ECE 435 – Network Engineering Lecture 12

Vince Weaver

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

vincent.weaver@maine.edu

12:30pm, Barrows 125

22 February 2024

Announcements

- Midterm on Tuesday March 5th
  More details when closer
- Project details were posted to website
- Updated the TCP congestion notes from last time

HW#4 Review – E-Mail headers

- First warning sign – says its from a bank, but the return address is from a Florida dental school
  Also not a bank of mine
- encrypted and verified from UFL, but sent from videotron.ca cablemodem
- Virus scanned and SPAM scanned, just sort of barely passed
- pop from deater.net via fetchmail (this isn't suspicious, it's the sender not receiver you have to look at)

- LMTP – local mail transport. LHLO. No mail queue, says right away whether deliver mail is possible.

HW#4 Review – E-Mail headers / PDF attachment

- pdf attached probably had some sort of exploit or phishing document. Didn't open.
- Can a PDF compromise your system? Modern browsers will open in-browser
- Note, the attachment being listed as "Application" does not mean it's an executable
  - Just the first part of the mime type
  - If it was an executable, is that an issue?

- Mention Phishing e-mails, Ransomware

HW#4 Review – E-Mail jpeg attachment

- was looking for MIME as what's going on
- Also was looking for base64 as the encoding

HW#4 Review – Domains

- maine.edu created
  - 2 December 1988
  - Fairly typical. Early schools 1985 or so
- Registrar is EDUCAUSE
  What is a Registrar?
  Note: registrar, not registrant

HW#4 Review – DNS

- A 130.111.218.23
- AAAA 2607:f8b0:4006:802::2004
- NS nameo.unet.maine.edu / namep
- MX ALT4.ASPMX.L.GOOGLE.COM
  - Also for MX have a priority value
  - Why google? Running mailserver difficult, at some point all universities let google/microsoft take over. For various reasons you gave, but also likely hoping to lock you into gmail.

- DNS security

The Network Layer

- Also “the internet protocol layer”
- Get packets from source to destination
- May require multiple hops
- Transport Layer runs mostly on the endpoint machine, but Network Layer happens along the routers along way
- Critical, and much more complicated than Link Layer
- Connectivity, Scalability, and Resource Sharing problems

Network Layer Design Issues

- Should be independent of router tech, should hide topology and num, type of routers

- Need to send packets between any two machines, globally:

1. How to identify a host globally (addressing)
2. How to connect different networks together
3. How to find a path between two hosts

Internetworking

- Connecting various types of networks (ethernet, 802.11, etc)
- A group of LANs connected together is an inter-network, or "Internet"

Connection vs Connectionless

Connectionless / Packet-Switched

- Packets sometimes called Datagrams
- Packets injected into network with no prior setup
- Router responsible for picking how it gets there, routing algorithm
- Router makes “best-effort”. Tries to get things there, but if packet gets lost, goes to wrong place, or arrives out of order it doesn’t have to do anything about it.
- Example: Internet
- Send/Receive packet primitives.

- Packet ordering/flow control by higher level
- Each packet carry full destination address, as may travel independently of predecessors

Connection-Oriented

- Virtual circuit created
- Avoid creating a new route for every packet
- A route from source to destination created in all routers along the way
- Each packet carries an ID saying what route it belongs to
- Example: Old POTS telephone land-line network
  older cell phones?

Connectionless vs Connection Tradeoffs

	Connectionless	Connection
Setup	none	required
Addressing	full source + dest	short virt circuit num
State	no router state	each virt circuit has state
Routing	each packet independent	routing done at startup
Router Failure	can route around	all virt circuits terminated
QoS	difficult	easy if allocated in advance
Congestion	difficult	easy if allocated in advance

Which won?

