ECE 435 – Network Engineering
Lecture 16

Vince Weaver

http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

26 October 2017

Announcements

- No homework this week!

Questions from Last Time

- Can BGP due IPv6?
  Yes, despite BGP4 being older than IPv6, it was designed with extensions in mind and can handle IPv6. It apparently gets complex when both IPv4 and IPv6 involved
- Programming routers – some classes will do this.
  I don't really have any. Somewhat boring too, can get Cisco certified, traditionally hook up a serial port to your laptop and enter a lot of arcane commands There are

simulators, even ones that use virtual machines.

- Actual Router
- Can install on your Linux machine
- Zebra was traditionally, discontinued
- Quagga
- BIRD
- OpenBGPD and OpenOSPFD
- Potentially dangerous to mess around with unless you isolate your network well

Broadcast Routing

Unicast/Multicast/Broadcast

- Unicast – send from one machine to another
- What if want to send to multiple?
  - Multi-unicast – open direct connection to each destination. Inefficient
  - Broadcast – send to *every* destination? Waste bandwidth, but also need to know all possible destinations
  - Flooding? Also too much bandwidth
  - Multi-destination routing

- Multicast Goals
  - Only send to users who want it
  - Each member only receives one copy
  - No loops
  - Path traveled should be optimal
- Spanning tree – tree with source as root and members as leaves
- Reverse-path forwarding

Why would you multicast?

- Live streams? Backups?
- Why not just multi-unicast?
  - More work on sender, many more packets sent
  - Latency between first and last packet sent

Multicast IP

- For IP, just join a class D network
- To both sender and receiver it's like sending/receiving a unicast packet
- all the hard work done by routers
- How do you join a multicast group?
- Router two tasks: group membership management, packet delivery.

Group Management

- IGMP (Internet Group Management Protocol)
  IGMPv3 RFC 3376
  query, report, leave
  querier and noquerier
  router with lowest IP is querier
  no real controls on who can join or send

Multicast Trees

- Steiner tree – NP complete, no one uses
- Heuristics, but none generate entire tree as need centralized and global knowledge
- DVMRP (Distance-Vector Routing Protocol) original protocol, MBONE (tell story)
- Reverse path Forwarding – flood packet out all interfaces except one it came in on. Can have loops; drop dupes.

Then forward on the one that has traveled the shortest path.
Is running the routing table backwards

- Reverse path Broadcast – avoid getting multiple packets
- Protocol Independent Multicast (PIM)
  DVRMP not scalable for multicast groups with sparse members
- MOSPF
- CBT

Other types of Routing

- Mobile – what do you do when machines can come and go?
  have a “home” location. Packets go there. When you get on network, update with actual location. Network gets packets at home location, encapsulates and sends to actual location

- Ad Hoc Routing
  Bunch of machines in an area, routers and devices can come or go more or less randomly.

route discovery

- Peer to Peer File Sharing
  - Centralized server? Napster? Easy to take down.
  - Want Distributed, no central control.
  - Flooding: connect to one other connected node. Floods requests (sort of like broadcast) until it finds who has file, then direct connect to transfer.

  - distributed hash table

- Secret routing

TOR / The onion Router

Packet encrypted multiple times, in layers. Randomly sent to next machine which decrypts that layer, passed on

At end comes out random “exit node” and drops onto regular internet

Data Link Layer

- All about frames.
- Transmitting values to nearby machines: ones/zeros go out to physical layer, same bits arrive back on other machine
- Design issues:

1. Well defined interface
2. Dealing with transmission errors
3. Regulating flow so not overwhelmed by fast senders
4. Propagation delay

- Packets from network layer are encapsulated into frames for transmission
- Frame has header, payload, and trailer
- Other layers also encapsulate in frames, but this is lowest level so we will talk about it here

Link Layer – Issues

- Addressing – specify destination
- Framing – split data into frames
- Error control and reliability
- Flow Control – stop from sending too fast
- Medium Access Control – method to decide which host gets to transmit (handle collisions)

Address

- Global or local? Only few extra bits of extra overhead so often global these days (MAC address?) IEEE 802 is 48-bits. Is that enough?

Framing

- Break up data stream into frames, checksum each on send and receive

- How do you break up into frames?

1. Character count – send a byte describing how many chars follow, followed by that many chars
   Trouble is, what if count affected by noise. Then the data gets out of sync, no way to resync
2. Flag bytes – special byte indicates start and stop, you can then use to find frame boundaries

What to do if flag byte appears in data you are sending? Use escape chars (sometimes called "byte stuffing"?)

