Computer Network

Santiago

May 1, 2020

摘要

Prof.David Wetherall from University of Washington
Computer Network Course

1 Week 1 - Goals and Motivation

1.1 网址

- Total Course:
  - https://media.pearsoncmg.com/ph/streaming/esm/tanenbaum5e_videonotes/tanenbaum_videoNotes.html
- Week1 Weel5
  - https://www.youtube.com/playlist?list=PLzmjQ4eaGEug9YlLvqpBTdVyIZhzQxTzu
- Week6 Week10
  - https://www.youtube.com/playlist?list=PLv4Qy0s26-PL2-tAs6mPaxUzq-wwMdv5Y

1.2 The Main Point

- To learn how the Internet works
  - What really happens when you ”browse the web”?
  - What are TCP/IP, DNS, HTTP, NAT, VPNs, 802.11 etc. anyway?
- To learn the fundamentals of computer network.
  - What hard problems must they solve?
  - What design strategies have proven valuable?

2 Week 1 - Uses of Networks

2.1 Statistical Multiplexing

- Sharing of network bandwidth between users according to the statistic of their demand
  - (Multiplexing just means sharing)
  - Useful because users are mostly idle and their traffic is bursty
- Key question:
  - How much does it help?
- See Figure **??**
- With 30 independent users, still unlikely (20& chance) to need more than 100 Mbps!
  - Binomial probabilities

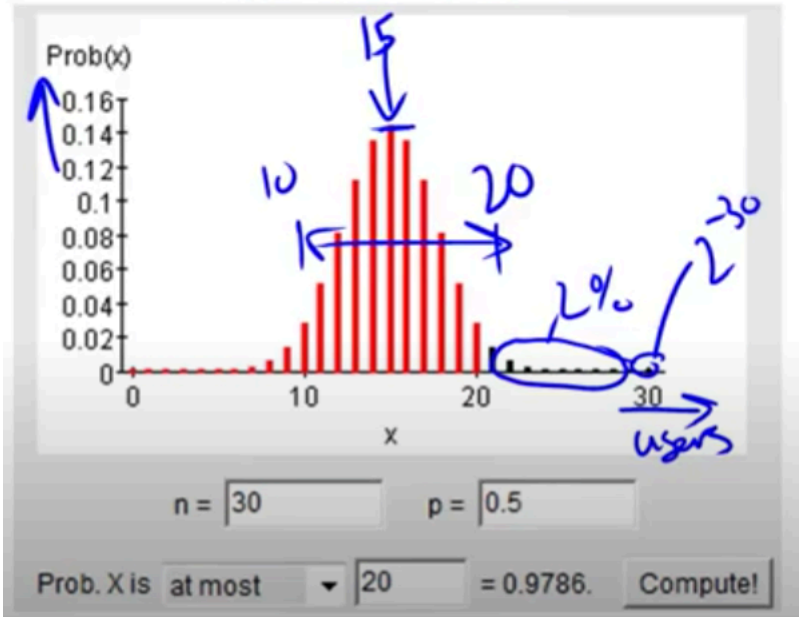


图 1: Binomial Calculator

- Can serve more users with the same size network
  - Statistical multiplexing gain is 30/20 or 1.5X
  - But may get unlucky; users will have degraded service

2.2 For Content Delivery

- Same content is delivered to many users
  - Videos (large), songs, apps and upgrades, web pages, ...
- More efficient then sending a copy all the way to each user
  - Uses replicas to the network
- Sending content from the source to 4 users takes $\underline{4 \times 3 = 12}$ "network hops" in the example, See Figure **??**

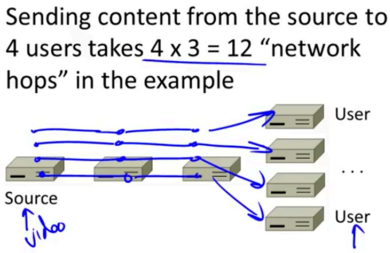


图 2: Sending content from the source to 4 users

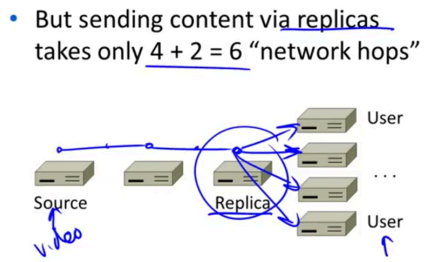


图 3: sending content via replicas

- But sending content via replicas takes only $4 + 2 = 6$ "network hops", See Figure **??**

2.3 The Value of Connectivity

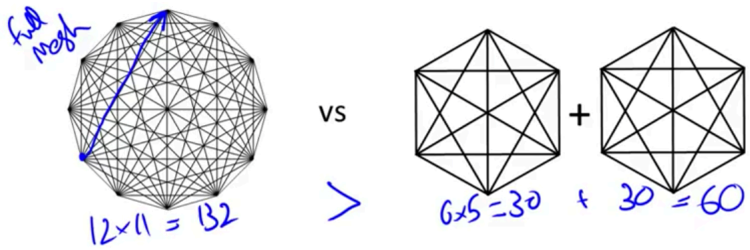


图 4: The Value of Connectivity

- "Metcalfe's Law" 1980:
  - The value of a network of N nodes is proportional to N^2
  - Large networks are relatively more valuable than small ones
- Example: both side have 12 nodes, but the left network has more connectivity, See Figure **??**

3 Week 1 - Network Components

3.1 Component Names

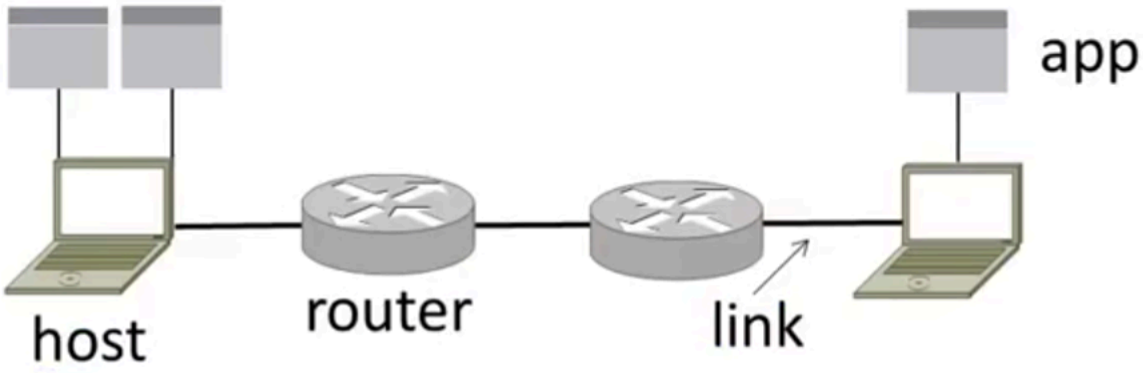


图 5: Parts of a Network

- See Figure **??**
- Component Names, See Table **??**

3.2 Types of Links

- Full-duplex
  - Bidirectional, See Figure **??**
- Half-duplex
  - Bidirectional, Figure **??**

Component	Function	Example
Application, or app, user	Uses the network	Skype, iTunes, Amazon
Host, or end-system, edge device, node, source, sink	Supports apps	Laptop, mobile, desktop
Router, or switch, node, hub, intermediate system	Relays messages between links	Access point, cable/DSL modem
Link, or channel	Connects nodes	Wires, wireless

表 1: Component Names

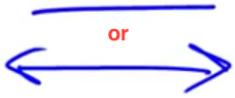


图 6: Full-duplex

图 7: Half-duplex

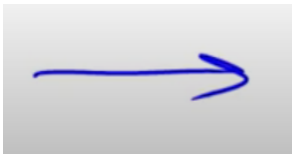


图 8: Simplex

- Simplex
  - unidirectional, See Figure **??**

3.3 Wireless Links

- Message is broadcast, See Figure **??**
  - Received by all nodes in range
  - Not a good fit with our model
- Often show logical links, See Figure **??**
  - Not all possible connectivity

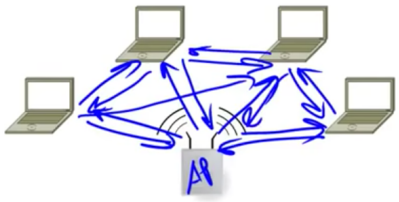


图 9: Message is broadcast

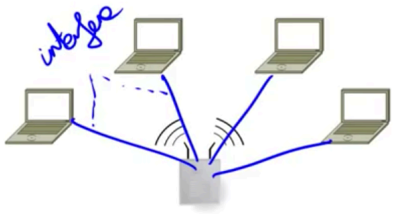


图 10: show logical links

4 Week 1 - Sockets

4.1 Network-Application Interface

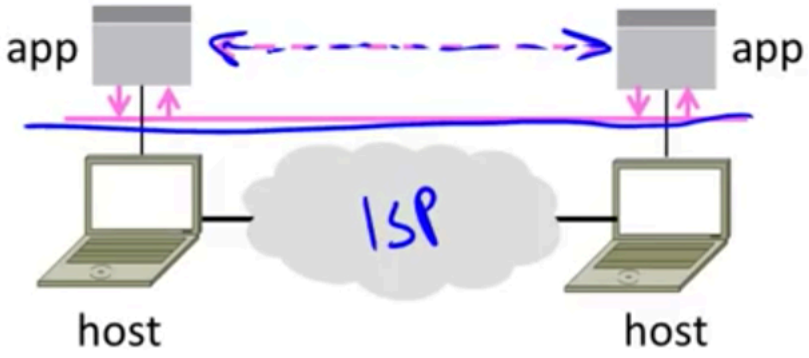


图 11: Network-Application Interface

- Defines how apps use the Network, See Figure **??**
  - Lets apps talk to each other via hosts; hides the detail of the net work

4.2 Socket API

- Simple abstraction to use the network

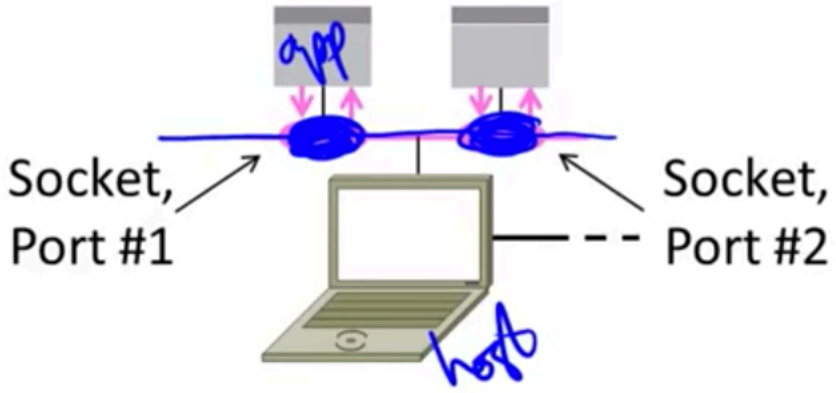


图 12: Socket API

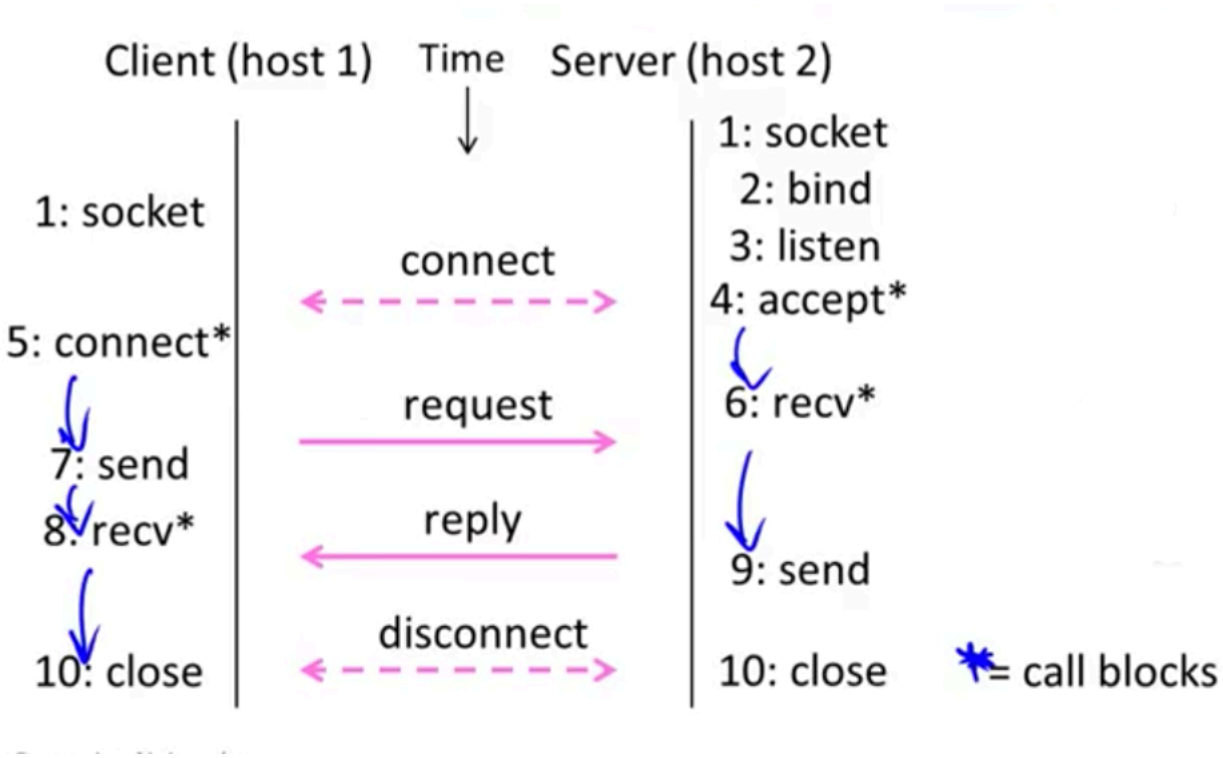


图 13: Using Sockets

  - The network service API used to write all Internet applications
  - Part of all major OSes and languages; originally Berkeley (Unix) 1983

- Supports two kinds of network services
  - Streams: reliably send a stream of bytes
  - Datagrams: unreliably send separate messages. (Ignore for now.)

- Sockets let apps attach to the local network at different ports , See Figure **??**

- Socket API Functions, See Table **??**

Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

表 2: Socket API Functions

- Using Sockets, See Figure **??**

5 Week 1 - Traceroute

5.1 Traceroute

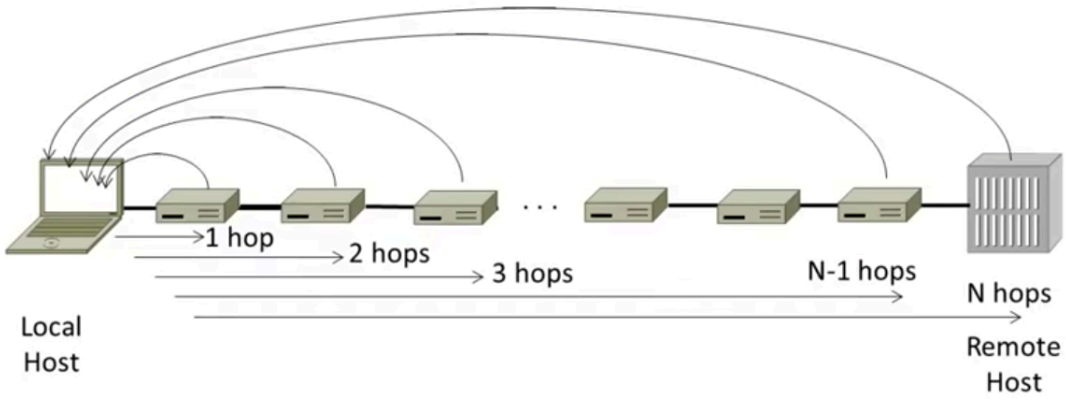


图 14: Traceroute

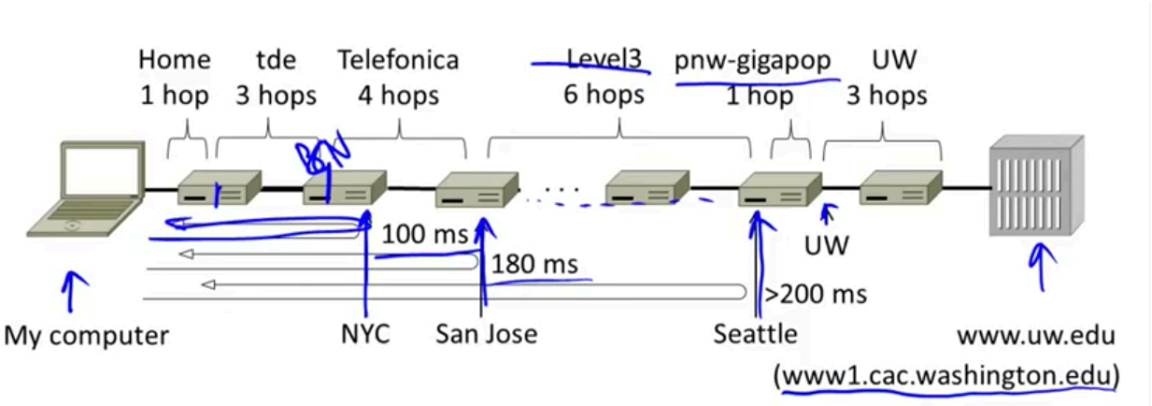


图 15: Using Traceroute

- Widely used command-line tool to let hosts peek inside the network
  - On all OSes (tracet on Windows)

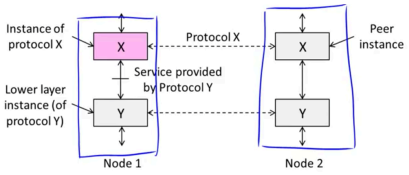


图 16: Protocols are horizontal, layers are vertical

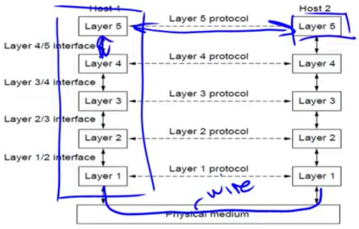


图 17: protocol stack

  - Developed by Van Jacobson 1987
  - Uses a network-network interface (IP) in ways we will explain later

- Probes successive hops to find network path, See Figure **??**
- For example: traceroute www.ahu.edu.cn
- ISP names and places are educated guesses, See Figure **??**

6 Week 1 - Protocol Layers

6.1 Protocol and Layers

- Protocols and layering is the main structuring method used to divide up network functionality
  - Each instance of a protocol talks virtually to its peer using the protocol
  - Each instance of a protocol uses only the services of the lower layer
- Protocols are horizontal, layers are vertical, See Figure **??**

- Set of protocols in use is called a protocol stack, See Figure **??**
- Protocols you've probably heard of:
  - TCP, IP, 802.11, Ethernet, HTTP, SSL, DNS,..., and many more
- An example protocol stack
  - Used by a web browser on a host that is wirelessly connected to the Internet

6.2 Encapsulation

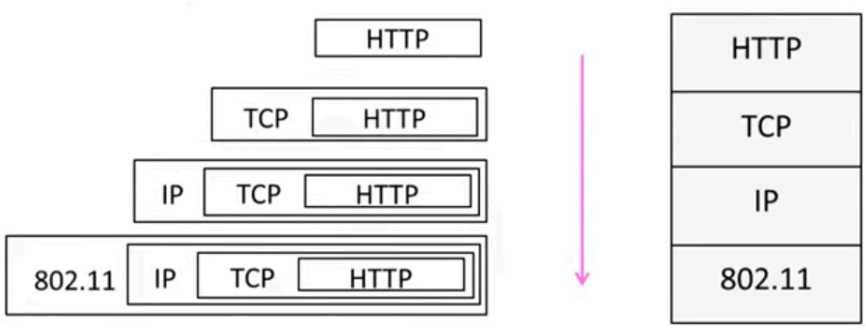


图 18: Encapsulation 1

- Encapsulation is the mechanism used to effect protocol layering
  - Lower layer wraps higher layer content, adding its own information to make a new message for delivery
  - Like sending a letter in an envelope; postal service doesn't look inside
- Message "on the wire" begins to look an onion, See Figure **??**

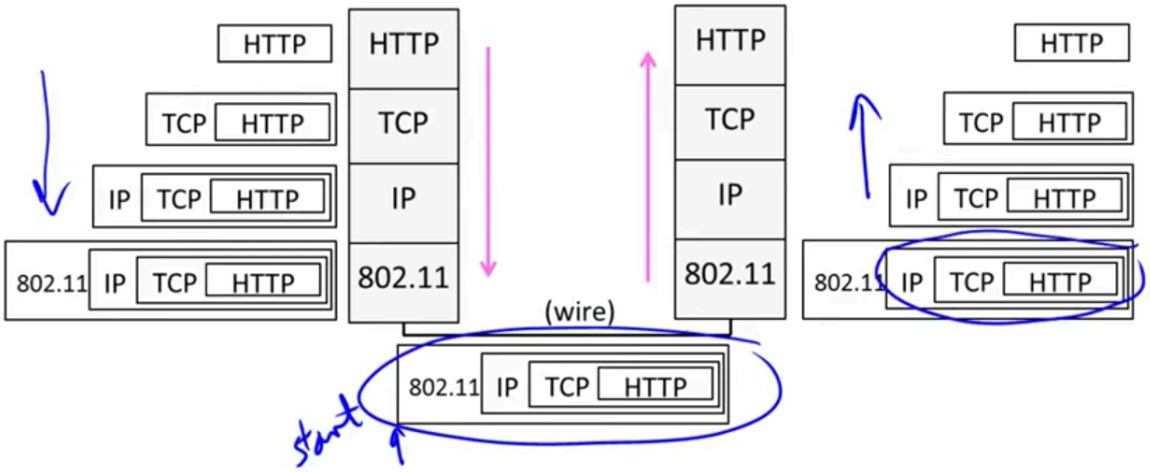


图 19: Encapsulation 2

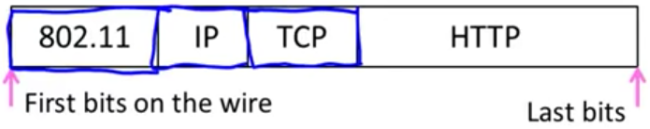


图 20: Encapsulation 3

  - Lower layers are outermost

- See Figure **??**
- Normally draw message like this: See Figure **??**
  - Each layer adds its own header
- More involved in practice
  - Trailers as well as headers, encrypt/compress contents
  - Segmentation (divide long message) and reassembly

6.3 Demultiplexing

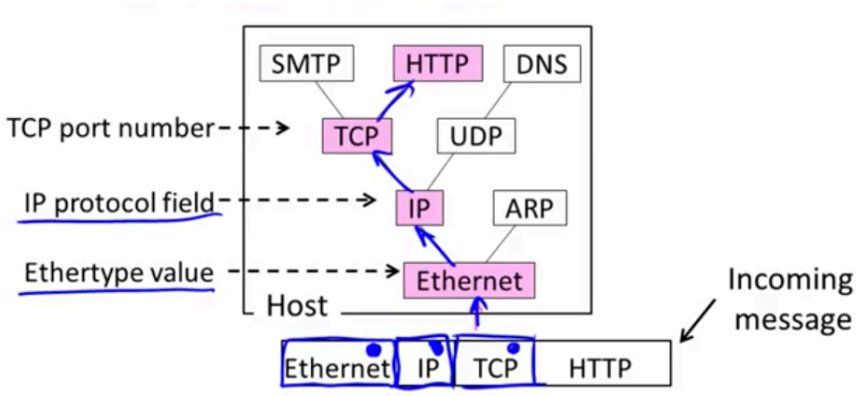


图 21: Demultiplexing

- Done with demultiplexing keys in the headers, See Figure **??**

6.4 Advantage of Layering

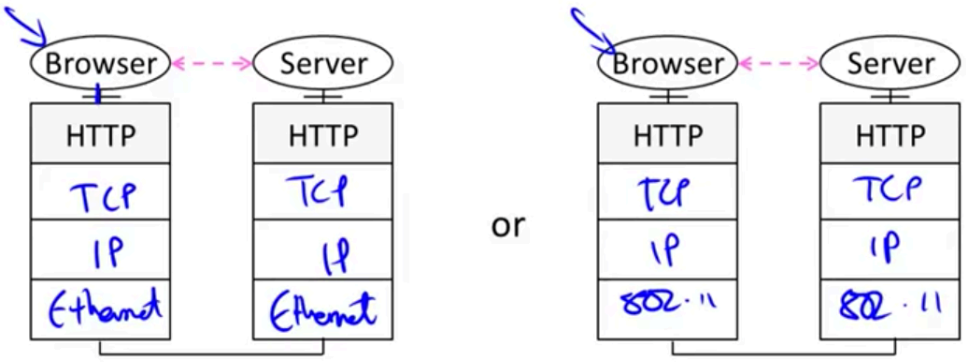


图 22: Information hiding and reuse

- Information hiding and reuse, See Figure **??**

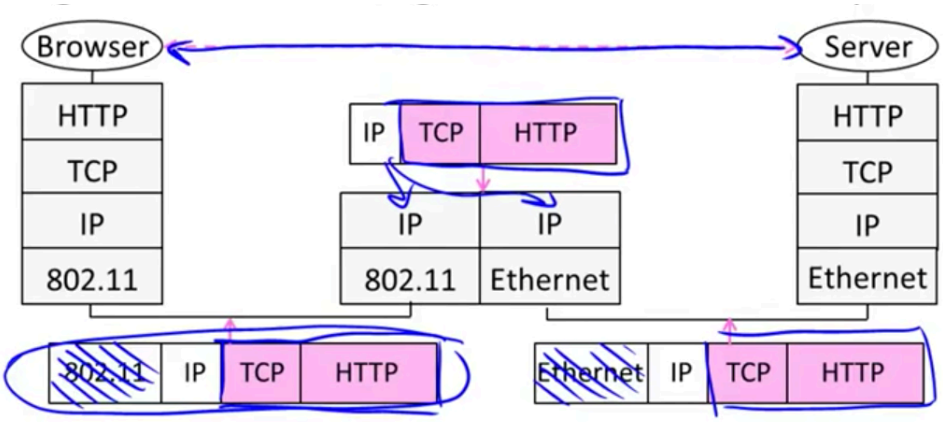


图 23: Using information hiding to connect different systems

- Using information hiding to connect different systems, See Figure **??**

6.5 Disadvantage of Layering

- Adds overhead
  - But minor for long messages
- Hides information
  - App might care whether it is running over wired or wireless!

7 Week 1 - Reference Models

7.1 OSI "7 layer" Reference Model

- A principled, international standard, to connect systems, See Figure **??**
  - Influential, but not used in practice. (Woops)

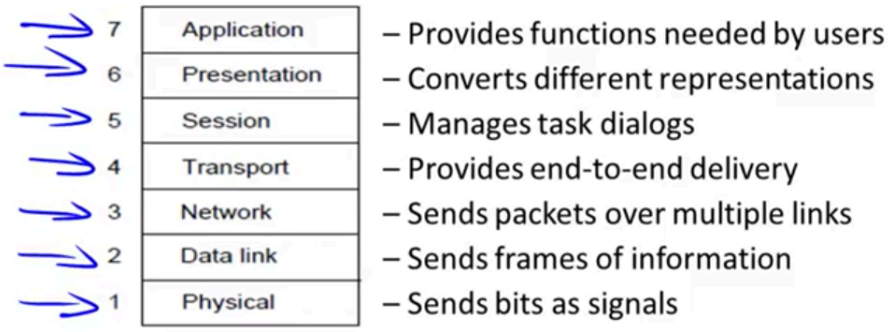


图 24: OSI "7 layer" Reference Model

7.2 Internet Reference Model

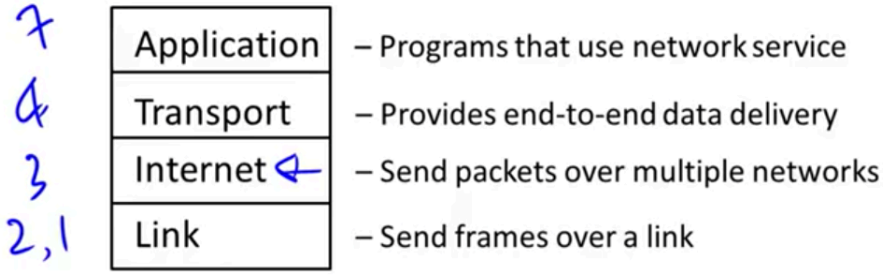


图 25: Internet Reference Model

- A four layer model based on experience; omits some OSI layers and uses IP as the network layer. See Figure **??**
- IP is the "narrow waist" of the Internet, See Figure **??**
  - Supports many different links below and apps above

7.3 Standards Bodies

- Where all the protocols come from! See Figure **??**

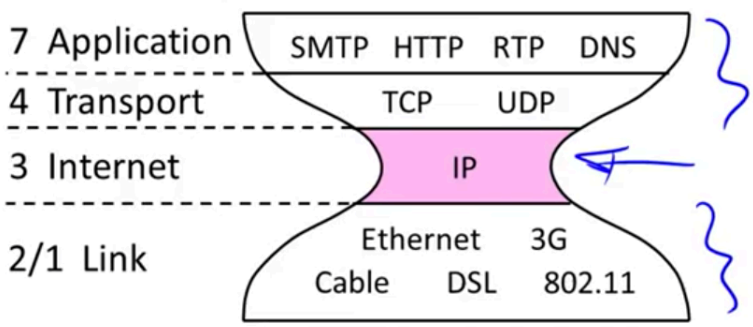


图 26: IP is the "narrow waist" of the Internet

Body	Area	Examples
ITU	Telecom	G.992, ADSL H.264, MPEG4
IEEE	Communications	802.3, Ethernet 802.11, WiFi
IETF	Internet	RFC 2616, HTTP/1.1 RFC 1034/1035, DNS
W3C	Web	HTML5 standard CSS standard

图 27: Standards Bodies

– Focus is on interoperability

7.4 Layer-based Names

- For units of data: See Figure **??**
- For devices in the network: See Figure **??** and Figure **??**

8 Week 1 - Internet History

略

Layer	Unit of Data
Application	Message
Transport	Segment
Network	Packet
Link	Frame
Physical	Bit

图 28: Units of data

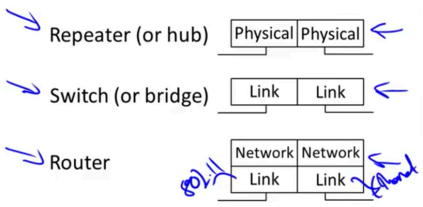


图 29: Devices in the network 1

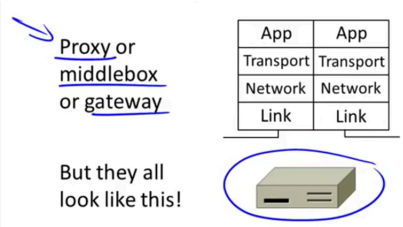


图 30: devices in the network 2

9 Week 1 - Lecture Outline

9.1 Textbook

- ”Computer Networks” 5/E (US or Intl, print or e-text) is optional but recommended
  - We’ll do our best to let you work without it (with the odd web search)
  - Read it for explanations, greater depth, extra topics, a reference, etc.

