client utilities such as ping
 - Software to perform the translation is known as a **name resolver** (or simply **resolver**)
 - In the socket API, for example, the resolver is invoked by calling function *gethostbyname*
 - The resolver becomes a client by contacting a DNS server
 - DNS server returns an answer to the caller
- The resolver forms a **DNS request** message
 - sends the message to the local server
 - waits for the server to send a **DNS reply** message for the answer
- A resolver can choose to use either the **stream** or **message paradigm** when communicating with a DNS server
 - most resolvers are configured to use a message paradigm because it imposes less overhead for a small request



Caching and Forwarding Servers

Workstations in organization



The value of a forwarding server

1. Workstation 1 enters www.technowidgets.com into a browser
2. Caching server 1 asks forwarding server to resolve it
3. Forwarding server resolves it and caches (saves) it
4. Caching server 1 caches it as well
5. The IP address is returned to workstation 1 and the page is requested
6. Workstation 2 enters www.technowidgets.com into a browser
7. It is resolved at caching server 1
8. Workstation 3 enters www.technowidgets.com into a browser
9. Caching server 2 cannot resolve it but the forwarding server can

Without a forwarding server, each caching server would have to access the Internet

Caching Server

- Resolves host names
- Caches (saves) the results
- Automatically installed when DNS is installed
- No configuration necessary

Forwarding Server

- Caching server that has access to the Internet and forwards traffic from other caching servers

Zones

- A zone is a part of the **domain namespace**
- For a domain as small as technowidgets.com, the domain name represents a single zone
- For large organizations (such as IBM), subdomains can be divided into separately maintained zones
 - Each zone typically has a separate DNS
- Zones must be contiguous
 - [admin.ssu.edu](#) can be combined with [ssu.edu](#)
 - [admin.ssu.edu](#) cannot be combined with [student.ssu.edu](#)
- There must be one primary/secondary DNS server in each zone
- Each zone can have multiple secondary DNS servers for load balancing, failure, etc.

Zone File Configuration

- Forward Lookup
 - These zones contain entries that map **names to IP addresses**
- Reverse Lookup
 - These zones contain entries that map **IP addresses to names**
- There are two primary files
 - Forward lookup is described by **named.technowidgets.com** file
 - It has the host names and how to handle e-mail
 - Reverse lookup is described by **named.0.168.192** file
 - Can be necessary for e-mail (SMTP) and security programs

Comparing Forward & Reverse Lookup

Forward lookup: /var/named.technowidgets.com

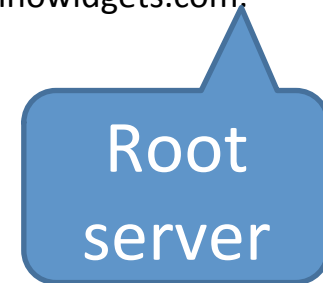
```
$TTL      86400
@ IN SOA  web1.technowidgets.com.
      admn.technowidgets.com. (
                2002072100 ; Serial
                28800   ; Refresh
                14400   ; Retry
                3600000 ; Expire
                86400 ) ; Minimum

      IN NS  web1
      IN A   192.168.0.100
      IN MX  10
      mail.technowidgets.com.
web1 IN A     192.168.0.100
www  IN CNAME web1
research IN A   192.168.0.150
      IN MX  10  mail
mail IN A     192.168.0.200
```

Reverse lookup: named.0.168.192

```
$TTL 86400
@ IN SOA web1.technowidgets.com.
      admn.technowidgets.com. (
                2002072100 ; Serial
                28800   ; Refresh
                14400   ; Retry
                3600000 ; Expire
                86400 ) ; Minimum

      IN NS  web1
100 IN PTR  web1.technowidgets.com.
150   IN PTR  research.technowidgets.com.
200   IN PTR  mail.technowidgets.com.
```



Types of DNS Entries

- Each entry in a DNS database consists of three items:
 - a domain name
 - a record type
 - The record type specifies how the value is to be interpreted
 - a value
- A query sent to a DNS server specifies both a domain name and a type
 - the server only returns a binding that matches the type of the query

Forward lookup: /var/named.technowidgets.com

```
$TTL      86400
@   IN   SOA  web1.technowidgets.com.
      admn.technowidgets.com. (
```

```
      2002072100 ; Serial
      28800      ; Refresh
      14400      ; Retry
      3600000    ; Expire
      86400 )    ; Minimum
```

```
      IN   NS   web1
```

```
      IN   A   192.168.0.100
```

```
      IN   MX  10
```

```
      mail.technowidgets.com.
```

```
web1   IN   A   192.168.0.100
```

```
www    IN   CNAME web1
```

```
research IN   A   192.168.0.150
```

```
      IN   MX  10  mail
```

```
mail   IN   A   192.168.0.200
```

Common DNS Record Types

DNS record	Function
Address (A)	Associates a host to an IP address.
Canonical name (CNAME)	Creates an alias for a specified host.
Internet (IN)	Identifies Internet records; precedes most DNS record entries.
Mail Exchanger (MX)	Identifies a server used for processing and delivering e-mail for the domain.
Name server (NS)	Identifies DNS servers for the DNS domain.
Pointer (PTR)	Performs reverse DNS lookups. Resolves an IP address to a host name.
Start of Authority (SOA)	Identifies the DNS server with the most current information for the DNS domain.