Routing

- How to you determine what path to take in a network?
- Routing protocols: lead to routing tables
- Routing table is destination paired with next hop
- goals
  - minimize routing table space (take up room, also pass around)
  - minimize control messages
  - robustness (don't want to misroute)
- choices:

- centralized vs distributed
- source-based v hop-by-hop. Source you specify entire path at beginning, hop decides each hop along away
- stochastic vs deterministic – deterministic each hop has one route, stochastic multiple routes, picks randomly
- single vs multiple path – one path or if alternate available
- state-dependent vs state-independent – whether you balance based on load. can be better, but can also lead to problems if choose poorly, also extra overhead

Routing and Forwarding

- Routing: which routes to use, find shortest path
- Forwarding: looking up which outgoing line to use
- Characteristics: simplicity/efficiency , robustness, stability, fairness, optimality
- Simplicity: packets stored on routers, efficient resource sharing
  maintain good performance (low delay and packet loss)

- Robustness: cope with changes w/o requiring all jobs stopped and rebooted
- Stability: routing eventually converges on an equilibrium
- Fairness and optimality often conflicting
- Fairness example?
- Unicast routing: point to point
- Multicast routing: one to many or many to many

Routing Algorithm Types

- Nonadaptive: not based on measurement, but computed in advance. Static routing. sysadmin sets them. Do not adapt well if routers fail.

- Adaptive: change routing decisions to reflect changes in topology and traffic
  centralized – require global information
  quasi-centralized (?)
  distributed – ?
  hop-by-hop (internet. source routing?)

Optimal Route?

- What do we optimize? Latency? Throughput? Number of hops?
- Something like ssh might want lowest latency
- Multimedia might want high bandwidth and low jitter
- Often a "cost" is defined based on the desired characteristics, and then this is optimized for

Optimality Principle

- If router J is on optimal path from I to K, then optimal path J to K is on same route
- Set of all optimal routes from all sources to a destination form a tree rooted at destination, called a "sink tree". Not necessarily unique
- Tree and not a loop, so packets delivered in finite number of hops
- Though routers can come and go so things can go wrong

(static) Shortest Path Routing

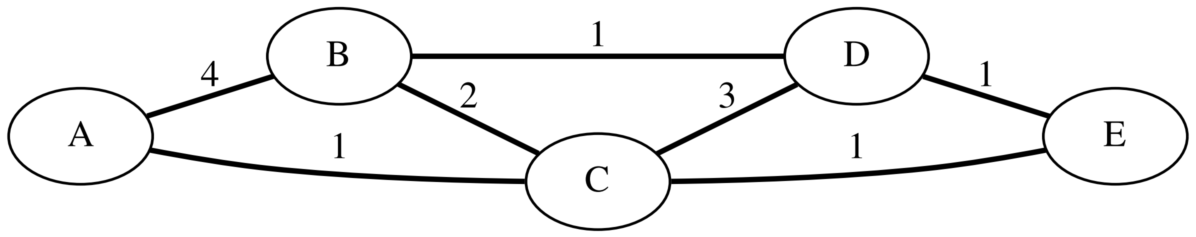
- Number of hops?
- Length (in meters?)
- Transmission delay?

(static) Link State Routing

- Requires global information, routers broadcast the info so all have consistent view
- Dijkstra Algorithm
  Form least spanning tree
  Find lowest cost iteratively
- Iterative algorithm, takes N-1 iterations

Dijkstra Example

Example based on one from Lin/Hwang/Baker



T=set of known machines, C(X)=cost of X, p(X)=previous hop

Iteration	T	C(B),p(B)	C(C),p(C)	C(D),p(D)	C(E),p(E)
0	A	4,A	1,A	∞	∞
1	AC	3,C		4,C	2,C
2	ACE	3,C		3,E	
3	ACEB			3,E	
4	ACEBD				

Dijkstra Example Explanation

- Iterative
- Start not knowing anything but direct connections.
- Pick shortest cost and add to set
- Update all the link costs.
- Repeat until all nodes added