10 Week 2 - Physical Layer Overview

10.1 Scope of the Physical Layer

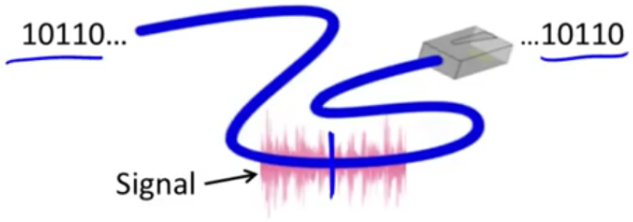


图 31: Transfer message bits over a link

- Concerns how signals are used to transfer message bits over a link, See Figure **??**
  - Wires etc. carry analog signals
  - We want to send digital bits

10.2 Topics

- 1. Properties of media
  - Wires, fiber optics, wireless
- 2. Simple signal propagation
  - Bandwidth, attenuation, noise
- 3. Modulation Schemes
  - Representing bits, noise
- 4. Fundamental limits
  - Nyquist, Shannon

10.3 Simple Link Model

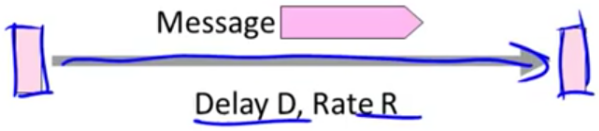


图 32: Simple Link Model

- We'll end with an abstraction of a physical channel , See Figure **??**
  - Rate (or bandwidth, capacity, speed) in bits/second
  - Delay in seconds, related to length
- Other important properties:
  - Whether the channel is broadcast, and its error rate

10.4 Message Latency

- Latency is the delay to send a message over a link
  - Transmission delay: time to put M-bit message "on the wire"

$$T - delay = M\,(bits)/Rate\,(bits/sec) = M/R\,seconds$$

  - Propagation delay: time for bits to propagate across the wire

$$P - delay = Length/speed\,of\,signals = \frac{Length}{\frac{2}{3}C} = D\,seconds$$

  - Combining the two terms we have: $L = M/R + D$

Prefix	Exp.	prefix	exp.
K(ilo)	10^3	m(illi) 毫秒	10^{-3}
M(ega)	10^6	μ(micro) 微秒	10^{-6}
G(iga)	10^9	n(ano)	10^{-9}

表 3: Socket API Functions

10.5 Metric Units

- The main prefixes we use: See Table **??**
- Use powers of 10 for rates, 2 for storage
  - 1 Mbps = 1,000,000 bps, 1 KB = 2^{10} bytes
- "B" is for bytes, "b" is for bits

10.6 Latency Example

- "Dialup" with a telephone modem:

$$\begin{aligned} &D = 5\ ms, R = 56\ Kbps, M = 1250\ bytes \\ &L = 5\ ms + (1250 \times 8)/(56 \times 10^3)\ sec = 148\ ms! \end{aligned} \tag{1}$$

- Broadband cross-country link:

$$\begin{aligned} &D = 50\ ms, R = 10\ Mbps, M = 1250\ bytes \\ &L = 50\ ms + (1250 \times 8)/(10 \times 10^6)\ sec = 51ms \end{aligned} \tag{2}$$

- A long link or a slow rate means high latency
  - Often, one delay component dominates

10.7 Bandwidth-Delay Product

- Messages take space on the wire!
- The amount of data in flight is the bandwidth-delay (BD) product

$$BD = R \times D$$

  - Measure in bits, or in messages
  - Small for LANs, big for "long fat" pipes
  - 表示第一个包到达接收端的时候，发送端总共发送进入网络的数据量，单位用 bit 表示。

11 Week 2 - Media

11.1 Types of Media

- Media propagate signals that carry bits of information
- We'll look at some common types:
  - Wires
  - Fiber (fiber optic cables)
  - Wireless

11.2 Wires - Twisted Pair

- Very common; used in LANs and telephone lines, Figure **??**
  - Twists reduce radiated signal

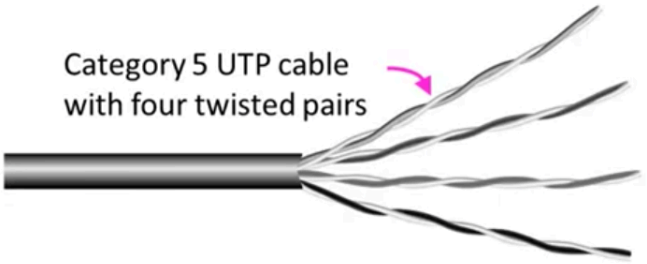


图 33: Wires - Twisted Pair

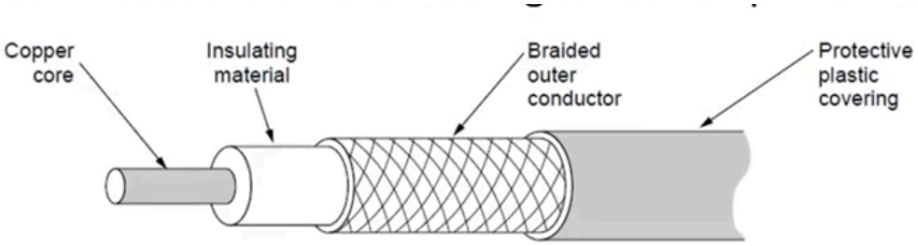


图 34: Wires - Coaxial Cable

11.3 Wires - Coaxial Cable 同轴电缆

- Also common. better shielding for better performance, See Figure **??**
- Other kinds of wires too: e.g., electrical power (§2.2.4)

11.4 Fiber

- Long, thin, pure strands of glass, See Figure **??**
  - Enormous bandwidth (high speed) over long distances
- Two varieties: multi-mode (shorter links, cheaper) and single-mode (up to 100 km), See Figure **??**

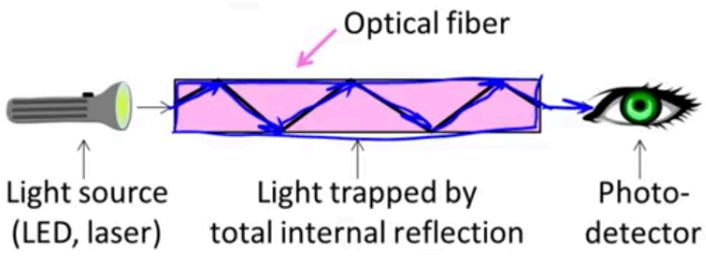


图 35: Fiber

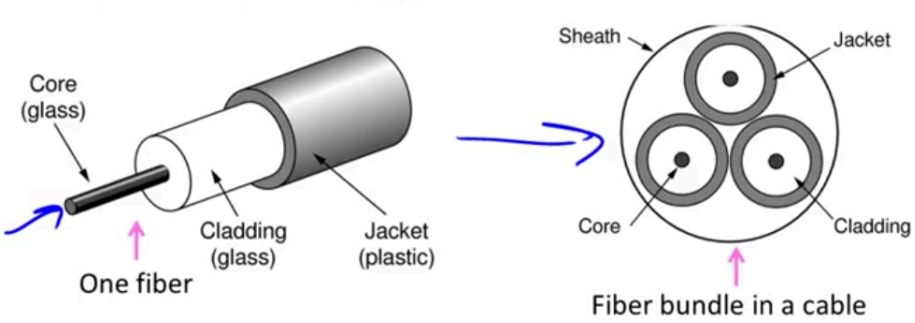


图 36: Fiber 2

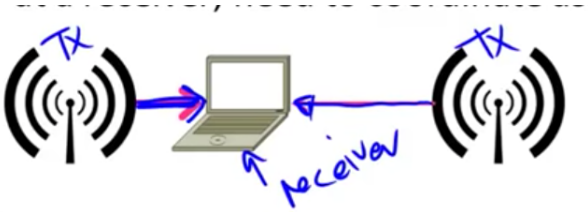


图 37: Wireless

11.5 Wireless

- Sender radiates signal over a region, See Figure **??**
  - In many directions, unlike a wire, to potentially many receivers
  - Nearby signal (same freq.) interface at a receiver; need to coordi-

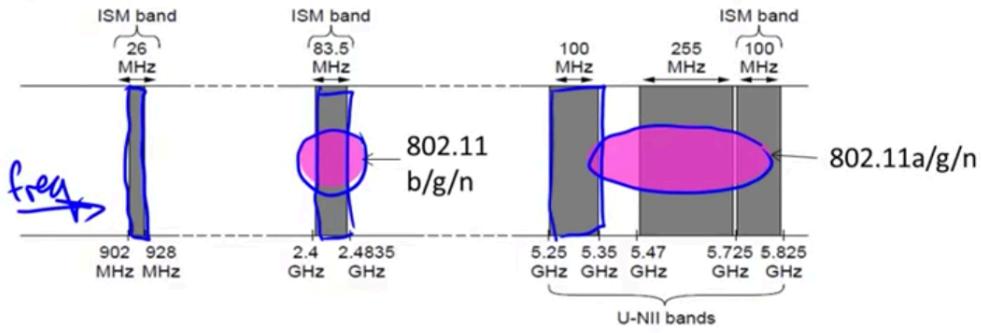


图 38: Wireless 2

nate use

- Microwave, e.g., 3G, and unlicensed (ISM) frequencies, e.g., WiFi, are widely used for computer networking, See Figure **??**

12 Week 2 - Signals

12.1 Topic

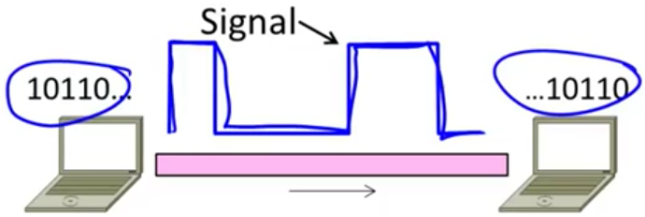


图 39: Topic of signals

- Analog signals encode digital bits. We want to know what happens as signals propagate over media, See Figure **??**

12.2 Frequency Representation

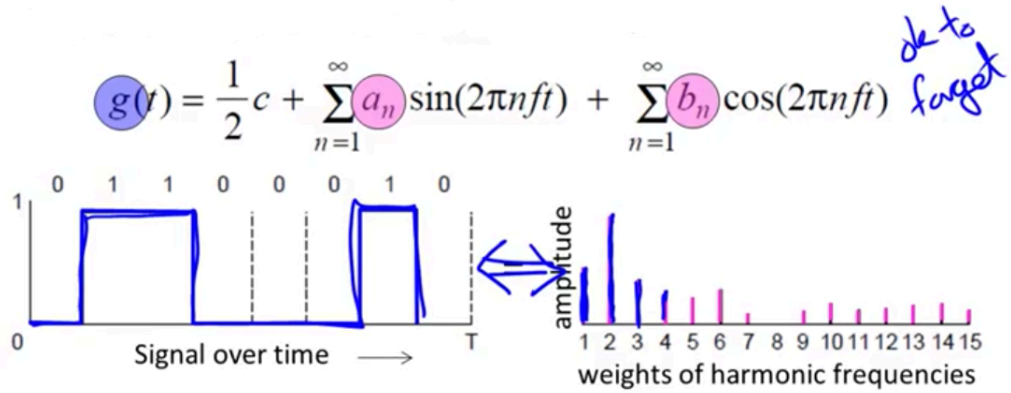


图 40: Frequency Representation

- A signal over time can be represented by its frequency components (called Fourier analysis) , See Figure **??**

12.3 Effect of Less Bandwidth

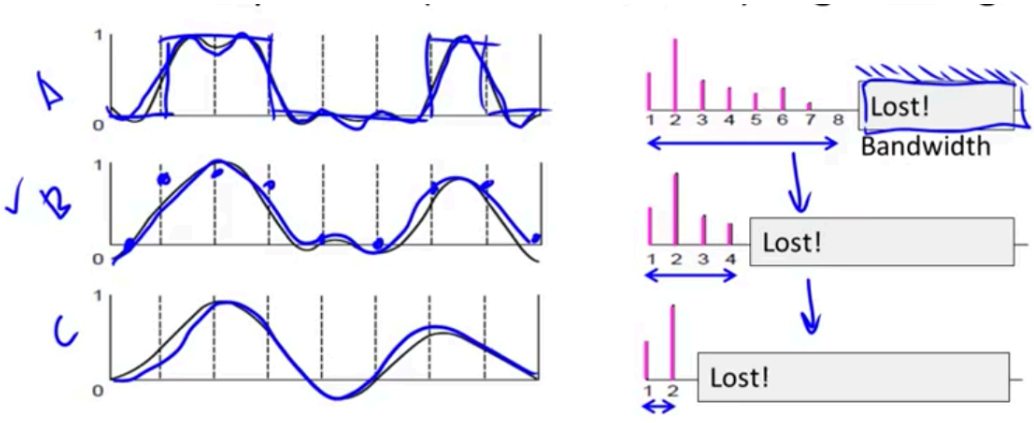


图 41: Effect of Less Bandwidth

- Fewer frequencies (=less bandwidth) degrades signal, See Figure **??**

12.4 Signals over a Wire

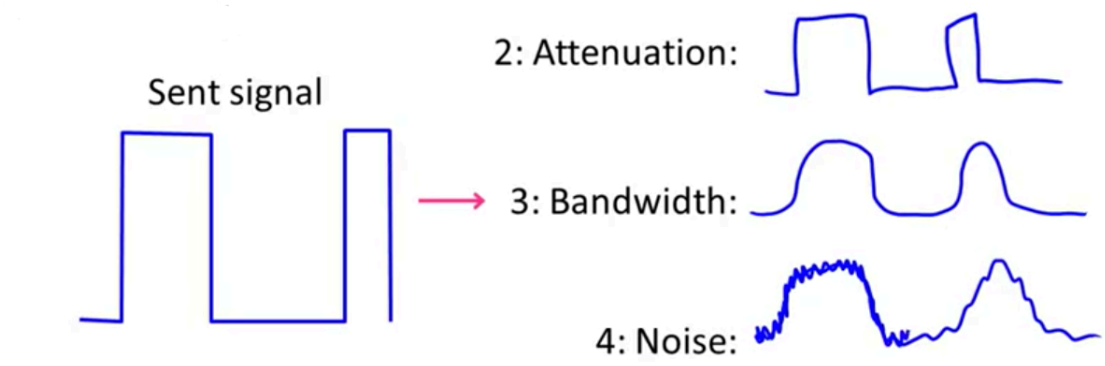


图 42: Signals over a Wire

- What happens to a signal as it passed over a wire?
  - 1. The signal is delayed (propagates at $\frac{2}{3}C$)
  - 2. The signal is attenuated (goes for m to km)
  - 3. Frequencies above a cutoff are highly attenuated
  - 4. Noise is added to the signal (later, causes errors)
- Example See Figure **??**
- Tips:
  - EE: Bandwidth = width of frequency band, measured in Hz
  - CS: Bandwidth = information carrying capacity, in bits/sec

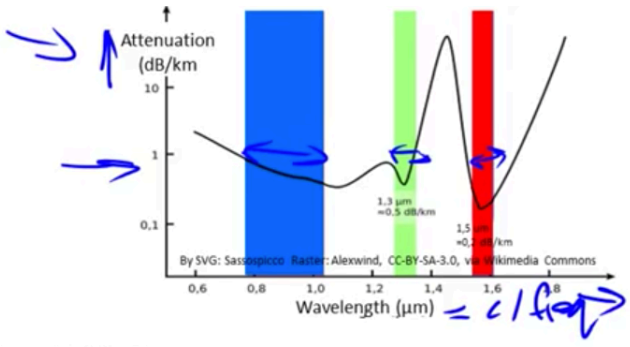


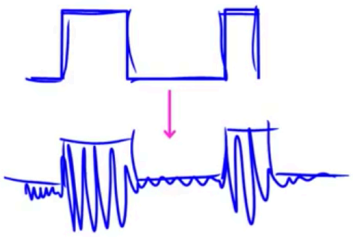
图 43: Signals over Fiber

12.5 Signals over Fiber

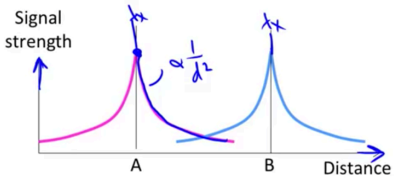
- Light propagates with very low loss in three very wide frequency bands, See Figure **??**
  - Use a carrier to send information

12.6 Signals over Wireless

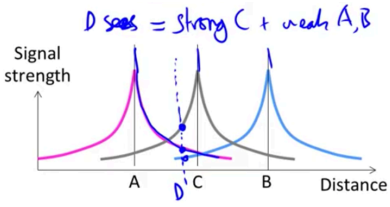
- Signals transmitted on a carrier frequency, like fiber (more later) , See Figure **??**
- Travel at speed of light, spread out and attenuate faster than $\frac{1}{dist^2}$, See Figure **??**
- Multiple signals on the same frequency interfere at a receiver, See Figure **??**
- Interference leads to notion of spatial reuse (of same freq.), See Figure **??**
- Various other effects too!
  - Wireless propagation is complex, depends on environment



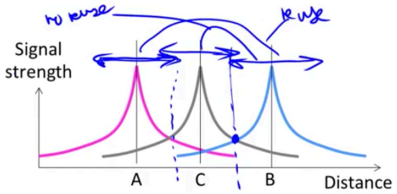
(a) Signals transmitted on a carrier frequency



(b) Attenuate



(c) Interfere



(d) Spatial reuse

图 44: Four Figures

- Some key effects are highly frequency dependent,
  - E.g., multipath at microwave frequencies

12.7 Wireless Multipath

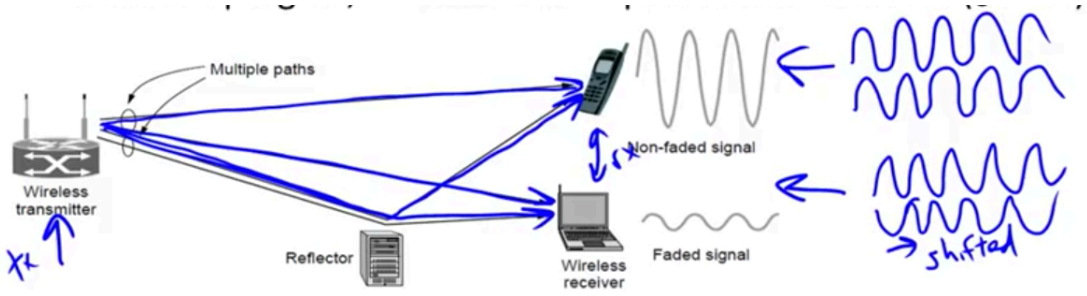


图 45: Wireless Multipath

- Signals bounce off objects and take multiple paths, See Figure **??**
  - Some frequencies attenuated at receiver, varies with location
  - Messes up signal; handled with sophisticated method (§2.5.3)

13 Week 2 - Modulation

13.1 Topic of Modulation

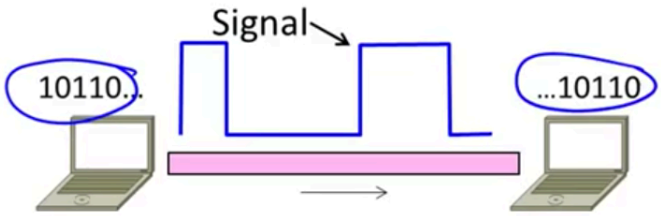


图 46: Topic of Modulation

- We've talked about signals representing bits. How, exactly? See Figure **??**
  - This is the topic of modulation

13.2 A Simple Modulation

- Let a high voltage (+V) represent a 1, and low voltage (-V) represent a 0, See Figure **??**
  - This is called NRZ (Non-Return to Zero)

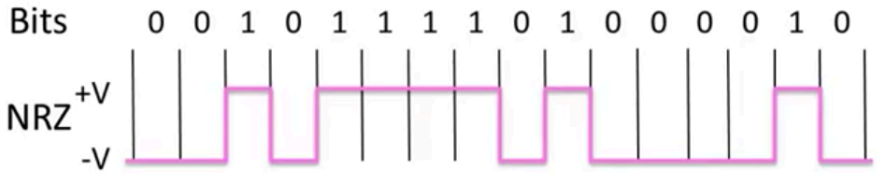


图 47: A Simple Modulation

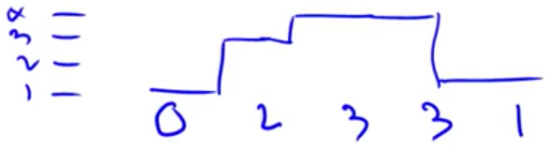


图 48: Many Other Schemes

13.3 Many Other Schemes

- Can use more signal levels, e.g., 4 levels in 2 bits per symbol, See Figure **??**
- Practical schemes are driven by engineering considerations
  - E.g., clock recovery

13.4 Clock Recovery

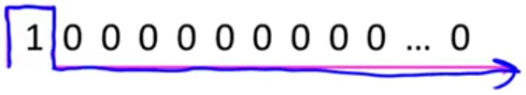


图 49: Clock Recovery

- Um, how may zeros was that? See Figure **??**

  - Receiver needs frequent signal transitions to decode bits

- Several possible designs
  - E.g.,. Manchester coding and scrambling (§2.5.1)

13.5 Clock Recovery - 4B/5B

- Map every 4 data bits into 5 code bits without long runs of zeros
  - 0000 → 11110, 0001 → 01001,
    1110 → 11100, ... 1111 → 11101
  - Has at most 3 zeros in a row
  - Also invert signal level on a 1 to break up long runs of 1s (called NRZI, §2.5.1) NRZI 中的 I 是 invert

- Message bits: 1 1 1 1 0 0 0 0 0 0 0 1 , See Figure **??**
  1 变 0 不变

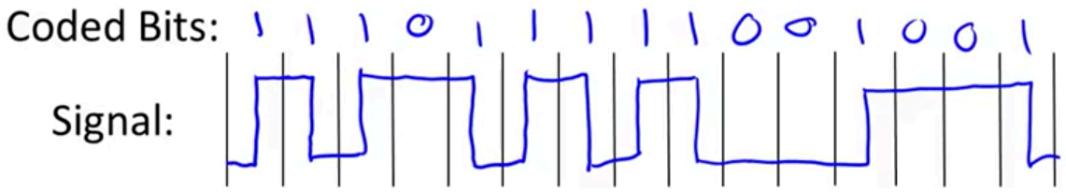


图 50: Message bits: 1 1 1 1 0 0 0 0 0 0 0 1

13.6 Passband Modulation

- What we have seen so far is baseband modulation for wires
  - Signal is sent directly on a wire

- These signals do not propagate well on fiber / wireless
  - Need to send at higher frequencies
- Passband modulation carries a signal by modulating a carrier
- Carrier is simply a signal oscillating at a desired frequency: See Figure **??**

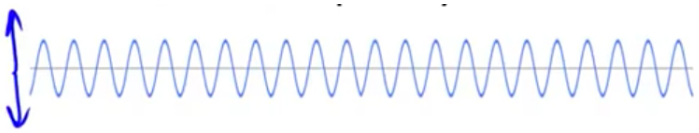


图 51: Carrier

- We can modulate it by changing:
  - Amplitude, frequency, or phase, See Figure **??**

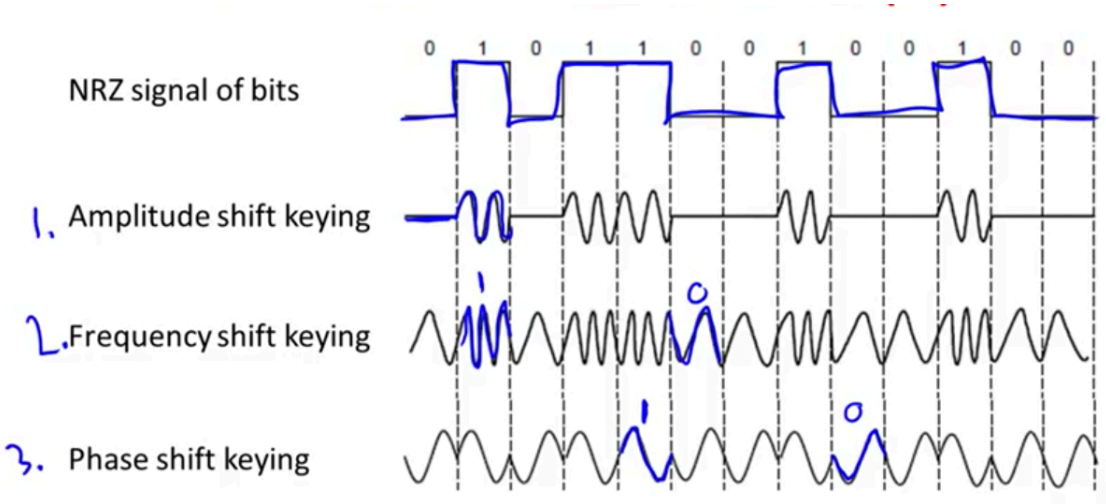


图 52: Modulate it by changing Amplitude, frequency, or phase

14 Week 2 - Limits

14.1 Topic of Limits

- How rapidly can we send information over a link?
  - Nyquist limit (1924)
  - Shannon capacity (1948)
- Practical systems are devised to approach these limits

14.2 Key Channel Properties

- The bandwidth (B), signal strength (S), and noise strength (N), See Figure **??**
  - B limits the rate of transitions
  - S and N limit how many signal levels we can distinguish

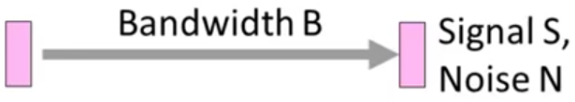


图 53: Key Channel Properties

14.3 Nyquist Limit

- The maximum symbol rate is 2B , See Figure **??**
- Thus if there are V signal levels, ignoring noise, the maximum bit rate is: $R = 2B \log_2 V \ bits/sec$

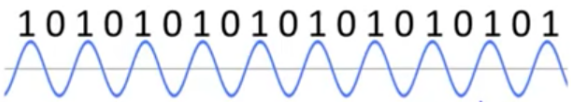


图 54: Nyquist Limit

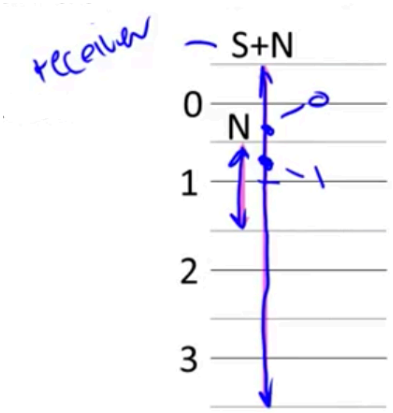


图 55: Shannon Capacity

14.4 Shannon Capacity

- How many levels we can distinguish depends on S/N, See Figure **??**
  - Or SNR, the Signal-to-Noise Ratio
  - Note noise is random, hence some errors
- SNR given on a log-scale in deciBels:
  - $SNR_{dB} = 10 \log_{10}(S/N)$
- Shannon limit is for capacity (C), the maximum information carrying rate of the channel: $C = B \log_2(1 + S/N)\ bits/sec$

14.5 Wired/Wireless Perspective

- Wires, and Fiber (Engineer SNR for data rate)
  - Engineer link to have requisite SNR and B → Can fix data rate
- Wireless (Adapt data rate to SNR)
  - Given B, but SNR varies greatly, e.g., up to 60 dB! → Can't design for worst case, must adapt data rate

14.6 Putting it all together - DSL

- DSL (Digital Subscriber Line, see §2.6.3) is widely used for broadband; many variants offer 10s of Mbps , See Figure **??**
  - Reuse twisted pair telephone line to the home; it has up to 2MHz of bandwidth but uses only the lowest 4kHz

图 56: Putting it all together - DSL

- DSL uses passband modulation (called OFDM §2.5.1) , See Figure **??**
  - Separate bands for upstream and downstream (larger)
  - Modulation varies both amplitude and phase (called QAM)
  - High SNR, up to 15 bits/symbol, low SNR only 1 bit/symbol

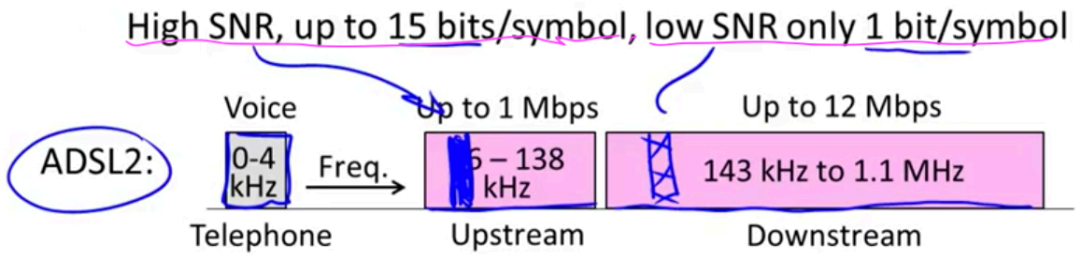


图 57: DSL 2

15 Week 2 - Link Layer Overview

15.1 Scope of the Link Layer

- Concerns how to transfer messages over one or more connected links, See Figure **??**
  - Messages are frames, of limited size
  - Builds on the physical layer

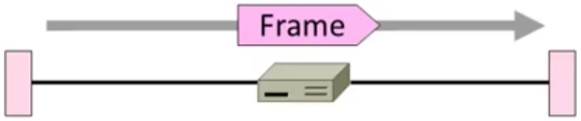


图 58: Scope of the Link Layer

15.2 In terms of layers

- See Figure **??**

15.3 Typical Implementation of Layers

- See Figure **??**

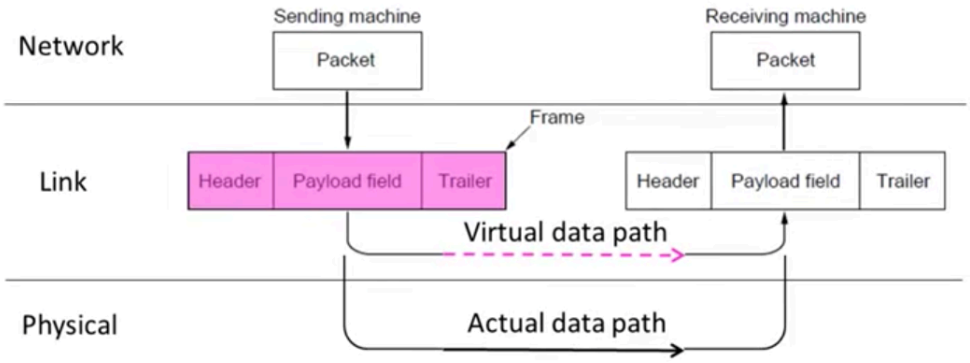


图 59: In terms of layers

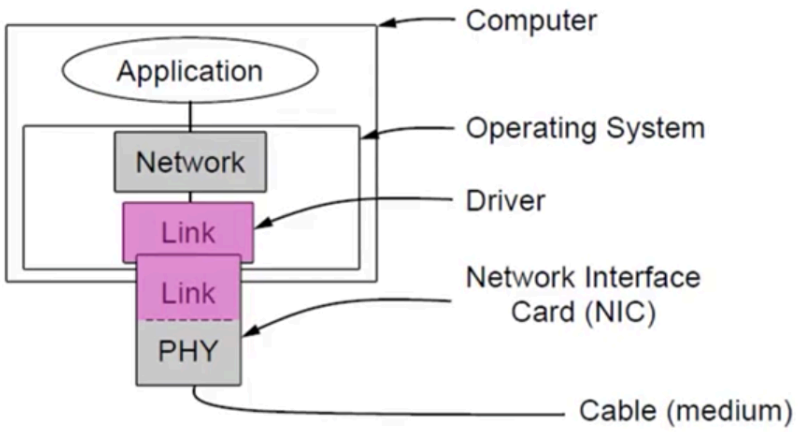


图 60: Typical Implementation of Layers

16 Week 2- Framing

16.1 Topic of Framing

- The Physical layer gives us a stream of bits. How do we interpret it as a sequence of frames? See Figure **??**

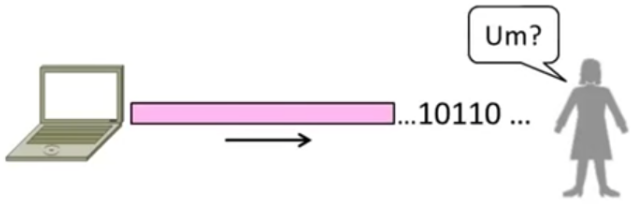


图 61: Topic of Framing

16.2 Framing Methods

- We'll look at:
  - Byte count (motivation)
  - Byte stuffing
  - Bit stuffing
- In practice, the physical layer often helps to identify frame boundaries
  - E.g., Ethernet, 802.11

16.3 Byte Count

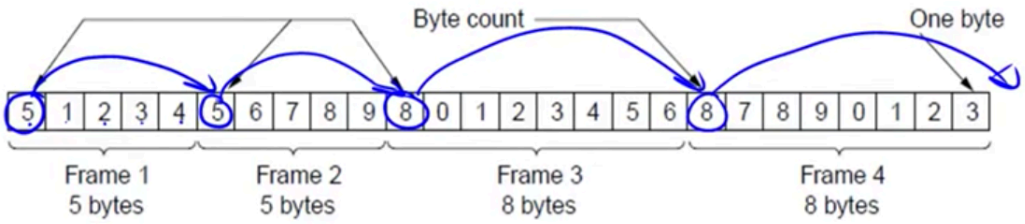


图 62: Byte Count 1

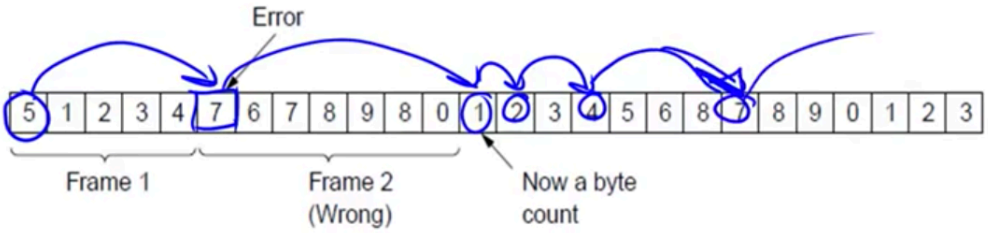


图 63: Byte Count 2

- First try:
  - Let's start each frame with length field!
  - It's simple, and hopefully good enough ...
  - How well do you think it works? See Figure **??**
  - Difficult to re-synchronize after framing error, See Figure **??**
    - Want a way to scan for a start of frame

16.4 Byte Stuffing

图 64: Byte Stuffing 1

- Better idea: See Figure **??**
  - Have a special flag byte value that means start/end of frame
  - Replace ("stuff") the flag inside the frame with an escape code
  - Complication: have to escape the escape code too!