3. Bitstuffing – instead of sending multiples of 8 bits, send arbitrary bit widths, with special bit patterns as flags
4. Physical layer coding – use some of the ones we discussed, where you can 4B/5B or such where you can use the unused values as frame markers

Flow Control

- What if sender tries to send faster than receiver can handle?
- Feedback based: receiver sends back info saying it is ready for more (serial with HW flow control)
- Rate-based flow control. The rate is set in the protocol. Not really used in the link layer

Error Control

- You detect an error, what can you do?
  - Drop it on the floor? ("Best Effort") Maybe hope another layer helps
  - Get an acknowledgement saying was correct?
  - What if something happens and the entire frame lost? Receiver never gets it one way or another. Sender waits forever?
  - Use a timer. If no response send again
  - What happens if you send multiple times and then

eventually both get there? Often have a sequence number to track if there are multiple.

- Very quickly end up re-implementing the net layer at this layer.

Error Detection/Correction

- Are errors a problem? If sending 1000 bit frames, and error rate is .001 per bit, then if even distributed on average each frame have an error. Are errors evenly distributed? What if 1000 in a row then none? (bursty)
- Error-Detection Codes – let you tell if an error happened what to do if error happens? resend. doable if errors infrequent (reliable connection)
- Error-Correcting Codes – let you fix an error

Hamming Distance

- Number of bits that differ
- Can calc by exclusive oring then counting the ones.
- 0101 1101 = 1000 = 1
- If hamming distance of N then takes N single-bit errors to convert between the two
- To detect N errors you need hamming distance of N+1 to ensure than N errors can create another valid code

- To correct N errors you need 2N+1 distance, that way even with N errors it is still closer to changed value than any other

- parity bit. Chosen so code word is always even (or odd) can detect single bit error

- Hamming code for detecting errors

Error Detecting Codes

- One way: arrange bits in rectangle, take parity bits across both rows and columns
- Polynomial codes: CRC (cyclic redundancy check)

CRC check

- Polynomial, 110001 means $x^5 + x^4 + x^0$
- Agree on generator in advance. High and low bits must be 1. Checksum is calculated. Value to check must be longer than generator
- Append checksum on end, and when run through the result is zero. Any remainder means an error
- IEEE 802 uses $x^{32} + x^{26} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$

which can detect any burst error less than 32 and all odd number bits

- might seem hard, but easy to make in hardware with a shift register and some xor gates.

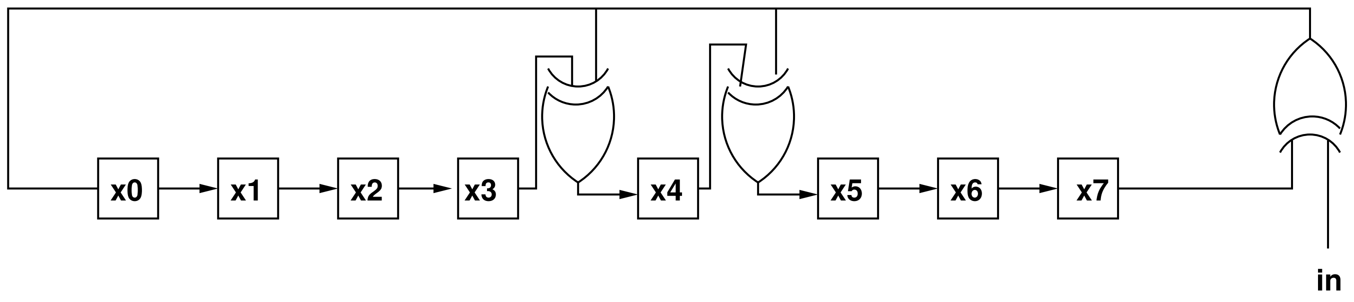
- CRC can find single bit errors, double bit errors, bursts of errors less than length of polynomial.

- Explaining how it works is “mathematically complex” says open source approach book

1-wire CRC check

- Usually used in hardware, harder to implement in software
- Can detect all double-bit errors, any double bit errors, any cluster within an 8-bit window
- if CRCs with itself gets 0 at the end, how hardware detects correct address.
- $X^8 + X^5 + X^4 + X^1$

- Fill with zero, shift values in.



  -