

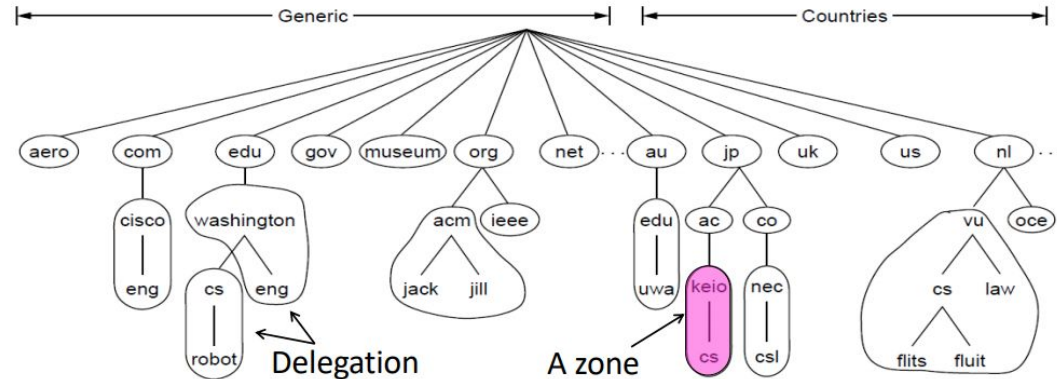
Final Review Cont.

Application Layer

- DNS
- HTTP
- CDNs

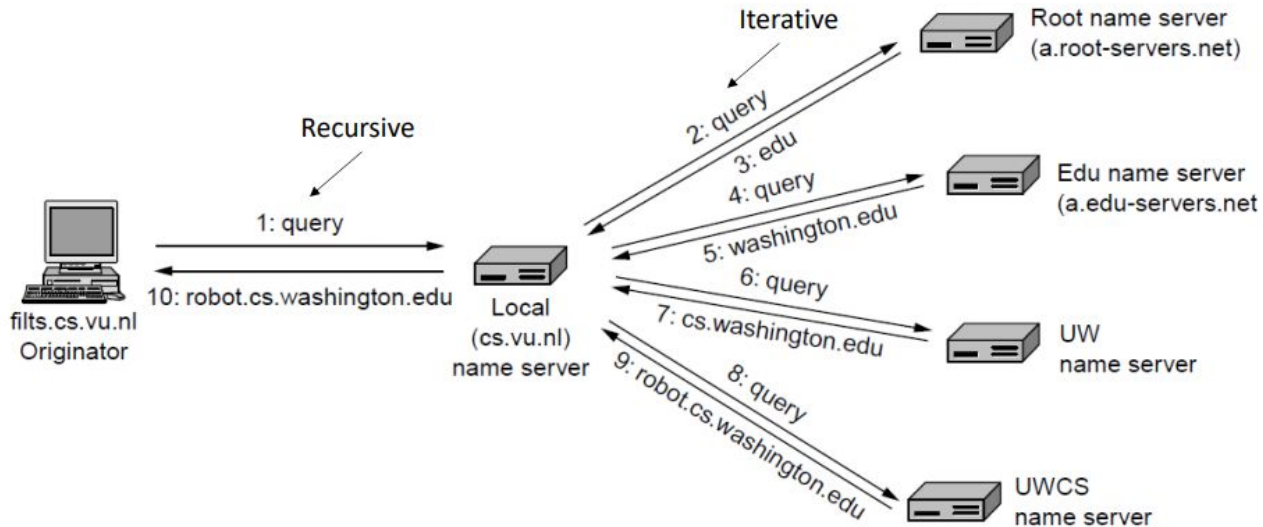
DNS

- A naming service to map between host names and their IP addresses (and more)
 - www.uwa.edu.au -> 130.95.128.140
- Hierarchical namespace, starting from “.”
 - **Zone** - contiguous portion of namespace, basis for distribution
 - Each zone has a **nameserver**