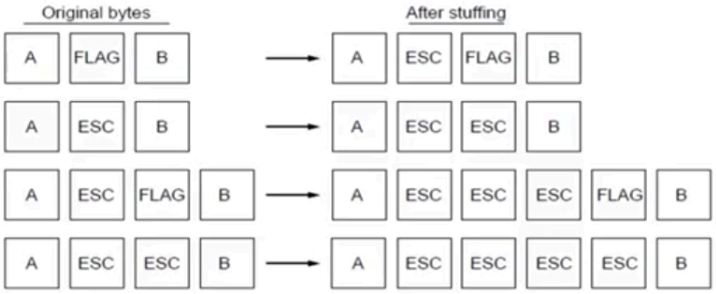


图 65: Byte Stuffing 2

- Rules: See Figure **??**
  - Replace each FLAG in data with ESC FLAG
  - Replace each ESC in data with ESC ESC
  - 这个和编程语言中的转义符合有异曲同工之妙
- Now any unescaped FLAG is the start/end of a frame

16.5 Bit Stuffing

- Can stuff at the bit level too
  - Call a flag six consecutive 1s
  - On transmit, after five 1s in the data, insert a 0
  - On receive, a 0 after five 1s is deleted
- Example: See Figure **??**
- So how does it compare with byte stuffing?

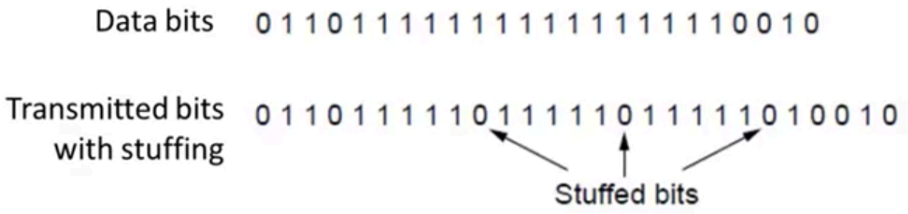


图 66: Bit Stuffing

16.6 Link Example: PPP over SONET

- PPP is Point-to-Point Protocol
- Widely used for link framing
  - E.g., it is used to frame IP packets that are sent over SONET optical links
- Think of SONET as a bit stream, and PPP as the framing that carries an IP packet over the link, See Figure **??**

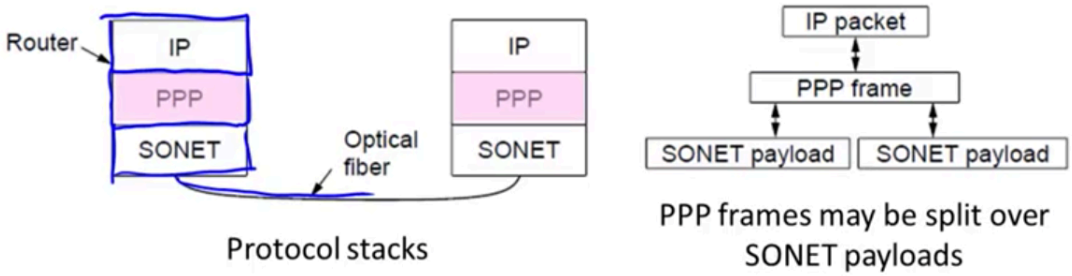


图 67: Link Example: PPP over SONET 1

2-7-8

- Framing uses byte stuffing, See Figure **??**

– FLAG is 0x7E and ESC is 0x7D

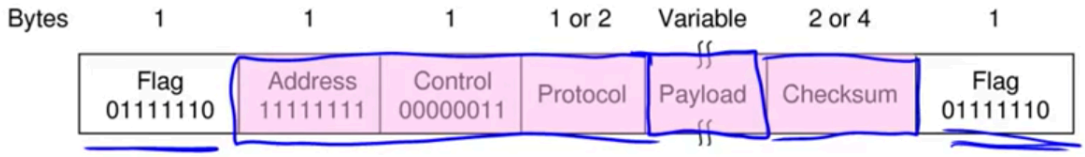


图 68: Link Example: PPP over SONET 2

- Byte stuffing method:
  - To stuff (unstuff) a byte, add (remove) ESC (0x7D), and XOR byte with 0x20, 翻转第 5 个 bit
  - Removes FLAG from the contents of the frame

$$\begin{aligned} 0x7E &\rightarrow 0x7D5E \\ 0x7D &\rightarrow 0x7D5D \end{aligned} \tag{3}$$

17 Week 2 - Error Overview

17.1 Topic of Error Overview

- Some bits will be received in error due to noise. What can we do?
  - Detect errors with codes
  - Correct errors with codes
  - Retransmit lost frame 这个会在后面讲
- Reliability is a concern that cuts across the layers - we'll see it again

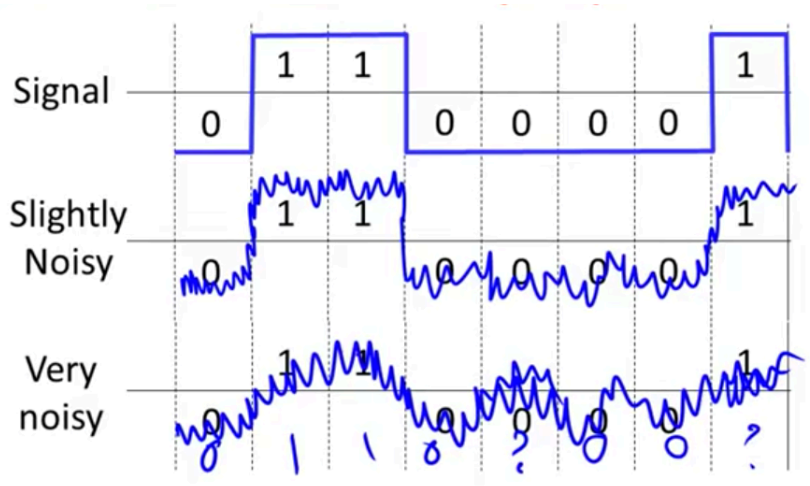


图 69: Problem - Noise may flip received bits

17.2 Problem - Noise may flip received bits

- See figure **??**

17.3 Approach - Add Redundancy

- Error detection codes
  - Add check bits to the message bits to let some errors be detected
- Error correction codes
  - Add more check bits to let some errors be corrected
- Key issue is now to structure the code to detect many errors with few check bits and modest computation （适当的计算量）

17.4 Using Error Codes

- Codeword consists of D data plus R check bits (= systematic block code)

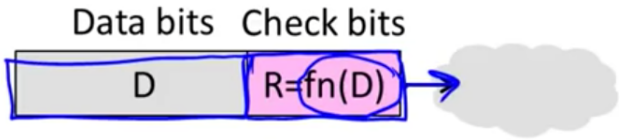


图 70: Using Error Codes - Sender

- Send: See Figure **??**
  - Compute R check bits based on the D data bits; send the codeword of D+R bits
- Receiver: See Figure **??**
  - Receive D+R bits with unknown errors
  - Recompute R check bits based on the D data bits; error if R doesn't match R'

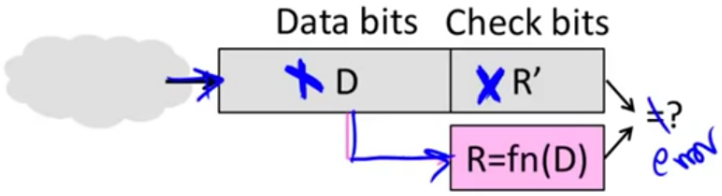


图 71: Using Error Codes - Receiver

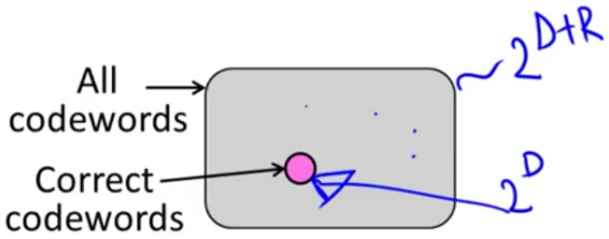


图 72: Intuition for Error Codes

17.5 Intuition for Error Codes

- For D data bits, R check bits: See Figure **??**
- Randomly chosen codeword is unlikely to be correct; overhead is low . 概率为 $\frac{1}{2^R}$

17.6 Hamming Distance

- Distance is the number of bit flips needed to change D_1+R_1 to D_2+R_2

$$1 \to 111 \qquad 0 \to 000 \qquad distance = 3 \tag{4}$$

- Hamming distance of a code is the minimum distance between any pair of codewords

$$HD(HammingDistance) = 3$$

- Error Detection:
  - For code of distance d+1, up to d errors will always be detected

$$\begin{aligned} &d+1=3 \quad \Rightarrow d=2 \\ &000 \qquad 111 \\ &\qquad \Downarrow \\ &001 \quad 010 \quad 100 \quad 011 \quad 101 \quad 110 \end{aligned} \tag{5}$$

- Error correction:
  - For code of distance 2d+1, up to d errors can always be corrected by mapping to the closest codeword

$$\begin{aligned} &HD = 3 \qquad 2d + 1 = 3 \\ &d = 1 \\ &\qquad \Downarrow \\ &010 \rightarrow 000 \qquad 110 \rightarrow 111 \end{aligned} \tag{6}$$

18 Week 2 - Error Detection

18.1 Topic of Error Detection

- Some bits may be received in error due to noise. How do we detect this?
  - Parity
  - Checksums
  - CRCs
- Detection will let us fix the error, for example, by retransmission (later).

18.2 Simple Error Detection - Parity Bit

- Take D data bits, add 1 check bit that is the sum of the D bits, See Figure **??**
  - Sum is modulo 2 or XOR
- How well does parity work?

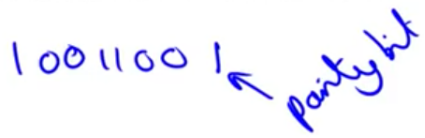


图 73: Parity Bit

- what is the distance of the code?

  2

- How many errors will it detect/correct?

  1 *and* 0

- What about larger errors?

  odd # errors

18.3 Checksums

- Idea: sum up data in N-bit words, See Figure **??**
  - Widely used in, e.g., TCP/IP/UDP

图 74: Checksums

- Stronger protection than parity

18.4 Internet Checksum

- Sum is defined in 1s complement arithmetic (must add back carries)
  - And It's the negative sum
- "The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words ..." -RFC 791
- Sending: See Figure **??**
  - 1. Arrange data in 16-bit words
  - 2. Put zero in checksum position, add
  - 3. Add any carry over back to get 16 bits
  - 4. Negate (complement) to get sum
- Receiving: See Figure **??**
  - 1. Arrange data in 16-bit words
  - 2. Checksum will be non-zero, add
  - 3. Add any carry over back to get 16 bits
  - 4. Negate the result and check it is 0
- How well does the checksum work?
  - What is the distance of the code?

    2
  - How many errors will it detect/correct?

    1 *and* 0
- What about larger errors? See Figure **??**

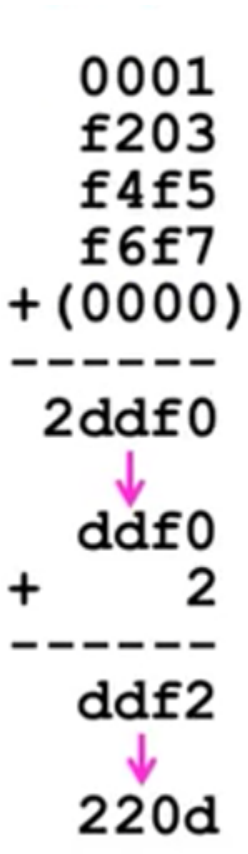


图 75: Internet Checksum - Sending

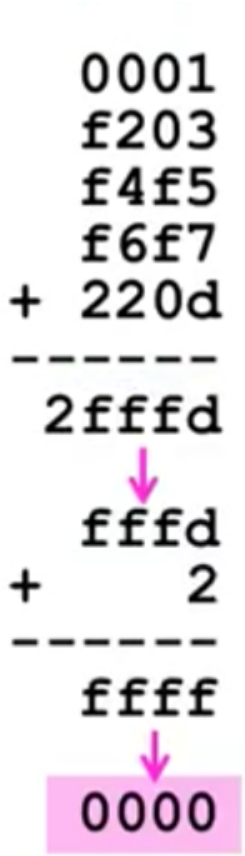


图 76: Internet Checksum - Receiving

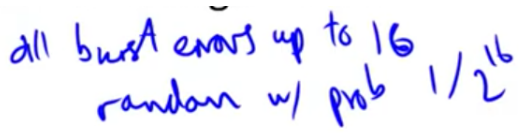


图 77: Internet Checksum - what about large errors?

18.5 Cyclic Redundancy Check (CRC)

- Even stronger protection
  - Given n data bits, generate k check bits such that n+k bits are evenly divisible by a generator C
- Example with numbers:
  - n=302, k= one digit, C=3 , See Figure **??**

图 78: CRC Example with numbers

- The catch:
  - It's based on mathematics of finite fields, in which "numbers" represent polynomials
  - e.g., 10011010 is $x^7 + x^4 + x^3 + x^1$
- What this means:
  - We work with binary values and operate using modulo 2 arithmetic

- Send Procedure: See Figure **??**
  - 1. Extend the n data bits with k zeros
  - 2. Divide by the generator value C
  - 3. Keep remainder, ignore quotient
  - 4. Adjust k check bits by remainder

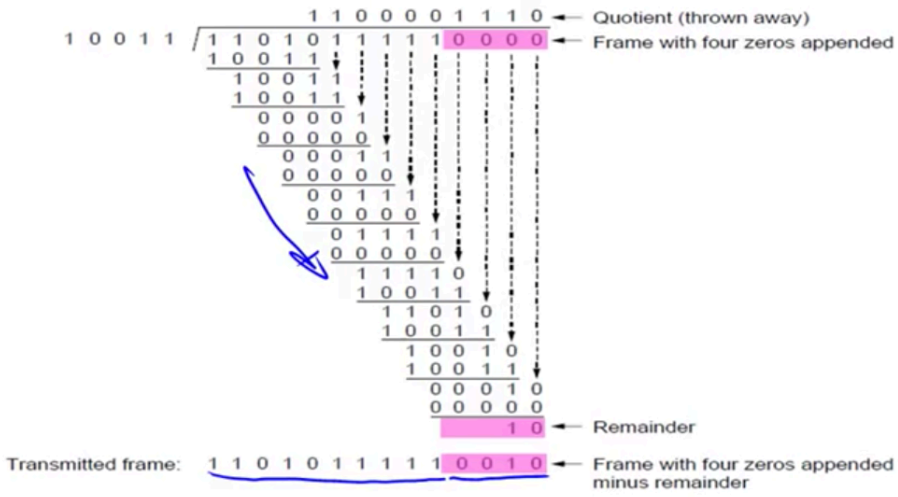


图 79: CRCs

- Receive Procedure:
  - 1. Divide and check for zero remainder
- Protection depend on generator
  - Standard CRC-32 is 10000010 01100000 10001110 110110111
- Properties:
  - HD=4, detects up to triple bit errors

  - Also odd number of errors
  - And bursts of up to k bits in error
  - Not vulnerable to systematic errors like checksums

18.6 Error Detection in Practice

- CRCs are widely used on links
  - Ethernet, 802.11, ADSL, Cable ...
- Checksum used in Internet
  - IP, TCP, UDP ... but it is weak
- Parity
  - Is little used

19 Week 2 - Error Correction

19.1 Topic of Error Correction

- Some bits may be received in error due to noise. How do we fix them?
  - Hamming code
  - Other code
- And why should we use detection when we can use correction?

19.2 Why Error Correction is Hard

- If we had reliable check bits we could use them to narrow down the position of the error

- Then correction would be easy

- But error could be in the check bits as well as data bits!
  - Data might even be correct

19.3 Intuition for Error Correcting Code

- Suppose we construct a code with a Hamming distance of at least 3
  - Need $3 \geqslant 3$ bit errors to change one valid codeword into another
  - Single bit errors will be closest to a unique valid codeword
- If we assume errors are only 1 bit, we can correct them by mapping an error to the closest valid codeword
  - Works for d errors if $HD \geqslant 2d + 1$
- Visualization of code: See Figure **??**

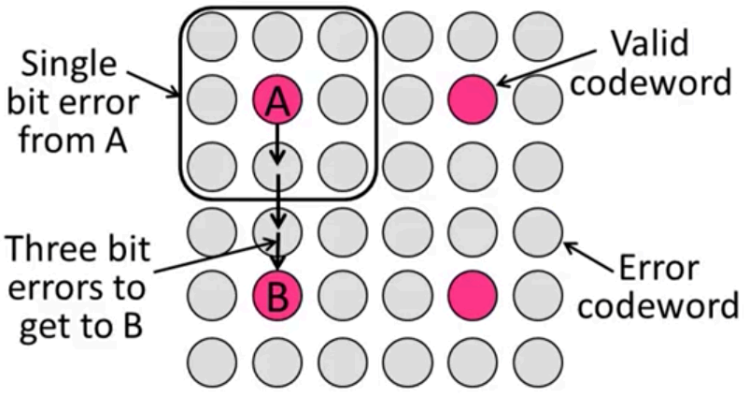


图 80: Intuition

19.4 Hamming Code

- Gives a method for constructing a code with a distance of 3
  - Use an $n = n^k - k - 1$, e.g., n=4, k =3
  - Put check bits in positions p that are powers of 2, starting with position 1
  - Check bit in position p is parity of positions with a p term in their values
- Plus an easy way to correct [soon]
- Example: data=0101, 3 check bits , See Figure **??**
  - 7 bit code, check bit positions 1, 2, 4
  - Check 1 covers positions 1, 3, 5, 7
  - Check 2 covers positions 2, 3, 6, 7
  - Check 4 covers positions 4, 5, 6, 7
  - 如上所示分配，比如说 1，给他分配 1，3，5，7 这几个都能出现在第一位上的 1；再比如说 2，给他分配 2，3，6，7，给他分配都能在第二位上出现 1。这么做便于定位错误所在，观察图 **??**

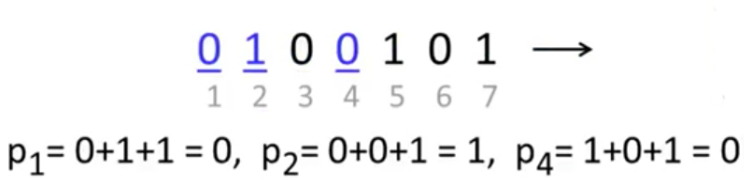


图 81: Hamming Code Example 1

- To decode:

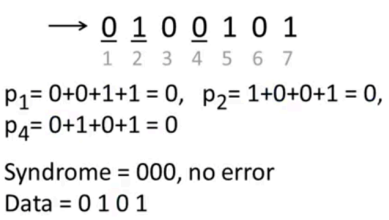


图 82: Hamming Code Example 2

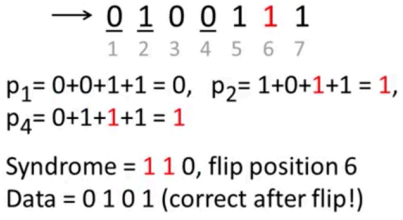


图 83: Hamming Code Example 3

  - Recompute check bits (with parity sum including the check bit)
  - Arrange as a binary number
  - Value (syndrome) tells error position
  - Value of zero means on error
  - Otherwise, flip bit to correct

- Example, continued, See Figure **??** and Figure **??**

19.5 Other Error Correction Code - LDPC

- Codes used in practice are much more involved then Hamming
- Convolutional codes (§3.2.3)
  - Take a stream of data and output a mix of the recent input bits
  - Makes each output bit less fragile
  - Decode using Viterbi algorithm (which can use bit confidence values)
- Low Density Parity Check (§3.2.3)
  - LDPC based on sparse matrices
  - Decoded iteratively using a belief propagation algorithm

  - State of the art today

- Invented by Robert Gallager in 1963 as part of his PhD thesis
  - Promptly forgotten until 1996 ...

19.6 Detection vs. Correction

- Which is better will depend on the pattern of errors. For example:
  - 1000 bit messages with a bit error rate (BER) of 1 in 10000
- Which has less overhead?
- 1. Assume bit errors are random
  - Messages have 0 or maybe 1 error
- Error correction:
  - Need 10 check bits per message
  - Overhead: 10
- Error detection:
  - Need 1 check bits per message plus 1000 bit retransmission 1/10 of the time
  - Overhead: $1 + \frac{1000}{10} \sim 101 bits$
- 2. Assume error come in bursts of 100
  - Only 1 or 2 messages in 1000 have errors
- Error correction:
  - Need » 100 check bits per message

  - Overhead: > 100?

- Error detection:
  - Need 32? check bits per message plus 1000 bit resend 2/1000 of the time
  - Overhead: $32 + \frac{1000}{1000} \times 2 \sim 34\ bits$

- Error correction:
  - Needed when errors are expected
  - Or when on time for retransmission

- Error detection:
  - More efficient when errors are not expected
  - And when errors are large when then do occur

19.7 Error Correction in Practice

- Heavily used in physical layer
  - LDPC is the future, used demanding links like 802.11, DVB, WiMAX, LTE, power-line, ...
  - Convolutional codes widely used in practice

- Error detection (w/ retransmission) is used in the link layer and above for residual errors

- Correction also used in the application layer
  - Called Forward Error Correction (FEC)
  - Normally with an erasure error model
  - E.g., Reed-Solomon (CDs, DVDs, etc.)

20 Week 3 - Overview of the Link Layer

略

21 Week 3 - Retransmissions

21.1 Topic of Retransmissions

- Two strategies to handle errors:
  - 1. Detect errors and retransmit frame (Automatic Repeat reQuest, QRQ)
  - 2. Correct errors with an error correcting code ←Done this

21.2 ARQ

- ARQ often used when errors are common or must be corrected
  - E.g., WiFi, and TCP(later)
- Rules at sender and receiver:
  - Receiver automatically acknowledges correct frames with an ACK
  - Sender automatically resends after a timeout, until an ACK is received
  - Normal operation (no less) , See Figure **??**
  - Loss and retransmission, See Figure **??**

21.3 So What's Tricky About ARQ

- Two non-trivial issues:

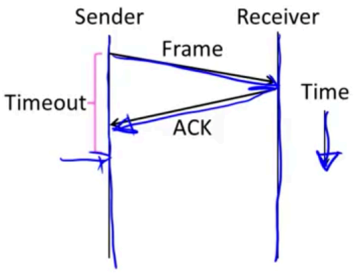


图 84: Normal operation (no less)

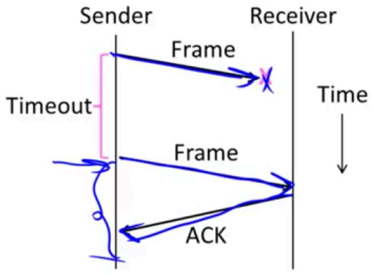


图 85: Loss and retransmission

  - How long to set the timeout?
  - How to avoid accepting duplicate frames as new frame

- Want performance in the common case and correctness always

21.4 Timeouts

- Timeout should be:
  - Not too big (link goes idle)
  - Not too small (spurious resend)
- Fairly easy on a LAN
  - Clear worst case, little variation
- Fairly difficult over the Internet
  - Much variation, on obvious bound
  - We'll revisit this with TCP (later)

21.5 Duplicates

- What happens if an ACK is lost? See Figure **??**

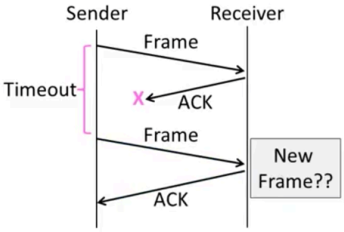


图 86: What happens if an ACK is lost?

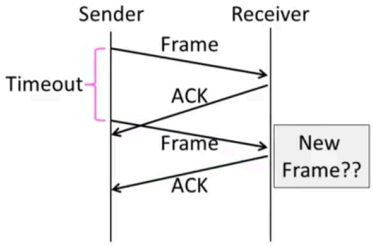


图 87: the timeout is early?

- the timeout is early? See Figure **??**

21.6 Sequence Numbers

- Frames and ACKs must both carry sequence numbers for correctness
- To distinguish the current frame from the next one, a single bit (two numbers) is sufficient
  - Called Stop-and-Wait
- In the normal case: See Figure **??**
- With ACK loss: See Figure **??**
- With early timeout: See Figure **??**

21.7 Limitation of Stop-and-Wait

- It allows only a single frame to be outstanding from the sender:
  - Good for LAN, not efficient for high BD (bandwidth-delay) , See Figure **??**

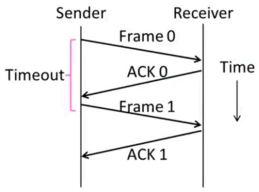


图 88: In the normal case

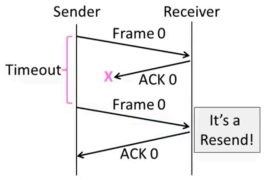


图 89: With ACK loss

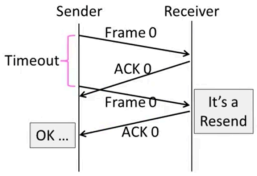


图 90: With early timeout

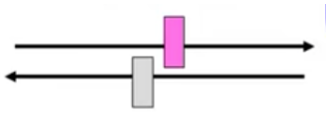


图 91: Limitation of Stop-and-Wait

- Ex: R=1 Mbps, D= 50 ms
  - How many frames/sec? If R=10 Mbps? $\approx 100\ Kpbs$

21.8 Sliding Window

- Generalization of stop-and-stop
  - Allows W frames to be outstanding
  - Can send W frames per RTT(Round Trip Time) (=2D) , See Figure **??**

图 92: Sliding Window

  - Various options for numbering frames/ACKs and handling loss

* Will look at long with TCP (later)

22 Week 3 - Multiplexing

22.1 Topic of Multiplexing

- Multiplexing is the network word for the sharing of resource
- Classic scenario is sharing a link among different users
  - Time Division Multiplexing (TDM)
  - Frequency Division Multiplexing

22.2 Time Division Multiplexing (TDM)

- Users take turns on fixed schedule, See Figure **??**

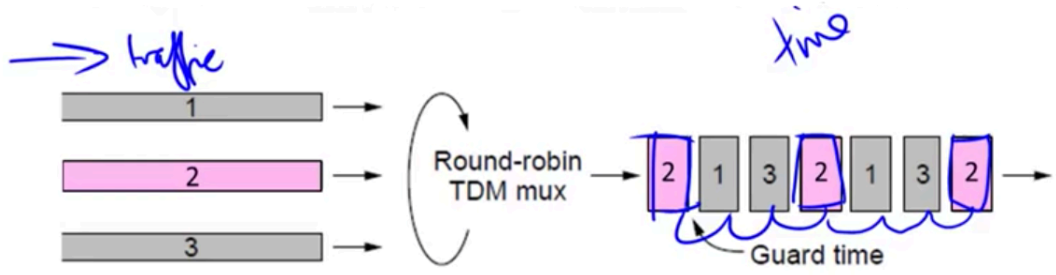


图 93: Time Division Multiplexing (TDM)

22.3 Frequency Division Multiplexing (FDM)

- Put different users on different frequency bands,See Figure **??**

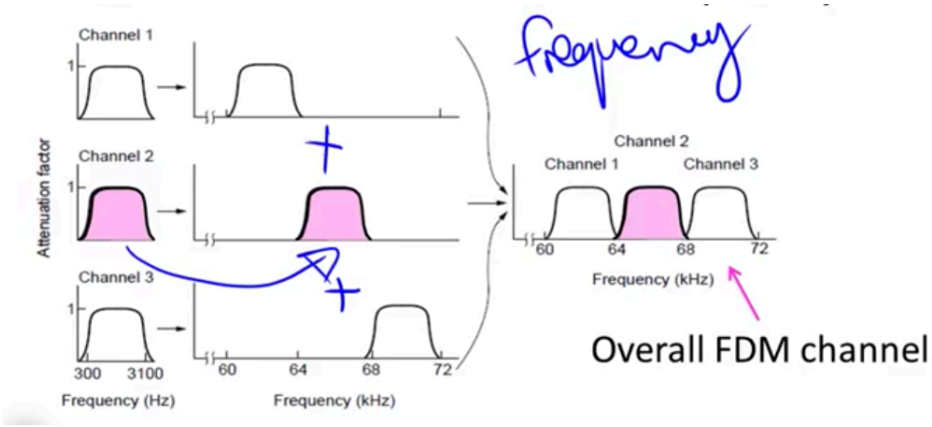


图 94: Frequency Division Multiplexing (FDM)

22.4 TDM versus FDM

- In TDM a user sends at a high rate a fraction of the time; in FDM, a user sends at a low rate all the time, See Figure **??**

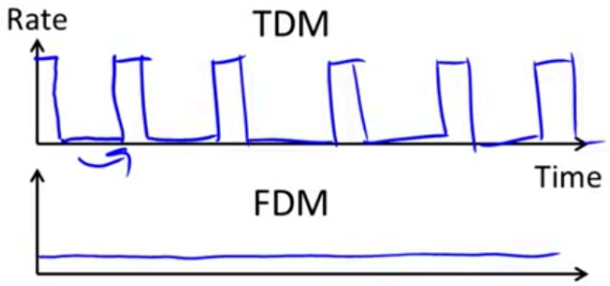


图 95: TDM versus FDM

22.5 TDM/FDM Usage

- Statically divide a resource
  - Suited for continuous traffic, fixed number of users

- Widely used in telecommunications
  - TV and radio stations (FDM)
  - GSM (2G cellular) allocates calls using TDM within FDM

22.6 Multiplexing Network Traffic

- Network traffic is bursty
  - Inefficient to always allocate user their ON needs with TDM/FDM , See Figure **??**

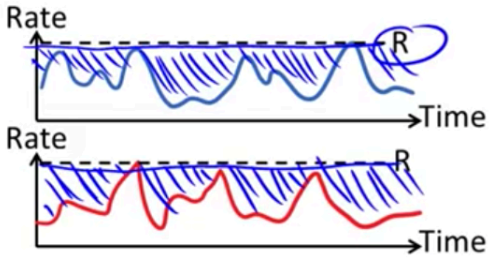


图 96: Inefficient to always allocate user their ON needs with TDM/FDM

- Multiple access schemes multiplex users according to their demands - for gains of statistical multiplexing , See Figure **??**

22.7 Multiple Access

- We will look at two kinds of multiple access protocols
- 1. Randomized. Nodes randomize their resource access attempts
  - Good for low load situations

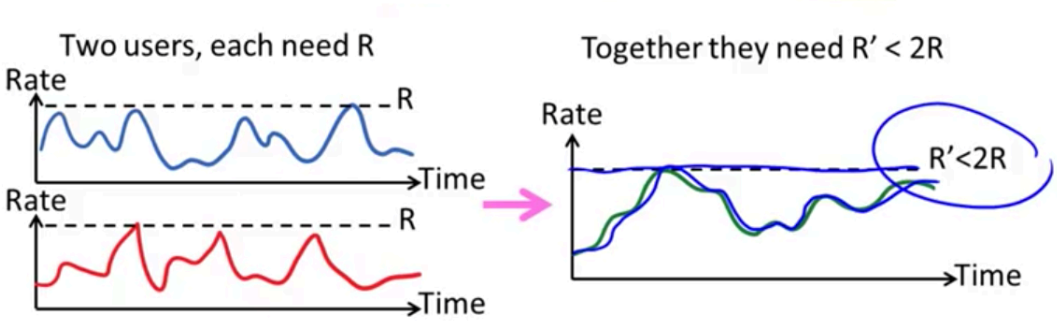


图 97: Multiple access schemes multiplex users according to their demands

- Contention-free. Nodes order their resource access attempts. 802.11
  - Good for high load or guaranteed quality of service situations

23 Week 3 - Randomized Multiple Access

23.1 Topic of Randomized Multiple Access

- How do nodes share a single link? Who sends when, e.g., in WiFi ?
  - Explore with a simple model
- Assume on-one is in charge; this is a distributed system
- Will explore random multiple access control (MAC) protocols
  - This is the classic Ethernet
  - Remember: data traffic is bursty

23.2 ALOHA Network

- Seminal computer network connecting the Hawaiian islands in the late 1960s , See Figure **??**

- When should nodes send?
- A new protocol was devised by Norm Abramson ...

