General Overview

Forwarding

* Router-local of transferring packet from input link interface to an outbound link interface
* Process:
  1. Receives packet on in-bound link
  2. Decodes back header and determines destination
  3. Finds out-bound link associated with destination in forwarding table
  4. Send packet to appropriated out-bound link

Parts of the router

* + Input ports
  + Switching fabric
    - Connects input and output ports
    - Three types:
      * Memory: Simplest form, routing processor reads value rom input, copies to output
        + Speed limited by memory bandwidth
      * Bus: All input/output ports connected to shared bus, input port addresses packet with output Id and sends it into bus
        + Everyone receives, ignored by unintended recipients
        + One packet on bus at time
        + Speed limited by bus speed
      * Crossbar: Interconnected network of addressable busses, switch controller closes connection between busses before sending
        + Each input can send packet simultaneously as long as not talking to same output
      * Dropp
  + Output ports
  + Routing processor
* Line Rate = Throughput of input /output lines

Two forwarding paradigms:

* + Destination-based forwarding: Based on IP address
    - Lookup tables map destination address ranges to output link interfaces
    - Managed by routing processor, copied to link interfaces
    - Might not be clean division between ranges though
      * When multiple entries match an address, use longest address prefix
      * Performed using ternary content addressable memories (TCAMs)
        + Lookups need to be extremely fast
        + 10 Gb line, full utilization, MSS of 1500 bytes:

New packet arrives every 1500B/10Gb a second

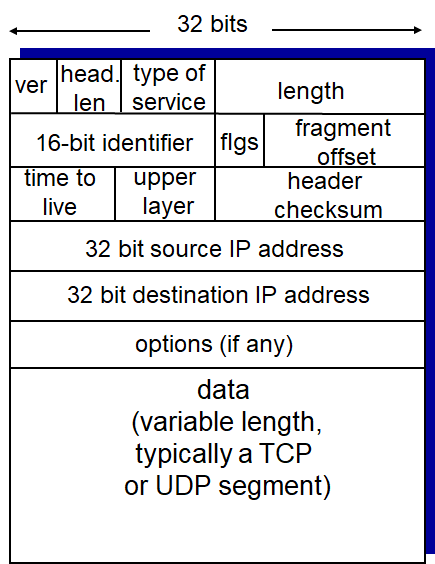
1.2 microseconds

* + Generalized forwarding: Using any header field
    - Not being discussed

Packet transfer:

* Switching Rate (Rswitching): Rate at which packets can be transferred
  + Multiple of input/output line rate
  + Rswitch > Rinc or packets will be queued
  + Ideally Rswitch >= N \* Rline, N = input ports
* Input Queuing:
  + Rswitch < N \* Rline, queuing
  + Rswitch > N \* Rline if:
    - HOL blocking
    - Packets at multiple ports go to same port
    - This can block packets further back
* Output port:
  + Even with an increase Rswitch, packets can be blocked at output ports
  + Multiple packets will be delivered to same output port before packet is sent
  + Needed buffer space:
    - Rule of thumb: Average bufferinging = RTT \* C (link capacity)
    - Recent recommendation: (RTT \* C) / sqrt(N)
* Packet dropping:
  + Can drop any packet in queue
  + Drop-tail: Drop incoming packet
  + Drop random packet
  + Drop packets based on priority (email vs VoIP)
  + AQM (active queue management), different techniques
    - Eg. Drop packets before queue is full as signal to sender that congestion is coming
* Packet scheduling:
  + First-in First-out
    - Send packets in order received
  + Priority queuing:
    - Create multiple queues for packets segmented by priority
    - When priority packets received in middle of lower priority packet being sent:
      * Can finish sending (pre-emptive queueing)
      * Can cancel
  + Round Robin
    - Different priorities queues, doesn’t necessarily send high priority first
      * Loop through queues sending packets one-by-one
  + Weighted fair queuing
    - Like roub robin, might send packets from queues at different rates

IP Protocol



Internet implements IPv4 and IPv6 at network layer (as well as ICMP)

Header fields:

* + Ver – IP version number (V4 / V6)
  + Head len – Length of header (decremented in v6)
  + Type of service (bits to distinguish of datagrams)
  + Legth – How long datagram is
  + Identifier, flags, fragment
    - Used for V4 fragmentation
    - Not in V6
  + Time To Live (TTL) – Records how many hops datagram has made, dropped when reaches 0
  + Upper Layer Protocol: Transport layer protocol
  + Header checksum: Only for header
  + Source and destination IP address
  + Options
  + Data

Header overhead

* + IPv4 + TCP = 20B + 20B
  + IPv4 + UDP = 20B + 8B
  + (Ethernet will add 14-18B and dead times before frames)

Fragmentation:

* + Packet will need to cross link with smaller effective MTU than 1500 B
    - Such as wide-area links, 576B
    - Other protocols needed to cross link
  + Packet fragmented, sent out in smaller chunks, reassembled at destination
  + Duplicates headers
  + Uses three header fields:
    - 16-bit ID increment will be shared
    - Flags (don’t fragment will drop packet, more fragment set if part of a fragment but not last)

Addressing:

* IP addresses trace origin back to Internet Corporation for Assigned Names and Numbers and their Regional Internet Registry
* Allocated in subnets
  + 223.11.1.1/26 -> 11011111 00001011 00000001 00000001
  + 26 bytes highlighted, address in this subnet have this prefix -> 26 possible addresses in unhighlighted block
  + Organizations (eg ISPs) allocated subnet blocks by ICANN
* Device IP allocation:
  + Static IPs – Manually assign MAC address to fixed IP addresses
  + DHCP – Dynamic Host Configuration Protocol
* DHCP
  + Automatically assigns IP addresses based on available addresses
  + Server must be hosted on network (often in routers)
  + Server discovery
    - New host sends DHCP discovery message to UDP port 67
    - Sends message with destination 255.255.255.255
    - Source address 0.0.0.0
  + Server offer(s)
    - Whe server receives discovery, makes offer to address
    - Server responds with broadcast message which includes transaction ID
  + Request
    - Client cooses from offers and responds with a request message that echos parameters
  + ACK
    - Server ACKs request, Finalizes whole process
* Network Address Translation
  + 4,294,967, 296 addresses , 23 billion devices
  + NAT routers connect to outside world, receives single IP address
  + Sends messages with this IP address, but receiving messages don’t know who sent it
    - Routers logs information to NAT table

Routing

Internet too large to run routing over as a whole – is segmented and running algorithms at different levels

* Edge networks (Clemson, Home networks)
* ISPs, which may subdivide nets
  + ISPs divided into autonomous systems
* Connections between ISPs
  + Border Gateway Protocol

Routing algorithms:

* Two paradigms:
  + Centralized (link-state routing algorithm)
    - Every router knows state of whole system
  + Decentralized (distance vector routing algorithm)
    - No node has complete knowledge
* Link-state
  + Centralized
  + Based on Dijkstra’s algorithm
    - Iterative
    - Builds up knowledge of shortest paths
    - Naïve implementation is O(n2), can achieve O(n log n)
  + OSPF is form of LS algorithm
* Distance-vector
  + Decentralized
  + Based on Bellman-Ford equation

Link Layer

Transmit frames along link between two+ nodes

Broadcast transmission

* Operate over shared medium
* All nodes attempt to send to all simultaneously
  + Results in collisions

Point-to-point tranmissions

* Non-shared mediums
  + Each link isolated in its own collision domain
* Modern LANs / Ethernets operate using switches that remove possibilities of collisions
  + Early Ethernet used bus design

MAC Addresses

* MAC address, 48 bit
* Assigned to specific interface element
* Burned into ROM device, sometimes programmable
  + All unique, unless programmable
* Six pairs of hexadecimal numbers
* Managed by IEEE
  + Manufacterers assigned 24-bit spaces of MAC addresses
  + 3 pairs fixed, last 3 vary
  + Number not relevant to network
* Address Resolution Protocol (ARP)
  + Get MACs from IP address using ARP
  + Each IP node has ARP table:
    - IP, MAC, TTL
  + Node A wants to send to B it looks up MAC address in table
    - If in table, we’re fine
    - If not, broadcast query packet containing B’s IP Packet, MAC = FFFFFFF
  + Only within given subset
    - Routers bridge subnets

Ethernet

* Early system used bus toplogy
* Modern ones use star topology
  + Active switch in center of LAN
  + Spokes run separate Ethernet protocols
* Frame structure
  + Sending adapter encapsulated IP datagram in Ethernet frame
  + Preamble
    - 7 bytes with pattern 101010 followed by byte pattern 10101011
    - Used to synchronize receive, sender clock rates
  + Addresses:
    - 6 byte source/destination MAC addresses
  + Type: indicates higher layer protocol (usually IP)
  + CRC: Cyclic redundancy check at receiver
* Unreliable, connectionless
  + No handshaking, ACKs, or NAKs
* Switch
  + Routers but don’t implement network layer
    - Stores and forwards Ethernet frames
    - Examines MAC addresses, slecively forwas frame
  + Transparent, hosts unaware of switches
  + Plug-and-play
  + Forwarding tables
    - Using switch table
    - MAC, Interface, TTL

Network Security

Key aspects:

* Confidentiality: Only sender and recipient understand contents
* Authentication: Sender and receiver confirm identity of each other
* Integrity: Ensure message not altered
* Access & availability: Services must be accessible and available to users

Risks:

* Eavesdropping: Intercept messages
* Impersonate: Fake source address
* Hijack: Take over ongoing connection
* Denial of service: Prevent service from being used

Cryptography:

* Prevents eavesdropping, impersonating, hijacking
* Symmetric key cryptography
  + KA and KB are same
  + Relies on sender and recipient knowing key, but no one else
  + Caesar cipher: Shift letters right by k places
    - Simple, easy to break
  + Monoalphabetic cipher:
    - Letter placed by another letter, random mapping
    - 26! Permutations
    - Cipertext-only: Use statistics of known language
    - Known-plaintext: Look for words you suspect are in message
    - Chosen-plaintext: Getting sender to send message and intercept its cypher
  + Poloyalphabetic cipher:
    - Multiple ciphers per message
  + Modern encryiption: Block ciphers
    - Break message into blocks of k
    - Repeatedly encrypt blocks with different keys
    - Shuffle bits of blocks
    - Confusion and diffusion
* Public key cryptography
  + Uses secret and public key
    - Senders encrypt using recipient’s private key
    - Message only decrypted with private key
  + RSA: Primary method
    - Relies on modular arithmetic, modulo operator represented as %
    - Too computationally expensive to hack (unless…quantum)
* Both worlds:
  + Problem with AES – Establishing shared secret key
  + Can use public key to exchange secret key with AES

Ensuring integrity:

* Cryptographic hash function
  + Takes in m, computed H(m)
  + Computationally infeasible to find any two different messages x and y such that H(x) = H(y)
  + Recipient computers H(m) and compares it to receives hash
* Message Digest 5 (MD5)
  + Produces 128 bit hasj
* Secure Hash Algorithm (SHA-1)
  + Produced 160 bit hash, more secure
  + Federally mandated for hased federal communication
* Guarantee integrity, not identity
  + Can append secret key to hashed message
  + Message Authentication Code
* Digital signatures
  + Without shared secret key
  + Bob will encrypt message using private key, recipients will decrypt using public key
  + Keys have certificate authorities