This time

Starting with Networking Basics

- A whirlwind tour of networking
- What is a protocol?
- What are the abstractions / mental models?
- Network stack
(1) Protocols

Agreement on how to communicate

• Syntax:
  • How the communication is specified and structured
  • Format, order of messages

• Semantics:
  • What the communication means
  • Actions that should be taken when transmitting, receiving, or when a timer expires.
(1) Protocols

Agreement on how to communicate

• Syntax:
  • How the communication is specified and structured
  • Format, order of messages

• Semantics:
  • What the communication means
  • Actions that should be taken when transmitting, receiving, or when a timer expires.

An algorithm for communicating.
And a “language” to speak.
The payload is the “data” that IP is delivering: May contain another protocol’s header & payload, and so on.
The network is “dumb”

- **End-hosts** are on the periphery of the network
  - They can *connect* to one another, even though they are *not physically connected* to one another

- **Routers** are the interior nodes that
  - “*Route*”: *determine how* to get to B
  - “*Forward*”: actually forward traffic from A to B

- **Principle**: the routers have no knowledge of ongoing connections through them
  - They do “destination-based” routing and forwarding
    - Given the destination in the packet, send it to the “next hop” that is best suited to help ultimately get the packet there
The network is “dumb”

- **End-hosts** are on the periphery of the network
  - They can *connect* to one another, even though they are *not physically connected* to one another

- **Routers** are the interior nodes that
  - “*Route*”: *determine how* to get to B
  - “*Forward*”: actually forward traffic from A to B

- Principle: the routers have no knowledge of ongoing connections through them
  - They do “*destination-based*” routing and forwarding
    - Given the destination in the packet, send it to the “*next hop*” that is best suited to help ultimately get the packet there

Mental model: The postal system
Postal system analogy

• Messages are self-contained
  • Post: a message in an envelope
  • Internet: data in a packet

• Interior routers forward based on destination address
  • Post: zip code, then street, then building, then apartment number (then the right individual)
  • Internet: progressively smaller blocks of IP addresses, then your computer (then the right application)

• Simple, robust.
  • More sophisticated things go at the ends of the network
(3) Layers

• The design of the Internet is strongly partitioned into **layers**
  • Each layer relies on the services provided by the layer **immediately** below it…
  • … and provides service to the layer **immediately** above it
(3) Layers

- The design of the Internet is strongly partitioned into layers
  - Each layer relies on the services provided by the layer immediately below it...
  - ... and provides service to the layer immediately above it

Analogy:
(3) Layers

• The design of the Internet is strongly partitioned into layers
  • Each layer relies on the services provided by the layer immediately below it…
  • … and provides service to the layer immediately above it

**Analogy:**

Code you write
(3) Layers

• The design of the Internet is strongly partitioned into layers
  • Each layer relies on the services provided by the layer immediately below it…
  • … and provides service to the layer immediately above it

**Analogy:**

- Code you write
- Run-time library
(3) Layers

- The design of the Internet is strongly partitioned into layers
  - Each layer relies on the services provided by the layer immediately below it...
  - ... and provides service to the layer immediately above it

**Analogy:**

- Code you write
- Run-time library
- System calls
(3) Layers

- The design of the Internet is strongly partitioned into **layers**
  - Each layer relies on the services provided by the layer *immediately* below it...
  - ... and provides service to the layer *immediately* above it

**Analogy:**

1. Code you write
2. Run-time library
3. System calls
4. Device drivers
(3) Layers

• The design of the Internet is strongly partitioned into layers
  • Each layer relies on the services provided by the layer immediately below it…
  • … and provides service to the layer immediately above it

**Analogy:**

- Code you write
- Run-time library
- System calls
- Device drivers
- Voltage levels, etc.
(3) Layers

• The design of the Internet is strongly partitioned into layers
  • Each layer relies on the services provided by the layer immediately below it…
  • … and provides service to the layer immediately above it

Analogy:

- Code you write
- Run-time library
- System calls
- Device drivers
  - Voltage levels, etc.

Isolated from user programs
(3) Layers

- The design of the Internet is strongly partitioned into layers:
  - Each layer relies on the services provided by the layer immediately below it…
  - … and provides service to the layer immediately above it

**Analogy:**

- Code you write
- Run-time library
- System calls
- Device drivers
- Voltage levels, etc.

Each layer has a well-defined role that builds off of the layer below it.

Isolated from user programs
(3) Layers

- The design of the Internet is strongly partitioned into layers.
  - Each layer relies on the services provided by the layer immediately below it…
  - … and provides service to the layer immediately above it.

**Analogy:**
- Code you write
- Run-time library
- System calls
- Device drivers
- Voltage levels, etc.

Isolated from user programs

Between each layer is a well-defined *interface*. Each layer has a well-defined *role* that builds off of the layer below it.
Internet layering = “Protocol stack”

7 Application
4 Transport
3 (Inter)network
2 Link
1 Physical
Layer 1: Physical layer

- **Encoding** of bits to send over a *single* physical link
- **Examples:**
  - Voltage levels
  - RF modulation
  - Photon intensities
Physical layer:
transmitting a single bit
over a physical link
(though not necessarily *wired* link)