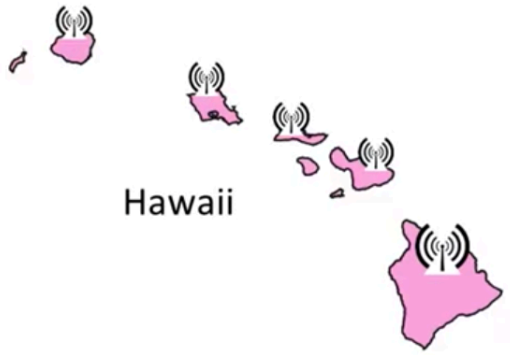


图 98: ALOHA Network

23.3 ALOHA Protocol

- Simple idea:
  - Node just sends when it has traffic.
  - If there was a collision (no ACK received) then wait a random time and resend
- That's it !
- Some frames will be lost, but many get through ... See Figure **??**
- good idea ?
- Simple, decentralized protocol that works well under low load !
- Not efficient under high load
  - Analysis shows at most 18% efficiency

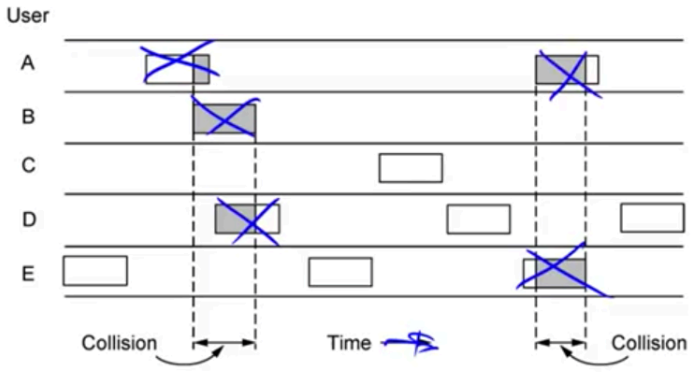


图 99: ALOHA Protocol

- Improvement: divide time into slots and efficiency goes up to 36%

- We'll look at other improvements

23.4 Classic Ethernet

- ALOHA inspired Bob Metcalfe to invent Ethernet for LANs in 1973 , See Figure **??**
  - Nodes share 10 Mbps coaxial cable
  - Hugely popular in 1980s, 1990s

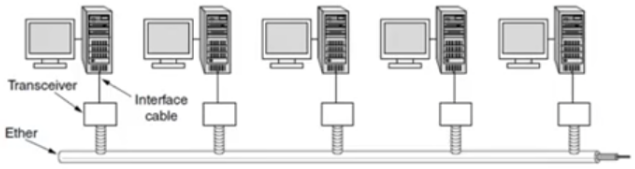


图 100: Classic Ethernet

23.5 CSMA (Carrier Sense Multiple Access) 载波监听多路访问协议

- Improve ALOHA by listening for activity before we send (Doh!)
  - Can do easily with wires, not wireless
- So does this eliminate collisions ?
  - Why or why not ?
- Still possible to listen and hear nothing when another node is sending because of delay
- CSMA is a good defense against collisions only when BD is small $\ll$ 1 *packet*

23.6 CSMA/CD (with Collision Detection)

- Can reduce the cost of collisions by detecting them and aborting (Jam) the rest of the frame time, See Figure **??**
  - Again, we can do this with wires

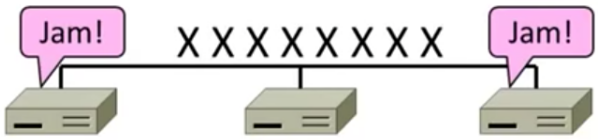


图 101: CSMA/CD (with Collision Detection

23.7 CDMA/CD Complications

- Want everyone who collides to know that it happened
  - Time window in which a node may hear of a collision is 2D second , **??**

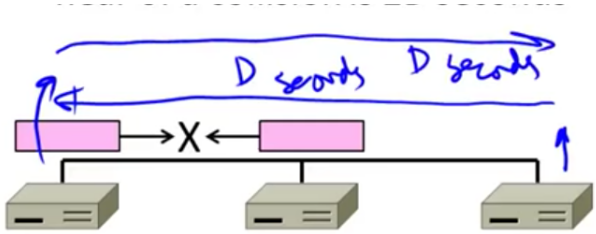


图 102: CDMA/CD Complications 1

- Impose a minimum frame size that lasts for 2D seconds , See Figure **??**
  - So node can't finish before collision
  - Ethernet minimum frame is 64 bytes

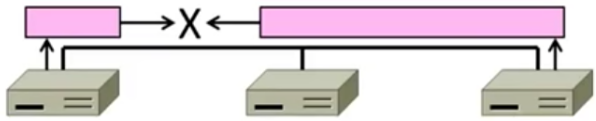


图 103: CDMA/CD Complications 2

23.8 CSMA "Persistence"

- What should a node do if another node is sending? See Figure **??**
- Ida: Wait until it is done, and send

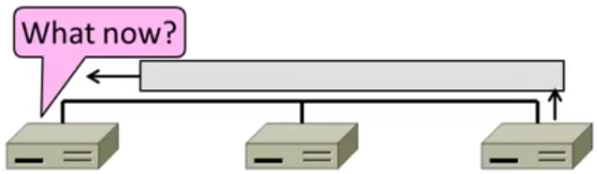


图 104: CSMA "Persistence" 1

- Problem is that multiple waiting nodes will queue up then collide, See Figure **??**
  - More load, more of a problem

图 105: CSMA "Persistence" 2

- Intuition for a better solution
  - If there are N queued senders, we want each to send next with probability 1/N, See Figure **??**

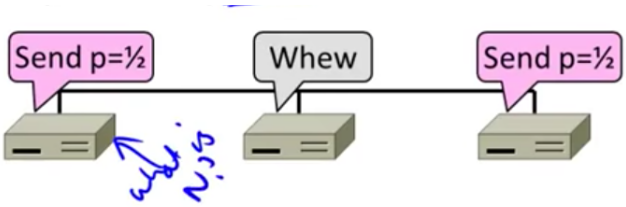


图 106: CSMA "Persistence" 3

23.9 Binary Exponential Backoff (BEB)

- Cleverly estimates the probability
  - 1st collision, wait 0 or 1 frame times
  - 2nd collision, wait from 0 to 3 times
  - 3rd collision, wait from 0 to 7 times ...
- BEB doubles interval for each successive collision
  - Quickly gets large enough to work
  - Very efficient in practice

23.10 Classic Ethernet, or IEEE 802.3

- Most popular LAN of the 1980s, 1990s, See Figure **??**
  - 10 Mbps over shared coaxial cable, with baseband signals
  - Multiple access with "1-persistent CSMA/CD with BEB"

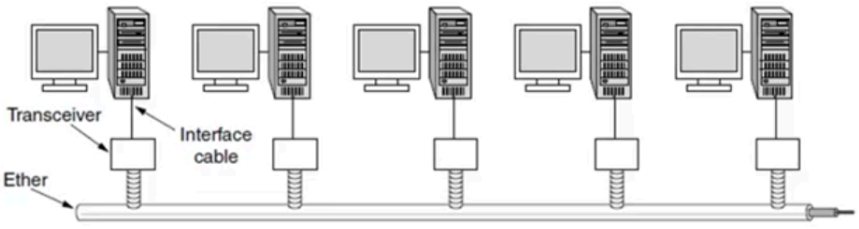


图 107: Classic Ethernet, or IEEE 802.3

23.11 Ethernet Frame Format

- Has addresses to identify the sender and receiver

- CRC-32 for error detection; no ACKs or retransmission
- Start of frame identified with physical layer preamble
- See Figure ??

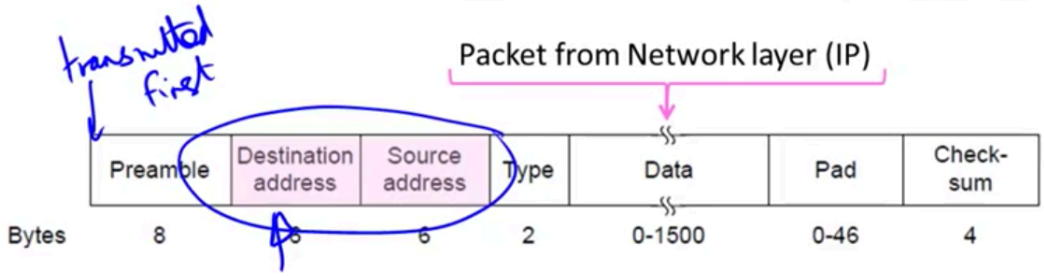


图 108: Ethernet Frame Format

23.12 Modern Ethernet

- Based on switches, not multiple access, but still called Ethernet, See Figure ??
  - We'll get to it in a later segment

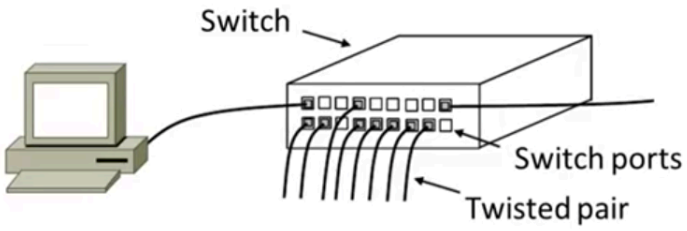


图 109: Modern Ethernet

24 Week 3 - Contention Free Multiple Access

24.1 Topic of Contention Free Multiple Access

- A new approach to multiple access , See Figure **??**
  - Based on turns, not randomization

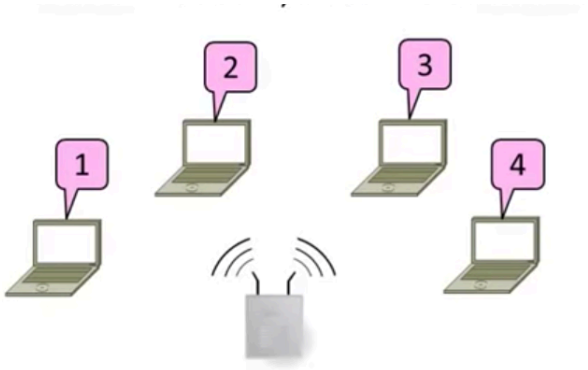


图 110: Topic of Contention Free Multiple Access

24.2 Issues with Random Multiple Access

- CSMA is good under low load:
  - Grants immediate access
  - Little overhead (few collisions)
- But not so good under high load:
  - High overhead (expect collisions)
  - Access time varies (lucky/unlucky)
- We want to do better under load !

24.3 Turn-Taking Multiple Access Protocols

- They define an order in which nodes get a chance to send
  - Or pass, if no traffic at present
- We just need some ordering ...
  - E.g., Token Ring
  - E.g., node addresses

24.4 Token Ring

- Arrange nodes in a ring; token rotates "permission to send" to each node in turn, See Figure **??**

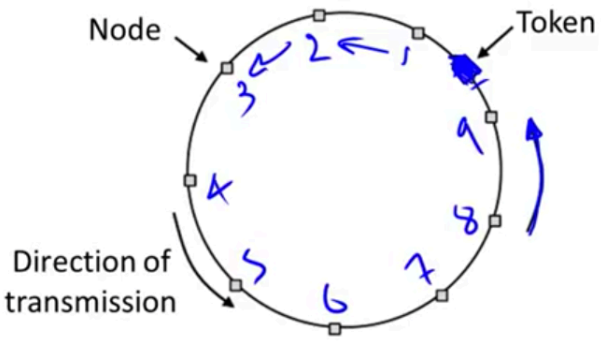


图 111: Token Ring

24.5 Turn-Taking Advantages

- Fixed overhead with no collisions
  - More efficient under load
- Regular chance to send with no unlucky nodes

- Predictable service, easily extended to guaranteed quality of service

24.6 Turn-Taking Disadvantages

- Complexity
- More things that can go wrong then random access protocols!
  - E.g., what if the token is lost ?
- Higher overhead at low load

25 Week 3 - Wireless Multiple Access

25.1 Wireless Complications

- Wireless is more complicated than the wired case (Surprise!)
  - 1. Nodes may have different areas of coverage - doesn't fit Carrier Sense
  - 2. Nodes can't hear while sending - can't Collision Detect
  - WiFi $\neq$ CSMA/CD

25.2 Different Coverage Areas

- Wireless signal is broadcast and received nearby, where is sufficient SNR, See Figure **??**

25.3 Hidden Terminals

- Nodes A and C are hidden terminals when seeding to B, See Figure **??**

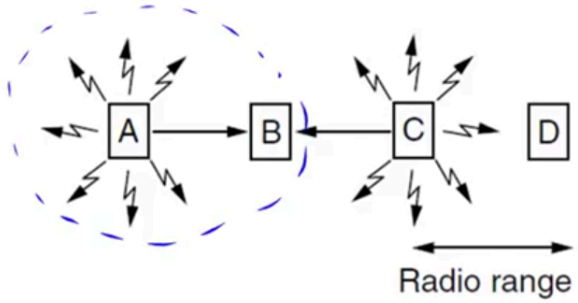


图 112: Different Coverage Areas

- Can't hear each other (to coordinate) yet collide at B
- We want to avoid the inefficiency of collisions

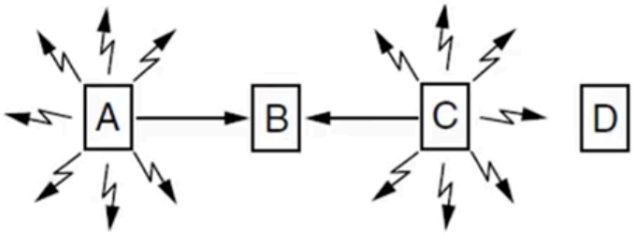


图 113: Hidden Terminals

25.4 Exposed Terminals

- B and C are exposed terminals when sending to A and D , See Figure **??**
  - Can hear each other yes don't collide at receivers A and D
  - We want to send concurrently to increase performance

25.5 Nodes Can't Hear While Sending

- With wires, detecting collisions (and aborting) lowers their cost

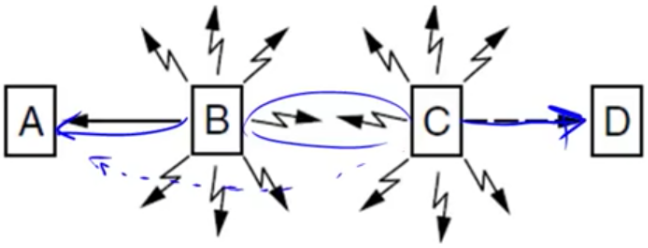


图 114: Exposed Terminals

- More wasted time the wireless
- See Figure **??**

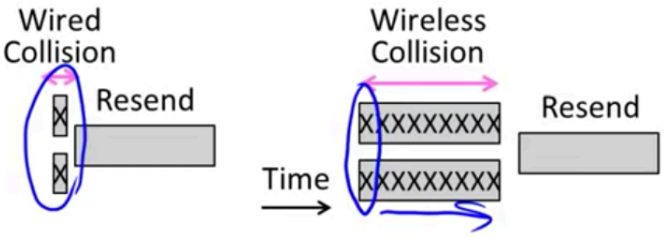
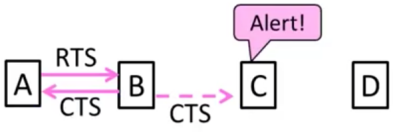


图 115: Nodes Can't Hear While Sending

25.6 Possible Solution: MACA

- MACA uses a short handshake instead of CSMA (Karn, 1990)
  - 802.11 uses a refinement of MACA (later)
- Protocol rules:
  - 1. A sender node transmits a RTS (Request-To-Send, with frame length)



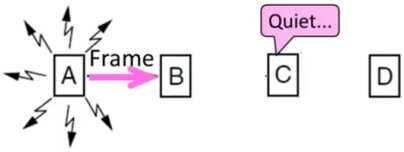


图 116: MACA - Hidden Terminals 1 图 117: MACA - Hidden Terminals 3 and 2

  - 2. The recerver replies with a CTS (Clear-To-Send, with frame length)
  - 3. Sender transmits the frame while nodes hearing the CTS stay silent
  - - Collisions on the RTS/CTS are still possible, but less likely

25.7 MACA - Hidden Terminals

- A → B with hidden terminal C
  - 1. A sends RTS, to B
  - 2. B sends CTS, to A, and C too
  - See Figure **??**
  - A sends frame while C defers
  - See Figure **??**
- $B \rightarrow A$, $C \rightarrow D$ as exposed terminals
  - B and C send RTS to A and D
  - See Figure **??**
  - See Figure **??**

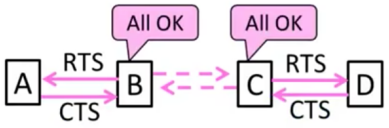


图 118: B and C send RTS to A and D 第一个图

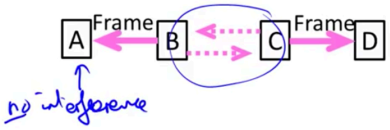


图 119: B and C send RTS to A and D 第二个图

25.8 802.11, or WiFi

- Very popular wireless LAN started in the 1990s
- Clients get connectivity from a (wired) AP (Access Point)
- It's a multi-access problem
- Various flavors have been developed over time
  - Faster, more features
- See Figure **??**

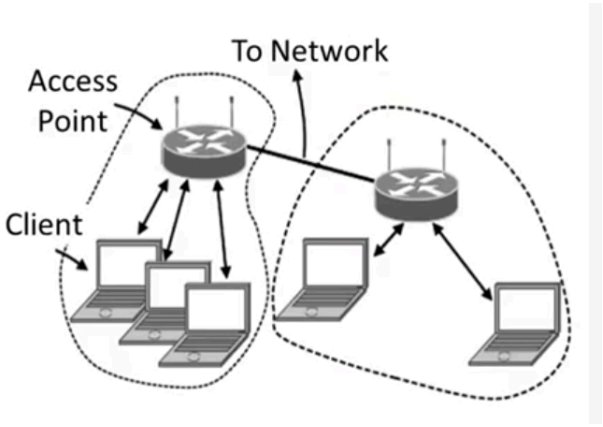


图 120: 802.11, or WiFi

25.9 802.11 Physical Layer

- Uses 20/40 MHz channels on ISM bands
  - 802.11 b/g/n on 2.4 GHz
  - 802.11 a/n on 5 GHz
- OFDM modulation (except legacy 802.11b)
  - Different amplitudes/phases for varying SNRs
  - Rates from 6 to 54 Mbps plus error correction
  - 802.11n uses multiple antennas; see "802.11 with Multiple Antennas for Dummies"

25.10 802.11 Link Layer

- Multiple access uses CSMA/CA (next); RTS/CTS optional
- Frames are ACKed and retransmitted with ARQ
- Funky addressing (three address!) due to AP
- Errors are detected with a 32-bit CRC
- Many, many features (e.g., encryption, power save)
- See Figure **??**

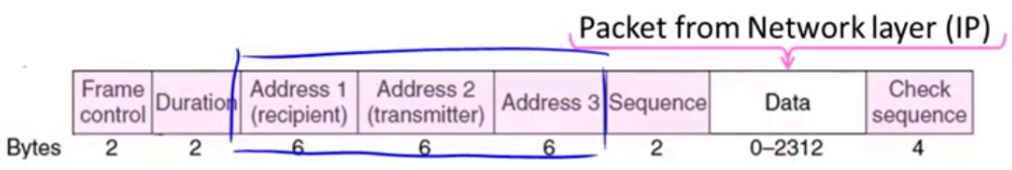


图 121: 802.11 Link Layer

25.11 802.11 CSMA/CA for Multiple Access

- Sender avoids collisions by inserting small random gaps
  - E.g., when both B and C send, C picks a smaller gap, goes first
  -
  - See Figure **??**

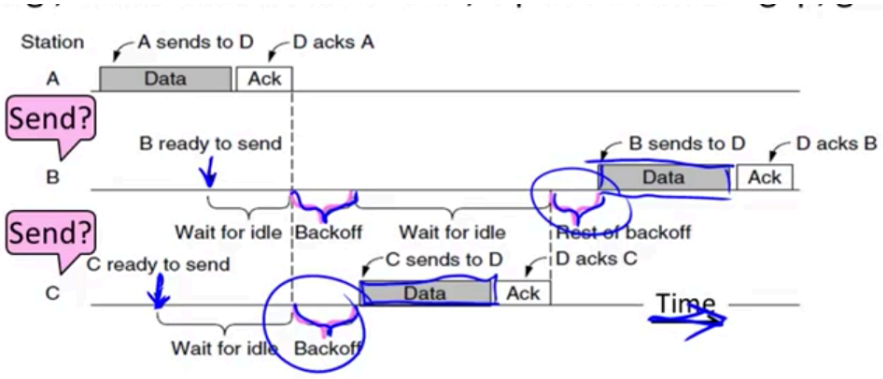


图 122: 802.11 CSMA/CA for Multiple Access

25.12 The Future of 802.11 (Guess)

- Likely ubiquitous for Internet connectivity
  - Greater diversity, from low-to high-end devices
- Innovation in physical layer drives speed
  - And power-efficient operation too
- More seamless integration of connectivity
  - Too manual now, and limited (e.g., device-to-device)

26 Week 3 - LAN Switches

26.1 Topic of LAN Switches

- How do we connect nodes with a switch instead of multiple access
  - Use multi links/wires
  - Basis of modern (switched) Ethernet, See Figure **??**

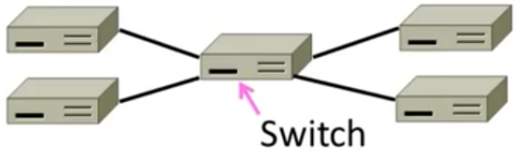


图 123: Topic of LAN Switches

26.2 Switched Ethernet

- Hosts are wired to Ethernet switches with twisted pair
  - Switch serves to connect the hosts
  - Wires usually run to a closet
  - See Figure **??**

26.3 What's in the box ?

- Remember from protocol layers:
- See Figure **??**

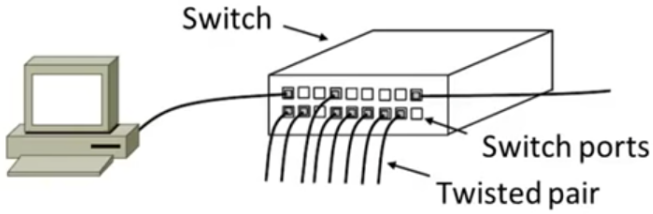


图 124: Switched Ethernet

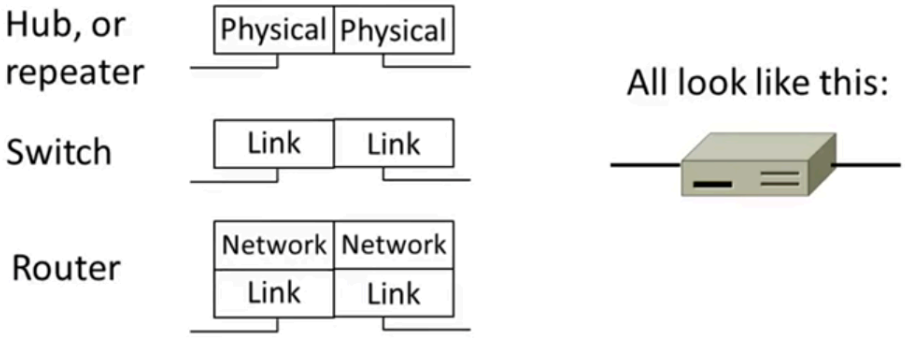


图 125: What's in the box ?

26.4 Inside a Hub

- All ports are wired together; more convenient and reliable than a single shared wire
- See Figure **??**

26.5 Inside a Switch

- Uses frame addresses to connect input to the right output port; multiple frames may be switched in parallel
- Port may be used for both input and output (full-duplex)

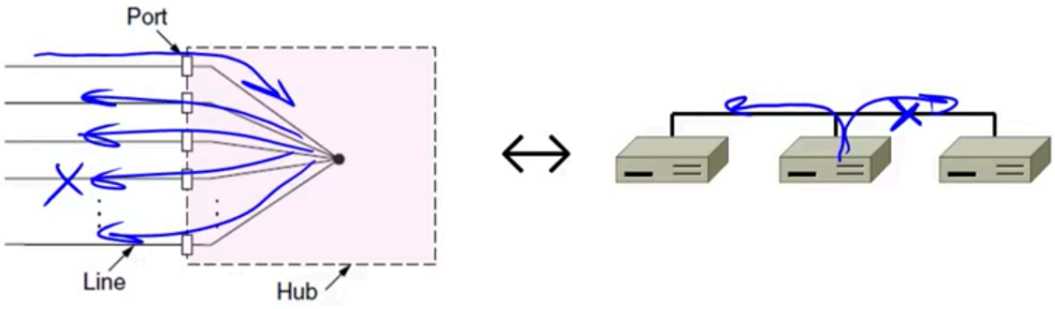


图 126: Inside a Hub

  - Just send, no multiple access protocol

- See Figure **??**

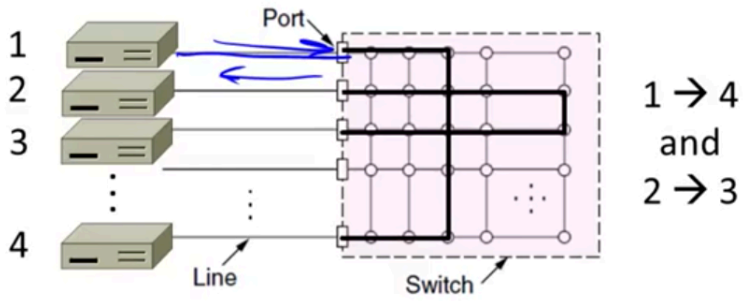


图 127: Inside a Switch 1

- Need buffers for multiple inputs to send to one output
- Sustained overload will fill buffer and lead to frame loss
- See Figure **??**

26.6 Advantages of Switches

- Switches and hubs have replaced the shared cable of classic Ethernet

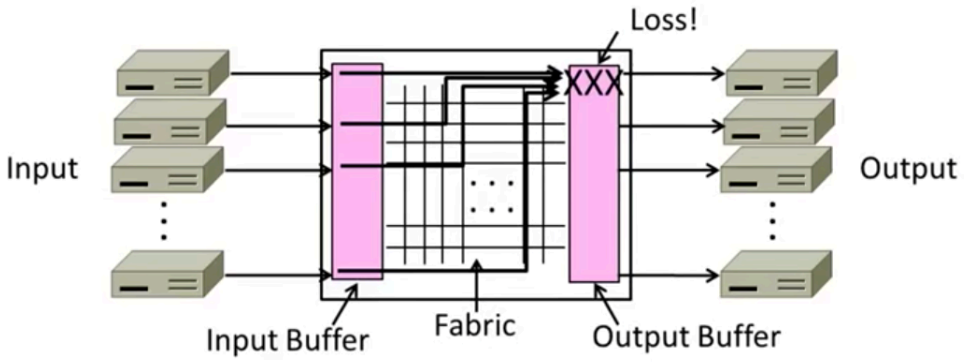


图 128: Inside a Switch 2

  - Convenient to run wires to one location
  - More reliable; wire cut is not a single point of failure that is hard to find

- Switches offer scalable performance
  - E.g., 100 Mbps per port instead of 100 Mbps for all nodes of shared cable / hub

26.7 Switch Forwarding

- Switch needs to find the right output port for the destination address in the Ethernet frame. How?
  - Want to let hosts be moved around readily; don't look at IP
- See Figure **??**

26.8 Backward Learning

- Switch forwards frames with a port/address table as follow:

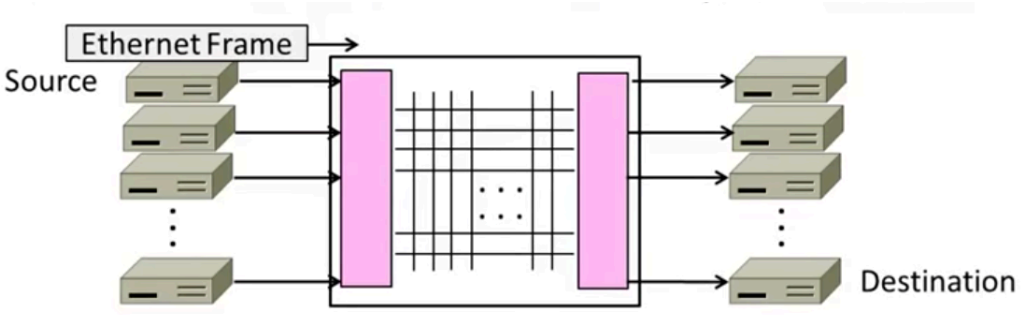


图 129: Switch Forwarding

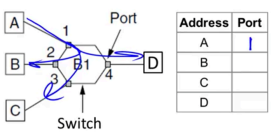


图 130: 1: A sends to D

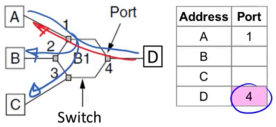


图 131: 2: D sends to A

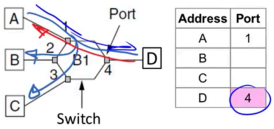


图 132: 3: A sends to D

- 1. To fill the table, it looks at the source address of input frames
- 2. To forward, it sends to the port, or else broadcasts to all ports

- 1: A sends to D , See Figure **??**
- 2: D sends to A , See Figure **??**
- 3: A sends to D , See Figure **??**

26.9 Learning with Multiple Switches

- Just works with multiple switches and a mix of hubs *assuming no loops*, e.g., A sends to D then D sends to A
- See Figure **??** and Figure **??**

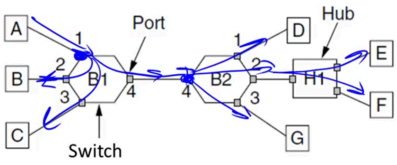


图 133: A sends to D

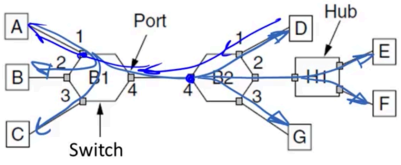


图 134: D sends to A

27 Week 3 - Switch Spanning Tree

27.1 Topic of Switch Spanning Tree

- How can we connect switches in any topology so then just work
  - This is part 2 of switched Ethernet
  - See Figure **??**

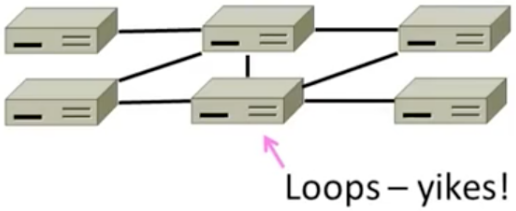


图 135: Topic of Switch Spanning Tree

27.2 Problem - Forwarding Loops

- May have a loop in the topology
  - Redundancy in case of failures
  - Or a simple mistake
- Want LAN switches to ”just work”

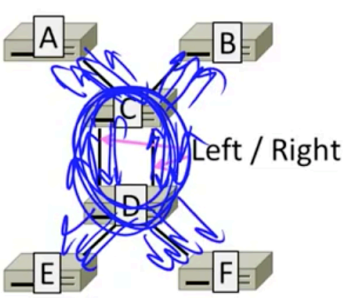


图 136: Problem - Forwarding Loops 1

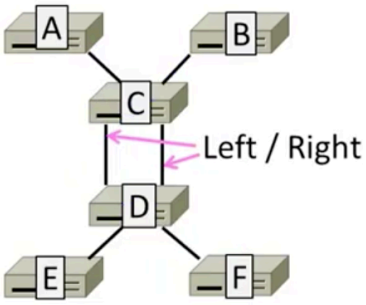


图 137: Problem - Forwarding Loops 2

  - Plug-and-play, no changes to hosts
  - But loops cause a problem ...

- Suppose the network is started and A sends to F. What happens ? See Figure **??** and Figure **??**
  - A → B → B, D-left, D-right
  - D-left → C-right, E, F
  - D-right → C-left, E, F
  - C-right → D-left, A, B
  - C-left → D-right, A, B
  - D-left → ...
  - D-right → ...

27.3 Spanning Tree Solution

- Switches collectively find a spanning tree for the topology , See Figure **??**

- A subset of links that is a tree (no loops) and reaches all switches
- They switches forward as normal on the spanning tree
- Broadcasts will go up to the root of the tree and down all the branches

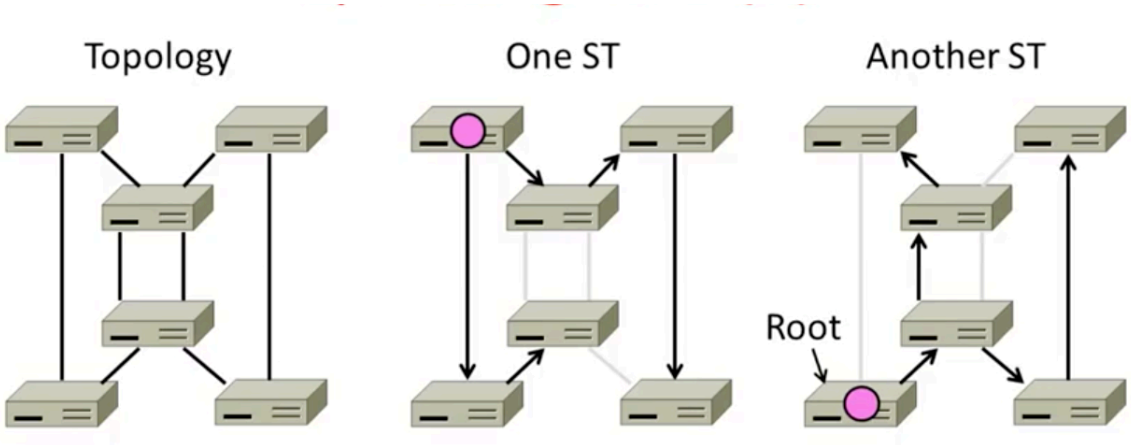


图 138: Spanning Tree Solution

27.4 Spanning Tree Algorithm

- Rules of the distributed game:
  - All switches run the same algorithm
  - They start with no information
  - Operate in parallel and send messages
  - Always search for the best solution
- Ensures a highly robust solution
  - Any topology, with no configuration
  - Adapts to link/switch failures, ...