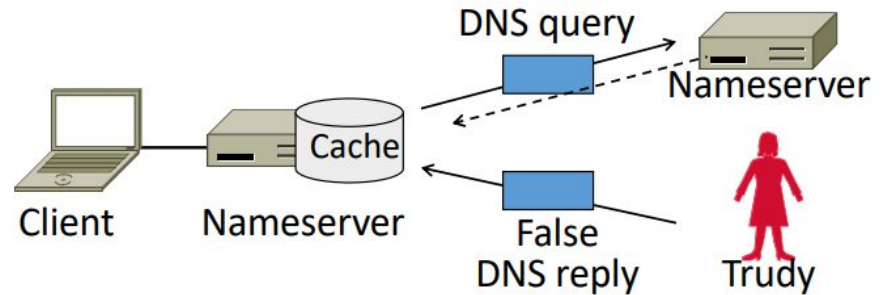
DNS Queries

- Iterative vs. Recursive Queries
 - Recursive query offloads client burden, can cache results
 - Iterative query run on high load servers



DNS Issues

- DNS Caching allows queries to be directly resolved
 - Also enables vector for **DNS Spoofing**
 - Attacker sends fake DNS reply to cache wrong IP binding
- DNSSEC (DNS Security Extensions)
 - Records have digital signatures
 - Public key authentication for DNS resolver
 - Bunch of other stuff



HTTP

Steps to fetch a web HTTP:

- Resolve the server IP
- Setup TCP connection (port 80)
- Send/Receive HTTP request over TCP
- Teardown TCP connection

Steps to fetch a web HTTPS:

- Resolve the server IP
- Setup TCP connection (port 443)
- SSL/TLS negotiation and key exchange
- Send Encrypted messages
- Teardown connection

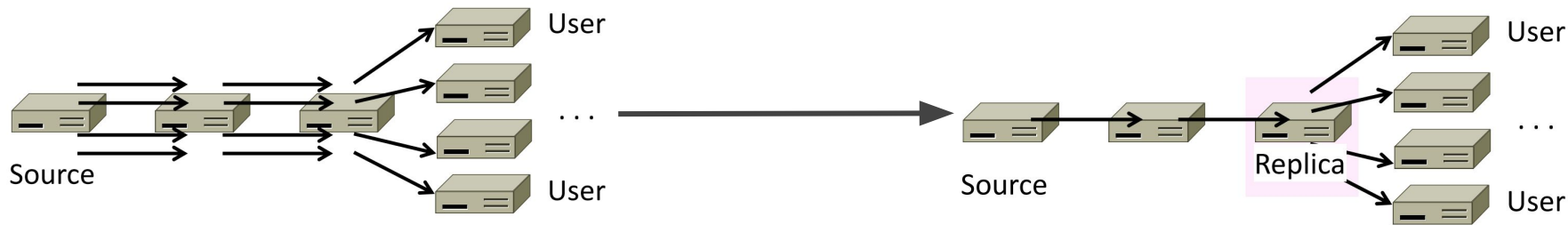
How to decrease Page Load Time (PLT)?

- Parallel connections and persistent connections
- HTTP caching and proxies
- Move content closer to client (CDNs)



CDNs

- Content Delivery Networks
 - *It's just a cache*
- Place popular content near clients
- Use DNS to place replicas across the Internet for use by all nearby clients



Security

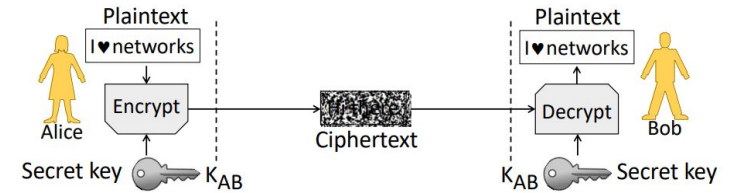
- Crypto
- TLS
- VPNs
- TOR
- DoS

Encryption

Two main kinds of encryption:

