COMS3200 Study Notes

Brae

Semester 1, 2018

1 Internet?

- Collection of billions of connected devices.
- Connected via `communication links` such as fiber, copper, radio and satellites.
- Controlled by `packet switches` such as routers and switches.
- Standardized by `protocols` such as TCP, IP, HTTP, Skype, 802.11
- Standards are made by organizations such as `RFC: Request for comments and IETF: Internet Engineering Task Force`

> Actually a network of networks (ISPs connected together)

2 Protocol?

`Protocols` define a guide for messages (packets) sent and received between network entities by defining the:

- `format` of messages
- `order` of messages
- `actions` taken when messages are transmitted or received

3 Network Edge/Core

`Network Edges` are host devices i.e. client machines or servers.

`Network Cores` are interconnected routers.

> `Frequency division multiplexing`: different channels transmitted in different frequency bands

4 Application Layer

The Application Layer provides the interface between the end-user and network communication.

Implementation aspects of network protocols

- transport-layer service models
- client-server paradigm

5 Network Applications

Network applications run on **different end systems** (network edges) and **communicate over the network.**

Network applications **do not** run on network cores.

Network applications allow for **rapid app development and propagation.**

6 Network Architectures

- Client-server
- Peer-to-peer (P2P)

> **Client-server Architecture** is the classical architecture consisting of communication between **multiple clients** and a **singular server.**
> The server is **always-on** with a **fixed address** that **can be scaled** to multiple devices.
> Clients communicate with directly with the server and **do not need to be always on or have a fixed address.** Clients **do not communicate with each other.**

> **Peer-to-peer Architecture** is a form of network communication where clients (now peers) do not connect to an always-on server and instead **communicate directly with each other.**
> Peers request service from other peer and provide service in return to other peers. Think torrents.
> Peers are **intermittently connected and can change addresses.**

7 Processes

A **Process** is a program running within a host.

Inter-process communication is two processes communicating on the same host.

Messages are exchanged by processes communicating on different hosts.

> **Client process:** initiates communcation
> **Server process:** waits for communcation from clients

> **P2P Applications have both client and server processes**

8 Sockets

Processes send and receive messages to and from sockets.

Sockets are connections between host devices.

9 Addressing Processes

Processes require **identifiers** so that messages can be sent back to the correct process.

Each **host** has a **32-bit IP address.**

A host can have **multiple processes** so IP addresses are combined with **port numbers** as **identifiers.**

10 App-Layer Protocol

App-Layer Protocol defines:

- **type of message** e.g. request, response

- **message syntax:** message fields and encoding
- **message semantics:** meaning of the fields
- **rules:** how processes should send/receive messages

Open protocols:

- defined in **RFCs**
- allows for **interoperability**

Proprietary protocols:

- normally implemented for a specific proprietary application

11 Transport Service Considerations

Data Integrity Reliability of data to reach the destination. Some applications require all data to reach the destination.

Timing Speed transportation takes. Some applications require fast transportation to work well.

Throughput Amount of data in a transfer. Some applications require large throughput while others require minimal throughput.

12 TCP & UDP

TCP

- **reliable** transport protocol
- **flow control** prevent overwhelming receiver
- **congestion control** prevent overwhelming network
- **no** timing, minimum throughput guarantee, security
- **setup required** connections need to be established

UDP

- **unreliable** transport protocol
- **no** flow control, congestion control, timing, throughput guarantee, security, or connection setup

13 Secure TCP

TCP & UCP connections have **no encryption.**

SSL connections are encrypted TCP connections.

SSL connections increase **data integrity** and offer **end-point authentication.**

SSL is an application layer protocol. Applications use SSL libraries.

14 Secure TCP

TCP & UCP connections have **no encryption.**

SSL connections are encrypted TCP connections.

SSL connections increase **data integrity** and offer **end-point authentication.**

SSL is an application layer protocol. Applications use SSL libraries.