- Outline:
  - 1. Elect a root node of the tree (switch with the lowest address)
  - 2. Grow tree as shortest distances from the root (using lowest address to break distance ties)
  - 3. Turn off ports for forwarding if they aren't on the spanning tree
- Details:
  - Each switch initially believes it is the root of the tree
  - Each switch sends periodic updates to neighbors with:
    - Its address, address of the root, and distance (in hops) to root
  - Switches favors ports with shorter distances to lowest root
    - Uses lowest address as a tie for distances
- See Figure **??**

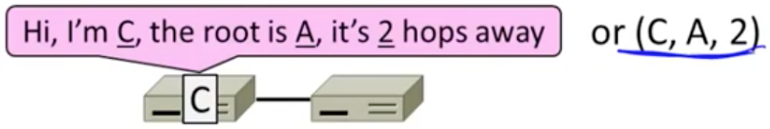
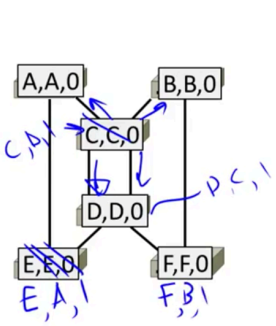


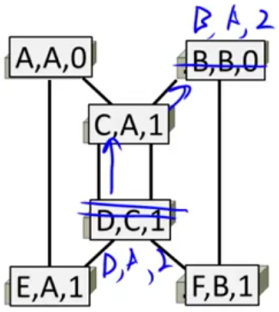
图 139: Spanning Tree Algorithm

27.5 Spanning Tree Example

- 1^{st} round, sending:
  - A sends (A,A, 0) to say it is root
  - B, C, D, E, and F do likewise

- 1^{st} round, receiving:
  - A still thinks is it (A, A, 0)
  - B still thinks (B, B, 0)
  - C updates to (C, A, 1)
  - D updates to (D, C, 1)
  - E updates to (E, A, 1)
  - F updates to (F, B, 1)
- 这是以 C 的视角来看的，A 和 B 一直坚持他们的地位，然后 A 与 D 相对于 E 的距离一样，但是就 A 为 (A,A,0), 而 D 为 (C,D,1). 此时 E 更倾向于与 A 建立链接，此时 E 为 (E, A, 1).
- See Figure **??**
- 2^{nd} round, sending
  - Nodes send their updated state
- 2^{nd} round receiving:
  - A remains (A, A, 0)
  - B updates to (B, A, 2) via C
  - C remains (C, A, 1)
  - D updates to (D, A, 2) via C
  - E remains (E, A, 1)
  - F remains (B, B, 1)
- 这时候，B 从 C 中得知它并不是最佳的人选，A 要更胜一筹，所以它也改变了自身的状态。





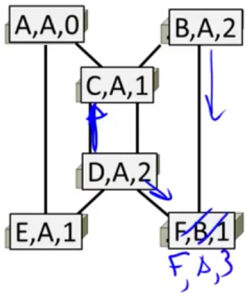


图 140: Spanning Tree Example 第一幅图

图 141: Spanning Tree Example 第二幅图

图 142: Spanning Tree Example 第三幅图

- See Figure **??**
- 3^{rd} round, sending
  - Nodes send their updated state
- 3^{rd} round receiving:
  - A remains (A, A, 0)
  - B updates to (B, A, 2) via C
  - C remains (C, A, 1)
  - D updates to (D, A, 2) via C
  - E remains (E, A, 1)
  - F updates to (F, A, 3) via B
- See Figure **??**
- 4^{th} round
  - Steady-state has been reached

  - Nodes turn off forwarding that is not on the spanning tree

- Algorithm continues to run
  - Adapts by timing out information
  - E.g., if A fails, other nodes forget it, and B will became the new root

- See Figure **??**
- Forwarding proceeds as usual on the ST
- Initially D sends to F
  - D $\rightarrow$ C-left
  - C $\rightarrow$ A, B
  - A $\rightarrow$ E
  - B $\rightarrow$ F

- And F sends back to D:
  - F $\rightarrow$ B
  - B $\rightarrow$ C
  - C $\rightarrow$ D
    (hm, not such a great route)

- See Figure **??**

28 Week 4 - Network Layer Overview

28.1 Shortcomings of Switches

- Don't scale to large networks

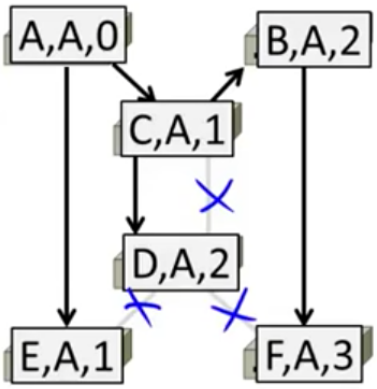


图 143: Spanning Tree Example 第四幅图

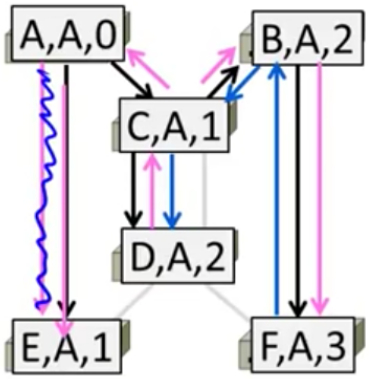


图 144: Spanning Tree Example 第五幅图

  - Blow up of routing table, broadcast

- Don't work across more than one link layer technology
  - Hosts on Ethernet + 3G + 802.11 ...
- Don't give much traffic control
  - Want to plan routes / bandwidth

28.2 Network Layer Approach

- Scaling:
  - Hierarchy, in the form of prefixes
- Heterogeneity:
  - IP for internetworking

- Bandwidth Control:
  - Lowest - cost routing
  - Later QOS (Quality of Service)

28.3 Topics of Network Layer

- Network service models
  - Datagrams (packets), virtual circuits
- IP (Internet Protocol)
  - Internetworking
  - Forwarding (Longest Matching Prefix)
  - Helpers: ARP and DHCP
  - Fragmentation and MTU discovery
  - Errors: ICMP (traceroute!)
- IPv6, the future of IP
- NAT, a ”middlebox”
- Routing algorithms — talk next time

28.4 Routing vs. Forwarding

- Routing is the process of deciding in which direction to send trafic
  - Network wide (global) and expansive
- Forwarding is the process of sending a packet on its way
  - Node process (local) and fast

29 Week 4 - Network Services

29.1 Topic of the Network Services

- What kind of service does the Network layer provide to Transport layer ?
  - How is it implemented at routers ?

29.2 Two Network Service Models

- Datagrams, or connectionless service
  - Like postal letters
  - (This one is IP)
- Virtual circuits, or connection-oriented service
  - Like a telephone call

29.3 Store-and-Forward Packet Switching

- Both models are implemented with store-and-forward packet switching
  - Routers receive a complete packet, storing it temporarily if necessary before forwarding it onwards
  - We use statistical multiplexing to share link bandwidth over time
- Switching element has internal buffering for connection
- See Figure **??**
- Simplified view with per port output buffering
  - Buffer is typically a FIFO (First In First Out) queue

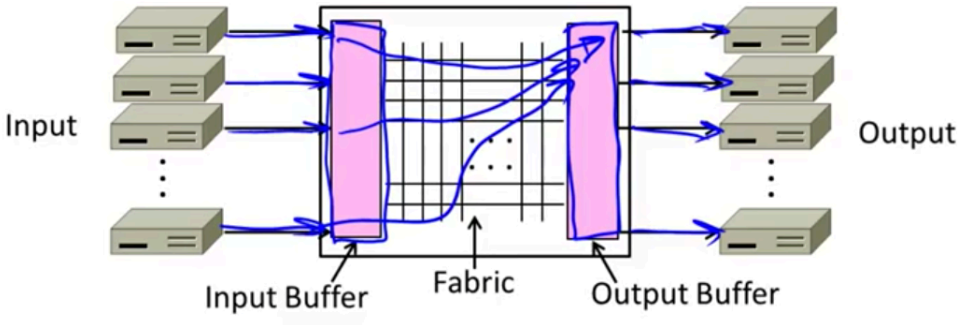


图 145: Switching element has internal buffering for connection

- If full, packets are discarded (congestion, later)

- See Figure **??**

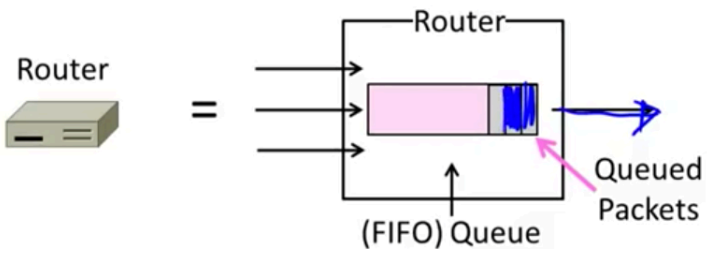


图 146: Simplified view with per port output buffering

29.4 Datagram Model

- Packets contain a destination address; each router uses it to forward each packet, possibly on different paths

- See Figure **??**

- Each router has a forwarding table keyed by address

  - Gives next hop for each destination address; may change

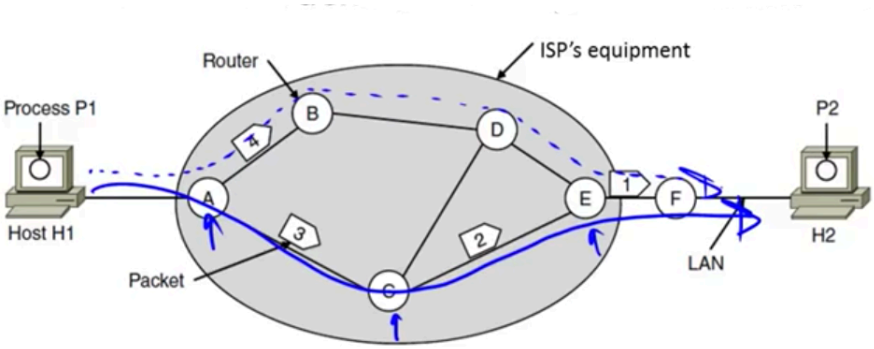


图 147: Datagram Model 1

- See Figure **??**

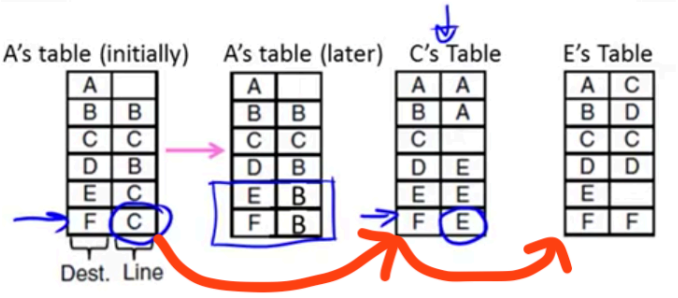


图 148: Datagram Model 2

29.5 IP (Internet Protocol

- Network layer of the Internet, uses datagrams (next)
  - IPv4 carries 32 bit addresses on each packet (often 1.5 KB)
- See Figure **??**

29.6 Virtual Circuit Model

- Three phases:

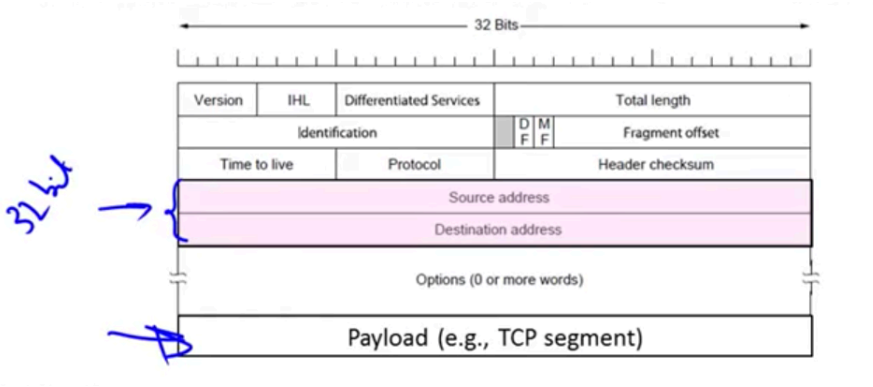


图 149: IP (Internet Protocol

  - 1. Connection establishment, circuit is set up
    - Path is chosen, circuit information stored in routers
  - 2. Data transfer, circuit is used
    - Packets are forwarded along the path
  - 3. Connection teardown, circuit is deleted
    - Circuit information is removed from routers

- Just like a telephone circuit, but virtual in the sense that no bandwidth need be reserved; statistical sharing of links
- Packets only contain a short label to identify the circuit
  - Labels don't have any global meaning, only unique for link
- See Figure **??**
- Each router has a forwarding table keyed by circuit
  - Gives output line and next label to place on packet

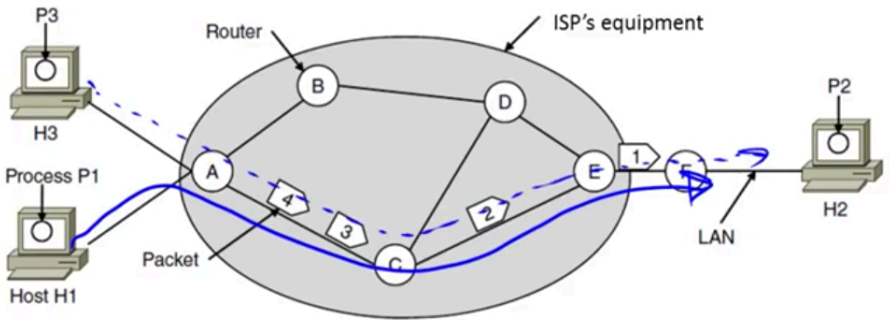


图 150: Virtual Circuit Model 1

- 下图中的数值，1、5，1，1 和 1，2，2，2 表示 Circuit number，个人估计是用来将稳定的传输通道表上序号，比如说 A 与 C 直接的管道，比如 H1 传输的数据包在该段是通过第五号通道进行传输，而 H3 的数据包在该段则通过第二号通道传输。这就是基于连接的信号传输，也是 TCP 的基础。

- See Figure **??**

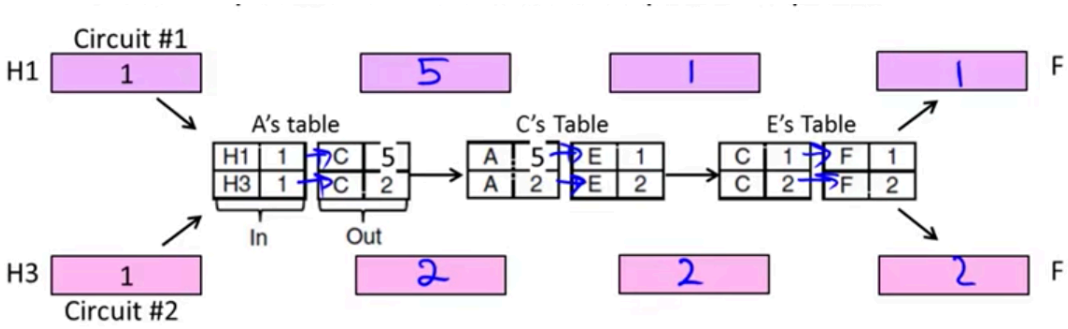


图 151: Virtual Circuit Model 2

29.7 MPLS (Multi-Protocol Label Switching), §5.6.5

- A virtual-circuit like technology widely used by ISPs

  - ISP sets up circuits inside their backbone ahead of time
  - ISP adds MPLS label to IP packet at ingress, undoes at egress

- See Figure **??**

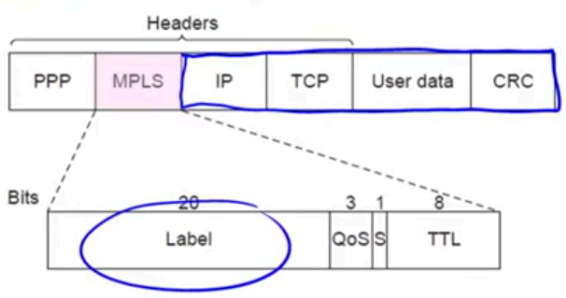


图 152: MPLS (Multi-Protocol Label Switching)

29.8 Datagrams vs. Virtual Circuits

- Complementary strengths
- See Figure **??**

Issue	Datagrams	Virtual Circuits
Setup phase	Not needed	Required
Router state	Per destination	Per connection
Addresses	Packet carries full address	Packet carries short label
Routing	Per packet	Per circuit
Failures	Easier to mask	Difficult to mask
Quality of service	Difficult to add	Easier to add

图 153: Datagrams vs. Virtual Circuits

30 Week 4 - Internetworking

30.1 Topic of the Internetworking

- How do we connect different networks together ?
  - This is called internetworking
  - We'll look at how IP does it

30.2 How Networks May Differ

- Basically, in a lot of ways:
  - Service model (datagram, VCs)
  - Addressing (what kind)
  - QOS (priorities, no priorities)
  - Packet sizes
  - Security (whether encrypted)
- Internetworking hides the differences with a common protocol. (Uh oh.)

30.3 Connecting Datagram and VC networks

- An example to show that it's not so easy
  - Need to map destination address to a VC and vice-versa
  - A bit of a "road bump", e.g., might have to set up a VC
- See Figure **??**

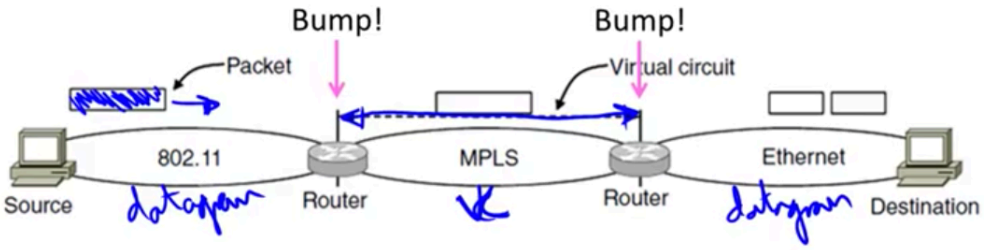


图 154: Connecting Datagram and VC networks

30.4 Internetworking - Cerf and Kahn

- Pioneered by Cerf and Kahn, the ”fathers of the Internet”
  - In 1974, later led to TCP/IP
- Tackled the problems of interconnecting networks
  - Instead of mandating a single network technology

30.5 Internet Reference Model

- IP is the ”narrow waist” of the Internet
  - Supports many different links below and apps above
- See Figure **??**

30.6 IP as a Lowest Common Denominator 最小公约数

- Suppose only some networks support QOS or security etc.
  - Difficult for internetwork to support
- Pushes IP to be a ”lowest common denominator” protocol

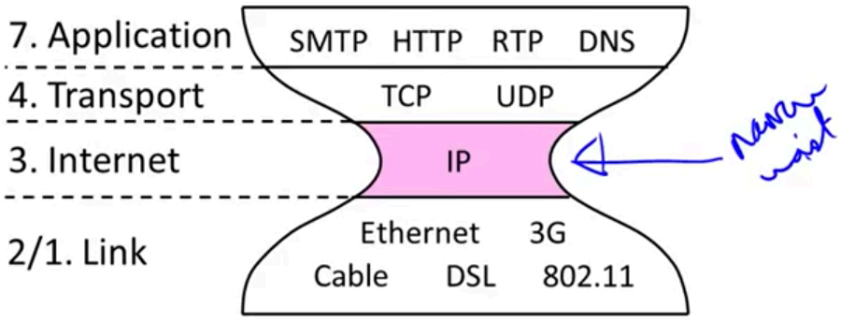


图 155: Internet Reference Model

- Asks little of lower-layer networks
- Gives little as a higher layer service

30.7 IPv4 (Internet Protocol

- Various fields to meet straightforward needs
  - Version, Header (IHL) and Total length, Protocol, and Header Checksum
- See Figure **??**
- Network layer of the Internet, uses datagrams
  - Provides a layer of addressing above link address (next)
- Some fields to handle packet size differences (later)
  - Identification, Fragment offset, Fragment control bits
- Other fields to meet other needs (later, later)
  - Differentiated Services, Time to live (TTL)

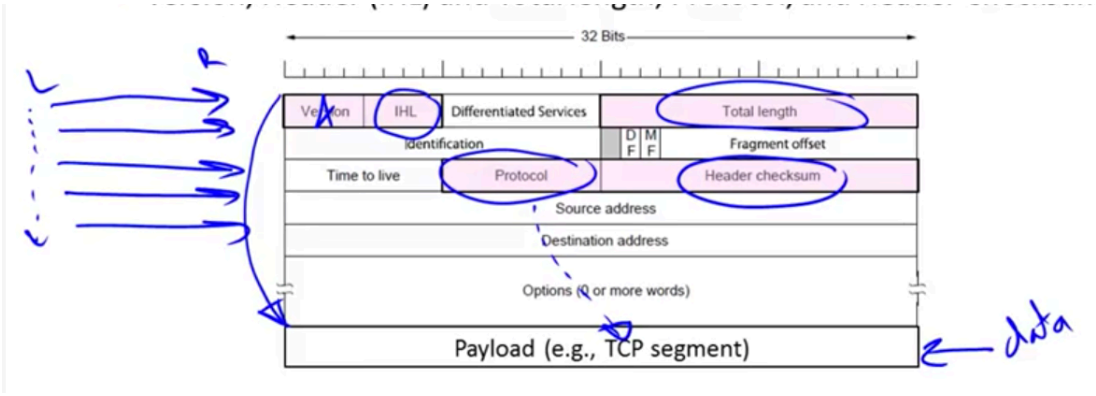


图 156: Internet Protocoll

31 Week 4 - IP Prefixes

31.1 Topic of the IP Prefixes

- What do IP addresses look like?
  - And IP prefixes, or blocks of address
  - (This is IPv4; we'll cover IPv6 later.)
- See Figure **??**

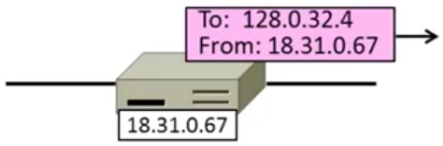


图 157: Topic of the IP Prefixes

31.2 IP Addresses

- IPv4 uses 32-bit address

- Later we'll see IPv6, which uses 128-bit addresses

- Written in "dotted quad: notation
  - Four 8-bit numbers separated by dots

- See Figure **??**

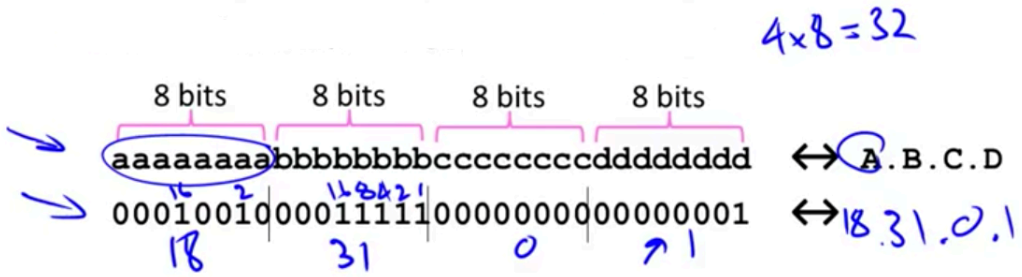


图 158: IP Addresses

31.3 IP Prefixes - Modern

- Addresses are allocated in blocks called prefixes
  - Addresses in an L-bit prefix have the same top L bits
  - There are 2^{32-L} addresses aligned on 2^{32-L} boundary

- See Figure **??**

- Written in "IP address/length" notation
  - Address is lowest address in the prefix, length is prefix bits
  - E.g., 128.13.0.0/16 is 128.13.0.0 to 128.13.255.255
  - So a /24 ("slash 24") is 256 addresses, and a /32 is one address

- See Figure **??**

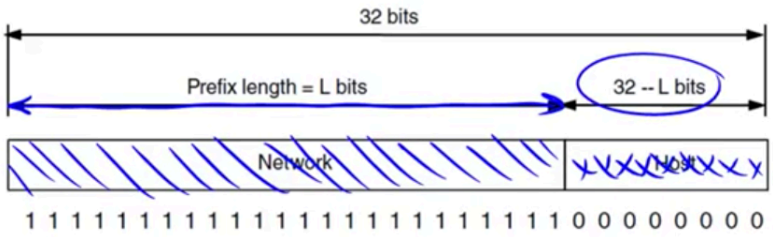


图 159: IP Prefixes - Modern

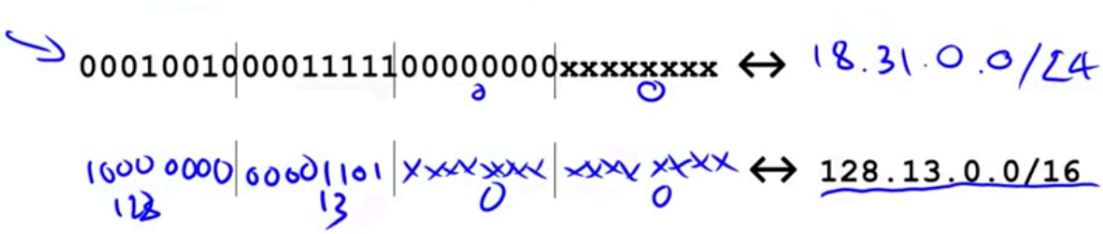


图 160: IP Prefixes - Modern Examples

- More specific prefix
  - Has longer prefix, hence a smaller number of IP addresses
- Less specific prefix
  - Has shorter prefix, hence a larger number of IP address
- See Figure **??**

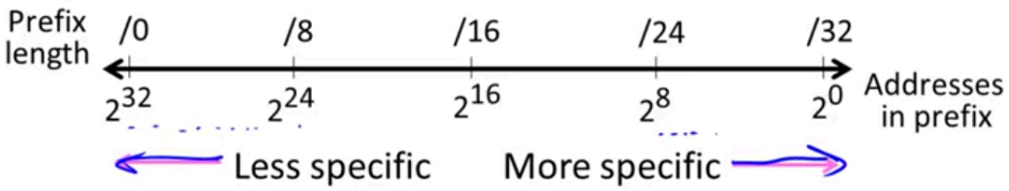


图 161: IP Prefixes - Modern 3

31.4 IP Address Classes - Historical

- Originally, IP addresses came in fixed size blocks with the class/size encoded in the high-order bits
  - They still do, but the classes are now ignored
- See Figure **??**

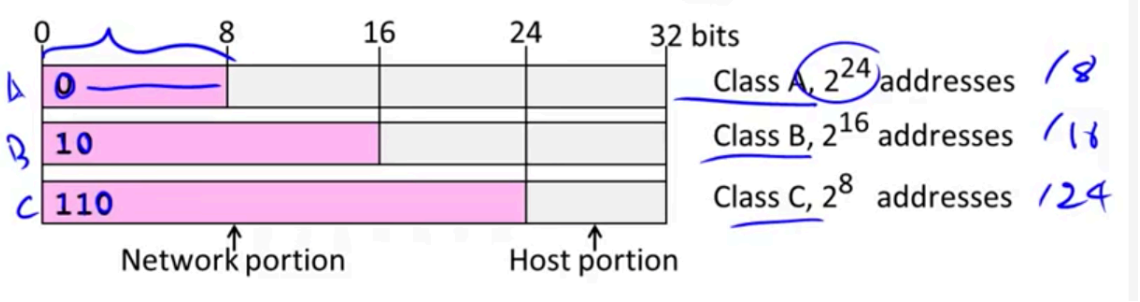


图 162: IP Address Classes - Historical

31.5 Public / Private IP Address

- Public IP address, e.g., 18.31.0.1
  - Valid destination on the global internet
  - Must be allocated to yo before use
  - Mostly exhausted ... time for IPv6!
- Private IP addresses
  - Can be used freely within private networks (home, small company)
  - 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
  - Need public IP address(es) and NAT to connect to global Internet

31.6 Allocating Public IP Address

- IANA delgates to reginal bodies (RIRs)
- RIRs delegate to companies in their region
- Companies assign to their customers/computers (later, DHCP)
- See Figure **??**

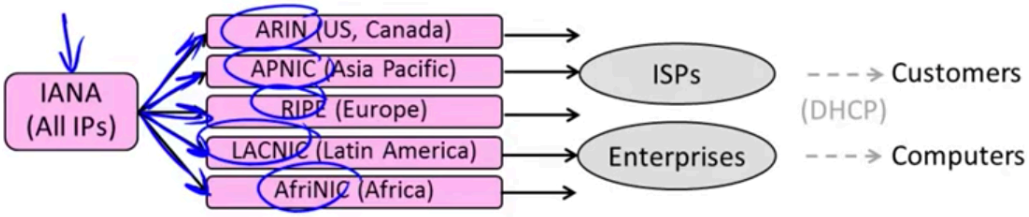


图 163: Allocating Public IP Address

32 Week 4 - IP forwarding

32.1 Topic of IP forwarding

- How do routers forward packets ?
  - We'll look at how IP does it
  - (We'll cover routing later)

32.2 Recap ⇒ We want the network layer to:

- Scale to large networks —this lecture
  - Using addresses with hierarchy

- Support diverse technologies —More later
  - Internetworking with IP
- Use link bandwidth well — Next time
  - Lowest-cost routing

32.3 IP Forwarding

- IP addresses on one network belong to the same prefix
- Node uses a table that lists the next hop for IP prefixes
- See Figure **??**

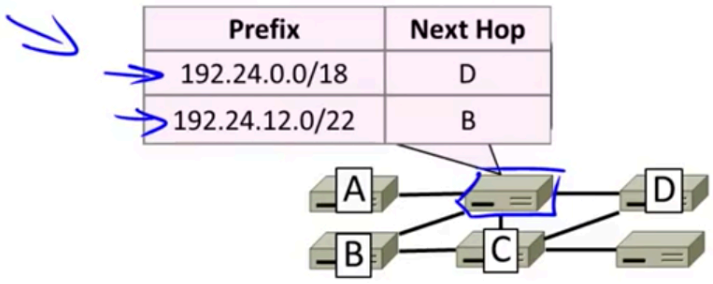


图 164: IP Forwarding

32.4 Longest Matching Prefix

- Prefixes in the table might overlap !
  - Combines hierarchy with flexibility
- Longest matching prefix forwarding rule:

  - For each packet, find the longest prefix that contains the destination address, i.e., the most specific entry
  - Forward the packet to the next hop router for that prefix

- 下图中的 192.24.15.255 的来历是前面 22 个 bit 固定，然后从后面从 00.00000000 到 11.11111111，而 192.24.63.255 则是从 000000.00000000 到 111111.11111111，其中中间粉红色的小矩形把整个区间掰成三部分，然后落在中间的部分调到 B，其他则是调到 D
- See Figure **??**

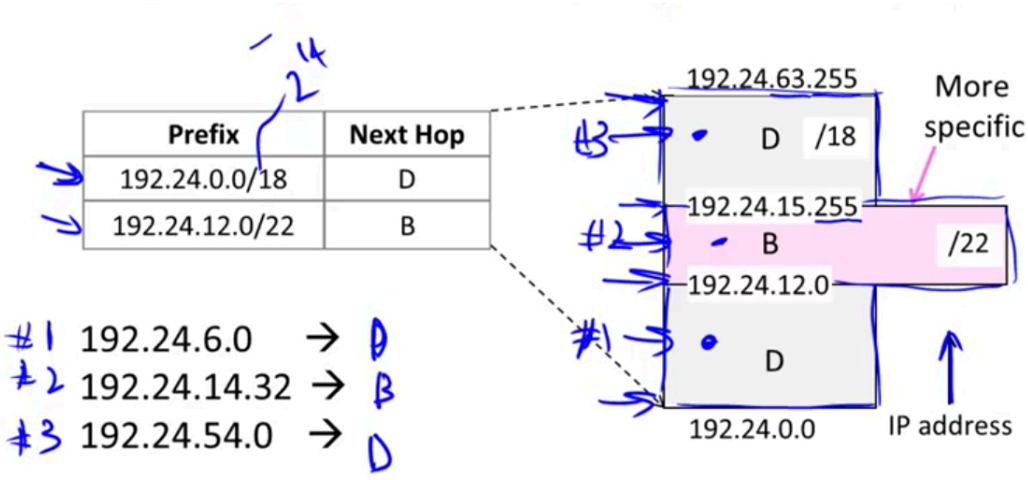


图 165: Longest Matching Prefix

32.5 Host/Router Distinction

- In the Internet:
  - Routers do the routing, know which way to all destinations
  - Hosts send remote traffic (out of prefix) to nearest router , 其中 traffic 的意涵是想象一下一般时刻的合肥道路上的情况。

32.6 Host Forwarding Table

- Give using longest matching prefix
  - 0.0.0.0/0 is a default route that catches all IP addresses
- See Figure **??**

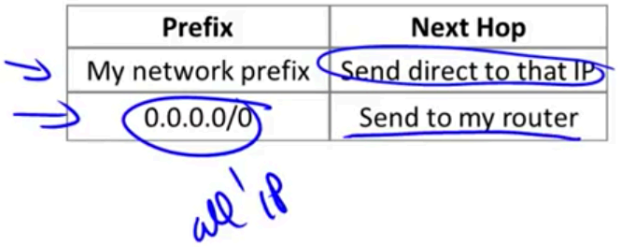


图 166: Host Forwarding Table

32.7 Flexibility of Longest Matching Prefix

- Can provide default behavior, with less specific prefixes
  - To send traffic going outside an organization to a border router
- Can special case behavior, with more specific prefixes
  - For performance, economics, security, ...