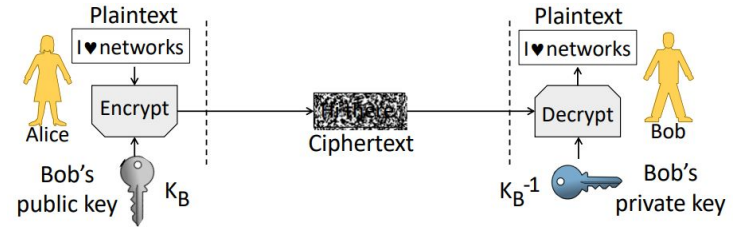
1. Symmetric key encryption, e.g. AES

- Alice and Bob share secret key to encrypt/decrypt



2. Public key encryption, e.g. RSA

- Alice and Bob have public/private key pairs
- Alice encrypts with Bob's public key, only Bob can then decrypt with their own private key



- MAC (Message Authentication Code) can be included to ensure ciphertext's integrity
 - i.e. what if an attacker randomly flipped bits in the ciphertext?

Encryption, Tradeoffs

Property	Symmetric	Public Key
Key Distribution	Hard – share secret per pair of users	Easier – publish public key per user
Runtime Performance	Fast – good for high data rate	Slow – few, small, messages

Combining the two:

- Use public key encryption to send a private key
- Then use that shared private key for symmetric encryption
 - Key is called a ***session key***



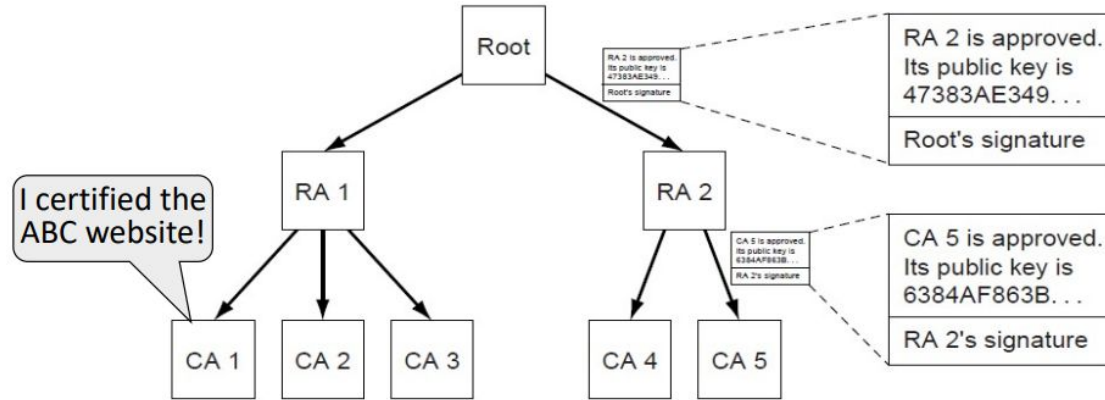
SSL/TLS

- Goal: provide secure transport, client must securely connect to servers not used before
 - Uses public key authentication - but how does a client get a public key from unknown server?
 - With **certificates**
- Certificate - binding of public key to identity/domain
 - Idea, public keys can be distributed when signed by a party you trust