Dijkstra Example, finding routing Table for Node A

- First iteration calculate values for just A
  - Cost A to B is 4
  - Cost A to C is 1
  - Cost to D and E unknown
- Next iteration add in C (as it's next shortest latency)
  - Cost A to B is 1 (A to C) plus 2 (C to B) = 3. This is less than previous so update
  - We don't recalc A to C as both in set

  - A to D is 1+3=4
  - A to E is 1+1=2
- Next iteration add in E (as it's next shortest latency)
  - Cost A to B doesn't change (obviously, but why?)
  - Cost A to D would be AtoE plus D, so 3, so update
- Next iteration add in B (why?)
  - Cost A to D would be AtoB plus D, so 5, which would be longer
- Next iteration add in D, last one, so done

Now Construct Routing Table

Final routing table for A.

Path	Cost	Next Hop
A-B	3	C
A-C	1	C
A-D	3	C
A-E	2	C

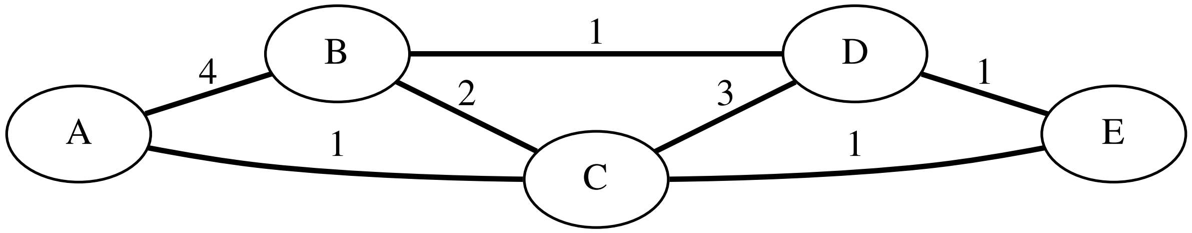
(static) Flooding

- Every packet sent out on every outgoing line, with a counter (set to the distance) so after so many hops discarded
- Selective flooding, only floods out the connections going in vaguely the right direction
- Very robust (can handle if routers dropping out constantly)
- Flooding always chooses shortest path, as it finds all

possible paths in parallel

(dynamic) Distance Vector Routing

- Used by ARPANET until 1979
- Asynchronous, distributed, uses local info
- Each router maintains a table (vector) giving best known distance to each destination and line to use to get there
- First line shows out starting info they all know.
  Each iteration shows as the info from neighbors is passed on and the routing tables are updated.


A
B
C
D
E
4
1
2
3
1
1
1
1

DVR example

1. Start with what you know
2. Send routing table to neighbor
3. Update if find shorter route. This is all happening simultaneously
4. Should converge on Dijkstra.

DVR example

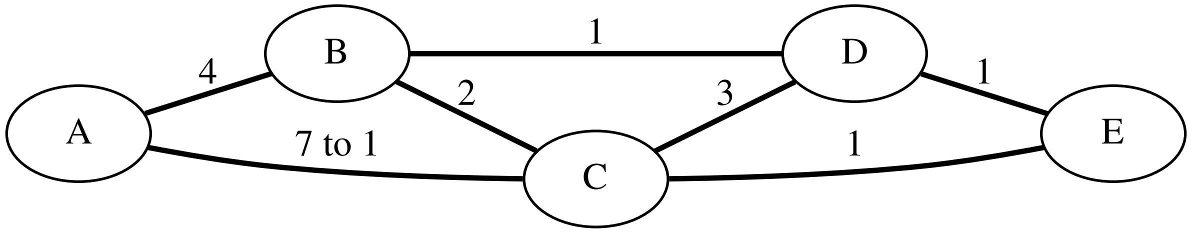
D=destination, C=cost, N=next hop

	A			B			C			D			E		
	D	C	N	D	C	N	D	C	N	D	C	N	D	C	N
1	B	4	B	A	4	A	A	1	A	B	1	B	C	1	C
	C	1	C	C	2	C	B	2	B	C	3	C	D	1	D
				D	1	D	D	3	D	E	1	E			
							E	1	E						
2	B	3	C	A	3	C	A	1	A	A	4	C	A	2	C
	C	1	C	C	2	C	B	2	B	B	1	B	B	2	D
	D	4	C	D	1	D	D	2	E	C	2	E	C	1	C
	E	2	C	E	2	D	E	1	E	E	1	E	D	1	D
3	B	3	C							A	3	E			
	C	1	C							B	1	B			
	D	3	C							C	2	E			
	E	2	C							E	1	E			