32.8 Performance of Longest Matching Prefix

- Uses hierarchy for a compact table
  - Benefits from less specific prefixes
- Lookup more complex then table

- Was a concern for fast routers, but not an issue in practice these days

32.9 Other Aspects of Forwarding

- It's not all about addresses ...
- See Figure **??**

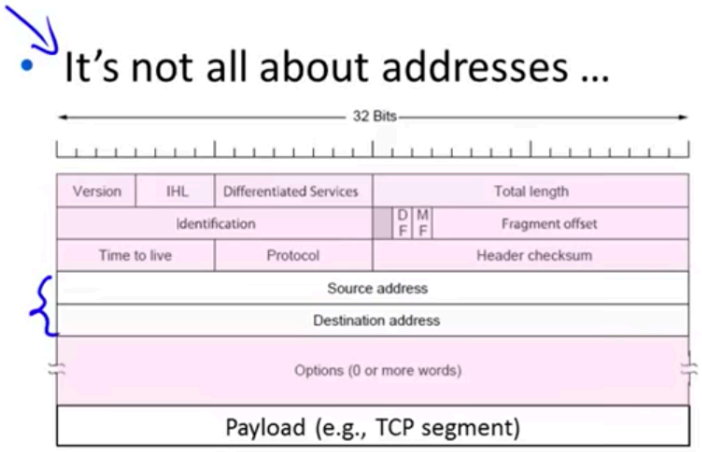


图 167: Other Aspects of Forwarding

- Decrement TTL value
  - Protects against loops
- Checks header checksum
  - To add reliability
- 下面的会在接下来进行讲解.....
- Fragment large packets

  - Split to fit it on next link

- Send congestion signals
  - Warns hosts of congestion
- Generates error messages
  - To help mange network
- Handle various options

33 Week 4 - IP Helpers ARP and DHCP

33.1 Topic of IP Helper ARP and DHCP

- Filling in the gaps we need to make for IP forwarding work in practice
  - Getting IP addresses (DHCP)
  - Mapping IP to link addresses (ARP)
- See Figure **??**

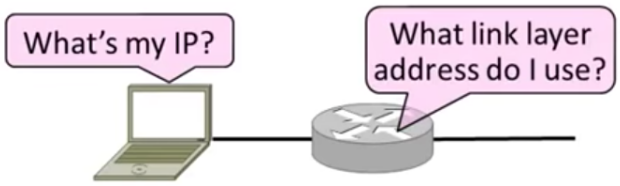


图 168: Topic of IP Helper ARP and DHCP

33.2 Getting IP Addresses

- Problem:
  - A node wakes up for the first time ...
  - What is its address ? What's the IP address of its router ? Etc.
  - At least Ethernet address is on NIC
- 1. Manual configuration (old days)
  - Can't be factory set, depends on use
- 2. A protocol for automatically configuring addresses (DHCP)
  - Shifts burden from users to IT folk

33.3 DHCP

- DHCP (Dynamic Host Configuration Protocol), from 1993, widely used
- It leases IP address to nodes
- Provides other parameters too
  - Network prefix
  - Address of local router
  - DNS server, time server, etc.

33.4 DHCP Protocol Stack

- DHCP is a client-server application
  - Uses UDP ports 67, 68

33.5 DHCP Addressing

- Bootstrap issue:
  - How does node send a message to DHCP server before it is configured ?
- Answer:
  - Node sends broadcast messages that delivered to all nodes on the network
  - Broadcast address is all 1s
  - IP (32 bit): 255.255.255.255
  - Ethernet (48 bit): ff:ff:ff:ff:ff:ff

33.6 DHCP Messages

- 如下图所示，Client 通过首先发送广播来寻找 DHCP Server；然后 Server 给该 Client 提供一个 IP 地址；然后 Client 申请使用该地址，由于此时还没有完成 IP 地址申请，所以还是广播；最后 Server 回复，表示同意，此时 Client 就有了 IP 地址。
- See Figure **??**
- To renew an existing lease, an abbreviated sequence is used:
  - REQUEST, followed by ACK
- Protocol also supports replicated servers for reliability

33.7 Sending an IP Packet

- Problem:

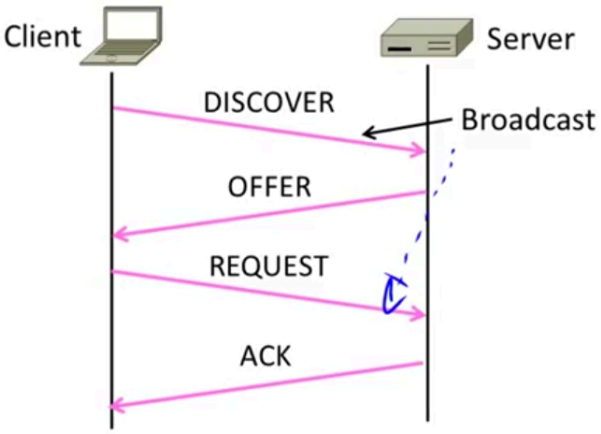


图 169: DHCP Messages

- A node needs Link layer address to send a frame over the local link
- How does it get the destination link address from a destination IP address ?

33.8 ARP (Address Resolution Protocol)

- Node uses to map a local IP address to its Link layer addresses
- See Figure **??**

33.9 ARP Protocol Stack

- ARP sits right on top of link layer
  - No servers, just asks node with target IP to identify it self
  - Uses broadcast to reach all nodes

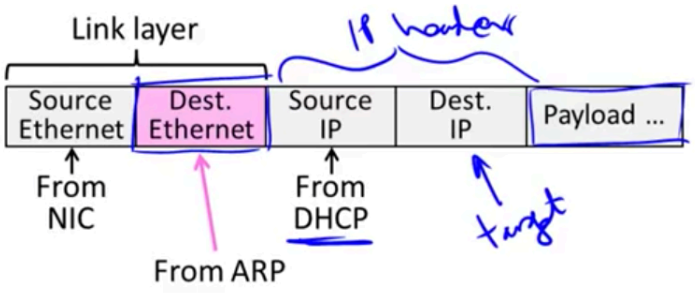


图 170: DHCP Messages

33.10 ARP Messages

- 向网络中所有的节点询问是否是目标节点，如果是目标节点的话，它会返回其 Link Address 地址，是一个 48 位的地址，这样 Node 就有了 Target 的链接地址
- See Figure **??**

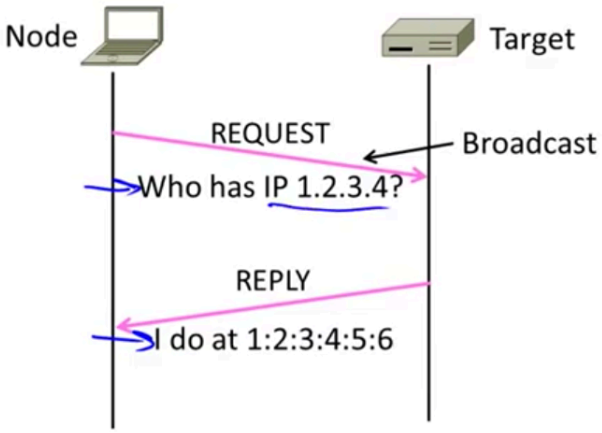


图 171: ARP Messages

33.11 Discovery Protocols

- Help nodes find each other
  - There are more of them !
    - E.g., zeroconf, Bonjour
- Often involve broadcast
  - Since nodes aren't introduced
  - Very handy glue

34 Week 4 - Packet Fragmentation

34.1 Topic of Packet Fragmentation

- How do we connect networks with different maximum packet sizes?
  - Need to split up packets, or discover the largest size to use

34.2 Packet Size Problem

- Different networks have different maximum packet sizes
  - Or MTU (Maximum Transmission Unit)
  - E.g. Ethernet 1.5K, WiFi 2.3K
- Prefer large packets for efficiency
  - But what size is large ?
  - Difficult because node does not know complete network path

34.3 Packet Size Solutions

- Fragmentation (now)
  - Split up large packets in the network if they are too big to send
  - Classic method, dates
- Discovery (next)
  - Find the largest packet that fits on the network path and use it
  - IP uses today instead of fragmentation

34.4 IPv4 Fragmentation

- Routers fragment packets that are too large to forward
- Receiving host reassembles to reduce load on routers
- See Figure **??**

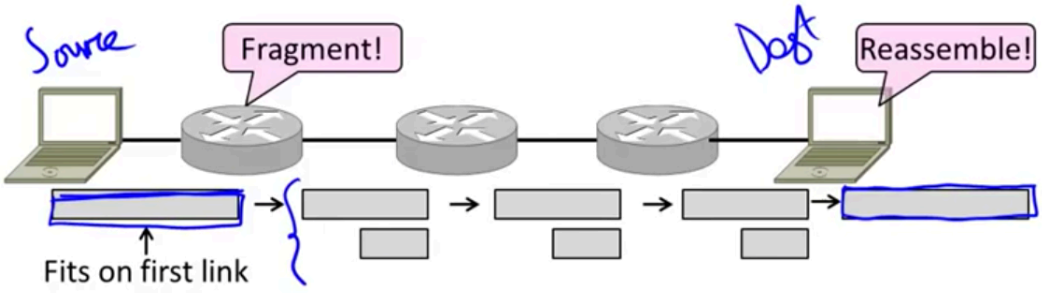


图 172: IPv4 Fragmentation

34.5 IPv4 Fragmentation Fields

- Header fields used to handle packet size differences

- Identification, Fragment offset, MF/DF control bits

- See Figure **??**

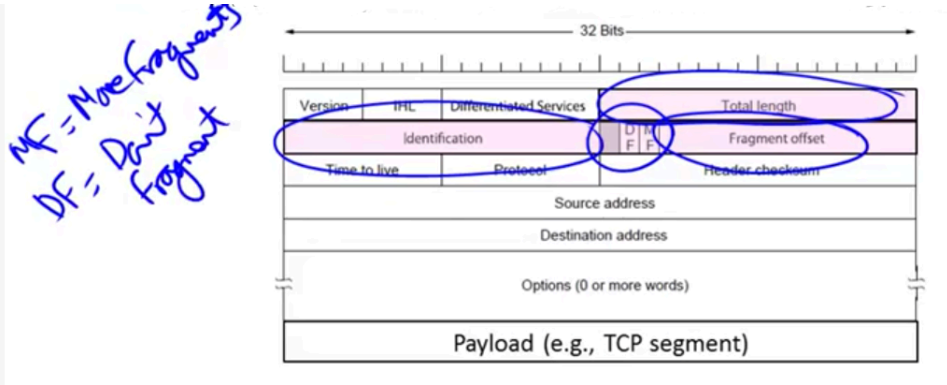


图 173: IPv4 Fragmentation Fields

34.6 IPv4 Fragmentation Procedure

- Routers split a packet that is too large:
  - Typically break into large pieces
  - Copy IP header to pieces
  - Adjust length on pieces
  - Set offset to indicate position
  - Set MF (More Fragments) on all pieces except last
- Receiving hosts reassembles the pieces:
  - Identification field links pieces together, MF tells receiver when it has all pieces
- See Figure **??**

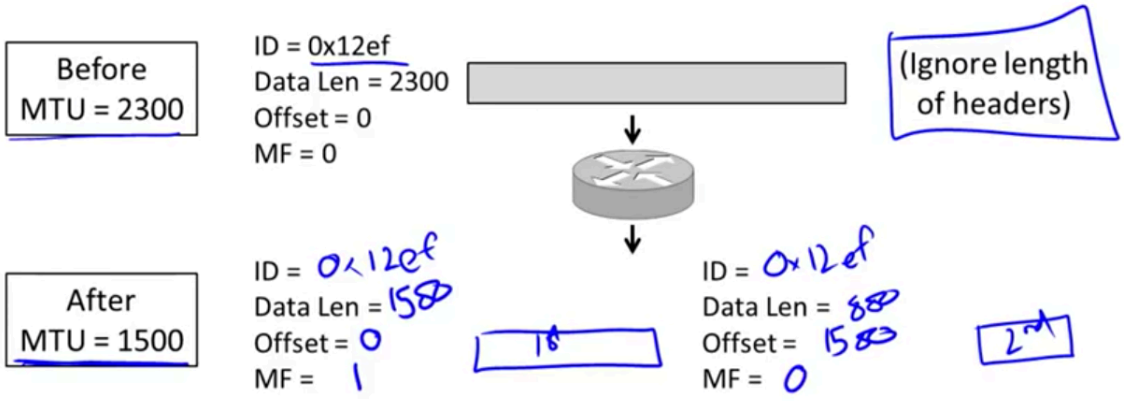


图 174: IPv4 Fragmentation Procedure

- It works !
  - Allows repeated fragmentation
- But fragmentation is undesirable
  - More work for routers, hosts
  - Tends to magnify loss rate
  - Security vulnerabilities too

34.7 Path MTU Discovery

- Discover the MTU that will fit
  - So we can avoid fragmentation
  - The method in use today
- Host tests path with large packet
  - Routers provide feedback if too large; they tell host what size would have fit

- See Figure **??**

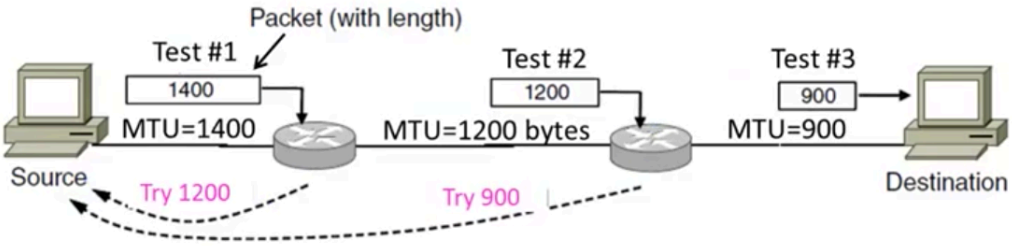


图 175: Path MTU Discovery

- Process may seem involved
  - But usually quick to find right size
- Path MTU depends on the path and so can change over time
  - Search is ongoing
- Implemented with ICMP (next)
  - Set DF (Don't Fragment) bit in IP header to get feedback messages

35 Week 4 - IP Errors ICMP

35.1 Topic of IP Errors ICMP

- What happens when something goes wrong during forwarding
  - Need to be able to find the problem

35.2 Internet Control Message Protocol

- ICMP is a companion protocol to IP
  - They are implemented together
  - Sits on top of IP (IP Protocol=1)
- Provides error report and testing
  - Error is at router while forwarding
  - Also testing that hosts can use

35.3 ICMP Errors

- It sends an ICMP error report back to the IP source address
  - It sends an ICMP error back to the IP source address
  - It discards the problematic packet; host needs to rectify
- See Figure **??**

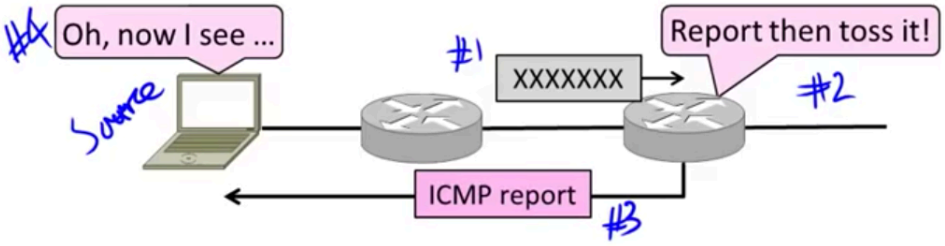


图 176: ICMP Errors

35.4 ICMP Message Format

- Each ICMP message has a Type, Code, and Checksum
- Often carry the start of the offending packet as payload
- Each message is carried in an IP packet
- See Figure **??**

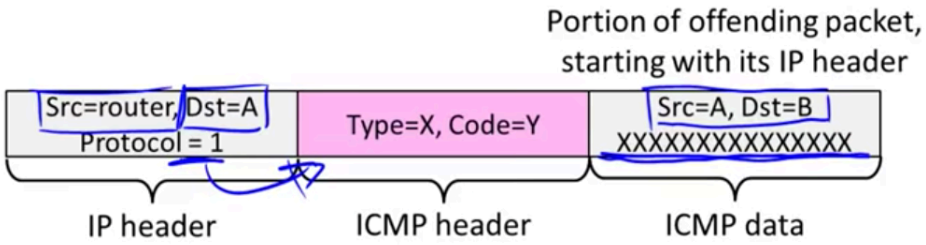


图 177: ICMP Message Format

35.5 Example ICMP Messages

- See Figure **??**

35.6 Traceroute

- IP header contains TTL (Time to live) field
  - Decremented every router hop, with ICMP error if it hits zero
  - Protects against forwarding loops 如果有 Loop 的话，这个 TTL 值也会在一定时间内消失，此时 Packet 失效。
- Traceroute repurposes TTL and ICMP functionality
  - Sends probe packets increasing TTL starting from 1

Name	Type / Code	Usage
Dest. Unreachable (Net or Host)	3 / 0 or 1	Lack of connectivity
Dest. Unreachable (Fragment)	3 / 4	Path MTU Discovery
Time Exceeded (Transit)	11 / 0	Traceroute
Echo Request or Reply	8 or 0 / 0	Ping

Testing, not a forwarding error: Host sends Echo Request, and destination responds with an Echo Reply

图 178: Example ICMP Messages

  - ICMP errors identify routers on the path

- See Figure **??**

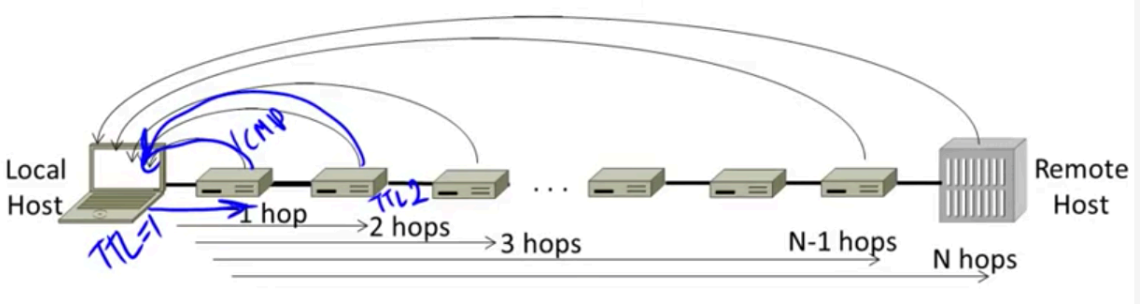


图 179: Traceroute

36 Week 4 - IP version 6

36.1 IP Version 6 to the Rescue

- Effort started by the IETF in 1994

  - Much larger addresses (128 bits)

  - Many sundry improvements

- Became an IETF standard in 1998
  - Nothing much happened for a decade
  - Hampered by deployment issues, and a lack of adoption incentives
  - Big push 2011 as exhaustion looms

36.2 IPv6

- Features large addresses
  - 128 bits, most of header
- New notation
  - 8 groups of 4 hex digits (16 bits)
  - Omit leading zeros, groups of zeros
- See Figure **??**
- See Figure **??**

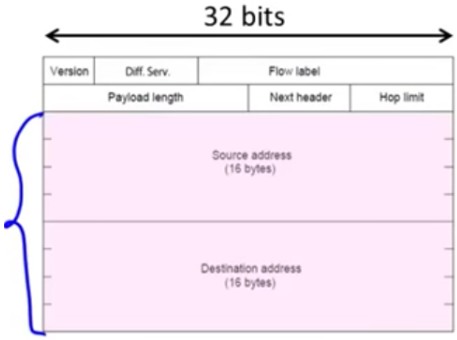


图 180: IPv6 第一幅图

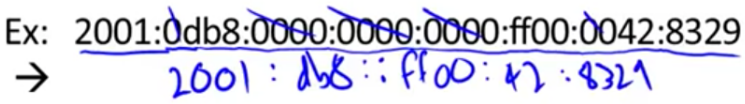


图 181: IPv6 第二幅图

- Lots of other, smaller changes
  - Streamlined header processing
  - Flow label to group of packets
  - Better fit with ”advanced” features (mobility, multicasting, security)

36.3 IPv6 Transition

- The Big Problem
  - How to deploy IPv6 ?
  - Fundamentally incompatible with IPv4
- Dozens of approaches proposed
  - Dual stack (speak IPv4 and IPv6)
  - Translators (covert packets)
  - Tunnels (carry IPv6 over IPv4)

36.4 Tunneling

- Native IPv6 islands connected via IPv4
  - Tunnel carries IPv6 packets across IPv4 network
- See Figure **??**

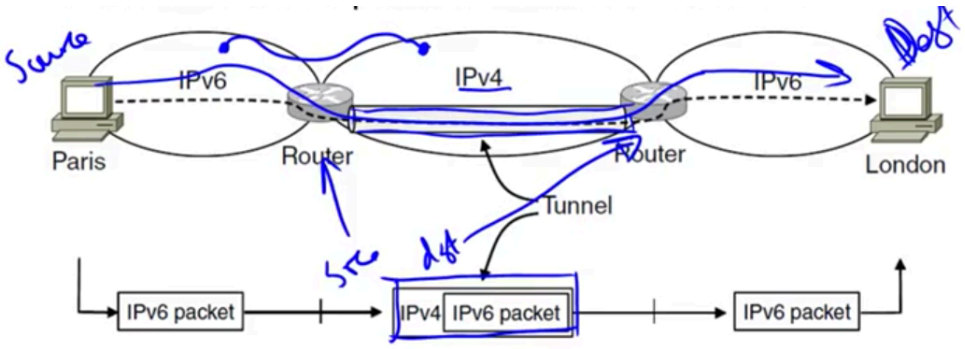


图 182: Tunneling

- Tunnel acts as a single link across IPv4 network
- See Figure **??**

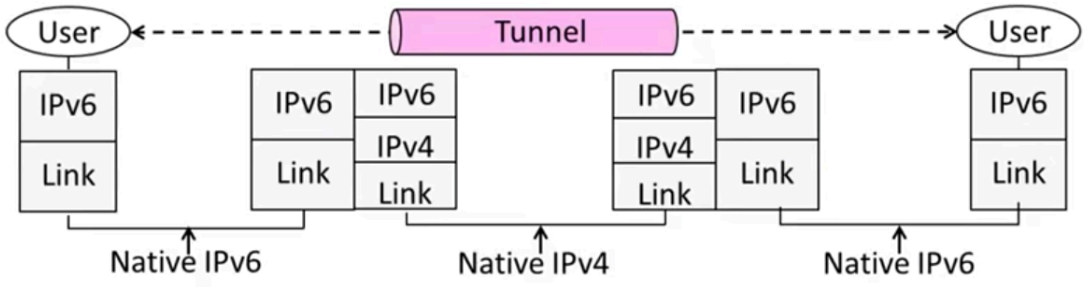


图 183: Tunnel acts as a single link across IPv4 network

37 Week 4 - Network Address Translation NAT

37.1 Topic of NAT

- What is NAT (Network Address Translation)? How does it work?
  - NAT is widely used at the edges of the network, e.g., homes

37.2 Layering Review

- Remember how layering is meant to work?
  - "Routers don't look beyond the IP header." Well ...
- See Figure **??**

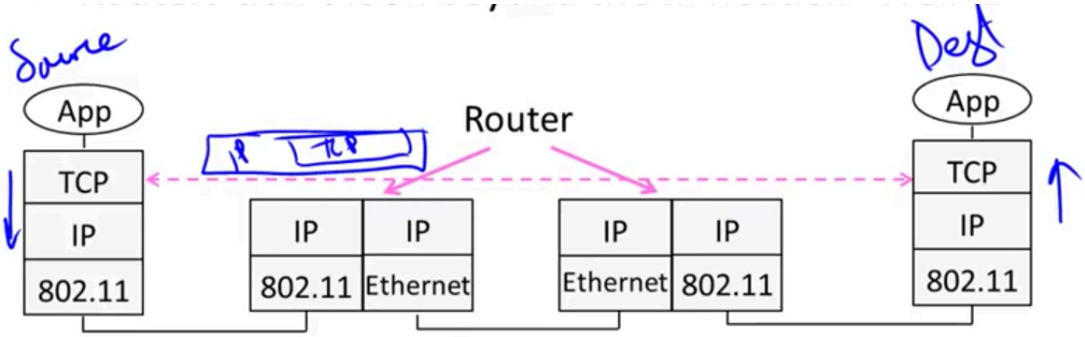


图 184: Layering Review

37.3 Middleboxes

- Sit "inside the network" but perform "more than IP" processing on packets to add new functionality
  - NAT box, Firewall / Intrusion Detection System
- See Figure **??**
- Advantages
  - A possible rapid deployment path when there is no other option
  - Control over many hosts (IT)
- Disadvantages

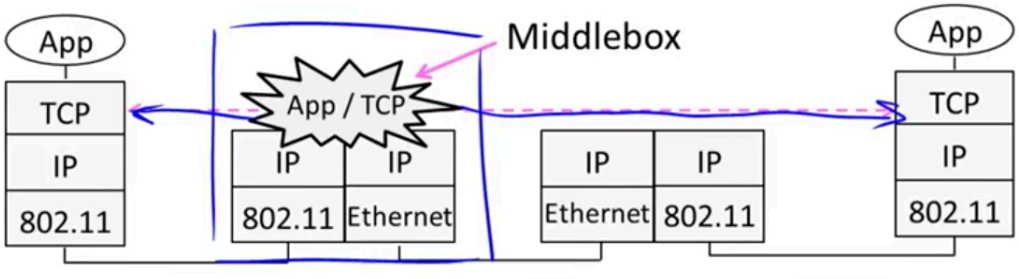


图 185: Middleboxes

- Breaking layering interferes with connectivity; strange side effects
- Poor vantage point for many tasks

37.4 NAT (Network Address Translation) Box

- NAT box connects an internal network to an external network
  - Many internal hosts are connected using few external addresses
  - Middlebox that "translates addresses"
- Motivated by IP address scarcity
  - Controversial at first, now accepted
- Common scenario
  - Home computers use "private" IP address
  - NAT (in AP/firewall) connects home to ISP using a single external IP address
- See Figure **??**

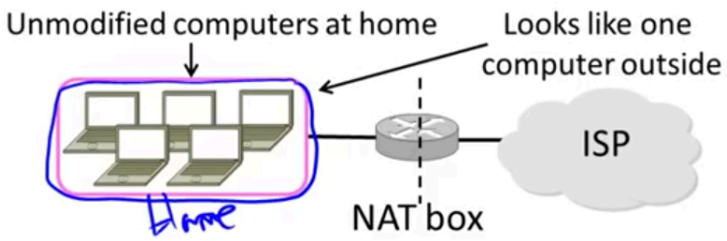


图 186: NAT

37.5 How NAT Works

- Keeps an internal/external table
  - Typically uses IP address + TCP port
  - This is address an port translation
- See Figure **??**

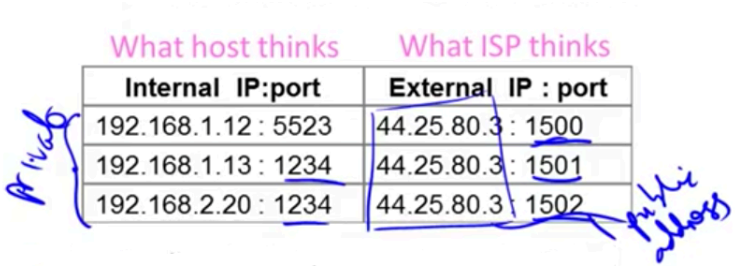


图 187: How NAT Works

- Need ports to make mapping 1-1 since there are fewer external IPs
- Internal → External:
  - Look up and rewrite Source IP/port
- See Figure **??**

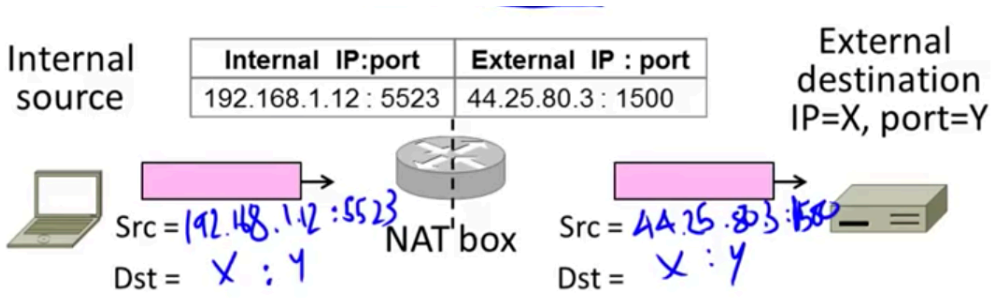


图 188: How NAT Works 发送

- External → Internal:
  - Look up and rewrite Destination IP/port
- See Figure **??**

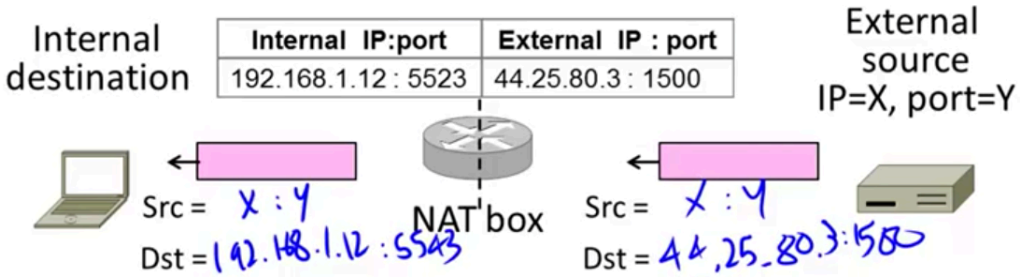


图 189: How NAT Works 回复

- Need to enter translations in the table for it to work
  - Create external name when host makes a TCP connection
- See Figure **??**

37.6 NAT Downsides

- Connectivity has been broken!

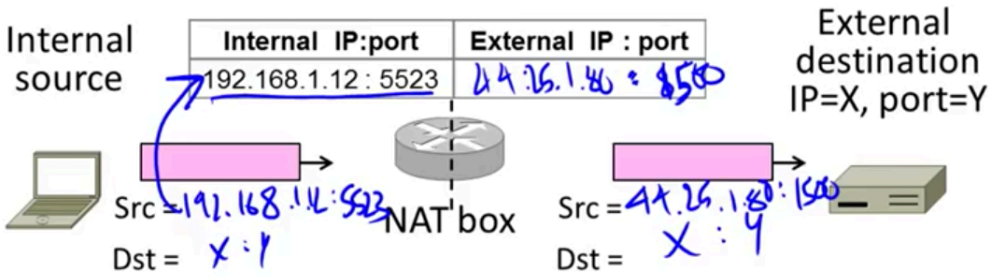


图 190: How NAT Works 第四幅图

  - Can only send incoming packets after an outgoing connection is set up
  - Difficult to run servers or peer-to-peer apps (Skype) at home
- Does't work so well when there are no connections (UDP apps)
- Breaks apps that unwisely expose there IP address (FTP)

37.7 NAT Upsides

- Relieve much IP address pressure
  - Many home hosts behind NATs
- Easy to deploy
  - Rapidly, and by you alone
- Useful functionality
  - Firewall, helps with privacy
- Kinks will get worked out eventually
  - "NAT Traversal" for incoming traffic

38 Week 5 - Routing Overview

38.1 Routing versus Forwarding

- See Figure ??

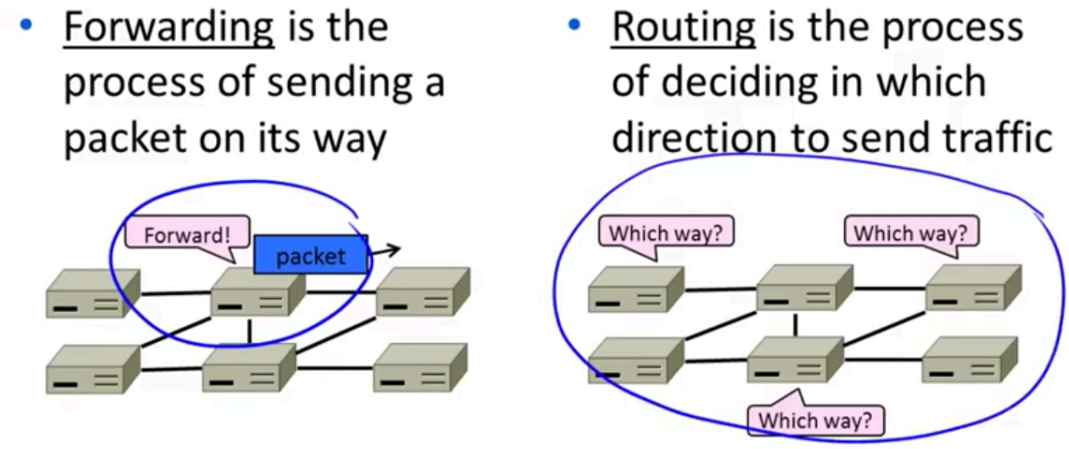


图 191: Routing versus Forwarding

38.2 Improving on the Spanning Tree

- See Figure ??

38.3 Perspective on Bandwidth Allocation

- Routing allocates network bandwidth adapting to failures; other mechanisms used at other timescales
- See Figure ??