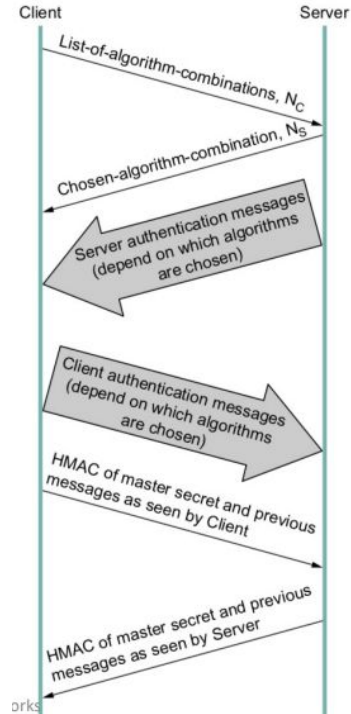
Public Key Infrastructure

- Adds hierarchy to certificates to let parties (Certificate Authorities) issue



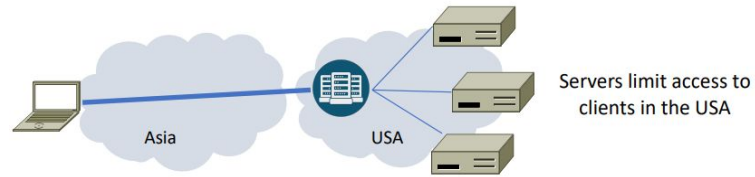
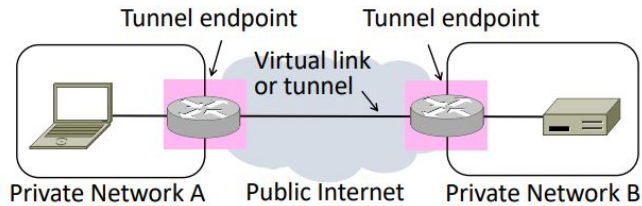
- If we have public key of Root, and trust in servers on path, client can verify public key of website ABC

TLS Handshake



Virtual Private Networks (VPNs)

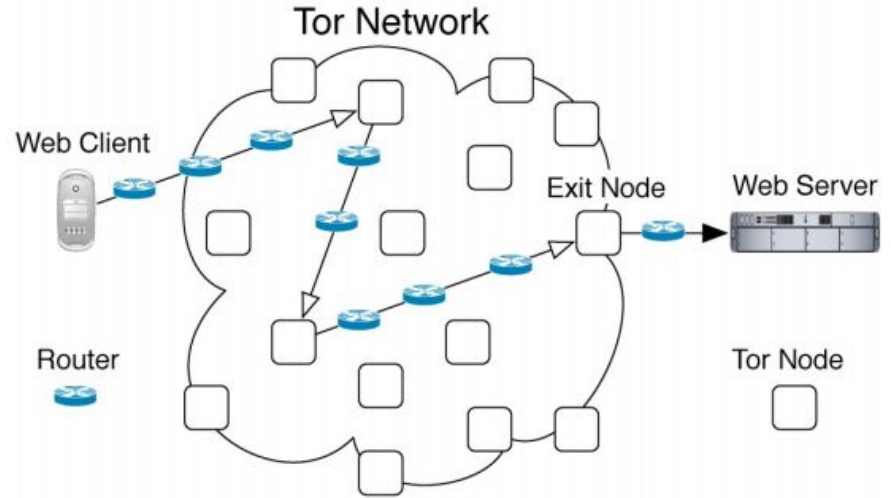
- Connects private networks over the public internet using tunnelling
 - Tunnel endpoints encapsulate IP packets (“IP in IP”) to send across tunnel
 - Uses IPSEC to secure tunnel



Tor: “The Onion Router”

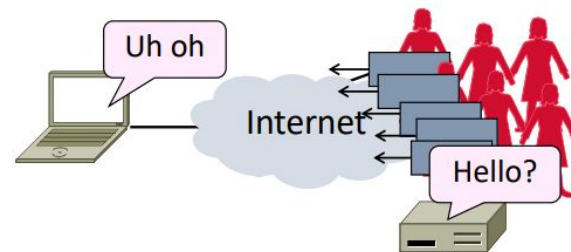
Basic idea:

1. Many volunteers act as routers in the overlay
2. Generate circuit of routers that will send packet
3. Encrypt the packet in layers for each router
4. Send the packet
5. Each router receives, decrypts their layer, and forwards based on new info
6. Routers maintain state about circuit to route stuff back to sender
 - But again, only know the next hop

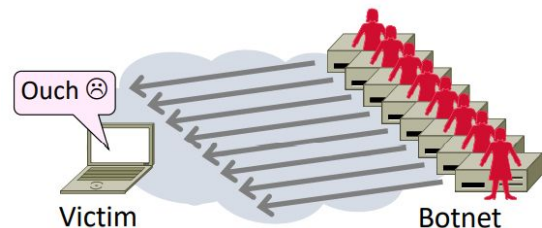


Resource Attacks

- Distributed Denial-of-Service (DDOS)
 - Attack on network availability
 - Flooding a host with many packets to interfere with its IP connectivity
 - Attackers can falsify their IP address, or spoof source IP addresses in replies
 - Not all ISPs do ingress filtering