Problems with DVR

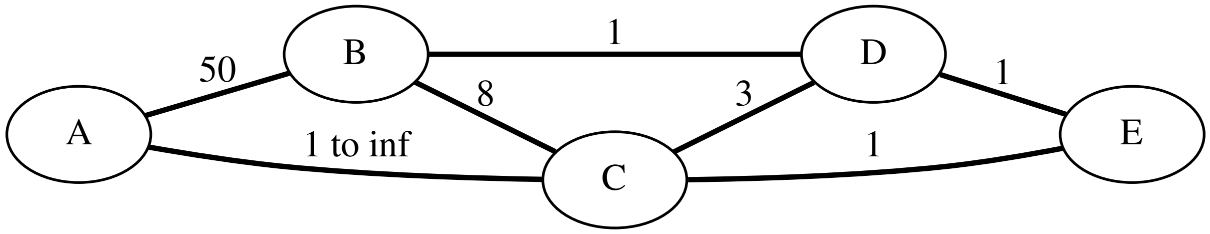
- Looping problems: packets can get stuck in loops.
- Good news travels fast, bad news travels slowly.

Good News Travels Fast



- Converges in two steps.

Bad News Travels Slowly



- A to C line goes down.
- Have bad timing. (Note: really need better description here)
- Everyone swaps routing info
- Initial situation
  - C thinks best path to A is E-D-B-A (53)

  - E still thinks best path to A is E-C-A (2)
- Swap tables
  - C hears E can get to A in 2, so it updates its table to say A is 3 away (C-E-C-A)
  - E learns C-A down so updates to E-D-B-A 52
- Swap tables
  - C hears E-A is 52, so it updates its table to 53
  - E hears C-A is 3, so it updates its table that E-C-A is 4
- Swap tables
  - C hears E-A is 4, so it updates its table so A is 5 away

(note counting up by one each time)

- ◦ E hears C-A is 52, so it updates its table 53 so updates to 52

- Note this will carry on for a while, “counting to infinity”
- As long as there is a valid route it will eventually find it but it might take a while

Solutions to Counting to Infinity

- Split horizon – a router should not tell neighbor back the least cost it just got from that neighbor
- Poison Reverse – instead of not telling back, should say the cost back to itself is infinity
- These only work for two hop loops. Other options to send additional "next hop" data, or have a "hold down timer" that lets things settle before updating info

(dynamic) Link State Routing

- Problems with DVR: did not take delay into account, took too long to converge
- Instead, send entire routing table to everyone. Each node then rebuilds own.
- Each router must:

1. Discover neighbors and learn network address
2. Measure delay or cost of each neighbor
3. Construct a packet telling all it learned
4. Send a packet to all other routers

5. Compute the shortest path to all other routers

- Learning about neighbors: sends HELLO packet at boot out all links
- Measure line cost: Send special ECHO packet and measure return. Take into account load?
- Building link-state packets
- Distributing
- Computing new routes

Hierarchical Routing

- At some point not possible for every router to know about every other
- Split into regions
- Example?

Internetworking

- Robert Metcalfe (one of inventors of Ethernet) Metcalfe's Law: networks value is the square of the nodes
- Joining networks together of different types
- Might have to convert packets at boundaries
- Or tunnel
- What if packets too big for size limit?
  - Fragmentation (difficult)
  - Path MTU (Maximum Transmit Unit) discovery