ICS 451: Today's plan

- BGP summary
- data link layer
- framing
- error detection and correction

BGP messages

- open: session initialization
- notification: session shutdown
- update
  - withdraw route (*withdraw* message)
  - new route (*update* message)
  - update existing route (*update* message)
- keepalive: avoid 90s timeout
  - sent if there is ever a 30s or longer gap in update messages

RIB and FIB (and MIB)

- Routing Information Base contains all acceptable routes
  - may include multiple routes to same destination
- Forwarding Information Base contains at most one route to each destination
  - used to actually forward packets
- originally, Management Information Base contains description of managed systems

Routing Policy

- routes with higher local preference used first
  - customer routes have high local pref
  - provider (expensive) routes have low local pref
- shortest AS path used when local pref is same
- prefer routes with smallest Multi-Exit Discriminator (MED) among same AS routes
- prefer routes going outside AS
- prefer routes with closest next-hop
- break ties with lower router ID

BGP summary

- used for interdomain routing
- policy more important than path optimality
- TCP connections allow focus on maintaining state rather than ensuring packet delivery
- connections set manually between trusted peers

Data Link Layer

- the layer below IP
- used for point-to-point links or Local Area Networks (LANs)
  - LANs often have a shared medium
  - so need Medium Access Control (MAC)
- uses MAC addresses (if any)
- examples include Ethernet and 802.11/WiFi

layering

- network layer sends and receives packets (datagrams)
- physical layer sends or receives collections of bits, called frames
  - frames may be fixed size, or have a range of sizes: n ≤ bytes ≤ m
    - ATM only has 53-byte cells
      - 5-byte header + 48-byte payload
  - an adaptation layer may allow sending packets smaller than n or greater than m

Physical Layer limitations

- bit errors
  - due to electromagnetic interference
  - or clock mis-match
- packet loss
- framing errors (e.g. receiving > m bits)

Framing Problem

How does a sender encode frames so that the receiver can efficiently extract them from the stream of bits that it receives from the physical layer?

(textbook, section 6.1.1)

Framing

- each bit (or set of bits) encoded by a symbol
  - e.g. a voltage level, a frequency, on/off, etc
- some encodings have extra symbols
  - e.g. in Manchester encoding
  - low->high voltage is 0
  - high->low voltage is 1
  - all high or all low voltage is not a valid symbol
    - can be used to mark the end of the frame

Bit and Byte stuffing

- if there are no extra symbols
- reserve one symbol for marking the frame, another symbol as an *escape*
- when sending a reserved symbol, escape it
  - by preceding it with the escape symbol
- an unescaped frame symbol marks the frame
- escaped symbols represent the value
- if escape is a bit, this is *bit stuffing*
- if escape is a byte, this is *byte stuffing*

Bit stuffing

- use symbols with many (e.g. 6) successive '1' bits as frame markers
- when sending data, add a 0 bit after any 5 successive bits of 1
- when receiving 5 successive bits of 1, if the next bit is:
  - 0, remove it and continue receiving
  - 1, this is a frame marker

Byte stuffing: slip

- serial line IP
- END character and ESC character
- if either occurs in the data, replace with ESC+x
  - x has one of two values to escape END or ESC
- so END never occurs in the data
  - only at the end (or start) of a frame
- receiver seeing ESC+x decodes sequence of two characters to one data character
  - either END or ESC

error detection

- add redundant data to a frame
  - computed from frame contents
  - usually at end or beginning of frame
- receiver recomputes redundant data
  - discards frame if no match
- detects errors in the data or in the error detection code

- TCP/UDP/IP checksum uses similar principles

parity

- add a single bit to a frame
  - even parity: add 1 if data has odd number of '1' bits, add 0 otherwise
  - odd parity: opposite
- detects any single-bit errors
  - and any errors flipping an odd number of bits
- fails to detect any errors flipping an even number of bits

CRC

- Cyclic Redundancy Check
- divide the bits in the frame by a fixed value
  - polynomial division with coefficients 0 or 1
  - easy to implement in hardware
    - with a special shift register and some XORs
- put the remainder (fixed # bits) in the frame
  - detects all odd numbers of bit errors, 2-bit errors, burst errors of less than N bits
- many standard CRCs, e.g. CRC-32
- Wes Peterson (University of Hawai'i), 1962

Forward Error Correction (FEC)

- send enough redundant bits that receiver can correct limited number of errors
- e.g. send 3 copies of each message
  - use majority voting to reconstruct original
- CRCs can be used to correct some errors
  - there are many error-correcting codes
  - send more bits to correct more errors
  - in general, some errors cannot be corrected