15 HTTP: Hypertext Transfer Protocol

Application protocol for websites.

Client requests web objects from server.

Server responds with web objects when requested.

HTTP uses TCP connections (port 80)

HTTP is a **stateless protocol**. Server does not maintain client information.

Persistent HTTP allows **multiple objects** per connection.

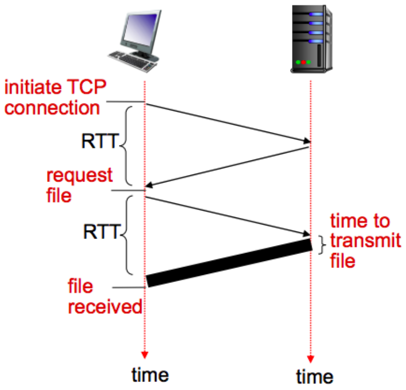
16 Non-persistent HTTP

Non-persistent HTTP restricts **one object** per connection.

1. Client makes TCP connection to port 80 using a socket
2. Server accepts incoming TCP connection
3. Client sends request message over socket to access a resource
4. Server responds with requested resource
5. Server closes connection
6. Client receives requested resource

17 Non-persistent HTTP Time

RTT: time for a packet to travel from a client to a server and back.



Non-persistent Response Time = initial RTT + request RTT + file transmission time

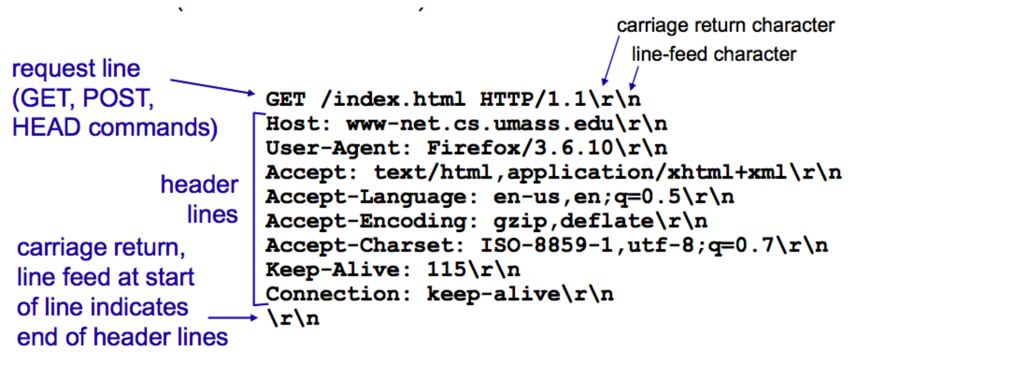
18 Persistent HTTP

Non-persistent HTTP requires 2 RTTs + OS overheard for each object.

Persistent HTTP leaves connections open allowing for as little as 1 RTT per object.

19 HTTP Messages

There are **request** and **response** HTTP messages



carriage return character

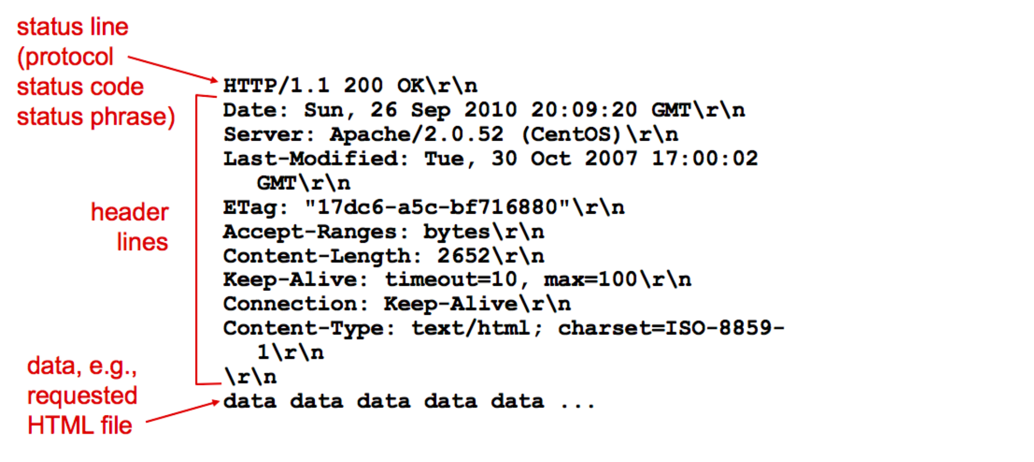
line-feed character

request line (GET, POST, HEAD commands)

