

# Operating Systems

## 16. Networking

Paul Krzyzanowski  
Rutgers University  
Spring 2015

4/6/2015 © 2014-2015 Paul Krzyzanowski 1

## Local Area Network (LAN)

LAN = communications network

- Small area (building, set of buildings)
- Same, sometimes shared, transmission medium
- High data rate: 1 Mbps – 100 Gbps
- Low latency
- Devices are peers
  - any device can initiate a data transfer with any other device

Most elements on a LAN are **workstations**

- endpoints on a LAN are called **nodes**

4/6/2015 © 2014-2015 Paul Krzyzanowski 2

## Connecting nodes to LANs

network



?

computer



4/6/2015 © 2014-2015 Paul Krzyzanowski 3

## Connecting nodes to LANs

network





computer



Adapter

- expansion slot (PCIx, USB dongle)
- usually integrated onto main board

Network adapters are referred to as **Network Interface Controllers (NICs) or adapters**  
(or **Network Interface Component, Network Interface Card**)

4/6/2015 © 2014-2015 Paul Krzyzanowski 4

## Multiple Access Protocols

Ways of enabling multiple devices to share one network

Three approaches

1. **Channel partitioning**
  - **Time Division Multiplexing (TDM)**
    - each node gets a time slot
  - **Frequency Division Multiplexing (FDM)**
    - each channel gets a frequency band

*Bandwidth allocated even if nothing to transmit!*
2. **Taking turns**
  - Polling protocol – master polls nodes in sequence
  - Token passing protocol – node needs a token to transmit

*Fails badly, Complex*
3. **Random access**
  - No scheduled time slots
  - Statistical multiplexing
  - Retransmit if there's a collision

*Sounds like it shouldn't work well ... but it does!*

4/6/2015 © 2014-2015 Paul Krzyzanowski 5

## Modes of connection

**Circuit-switching (virtual circuit)**

- Dedicated path (route) – established at setup
- Guaranteed (fixed) bandwidth – routers commit to resources
- Typically fixed-length packets (cells) – each cell only needs a virtual circuit ID
- Constant latency

**Packet-switching (datagram)**

- Shared connection; competition for use with others
- Data is broken into chunks called packets
- Each packet contains a destination address
- Available bandwidth ≤ channel capacity
- Variable latency

4/6/2015 © 2014-2015 Paul Krzyzanowski 6

# Client-Server Networking

4/6/2015 © 2014-2015 Paul Krzyzanowski 7

## What's in the data?

For effective communication

- same language, same conventions

For computers:

- electrical encoding of data
- where is the start of the packet?
- which bits contain the length?
- is there a checksum? where is it? how is it computed?
- what is the format of an address?
- byte ordering

These instructions and conventions are known as **protocols**

4/6/2015 © 2014-2015 Paul Krzyzanowski 8

## Layering

To ease software development and maximize flexibility:

- Network protocols are generally organized in **layers**
- Replace one layer without replacing surrounding layers
- Higher-level software does not have to know how to format an Ethernet packet ... or even know that Ethernet is being used

4/6/2015 © 2014-2015 Paul Krzyzanowski 9

## Layering

Most popular model of guiding (not specifying) protocol layers is

### OSI reference model

Created & adopted by ISO

Defines 7 layers of protocols

4/6/2015 © 2014-2015 Paul Krzyzanowski 10

## OSI Reference Model: Layer 1

Transmits and receives raw data to communication medium

Does not care about contents

Media, voltage levels, speed, connectors

Deals with representing bits

1

Physical

Examples: USB, Bluetooth, 1000BaseT, Wi-Fi

4/6/2015 © 2014-2015 Paul Krzyzanowski 11

## OSI Reference Model: Layer 2

Detects and corrects errors

Organizes data into frames before passing it down. Sequences packets (if necessary)

Accepts acknowledgements from receiver

Deals with frames

2

Data Link

Examples: Ethernet MAC, PPP

1

Physical

4/6/2015 © 2014-2015 Paul Krzyzanowski 12

### OSI Reference Model: Layer 2

An **ethernet switch** is an example of a device that works on layer 2

It forwards **ethernet frames** from one host to another as long as the hosts are connected to the switch (switches may be cascaded)

This set of hosts and switches defines the **local area network (LAN)**



2 **Data Link**

1 **Physical**

4/6/2015 © 2014-2015 Paul Krzyzanowski 13

### OSI Reference Model: Layer 3

Relay and route information to destination

Manage journey of **datagrams** and figure out intermediate hops (if needed)

3 **Network**

2 **Data Link**

1 **Physical**

Deals with datagrams

Examples: IP, X.25

4/6/2015 © 2014-2015 Paul Krzyzanowski 14

### OSI Reference Model: Layer 3

An IP router is an example of a device that works on layer 3

A router takes an incoming IP packet and determines which interface to send it out