- Botnet
 - Provides many attackers from compromised hosts

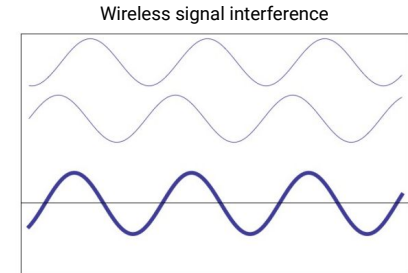
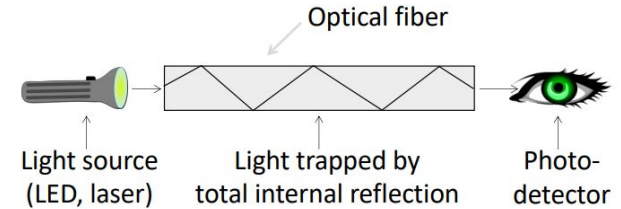


Physical Layer

- Media Types
- Coding and Modulation schemes
- Fundamental limits

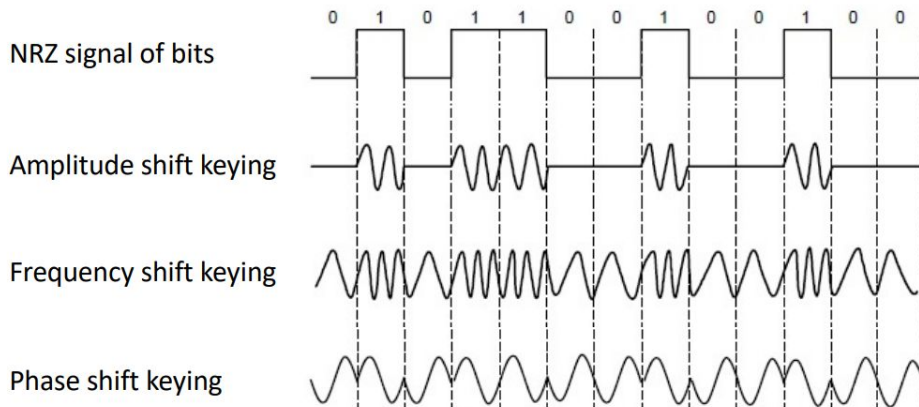
Physical Layer

- Physical layer - translate analog signal on wires into digital bits
- Media types
 - Wired - twisted pair, coaxial
 - Fiber - long, thin strands of glass
 - Wireless - broadcasted signal
 - Potentially many receivers
 - Nearby signals *interfere* at a receiver



Coding and Modulation

- How can we send information across a link?
- Coding - directly on a wire
 - Simplest scheme - Non-Return to Zero, high voltage represents 1, low voltage represents 0
 - Better schemes account for **clock recovery** - 4B/5B coding, Manchester encoding, etc.
- Modulation - carrying RF signals by modulating a carrier signal

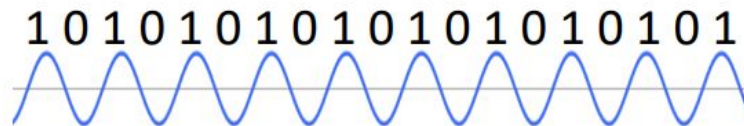


Fundamental Limits

- Key Channel Properties - B: Bandwidth (hertz), S: Signal strength, N: Noise

- Nyquist Limit

- Maximum **symbol rate** is $2B$
- With V signal levels, *ignoring noise*, max bit rate $R = 2B \log_2(V)$ bits/sec



- Shannon Capacity

- Signal-to-Noise ratio (S/N) determines number of distinguishable voltage levels
- Max carrying rate $C = B \log_2(1 + S/N)$ bits/sec
 - Max rate at transmitting *without loss* over a random channel
 - Increasing signal power gives diminishing returns
 - Increasing bandwidth increases capacity linearly
- How does Shannon Capacity affect wired/wireless designs?

