COMS3200 Study Notes

Brae

June 4, 2018

Semester 1, 2018

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`

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)

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

Network Edge/Core

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

`Network Cores` are interconnected routers.

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

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

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.**

Network Architectures

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

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.**

Network Architectures

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

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.**

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.

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

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

Sockets

Processes send and receive messages to and from sockets.

Sockets are connections between host devices.

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.**

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

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**

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

Transport Service Considerations

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

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.

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.

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

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.