It enables multiple networks to be connected together



3 **Network**

2 **Data Link**

1 **Physical**

4/6/2015 © 2014-2015 Paul Krzyzanowski 15

### OSI Reference Model: Layer 4

Provides an interface for end-to-end (application-to-application) communication: sends & receives **segments** of data. Manages flow control. May include end-to-end reliability

Network interface is similar to a mailbox

4 **Transport**

3 **Network**

2 **Data Link**

1 **Physical**

Deals with segments

Examples: TCP, UDP

4/6/2015 © 2014-2015 Paul Krzyzanowski 16

### OSI Reference Model: Layer 5

Services to coordinate dialogue and manage data exchange

Software implemented switch

Manage multiple logical connections

Keep track of who is talking: establish & end communications

5 **Session**

4 **Transport**

3 **Network**

2 **Data Link**

1 **Physical**

Deals with data streams

Examples: HTTP 1.1, SSL

4/6/2015 © 2014-2015 Paul Krzyzanowski 17

### OSI Reference Model: Layer 6

Data representation

Concerned with the meaning of data bits

Convert between machine representations

6 **Presentation**

5 **Session**

4 **Transport**

3 **Network**

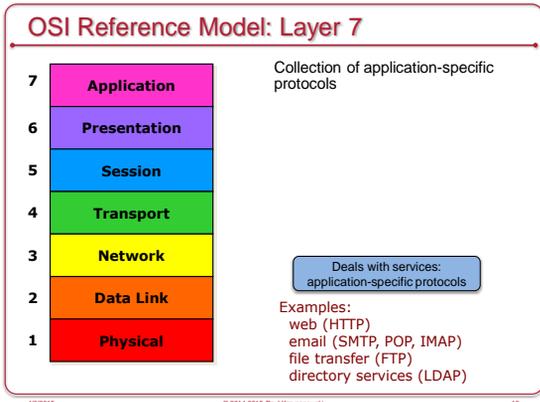
2 **Data Link**

1 **Physical**

Deals with objects

Examples: XDR, ASN.1, MIME, JSON, XML

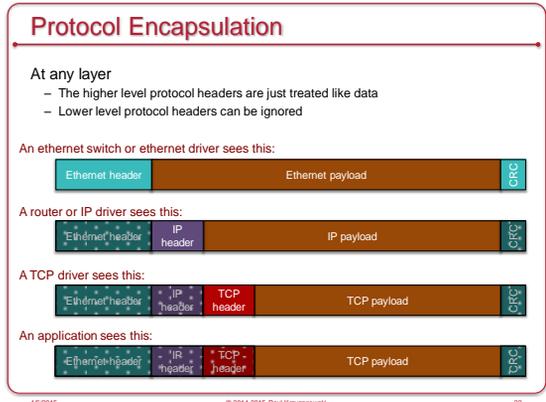
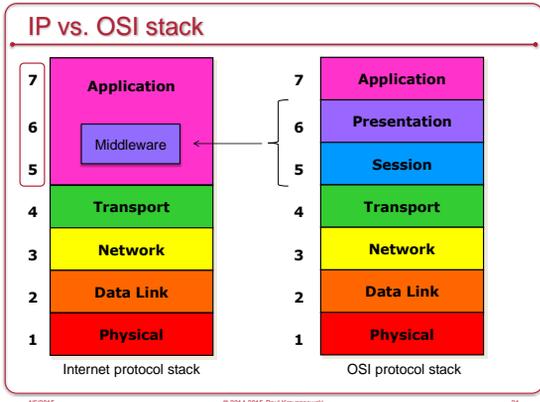
4/6/2015 © 2014-2015 Paul Krzyzanowski 18



### Internet Protocol

- A set of protocols designed to handle the interconnection of a large number of local and wide-area networks that comprise the Internet
- IPv4 & IPv6: **network layer**
  - Other protocols include **TCP, UDP, RSVP, ICMP**, etc.
  - Relies on **routing** from one physical network to another
  - IP is **connectionless**
    - No state needs to be saved at each router
  - Survivable** design: support multiple paths for data
    - ... but packet delivery is not guaranteed!