Fundamental Limits

- Bandwidth-Delay Product
 - Describes current amount of data in transit
 - $BD = R \times D$, measures in bits, or in messages



Link Layer

- Framing
- Error detection, correction
- Multiple Access
- Switching

Link Layer, Framing

- Link layer - transfer frames over one or more connected links
- Framing - how do we interpret a stream of bits as a sequence of frames?
 - Fixed-size frames (motivation), wasteful for small payloads
 - Byte count (motivation), store length of frame as header for frame
 - Byte stuffing/bit stuffing, encode flag at byte/bit level at start and end of frame




Error Detection, Correction

- Noise may flip some received bits in message
- Key idea: error detection/correction codes
 - Add check bits to the message to let some errors be detected (and more required for correction)
- Tradeoffs:
 - Adding check bits -> larger bandwidth requirements
 - Need modest computation to detect/correct error

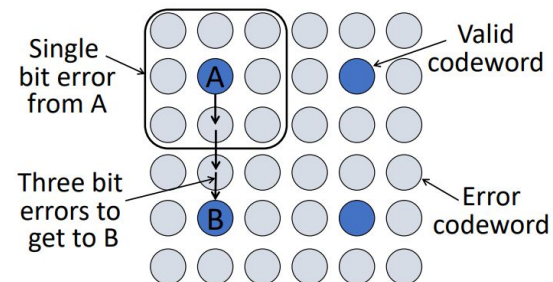


Error Detection, Correction cont.

- **Hamming distance** of a coding - minimum error distance between any pair of codewords that cannot be detected
 - e.g. 1-dimensional parity has Hamming distance of 2, since 2 bit-flips required to get to next valid codeword
 - **1 0 0 0 | 1** -> **0 1 0 0 | 1**, both pass error detection
 - Error detection:
 - For a coding of distance $d + 1$, up to d errors will always be detected
 - Error correction:
 - For a coding of distance $2d + 1$, up to d errors can always be corrected by mapping to closest valid keyword
 - Larger messages use **checksums** for error detection
- 

Error Detection, Correction cont.

- Why is error correction harder?
 - Need to narrow down position of the error, but errors can also occur in the check bit
 - Two-dimensional parity can correct 1-bit errors by identifying row/column with error
- With a code with a Hamming distance of at least 3,
 - Single bit errors will be closest to a unique valid codeword
- Error correction:
 - Needed when errors are expected
 - Or no time for retransmissions
 - Used in physical layer (802.11, DVB, WiMAX), application layer (FEC)
- Error detection:
 - More efficient when errors not expected
 - And errors are large when they do occur
 - Used with retransmission in link layer and above layers for residual errors



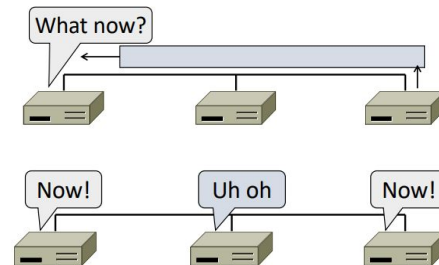
Multiple Access

- Multiplexing shared link usage
 - Time Division Multiplexing (TDM) - users take turns on a fixed schedule
 - Frequency Division Multiplexing (FDM) - users placed on different frequency bands
 - Tradeoffs?
- Centralized vs. Distributed Multiple Access
 - Centralized, use a “scheduler” to choose who transmits and when
 - E.g. cellular networks (tower coordinates)
 - Scales well + efficient, but long setup and management
 - Distributed, have participants “figure it out” somehow
 - E.g. WiFi networks
 - Operates well in low load and easy setup, but hard to scale



Distributed (random) Access

- How do nodes share a single link? Who sends when?
 - Aloha - just send it whenever
 - CSMA (Carrier Sense Multiple Access) - listen for activity before we send
 - CSMA/CD (with Collision Detection) - receiver detects collision and aborts (jams)
- CSMA “Persistence”
 - Each node waits $2 * \text{delay}$ seconds to hear of a collision
 - Problem - multiple waiting nodes will queue up, then collide
 - Ideally want N senders to send with probability $1/N$
 - Solution: Binary Exponential Backoff (BEB)
 - 1st collision, wait 0 or 1 frame times
 - 2nd collision, wait from 0 to 3 times
 - 3rd collision, wait from 0 to 7 times