header lines

carriage return, line feed at start of line indicates end of header lines

```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```



status line (protocol status code status phrase)

header lines

data, e.g., requested HTML file

```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02
   GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-
   1\r\n
\r\n
data data data data data ...
```

Method Types

HTTP 1.0 Methods: GET, POST, HEAD

HEAD asks the server to not send back the requested object.

HTTP 1.1 Methods: GET, POST, HEAD, PUT, DELETE

PUT uploads object to the given URL

DELETE deletes object at given URL

Response Codes

- 200 OK
- 301 Moved Permanately
- 400 Bad Request
- 404 Not Found
- 505 HTTP Version Not Supported

20 Cookies and Caches and Conditional GETs

... oh my!

Skipping for now

21 Electronic Mail

User Agents

A mail reader. Composing, editing, reading. Emails are temporarily stored on the client.

Mail Servers

Mailbox for each user which stores incoming emails

Message queue of outgoing emails

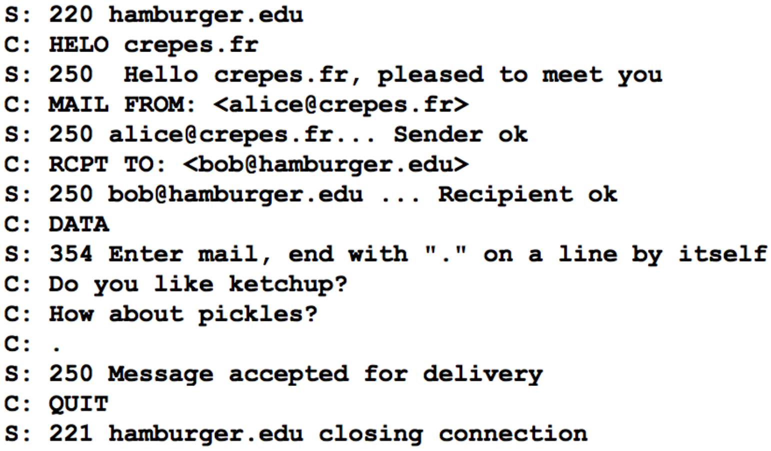
22 SMTP

Uses TCP to send emails on port 25.
Direct transporation - sending server to receiving server
Three phases of transfer

1. handshaking (greeting)
2. transfer of messages
3. closure

SMTP is a persistent connection and requires 7-bit ascii messages.

23 SMTP Transaction



```
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
```

24 Mail Message Format

Mail messages consist of a **header** containing To, From, Subject, etc fields.
Also contains a **Body** section in ASCII characters only.

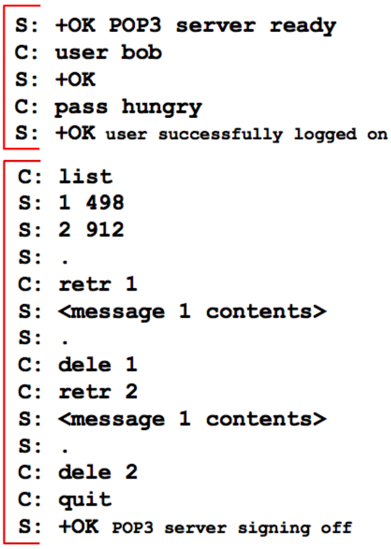
25 POP Protocol

Post Office Protocol [RFC 1939]: authorization and download

Protocol has two phases: **authentication** and **transaction**.

Commands: user, pass, list, retr, dele, quit

Responses: +OK, -ERR



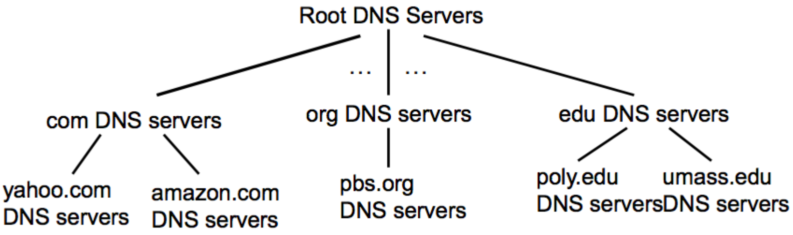
```
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on