4/6/2015 © 2014-2015 Paul Krzyzanowski 20



### Client – Server Communication

4/6/2015 © 2014-2015 Paul Krzyzanowski 23

### Addressing machines (data link layer)

Each interface on a host has a unique MAC address

- E.g., `aramis.rutgers.edu`: 48-bit ethernet address = `00:03:ba:09:1b:b0`

This allows us to address ethernet frames to machines on the same LAN

MAC = Media Access Control

April 6, 2015 © 2014 Paul Krzyzanowski 24

## Addressing machines (network layer)

Each interface on a host is given a unique IP address

- IPv4 (still the most common in the U.S.): 32-bit number
  - Example, `cs.rutgers.edu = 128.6.4.2 = 0x80060402`
- IPv6: 128-bit number
  - Example, `cs.rutgers.edu = 0:0:0:0:0:FFFF:128.6.4.2 = ::FFFF:8006:0402`
- **Routable across networks**
  - We can send IP packets to machines on the Internet
  - **BUT** ... this only gets us to the machine, not the application

4/6/2015

© 2014-2015 Paul Krzyzanowski

25

## Address translation

- **Domain name → IP address translation**
  - **Domain Name System (DNS)**
    - Hierarchical human-friendly names (e.g., `www.cs.rutgers.edu`)
  - User-level network service to look up IP domain names
  - Cache results to avoid future look-ups
  - The kernel's network drivers do not handle domain names
- **IP → Ethernet MAC address translation**
  - **Address Resolution Protocol (ARP)**
  - How does the OS know which ethernet address to use?
  - Broadcast an ARP query and wait for a response  
"Who has 128.6.4.2?"
  - Cache results to avoid future look-ups

4/6/2015

© 2014-2015 Paul Krzyzanowski

26

## Ethernet & IP – Message Reliability

- **Ethernet**
  - LAN connectivity
  - Higher-level protocols (IP) encapsulated inside
  - Unreliable delivery
    - Frames may be lost to congestion, errors, or collision
- **IP**
  - Datagram delivery is also unreliable
  - Frames may be lost due to dropped ethernet frames, errors, congestion, or time-to-live expiration

4/6/2015

© 2014-2015 Paul Krzyzanowski

27

## IP transport layer

IP give us two **transport-layer** protocols

- **TCP: Transmission Control Protocol**
  - **Connection-oriented service**
    - Operating system keeps state: *simulates* a virtual circuit over a datagram network
  - **Full-duplex connection**: both sides can send messages over the same link
  - **Reliable data transfer**: the protocol handles retransmission
  - **In-order data transfer**: the protocol keeps track of sequence numbers
- **UDP: User Datagram Protocol**
  - **Connectionless service**: lightweight transport layer over IP
  - Data may be lost
  - Data may arrive out of sequence
  - Checksum for corrupt data: operating system drops bad packets

4/6/2015

© 2014-2015 Paul Krzyzanowski

28

## Addressing applications (transport layer)

**Communication endpoint at the machine**

- **Port number**: 16-bit value
- Port number = transport endpoint
  - Allows application-application communication
  - Identifies a specific data stream
- Some services use well-known port numbers (0 – 1023)
  - IANA: Internet Assigned Numbers Authority ([www.iana.org](http://www.iana.org))
  - Also see the file `/etc/services`  
`ftp: 21/tcp ssh: 22/tcp smtp: 25/tcp http: 80/tcp ntp: 123/udp`
- Ports for proprietary apps: 1024 – 49151
- Dynamic/private ports: 49152 – 65535
- **To communicate with applications, we use a transport layer protocol and an IP address and port number**

4/6/2015

© 2014-2015 Paul Krzyzanowski

29

## Network API

- App developers need access to the network
- A **Network Application Programming Interface (API)** provides this
- Core services provided by the operating system
  - Operating System controls access to resources
- Libraries may handle the rest
- We will only look at IP-based communication

4/6/2015

© 2014-2015 Paul Krzyzanowski

30

### Programming: connection-oriented protocols

- |                         |                                |
|-------------------------|--------------------------------|
| 1. establish connection | <i>analogous to phone call</i> |
| 2. [negotiate protocol] | <i>dial phone number</i>       |
| 3. exchange data        | <i>[decide on a language]</i>  |
| 4. terminate connection | <i>speak</i>                   |
|                         | <i>hang up</i>                 |

#### virtual circuit service

- provides illusion of having a dedicated circuit
- messages guaranteed to arrive in-order
- application does not have to address each message

#### Not to be confused with virtual circuit networks

- Which provide constant latency & guaranteed bandwidth
- TCP simulates a virtual circuit network ... sort of

4/6/2015

© 2014-2015 Paul Krzyzanowski

31

### Programming: connectionless protocols

- |                         |                                |
|-------------------------|--------------------------------|
| - no call setup         | <i>analogous to mailbox</i>    |
| - send/receive data     |                                |
| (each packet addressed) | <i>drop letter in mailbox</i>  |
| - no termination        | <i>(each letter addressed)</i> |

#### datagram service

- client is not positive whether message arrived at destination
- no state has to be maintained at client or server
- cheaper but less reliable than virtual circuit service

4/6/2015

© 2014-2015 Paul Krzyzanowski

32

The End

4/6/2015

© 2014-2015 Paul Krzyzanowski

33