End-host C
Layer 2: Link layer

- Framing and transmission of a collection of bits into individual messages sent across a single subnetwork (one physical topology)
- Provides local addressing (MAC)
- May involve multiple physical links
- Often the technology supports broadcast: every “node” connected to the subnet receives
- Examples:
  - Modern Ethernet
  - WiFi (802.11a/b/g/n/etc)
Link layer
- transmitting messages
- over a subnet
- src/dst identified by **globally unique MAC addrs**
Link layer
- transmitting messages
- over a *subnet*
- src/dst identified by **globally unique** MAC *addrs*
Link layer
- transmitting messages
- over a *subnet*
- src/dst identified by **globally unique MAC addr**s

Because you need to be able to join any subnet and be uniquely distinguishable
Layer 3: (Inter)network layer

- Bridges multiple "subnets" to provide *end-to-end internet* connectivity between nodes
- Provides *global* addressing (IP addresses)
- Only provides *best-effort* delivery of data (i.e., no retransmissions, etc.)
- Works across different link technologies
Layer 3: (Inter)network layer

- Bridges multiple “subnets” to provide *end-to-end* internet connectivity between nodes
- Provides *global* addressing (IP addresses)
- Only provides *best-effort* delivery of data (i.e., no retransmissions, etc.)
- Works across different link technologies

Different for each Internet “hop”
Layer 3: (Inter)network layer

- Bridges multiple “subnets” to provide end-to-end internet connectivity between nodes
- Provides global addressing (IP addresses)
- Only provides best-effort delivery of data (i.e., no retransmissions, etc.)
- Works across different link technologies

Different for each Internet “hop”

Lowercase-i “internet” = network of networks.
Uppercase-i Internet = “the Internet”
Network layer
- transmitting packets
- within or across subnets
- src/dst identified by *locally* unique **IP addr**s
Network layer
- transmitting packets
- within or across subnets
- src/dst identified by locally unique IP addrs
Network layer
- transmitting packets
- within or across subnets
- src/dst identified by **locally unique IP addrs**
Network layer
- transmitting packets
- within or across subnets
- src/dst identified by locally unique **IP addrs**

Routers connect multiple subnets
Network layer
- transmitting packets
- within or across subnets
- src/dst identified by locally unique IP addrs

Routers connect multiple subnets

End-host A

End-host B

End-host E

Router 1

Router 2

Router 3

Router 4

Router 5

Router 6
Network layer
- transmitting packets
- within or across subnets
- src/dst identified by **locally** unique IP addrs

Routers connect multiple subnets
Local uniqueness is often enough

There are only $2^{32}$ IP addr

Many machines don’t need to be publicly reachable

Some addresses are “private” addresses

The router performs “Network Address Translation”: changes outgoing packets’ src from 192.168.1.100 to 63.14.2.33, and vice versa for incoming packets
Local uniqueness is often enough

There are only $2^{32}$ IP addrs

Many machines don’t need to be publicly reachable

Some addresses are “private” addresses

The router performs “Network Address Translation”: changes outgoing packets’ src from 192.168.1.100 to 63.14.2.33, and vice versa for incoming packets
Local uniqueness is often enough

Rest of the Internet

Router 6

192.168.1.1

192.168.1.100

192.168.1.101

There are only $2^{32}$ IP addr.

Many machines don’t need to be publicly reachable

Some addresses are "private" addresses

The router performs "Network Address Translation": changes outgoing packets’ src from 192.168.1.100 to 63.14.2.33, and vice versa for incoming packets.
Local uniqueness is often enough

There are only $2^{32}$ IP addresses.

Many machines don’t need to be publicly reachable.

Some addresses are "private" addresses.

The router performs "Network Address Translation": changes outgoing packets’ src from 192.168.1.100 to 63.14.2.33, and vice versa for incoming packets.

Rest of the Internet

Router 6

63.14.2.33

192.168.1.1

192.168.1.100

192.168.1.101
Layer 4: Transport layer

- End-to-end communication between processes
- Different types of services provided:
  - UDP: unreliable datagrams
  - TCP: reliable byte stream
- “Reliable” = keeps track of what data were received properly and retransmits as necessary
Layer 7: Application layer

- Communication of whatever you want
- Can use whatever transport(s) is(are) convenient/appropriate
- Freely structured
- Examples:
  - Skype (UDP)
  - SMTP = email (TCP)
  - HTTP = web (TCP)
  - Online games (TCP and/or UDP)
Internet layering = “Protocol stack”

Implemented only at end hosts, not at interior routers (this is our “dumb network”)
Internet layering = “Protocol stack”

Implemented everywhere

The network is “dumb” but it needs to know precisely this much to do its job.
Internet layering = “Protocol stack”

~Same for each Internet “hop”

Can be different for each Internet “hop”
Hop-by-hop vs. end-to-end layers

Host C communicates with host A
Hop-by-hop vs. end-to-end layers

Different physical & link layers

End-host A

End-host B

Router 1

Router 2

Router 3

Router 4

Router 5

Router 6

Ethernet

WiFi

End-host C

End-host D

End-host E
Hop-by-hop vs. end-to-end layers

Same network, transport, and application layers (3/4/7)
Routers *ignore* transport & application

E.g., HTTP over TCP over IP
Next time

• You now know the overall design:
  • What each layer is responsible for
  • What the predominant protocols are at each layer

• The overall design principles have been absolutely critical to making an Internet that can evolve with changing needs… for the most part

• But the devil’s in the details

• We will dig into specific protocols to understand the kinds of attacks that can happen at the networking layer and how to protect against them