C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

26 IMAP Protocol

Internet Mail Access Protocol [RFC 1730]
Keeps all messages on the **server**
Adds **folders** to store messages
Holds user states between sessions

27 Domain Name System

Distributed, hierarchical database of **name servers** to map hostnames to IP addresses.



28 Name Servers

Root Name Servers 13 root name servers worldwide map to TLD servers.

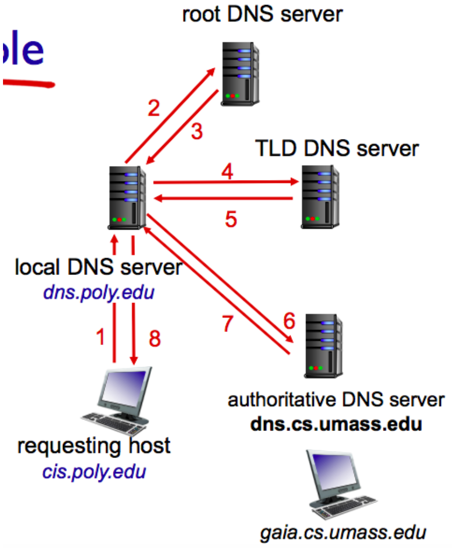
TLD Name Servers Responsible for mapping top-level domains such as com, org, edu, net to authoritative name servers.

Authoritative Name Servers An organizations name server, handles all domains below top-level.

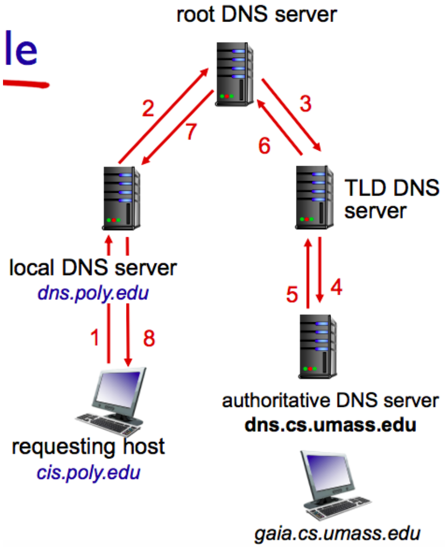
Local Name Servers The default name server, stores local caches of DNS records for fast lookup. If a record cannot be founds, queries hierarchy.

29 DNS Resolution

Iterative Resolution Local server queries each server in the hierarchy for where to look next.



Recursive Resolution Hierachy name servers recursively query lower name servers.



DNS records are cached by name servers for the particular records TTL.

30 DNS Records

DNS records are stored in a database as **resource records (RR)**

Format: (name, value, type, ttl)

Type A: name=hostname, value=ip address **Type NS:** name=domain, value=authoritative name server for this domain **Type CNAME:** name=alias hostname, value=real hostname **Type MX:** name=hostname, value=mailserver

31 DNS Attacks

1. bombard root name servers
2. bombard tld name servers
3. man-in-the-middle attacks: intercept queries
4. DNS poisoning: send bogus information to DNS servers
5. send queries with spoofed source address: target IP