38.4 Delivery Models

- Different routing used for different delivery models

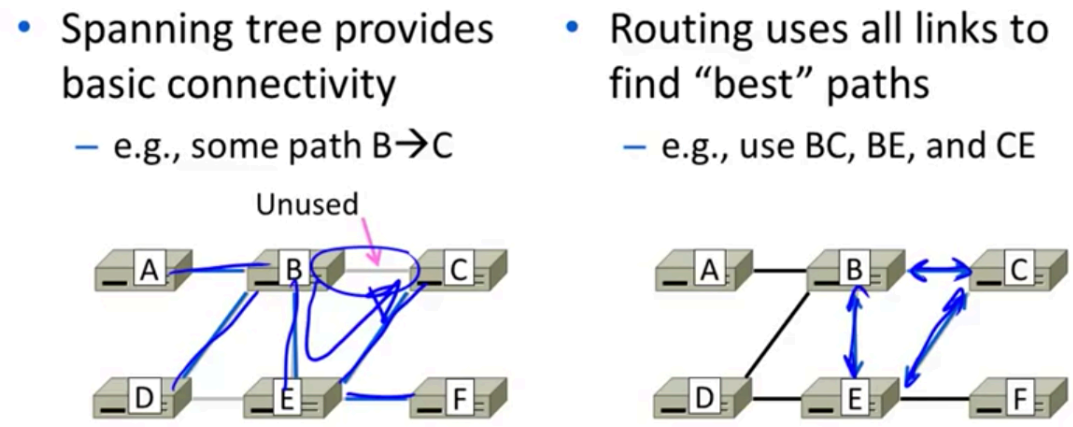


图 192: Improving on the Spanning Tree

Mechanism	Timescale / Adaptation
Load-sensitive routing	Seconds / Traffic hotspots
Routing	Minutes / Equipment failures
Traffic Engineering	Hours / Network load
Provisioning	Months / Network customers

图 193: Perspective on Bandwidth Allocation

- See Figure **??**

38.5 Goals of Routing Algorithms

- We want several properties of any routing scheme:
- See Figure **??**

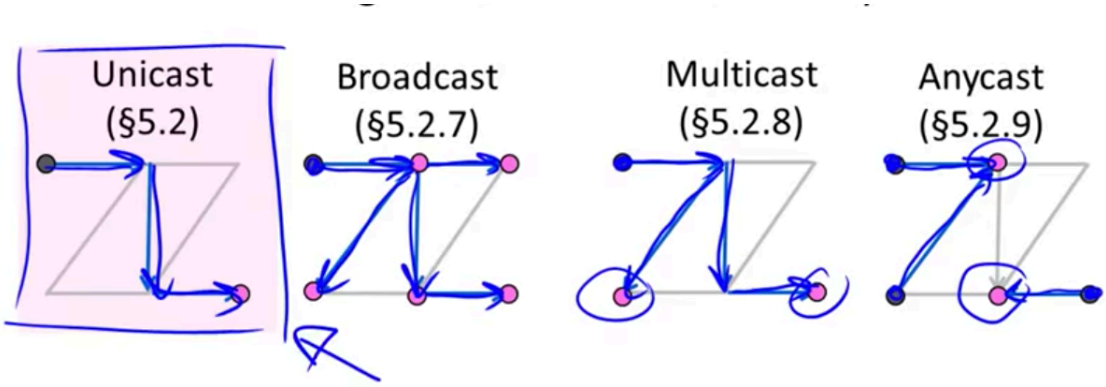


图 194: Perspective on Bandwidth Allocation

Property	Meaning
Correctness	Finds paths that work
Efficient paths	Uses network bandwidth well
Fair paths	Doesn't starve any nodes
Fast convergence	Recovers quickly after changes
Scalability	Works well as network grows large

图 195: Goals of Routing Algorithms

38.6 Rules of Routing Algorithms

- All nodes are alike; no controller
- Nodes only know what they learn by exchanging messages with neighbors
- Nodes operate concurrently
- May be node/link/message failures

38.7 Topics

- IPv4, IPv6, NATs, and all that
- 上面内容之前已经说过，下面是本章内容。
- Shortest path routing
- Distance Vector routing
- Flooding
- Link-state routing
- Equal-cost multi-path
- Inter-domain routing (BGP)

39 Week 5 - Shortest Path Routing

39.1 Topic of the Shortest Path Routing

- Defining "best" paths with link costs
  - These are shortest path routes
- See Figure **??**

39.2 What are "Best" paths anyhow ?

- Many possibilities:
  - Latency, avoid circuitous paths
  - Bandwidth, avoid slow links
  - Money, avoid expensive links

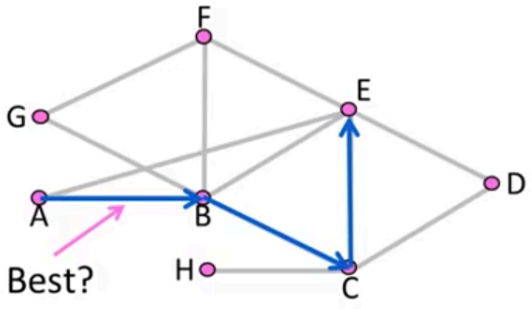


图 196: Topic of the Shortest Path Routing

- Hops, to reduce switching

- But only consider topology
  - Ignore workload, e.g., hotspots

- See Figure **??**

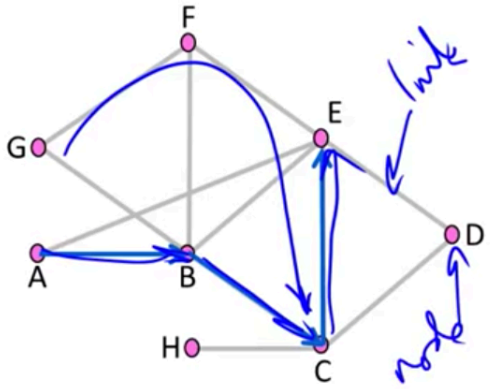


图 197: What are "Best" paths anyhow ?

39.3 Shortest Paths

- We'll approximate "best" by a cost function that captures the factors

  - Often call lowest "shortest"
- 1. Assign each link a cost (distance)
- 2. Define best path between each pair of nodes as the path that has the lowest total cost (or is shortest)
- 3. Pick randomly to any break ties
- Find the shortest path A $\rightarrow$ E
- All links are bidirectional, with equal costs in each direction
  - Can extend model to unequal costs if needed
- ABCE is a shortest path
- $dist(ABCE) = 4 + 2 + 1 = 7$
- This is less then:
  - $dist(ABE) = 8$
  - $dist(ABFE) = 9$
  - $dist(AE) = 10$
  - $dist(ABCDE) = 10$
- See Figure **??**
- Optimality property:
  - Subpaths of shortest paths are also shortest paths
- ABCE is a shortest path
  - So are ABC, AB, BCE, BC, CE

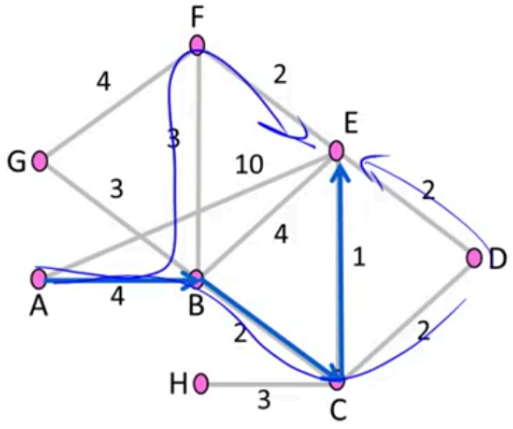


图 198: Shortest Paths

39.4 Sink Trees 汇集树

- Sink tree for a destination is the union of all shortest paths towards the destination
  - Similarly source tree
- Find the sink tree for E
- 如下图所示，sink tree 就是要找出除了 destination 以外其他任意节点和 destination 之间的最短路径，sink tree 是它们的集合。
- See Figure **??**
- Implications:
  - Only need to use destination to follow shortest paths
  - Each node only need to send to the next hop
- Forwarding table at a node
  - Lists next hop for each destination

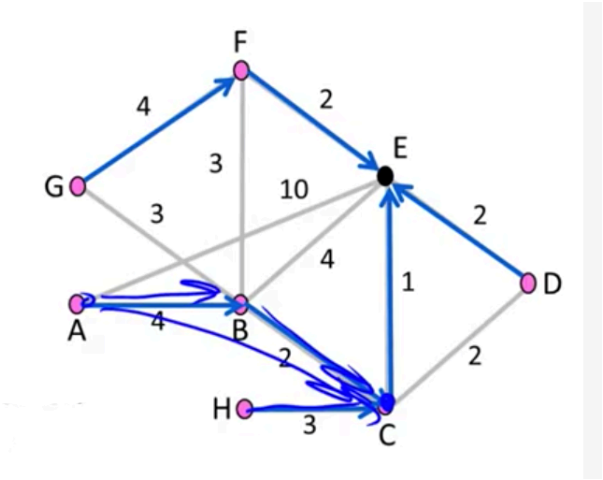


图 199: Sink Trees 汇集树

– Routing table may know more

40 Week 5 - Dijkstras Algorithm

40.1 Topic of Dijkstras Algorithm

- How to compute shortest paths given the network topology
  - With Dijkstra's algorithm

40.2 Dijkstra's Algorithm

- Algorithm:
  - Mark all nodes tentative, set distances from source to 0 (zero) for source, and ∞ (infinity) for all other nodes
- While tentative nodes remain:
  - Extract N, a node with lowest distance

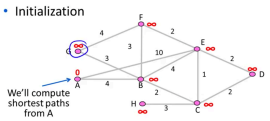


图 200: Dijkstra's Algorithm 1

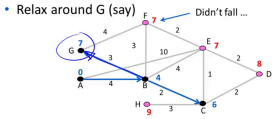


图 201: Dijkstra's Algorithm 2

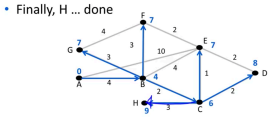


图 202: Dijkstra's Algorithm 3

  - Add link to N to the shortest path tree
  - Relax the distances of neighbors of N by lowering any better distance estimates

- 应该可以用层序遍历的思维去做:
  - 开始的时候，起点设置为 0，其他节点设置为 ∞
  - 再遍历起始节点所有的连接节点，从权值最小的开始遍历
    * 遍历时候节点设置一个 flag，表示已经经过该节点。
    * 遍历第一个节点, 更新其他与之相关的节点的值。
  - 再遍历次小权值的节点，如果其他它遇到的节点，它们之间的权值大于之前的值，这个路径就废弃了，如下图中 AE。
  - 然后再往下一层走，按照同样的方式，直到他们全都标记为 flag
- See Figure **??** , Figure **??** , Figure **??**

40.3 Dijkstra Comments

- Finds shortest paths in order of increasing distance from source
  - Leverages optimality property
- Runtime depends on efficiency of extracting min-cost node
  - Superlinear in network size (grows fast)

- Gives complete source/sink tree
  - More then needed for forwarding!
  - But requires complete topology

41 Week 5 - Distance Vector Routing

41.1 Topic of Distance Vector Routing

- How to compute shortest paths in a distributed network
  - The Distance Vector (DV) approach

41.2 Distance Vector Routing

- Simple, early routing approach
  - Used in ARPANET, and RIP
- One of two main approaches to routing
  - Distributed version of Bellman-Ford
  - Works, but very slow convergence after some failures
- Link-state algorithms are now typically used in practice
  - More involved, better behavior

41.3 Distance Vector Setting

- Each node computes its forwarding table in a distributed setting
  - 1. Nodes know only the cost to their neighbors; not the topology

- 2. Nodes can talk only to their neighbors using messages
- 3. All nodes run the same algorithm concurrently
- 4. Nodes and links may fail, messages may be lost

41.4 Distance Vector Algorithm

- Each node maintains a vector of distances (and next hops) to all destinations
  - 1. Initialize vector with 0 (zero) cost to self, ∞ (infinity) to other destinations
  - 2. Periodically send vector to neighbors
  - 3. Update vector for each destination by selecting the shortest distance heard, after adding cost of neighbor link
    - Use the best neighbor for forwarding

41.5 Distance Vector Example

- Consider a simple network. Each node runs on its own
  - E.g., node A can only talk to nodes B and D
- See Figure **??**
- First exchange, A hears from B, D and finds 1-hop routes
  - A always learns min(B+3, D+7)
- See Figure **??**
- First exchange for all nodes to find best 1-hop routes
  - E.g., B learns min(A+3, C+6, D+3)

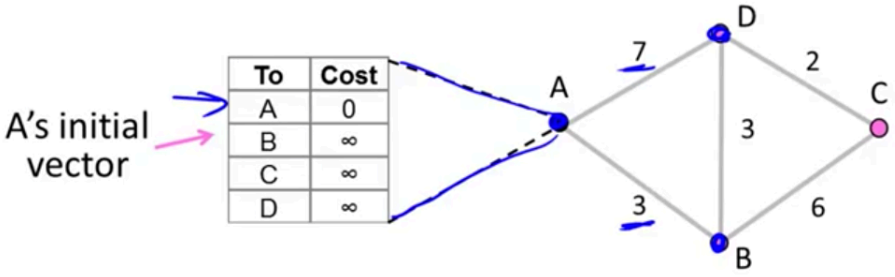


图 203: Distance Vector Example 图一

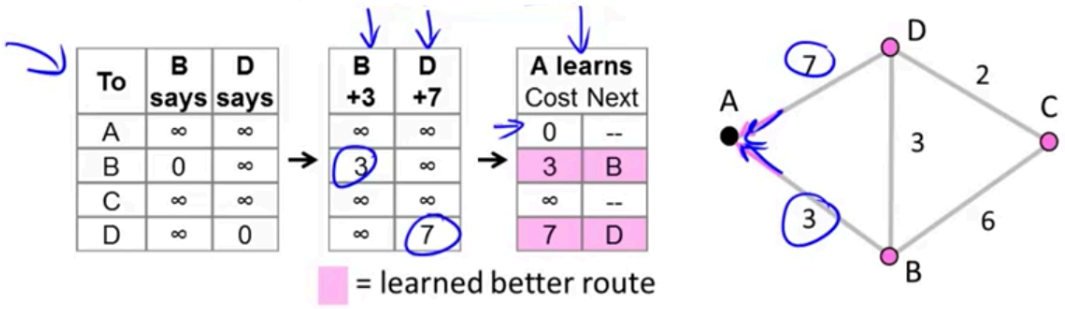


图 204: Distance Vector Example 图二

- See Figure **??**

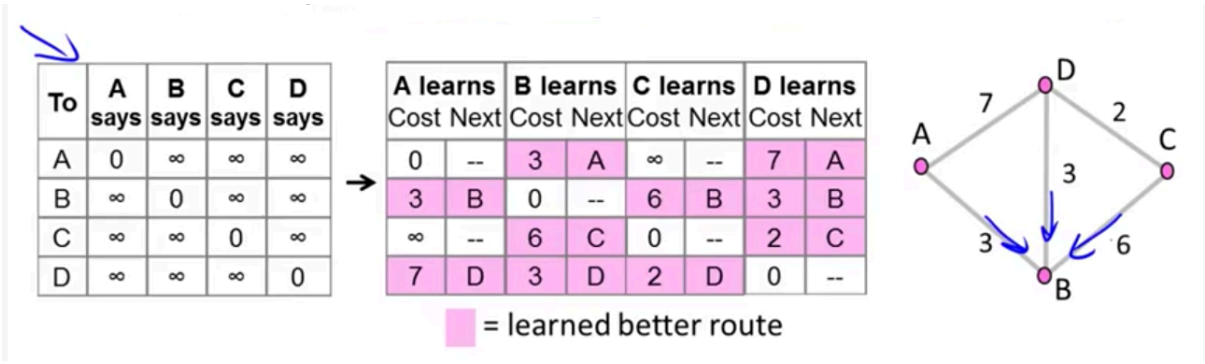


图 205: Distance Vector Example 图三

- Second exchange for all nodes to find best 2-hop routes

- 下图表示 2 跳到达各个点的距离，比如第一列，A 学到的东西的第三行，9 就是经过下一跳 D 达到 C 所需要的距离；第四行表示经由下一跳 B 到达 D 的距离。
- See Figure ??

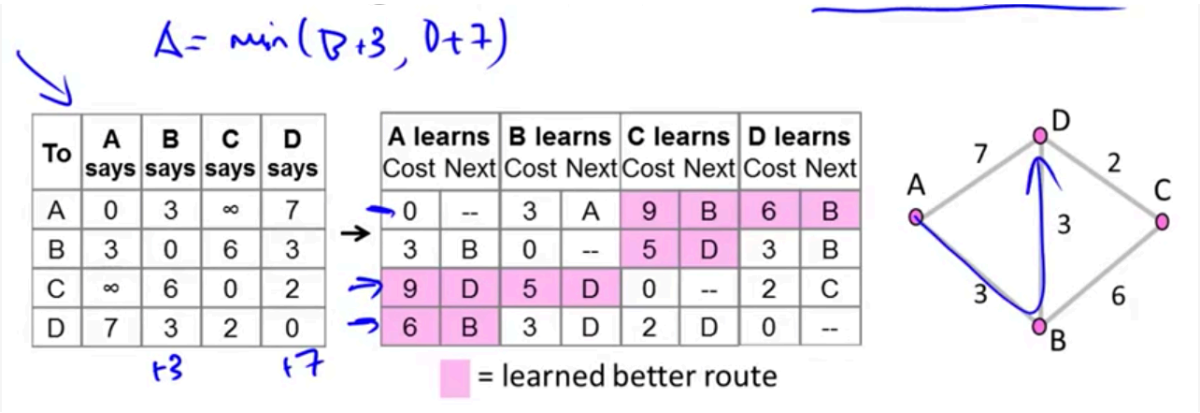


图 206: Distance Vector Example 图四

- Third exchange for all nodes to find best 3-hop routes
- See Figure ??

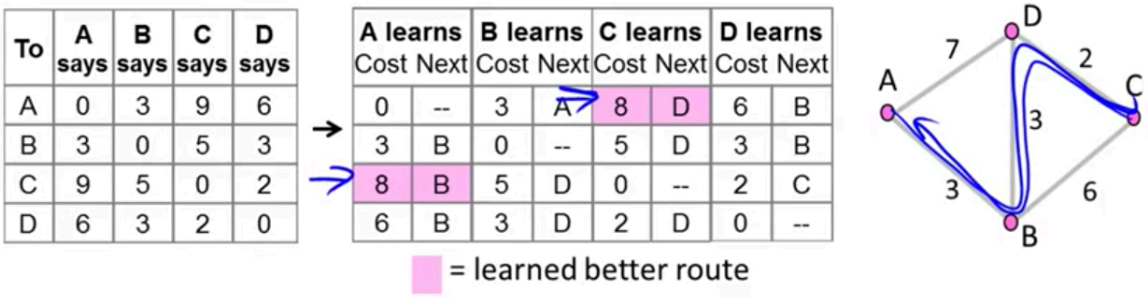


图 207: Distance Vector Example 图五

- Fourth and subsequent exchanges; converged

- See Figure **??**

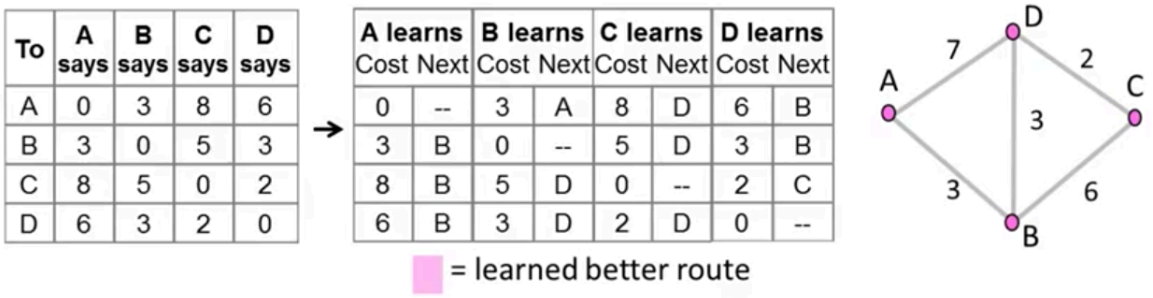


图 208: Distance Vector Example 图六

41.6 Distance Vector Dynamics

- Adding routes:
  - News travels on hop per exchange
- Removing routes
  - When a node fails, no more exchanges, other nodes forget
- But partitions (unreachable nodes in divided network) are a problem
  - "Count to infinity" scenario
- Good news travels quickly, bad news slowly (inferred)
- 如下图所示，第二幅图在第一幅的基础上，A 与 B 之间断线了。此时 B 找不到 A，但是 C 的表中认为自己可以找到 A，原因是 C 和 A 不直接接触。然后一路向下。。。
- See Figure **??**
- Various heuristics to address

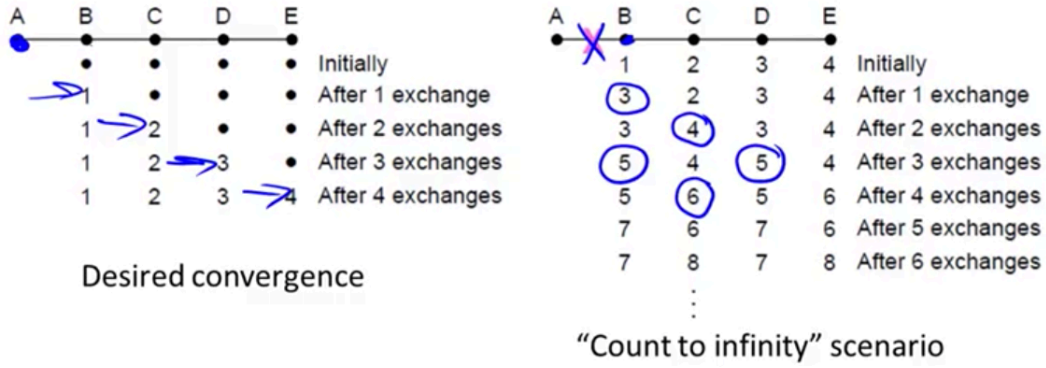


图 209: Distance Vector Dynamics

- e.g., "Split horizon, poison reverse" (Don't send route back to where you learned it from.)

- But one none are very effective
  - Link state now favored in practice
  - Except when very resource-limited

41.7 RIP (Routing Information Protocol

- DV protocol with hop count as metric
  - Infinity is 16 hops; limits network size
  - Includes split horizon, poison reverse 水平分割，毒性逆转
  - 水平分割是指，从一端收到的路由信息，不能再从原路被发送回去。
  - 带毒性逆转的水平分割 (split-horizon with poisoned reverse) : 但是能收到哪怕是坏消息总是比收不到消息要好得多。我们可以用带毒性逆转的水平分割来实现这一点。

- Routers send vectors every 30 secs
  - Runs on top of UDP
  - Timeout in 180 secs to detect failures
- RIPv1 specified in RFC1058 (1988)

42 Week 5 - Flooding

42.1 Topic of Flooding

- How to broadcast a message to all nodes in the network with flooding
  - Simple mechanism, but inefficient
- See Figure **??**

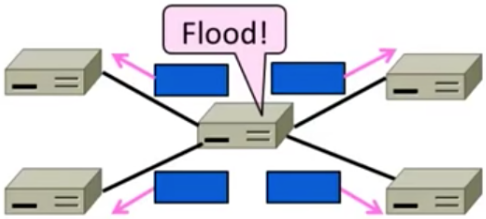


图 210: Topic of Flooding

42.2 Flooding

- Rule used at each node:
  - Sends an incoming message on to all other neighbors
  - Remember the message so that it is only flood once

- Inefficient because one node may receive multiple copies of message
- Consider a flood from A; first reaches B via AB, E via AE
- See Figure **??**

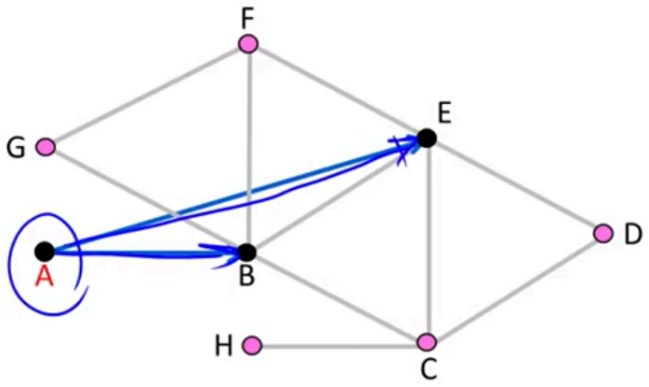


图 211: Flooding 图一

- Next B floods BC, BE, BF, BG, and E floods EB, EC, ED, EF
- See Figure **??**

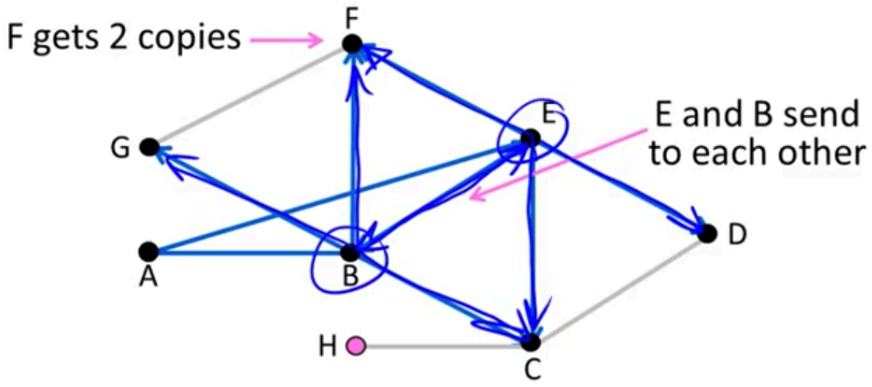


图 212: Flooding 图二

- C floods CD, CH; D floods DC; F floods FG; G floods GF
- See Figure **??**

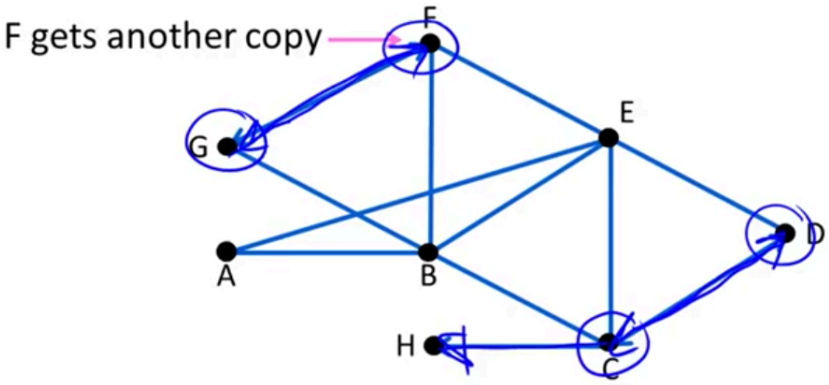


图 213: Flooding 图三

- H has no-one to flood ... and we're done
- See Figure **??**

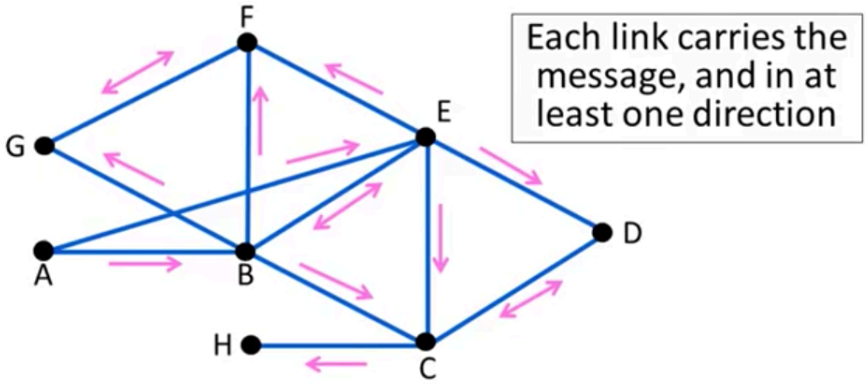


图 214: Flooding 图四

42.3 Flooding Details

- Remember message (to stop flood) using source and sequence number
  - So next message (with higher sequence number) will go through
- To make flooding reliable, use ARQ
  - So receiver acknowledges, and sender resends if needed

43 Week 5 - Link State Routing

43.1 Topic of Link State Routing

- How to compute shortest paths in a distributed network
  - The Link-State (LS) approach

43.2 Link-State Routing

- One of two approaches to routing
  - Trades more computation than distance vector for better dynamics
- Widely used in practice
  - Used in Internet/ARPANET from 1979
  - Modern networks use OSPF and IS-IS

43.3 Link-State Setting

- Nodes compute their forwarding table in the same distributed setting as for distance vector:

- 1. Nodes know only the cost to their neighbors; not the topology
- 2. Nodes can talk only to their neighbors using messages
- 3. All nodes run the same algorithm concurrently
- 4. Nodes/links my fail, messages may be lost

43.4 Link-State Algorithm

- Proceeds in two phases:
  - 1. Nodes flood topology in the form of link state packets
    - Each node learns full topology
  - 2. Each node computes its own forwarding table
    - By running Dijkstra (or equivalent)

43.5 Phase 1: Topology Dissemination

- Each node floods link state packet (LSP) that describes their portion of the topology
- See Figure **??**

43.6 Phase 2: Route Computation

- Each node has full topology
  - By combining all LSPs
- Each node simply runs Dijkstra
  - Some replicated computation, but finds required routes directly
  - Compile forwarding table from sink/source tree
  - That's it folks!

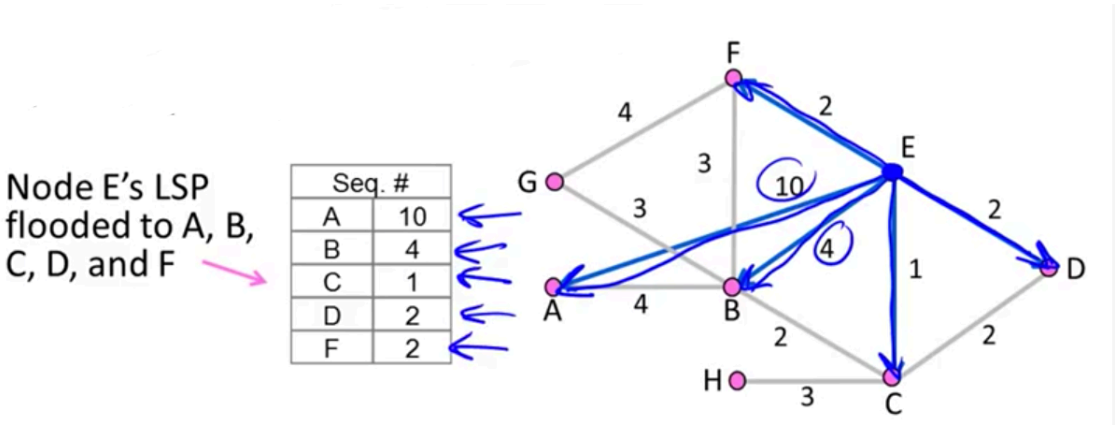


图 215: Phase 1: Topology Dissemination

43.7 Forwarding Table

- See Figure **??**

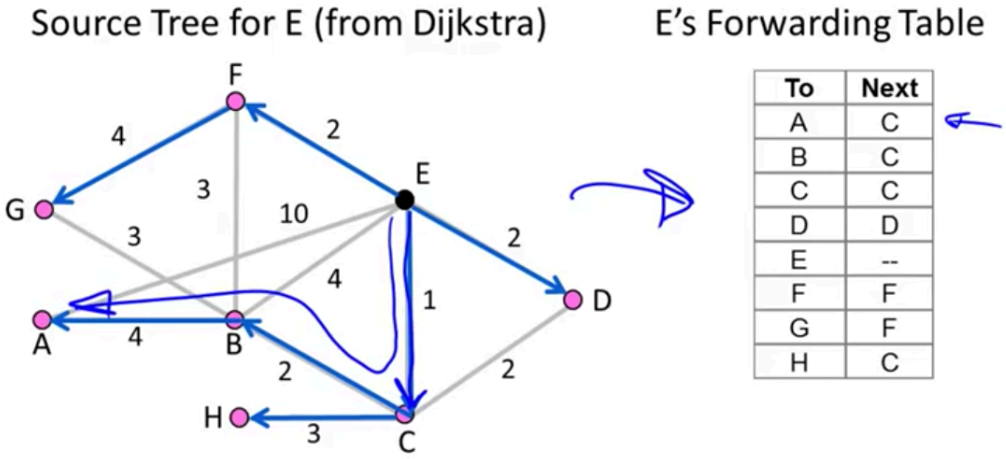


图 216: Forwarding Table

43.8 Handling Changes

- On change, flood updated LSPs, and re-compute routes

– E.g., nodes adjacent to failed link or node initiate

- See Figure **??**

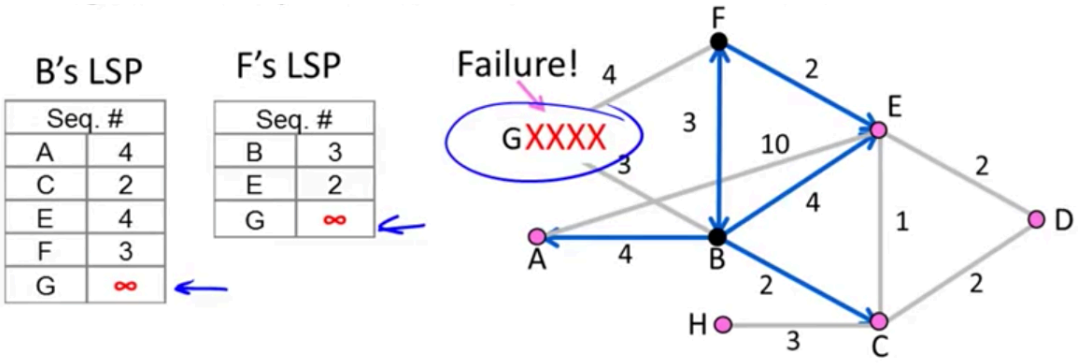


图 217: Handling Changes

- Link failure
  - Both nodes notice, send updated LSPs
  - Link is removed from topology
- Node failure
  - All neighbors notice a link has failed
  - Failed node can't update its own LSP
  - But it is OK: all links to node removed
- Addition of a link or node
  - Add LSP of new node to topology
  - Add LSPs are updated with new link
- Additions are the easy case ...