Testing DNS

- Use Use ping, nslookup, and dig to troubleshoot DNS
 - nslookup sonoma.edu // where is YOU DNS server

```
C:\Users\farahman>nslookup sonoma.edu
Server: ns.sonoma.edu
Address: 130.157.5.100

Name: sonoma.edu
Address: 130.157.5.226

C:\Users\farahman>dig sonoma.edu
'dig' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\farahman>nslookup 130.157.5.100
Server: ns1.sonoma.edu
Address: 130.157.5.100

Name: ns.sonoma.edu
Address: 130.157.5.100
```

ns1 is the alias; Sonoma has an DNS server

dig osnoma.edu

Try: dig sonoma.edu ns

About dig itself

```
; <<>> DiG 9.6.0-APPLE-P2 <<>> sonoma.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43351
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 4
```

```
;; QUESTION SECTION:
```

```
;sonoma.edu.                IN      A
```

This is what you asked & this is the answer:

```
;; ANSWER SECTION:
```

```
sonoma.edu.                3600    IN      A      130.157.10.66
```

what DNS servers can provide an authoritative answer to our query

```
;; AUTHORITY SECTION:
```

```
sonoma.edu.                86400   IN      NS     ns3.csu.net.
sonoma.edu.                86400   IN      NS     authns-a.sonoma.edu.
sonoma.edu.                86400   IN      NS     authns-b.sonoma.edu.
sonoma.edu.                86400   IN      NS     authns-c.sonoma.edu.
```

There are FOUR name servers

```
;; ADDITIONAL SECTION:
```


```
ns3.csu.net.               21079   IN      A      137.145.204.10
authns-a.sonoma.edu.       86400   IN      A      130.157.3.80
authns-b.sonoma.edu.       86400   IN      A      130.157.3.81
authns-c.sonoma.edu.       86400   IN      A      130.157.3.82
```

...and these are their IP addresses!

```
;; Query time: 6 msec
;; SERVER: 130.157.27.50#53(130.157.27.50)
;; WHEN: Thu Sep 15 15:39:23 2011
;; MSG SIZE rcvd: 202
```

Statistics about query – remove by using +short

IP Address and Location

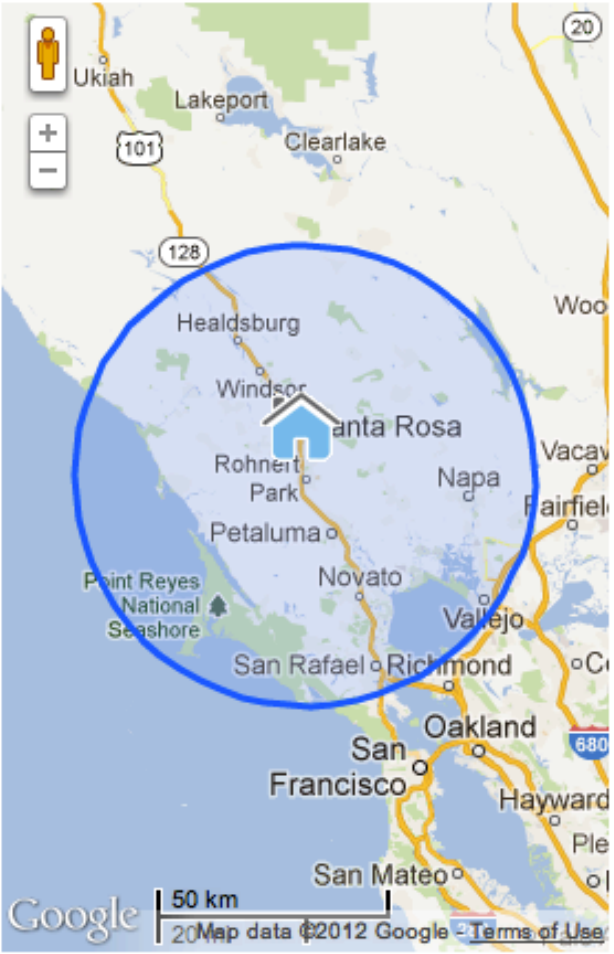
IP Address:	130.157.3.80
IP Host:	authns-a.sonoma.edu
Find IP Address Location for 'My IP' 130.157.3.80	
Continent:	North America (NA)
Country:	United States  (US)
State:	California
City:	Rohnert Park
Postal Code:	94928
Area Code:	707
Metro Code:	807
ISP:	Sonoma State University
Organization:	Sonoma State University
Time zone:	America/Los_Angeles
IP Address Lookup related for 'My IP' 130.157.3.80	
Continent Lat/Lon:	46.07305 / -100.546
Country Lat/Lon:	38 / -98
City Lat/Lon:	(38.3433) / (-122.7041)
IP Language:	English
IP Currency:	United States dollar(\$) (USD)
IDD Code:	+1

Ads by Google

[IP Address Lookup](#)

[Track an IP Address](#)

[Trace IP Address](#)



Class Exercise:

- www.whois.net → Give it a domain name / who it is
 - You can check who owns the domain name.
- www.arin.net/whois → IP Address (requires registration)
- <http://www.dnsstuff.com/> → Very interesting
- <http://www.ip-address.org/lookup/ip-locator.php>
→ Maps the IP address

Where is Richland College?