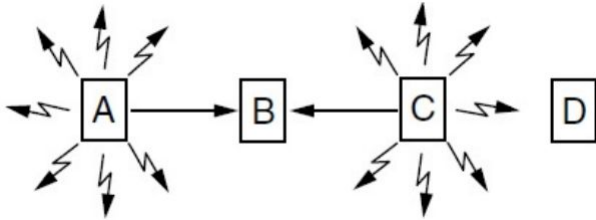


Wireless Complications

1. Media is infinite - can't Carrier Sense
2. Nodes (usually) can't hear while sending - can't Collision Detect

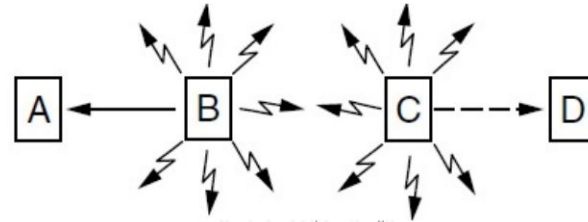
Hidden Terminal Problem:

- A and C can't hear each other, but collide at B



Exposed Terminal Problem:

- B and C can hear each other, but don't collide at A and D



MACA

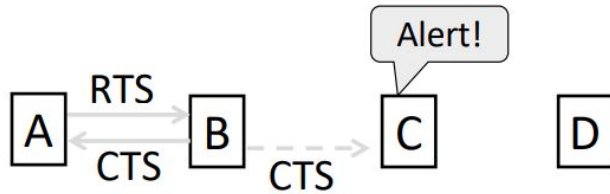
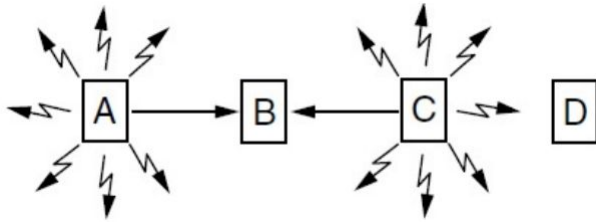
- Uses short handshake instead of CSMA
- Protocol rules:
 1. A sender node transmits a RTS (Request-To-Send, with frame length)
 2. The receiver replies with a CTS (Clear-To-Send, with frame length)
 3. Sender transmits the frame while nodes hearing the CTS stay silent



MACA

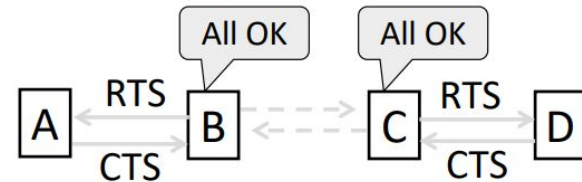
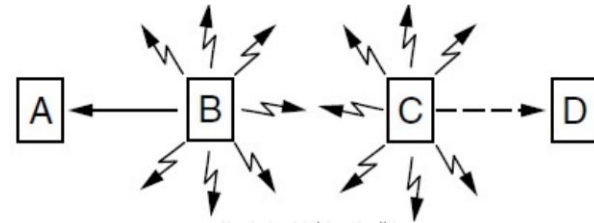
Hidden Terminal Problem:

- A and C can't hear each other, but collide at B



Exposed Terminal Problem:

- B and C can hear each other, but don't collide at A and D



Switching

- Modern Ethernet uses switches instead of multiple access
 - Convenience running wires to one location
 - More reliable - wire cut is not a single point of failure
- Switch Forwarding
 - Switch needs to find right output port for destination address
 - Backward learning
 - To fill table, looks at source address of input frames
 - To forward, looks up address in table or broadcasts to all ports if not found
- Issue - forwarding loops
 - Solution, each switch runs distributed *spanning tree* algorithm
 - Finds subset of links w/ no loops and reaches all switches