43.9 Link-State Complications

- Things that can go wrong:
  - Seq. number reaches max, or is corrupted
  - Node crashes and loses seq. number
  - Network partitions then heals
- Strategy:
  - Include age on LSPs and forget old information that is not refreshed
- Much of the complexity is due to handling corner cases (as usual !)

43.10 DV/LS Comparision

- See Figure **??**

Goal	Distance Vector	Link-State
Correctness	Distributed Bellman-Ford	Replicated Dijkstra
Efficient paths	Approx. with shortest paths	Approx. with shortest paths
Fair paths	Approx. with shortest paths	Approx. with shortest paths
Fast convergence	Slow – many exchanges	Fast – flood and compute
Scalability	Excellent – storage/compute	Moderate – storage/compute

图 218: DV/LS Comparision

43.11 IS-IS and OSPF Protocols

- Widely used in large enterprise and ISP networks
  - IS-IS = Intermediate System to Intermediate system
  - OSPF = Open Shortest Path First
- Link-state protocol with many added features
  - E.g., "Areas" for scalability

44 Week 5 - Equal Cost Multi Path Routing

44.1 Topic of Equal Cost Multi Path Routing

- More on shortest path routes
  - Allow multiple shortest paths
- See Figure **??**

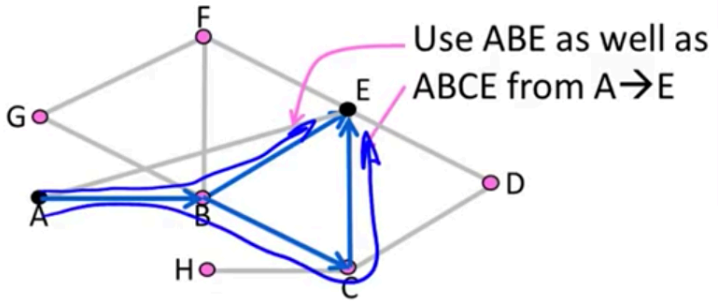


图 219: Topic of Equal Cost Multi Path Routing

44.2 Multipath Routing

- Allow multiple routing paths from node to destination be used at once
  - Topology has them for redundancy
  - Using them can improve performance
- Questions:
  - How do we find multiple paths?
  - How do we send traffic along them?

44.3 Equal-Cost Multipath Routes

- One form of multipath routing
  - Extends shortest path model by keeping set if there are ties
- Consider A $\rightarrow$ E
  - ABE = 4 + 4 =8
  - ABCE = 4 + 2 + 2 = 8
  - ABCED = 4 + 2 + 1 +1 = 8
  - Use them all !
- See Figure **??**

44.4 Source "Trees"

- With ECMP, source/sink "tree" is a directed acyclic (DAG)
  - Each node has set of next hops
  - Still a compact representation

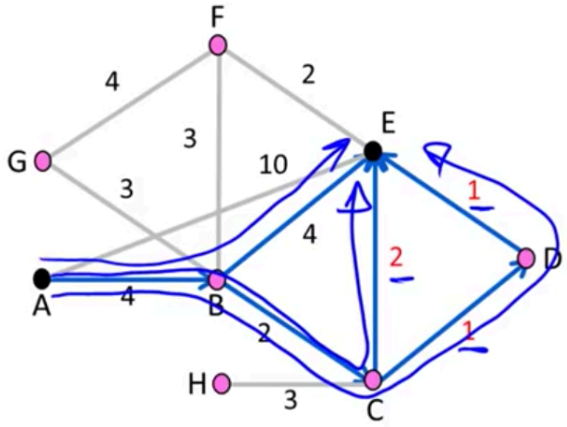


图 220: Equal-Cost Multipath Routes

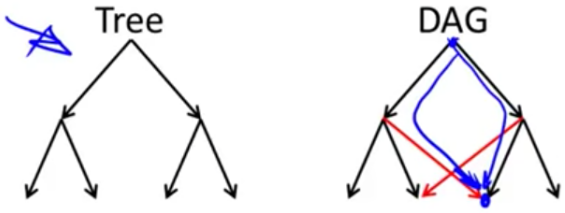


图 221: Source "Trees" 图一

- See Figure **??**
- Find the source "tree" for E
  - Procedure is Dijkstra, simply remember set of next hops
  - Compile forwarding table similarly, my have set of next hops
- Straightforward to extend DV too
  - Just remember set of neighbors
- See Figure **??**

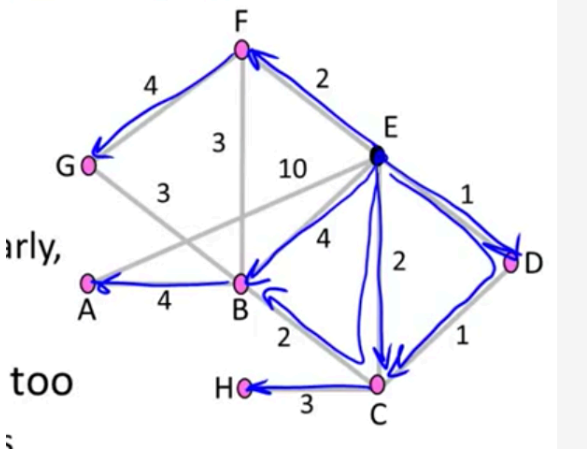


图 222: Source "Trees" 图二

- See Figure **??**

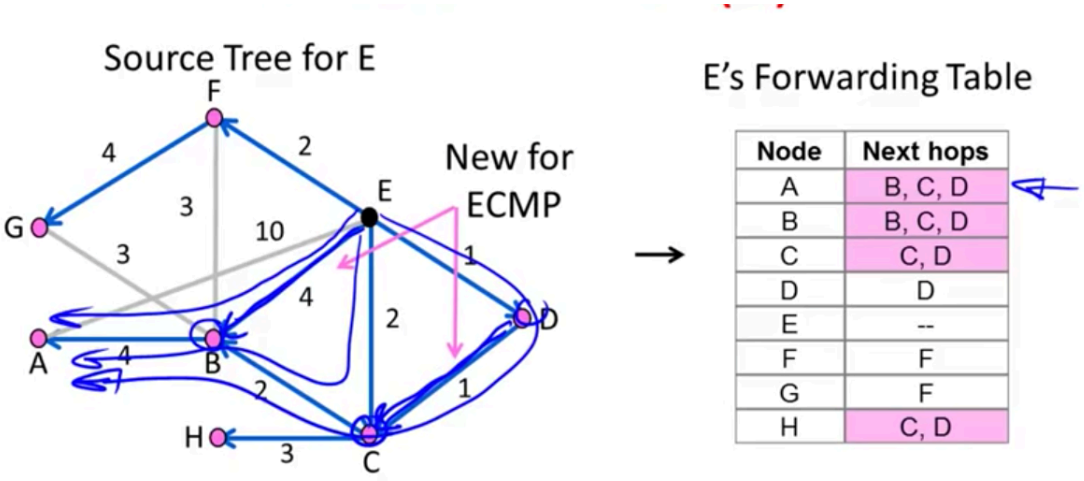


图 223: Source "Trees" 图三

44.5 Forwarding with ECMP

- Could randomly pick a next hop for each packet based on destination

- Balance load, but adds jitter

- Instead, try to send packets from a given source/destination pair on the same path
  - Source/destination pair is called a flow
  - Map flow identifier to single next hop
  - No jitter within flow, but less balanced

- See Figure **??**

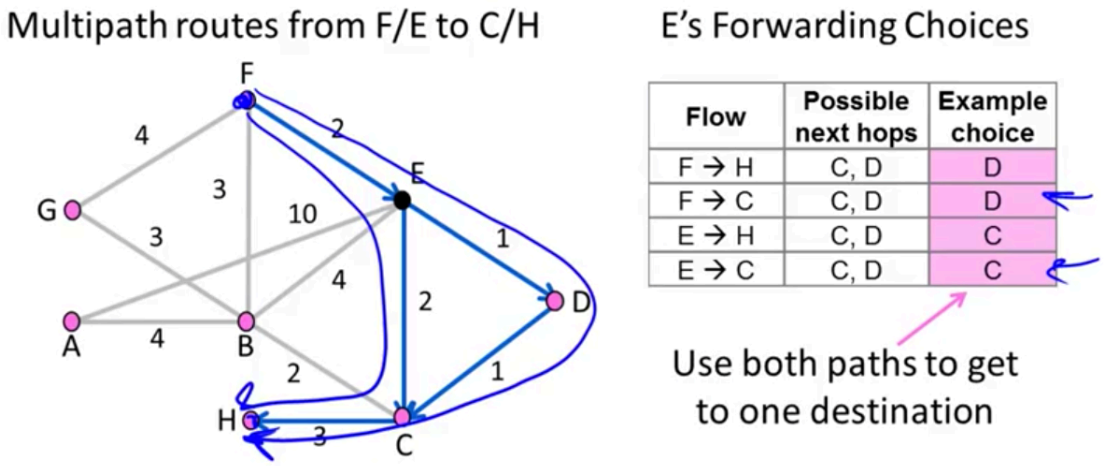


图 224: Forwarding with ECMP

45 Week 5 - Hosts and Routers

45.1 Topic of Hosts and Routers

- How routing protocols work with IP
  - The Host/Router distinction

45.2 Recap

- In the Internet:
  - Host on same network have IP address in the same IP prefix
  - Hosts just send off-network traffic to the nearest router to handle
  - Routers discover the routes to use
  - Routers use longest prefix matching to send packets to the right next hop

45.3 Host/Router Combination

- Hosts attach to routers as IP prefixes
  - Router needs table to reach all hosts
- See Figure **??**

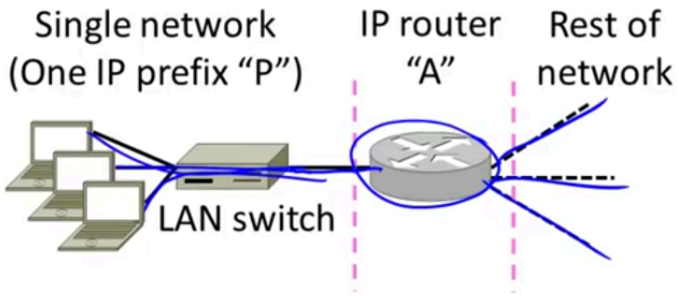


图 225: Host/Router Combination

45.4 Network Topology for Routing

- Group hosts under IP prefix connected directly to router
  - One entry for all hosts

- See Figure **??**

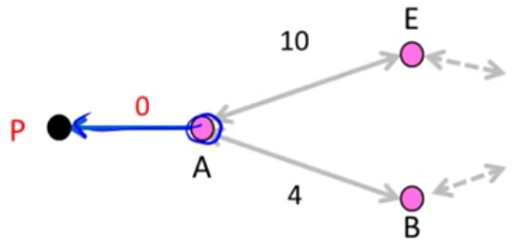


图 226: Network Topology for Routing

- Routing now works as before!
  - Routers advertise IP prefixes for hosts
  - Router addresses are "/32" prefixes
  - Lets all routers find a path to hosts
  - Hosts find by sending to their router

46 Week 5 - Hierarchical Routing

46.1 Topic of Hierarchical Routing

- How to scale routing with hierarchy in the form of regions
  - Route to regions, not individual nodes

46.2 Impact of Routing Growth

- Forwarding table grow
  - Larger router memories, may increase lookup time
- Routing messages grow

  - Need to keeps all nodes informed of larger topology

- Routing computation grows
  - Shortest path calculations grow faster then the size of the network

46.3 Techniques to Scale Routing

- 1. IP prefixes , 之前聊过
  - Route to blocks of hosts
- 2. Network hierarchy , 本次聊
  - Route to network regions
- 3. IP prefix aggregation , 下次聊
  - Combine, and split, prefixes

46.4 Hierarchical Routing

- Introduce a larger routing unit
  - IP prefix (hosts) ← from one host
  - Region, e.g., ISP network
- Route first to the region, then to the IP prefix within the region
  - Hide details within a region from outside of the region
- See Figure **??**
- Penalty is longer paths ， 意思是不好的地方是路径更长。
- See Figure **??**

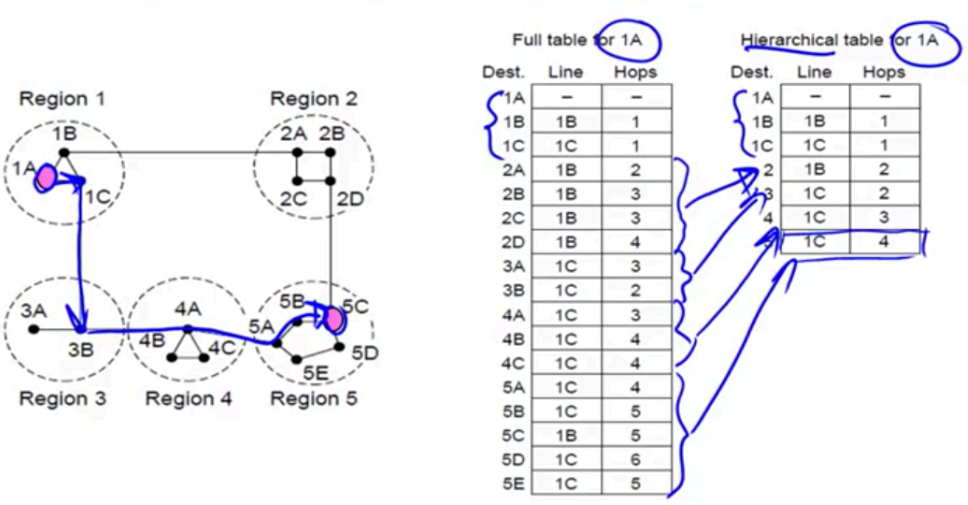


图 227: Hierarchical Routing 图一

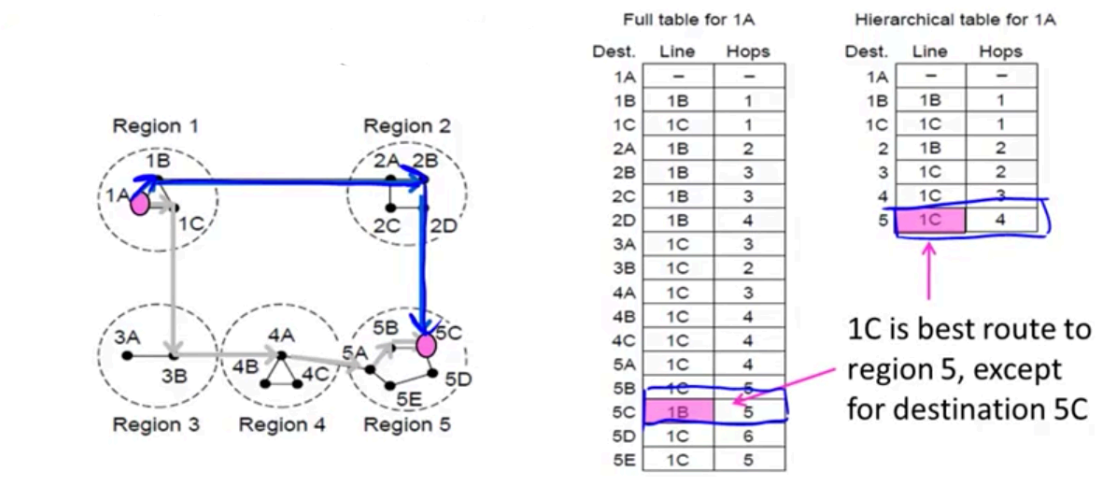


图 228: Hierarchical Routing 图二

46.5 Observations

- Outside a region, nodes have one route to all hosts within the region
  - This gives savings in table size, messages and computation

- However, each node may have a different route to an outside region
  - Routing decisions are still made by individual nodes; there is no single decision made by a region

47 Week 5 - Prefix Aggregation and Subnets

47.1 Topic of Prefix Aggregation and Subnets

- How to help scale routing by adjusting the size of IP prefixes
  - Split (subnets) and join (aggregation)
- See Figure **??**

图 229: Topic of Prefix Aggregation and Subnets

47.2 Recall

- IP addresses are allocated in blocks called IP prefixes, e.g., 18.31.0.0/16
  - Hosts on one network in same prefix
- A "/N" prefix has the first N bits fixed and contain 2^{32-N} addresses
  - E.g., "/24"
  - E.g., "/16"

47.3 Key Flexibility

- Routers keep track of prefix lengths
  - Use it for longest prefix matching

Routers can change prefix lengths without affecting hosts

- More specific IP prefix
  - Longer prefix, fewer IP addresses
- Less specific IP prefix
  - Shorter prefix, more IP addresses

47.4 Prefixes and Hierarchy

- IP prefixes already help to scale routing, but we can go further
  - Can use a less specific prefix to name a region made up of several prefixes
- See Figure **??**

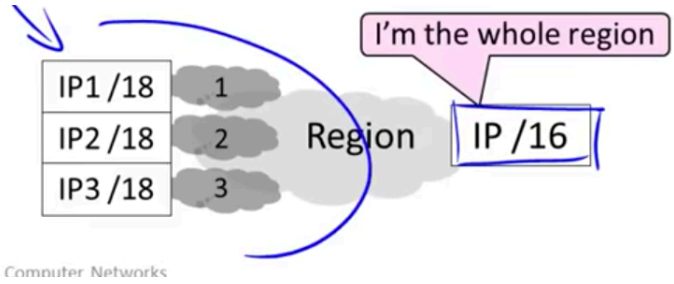


图 230: Prefixes and Hierarchy

47.5 Subnets and Aggregation

- Two use cases for adjusting the size of IP prefixes; both reduce routing table
- 1. Subnets
  - Internally split one less specific prefix into multiple more specific prefixes
- 2. Aggregation
  - Externally join multiple more specific prefixes into one large prefix

47.6 Subnets

- Internally split up one IP prefix
- See Figure **??**

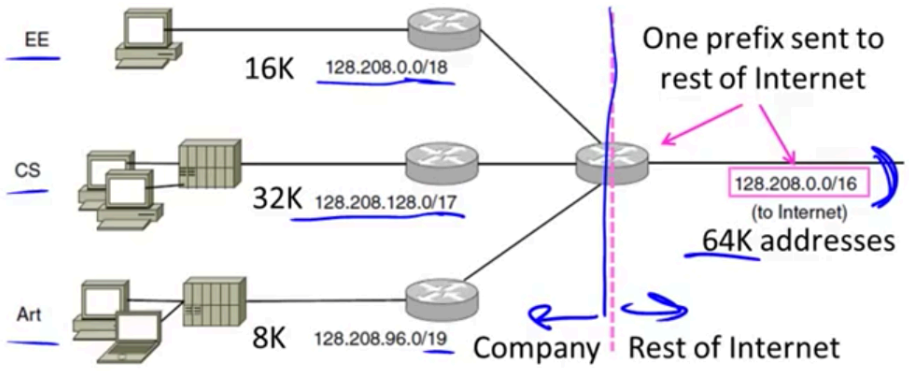


图 231: Subnets

47.7 Aggregation

- Externally join multiple separate IP prefixes
- See Figure ??

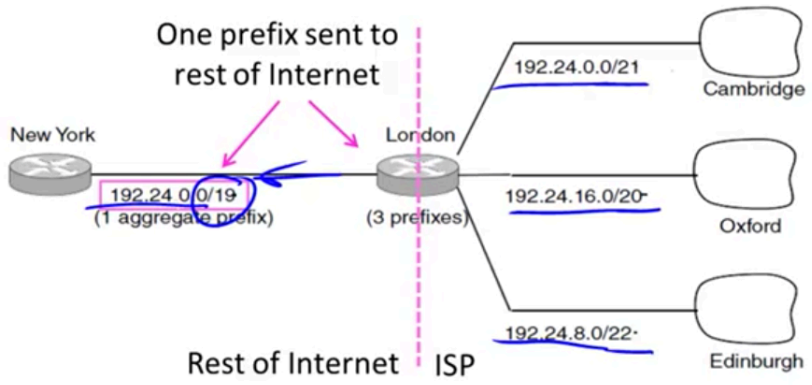


图 232: Aggregation

48 Week 5 - Routing with Multiple Parties

48.1 Topic of Routing with Multiple Parties

- Routing when there are multiple parties, each with their own goals
  - Link Internet routing accross ISPs ...
- See Figure ??

48.2 Structure of the Internet

- Networks (ISPs, CDNs, etc.) group hosts as IP prefixes
- Networks are richly interconnected, often using IXPs
- See Figure ??

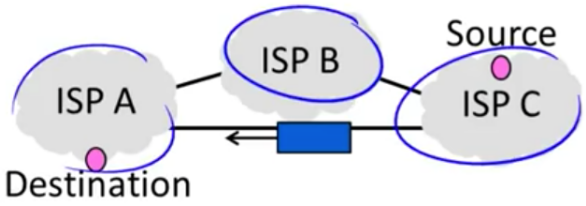


图 233: Topic of Routing with Multiple Parties

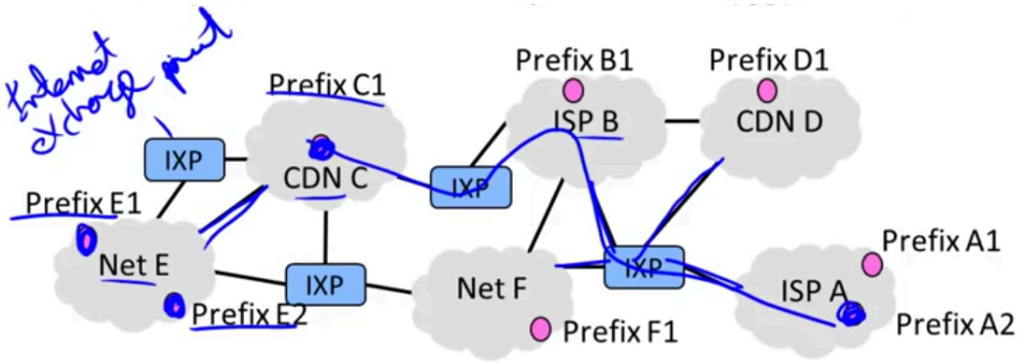


图 234: Structure of the Internet

48.3 Internet-wide Routing Issues

- Two problems beyond routing within an individual network
- 1. Scaling to very large networks
  - Techniques of IP prefixes, hierarchy, prefix aggregation
- 2. Incorporating policy decisions
  - Letting different parties choose their routes to suit their own needs

48.4 Effects of Independent Parties

- Each party selects routes to suit its own interests

- e.g., shortest path in ISP

- What path will be chosen for A2 → B1 and B1 → A2 ?
  - What is the best path ?
- Selected paths are longer than overall shortest path
  - And asymmetric too !
- This is a consequence of independent goals and decisions, not hierarchy
- See Figure **??**

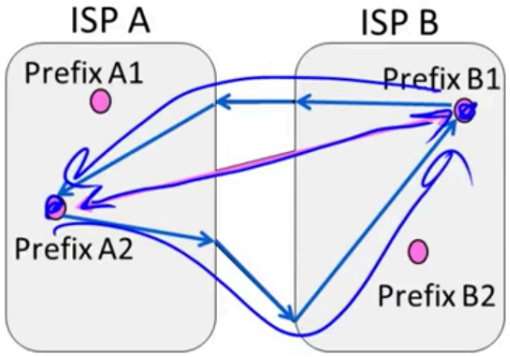


图 235: Effects of Independent Parties

48.5 Routing Policies

- Capture the goals of different parties - could be anything
  - E.g., Internet2 only carries non-commercial traffic
- Common policies we'll look at:
  - ISPs give TRANSIT service to customers
  - ISPs give PEER service to each other

48.6 Routing Policies - Transit

- One party (customer) gets TRANSIT service from another party (ISP)
  - ISP accepts traffic from customer to deliver to the rest of Internet
  - ISP accepts traffic from the rest of the Internet to delivery to customer
  - Customer pays ISP for the privilege
- See Figure **??**

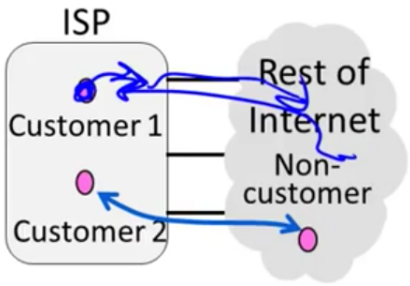


图 236: Routing Policies - Transit

48.7 Routing Policies - Peer

- Both party (ISPs in example) get PEER service from each other
  - Each ISP accepts traffic from the other ISP only for their customers
  - ISPs do not carry traffic to the rest of the Internet for each other
  - ISPs don't pay each other
- See Figure **??**

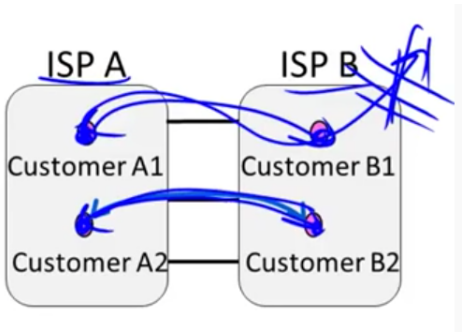


图 237: Routing Policies - Peer

49 Week 5 - Border Gateway Protocol

49.1 Topic of Border Gateway Protocol

- How to route with multiple parties, each with their own routing policies
  - BGP computes Internet-wide routes
- See Figure **??**

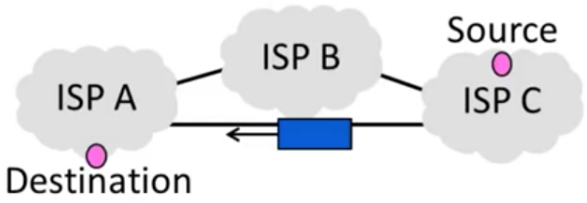


图 238: Topic of Border Gateway Protocol

49.2 Recall

- Internet is made up of independently run networks
- Each network has its own route preferences (policies)

- See Figure **??**

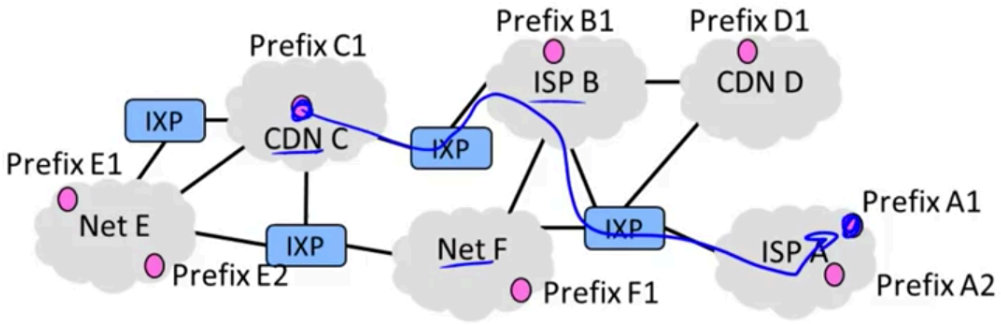


图 239: Recall

49.3 BGP (Border Gateway Protocol)

- BGP is the protocol that computes interdomain routes in the Internet
  - Path vector, a kind of distance vector
- See Figure **??**

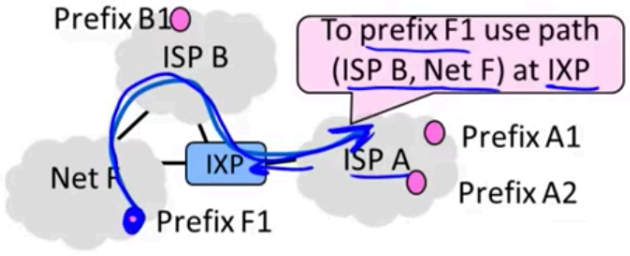


图 240: BGP (Border Gateway Protocol)

- Different parties like ISPs are called AS (Autonomous Systems)
- Border routers of ASes announce BGP routes to each other

- Route announcements contain an IP prefix, path vector, next hop
  - Path vector is list of ASes on the way to the prefix; list is to find loops
- Route announcements move in the opposite direction to traffic
- See Figure **??**

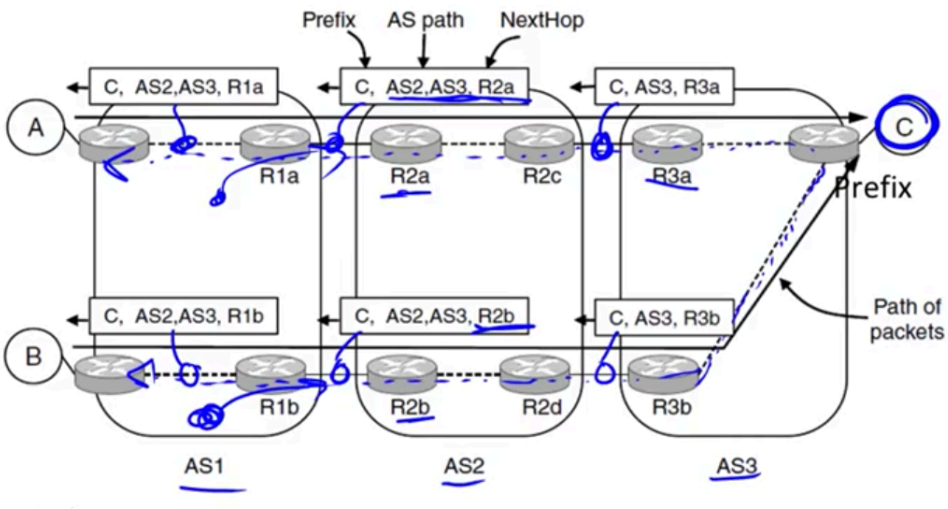


图 241: BGP (Border Gateway Protocol) 图二

- Policy is implemented in two ways
- 1. Border routers of ISP announce paths only to other parties who may use those paths
  - Filter out paths others can't use
- 2. Border routers of ISP select the best path of the ones they hear in any, non-shortest way

49.4 BGP example

- AS2 buys TRANSIT service from AS1 and PEER service from AS3
- See Figure **??**

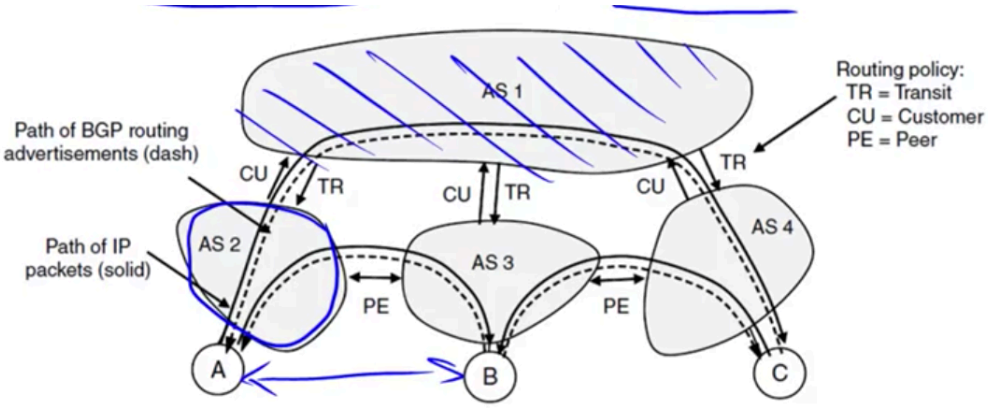


图 242: BGP example

- CUSTOMER (other side of TRANSIT) : AS2 says [A, (AS2)] to AS1
- TRANSIT: AS1 says [B, (AS1, AS3)], [C, (AS1, AS4)] to AS2
- AS2 hears one route to C, and two routes to B (chooses AS3!)

49.5 Closing Thoughts

- Much more beyond basics to explore !
- Policy is a substantial factor
  - Can we be sure independent decisions will yield sensible overall routes ?
- other important factors:

- Convergence effects
- How well it scales
- Integration with routing within ISPs
- And more ...