Routing Fundamentals

Sylvia Ratnasamy
Spring 2022

Acknowledgement: Based on slides from Murphy McCauley
Today

- Give you a chance to ask questions
- Make sure you understood the key concepts
- Stretch your understanding to new designs
Concepts

- Routers as intermediate nodes
- Valid routing state
- Least-cost routing
- Route convergence
- Distance vector
  - Split horizon
  - Counting to infinity

Note: we use path and route interchangeably
Recall Murphy: “Why is a router?”

Observe: interconnecting routers enables a variety of graph topologies
Recall Murphy: “Why is a router?”

**Question:** what would you call a “good” topology?
Recall: Two points to note

- Hosts generally do not participate in routing
  - In common case, hosts:
    - Have a single link to a single router
    - Have a default route that sends everything to that router
      - (unless they’re the destination!)
  - They’re not interesting, so we often ignore them except as destinations

- Routers might be legal destinations (in addition to hosts)
  - Depends on the network design
  - Internet Protocol routers can be!
  - But how often have you wanted to talk to a specific router?
  - Host-to-host communication much more common; we’ll often ignore routers as destinations
  - But do think of all routers as potential sources (packets may arrive in unexpected ways!)
Recall: Routing State Validity

- A necessary and sufficient condition for validity

- Global routing state is valid *if and only if*:
  - For each destination…
    - There are no dead ends
    - There are no loops

**Question:** difference between valid vs. least-cost paths?
Valid vs. least-cost paths to A
Can we have more than one least-cost path from E to A?

Does it matter which one E picks?

What if we wanted to allow E to use both?
Can we have more than one least-cost path from E to A?

Can we use the red for E and the green for H?
Recall: Checking routing state validity

- Very easy to check validity of routing state for a particular destination...
- Dead ends are obvious
  - A node with no outgoing arrow can’t reach destination
- Loops are obvious
  -Disconnected from destination (and entire rest of graph!)
- .. now just repeat for each destination!

How would you build a practical system to check the validity of routing state?
Checking routing state validity (in practice)

Validating datacenters at scale

Authors: Karthick Jayaraman, Nikolaj Bjørner, Jitu Padhye, Amar Agrawal, Ashish Bhargava, Paul-Andre C Bissonnette, Shane Foster, Andrew Helwer, Mark Kasten, Ivan Lee, +7

VeriFlow: Verifying Network-Wide Invariants in Real Time
Authors: Ahmed Khurshid, Xuan Zou, Wenxuan Zhou, Matthew Caesar, and P. Brighten Godfrey, U. Champaign

Move fast, don't break the network
Predict and test the impact of changes before deploying to production

How would you build a practical system to do this validation?
Checking routing state validity (in practice)

- Collect all forwarding tables at a central server
  - ..... 
- Network validation and verification is currently a hot topic!

How would you build a practical system to do such validation?
Recall: Least-Cost Routing

- Last time, we said we wanted “good” routes

- Goal #1: Routes that work!
  - State must not have any loops. Must not have any dead ends. Both of these.

- Goal #2: Routes that are in some way “good”
  - Commonly this is done by minimizing some “bad” quantity which we might call a cost
  - Hence least-cost routing!

**Question:** limitations of least-cost routing?
See a potential problem?
What if A doesn’t want her traffic routed via Russia?
Examining Cost metrics

**Question:** what are some cost metrics you can think of?
What happens if cost = 1/(link BW) & least-cost paths?

- R1: BW = 4 Gbps (cost = 0.25)
- R2: BW = 10 Gbps (cost = 0.1)
- R3: BW = 5 Gbps (cost = 0.2)
- R4: BW = 5 Gbps (cost = 0.2)

A → R1 → R2 → R3 → R4 → B
What happens if link cost = 1/(available link BW)?
Recall: Murphy said

Does everyone understand this?

Distance-Vector

<table>
<thead>
<tr>
<th>Dst</th>
<th>Nxt, Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>R1,2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dst</th>
<th>Nxt, Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Direct,1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dst</th>
<th>Nxt, Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>R2,3</td>
</tr>
</tbody>
</table>

We’ve converged! 😊 Does everyone understand this?
Reasons routes may change?
Distance-Vector: one-slide recap

- Periodically, router R1 tells each neighbor about its least-cost to each destination D
  - Exception: if R2 is R1’s next-hop to D, then don’t tell R2 about D

- When R1 receives an update from a neighbor R2 advertising a cost of X to dest. A
  - If R1 has no entry for A: add entry for dst=A with cost=minimum(X+1, INF), next-hop=R2, TTL=max
  - Else, if R2 is my current next-hop to A: cost=minimum(X+1, INF), TTL=max
  - Else, if (X+1) < current cost to A → replace entry for A: cost=X+1, next-hop=R2, TTL=max

- Update TTLs as time goes by; delete expired forwarding entries
Distance-Vector: one-slide recap

- Periodically, router R1 tells each neighbor about its least-cost to each destination D
  - Exception: if R2 is R1’s next-hop to D, then don’t tell R2 about D

- When R1 receives an update from a neighbor R2 advertising a cost of X to dest. A
  - If R1 has no entry for A: add entry for dst=A with cost=minimum(X+1, INF), next-hop=R2, TTL=max
  - Else, if R2 is my current next-hop to A: cost=minimum(X+1, INF), TTL=max
  - Else, if (X+1) < current cost to A → replace entry for A: cost=X+1, next-hop=R2, TTL=max

- Update TTLs as time goes by; delete expired forwarding entries

All routers run the above independently

This is a complete and correct routing solution!
Though not the most efficient: techniques for faster convergence next lectures
Sources of convergence delay?

- Timers
- TTLs
- Time to detect failure
- Time to recompute paths
- Packet loss
- Value of “infinity”
- ...
Can counting-to-infinity happen if we use split horizon?

Consider destination D

Link fails

Does B advertise its route to D to neighbor C?

What happens next?
Can counting-to-infinity happen if we use split horizon?
Can counting-to-infinity happen if we use split horizon?

C advertises as yet unexpired route
Can counting-to-infinity happen if we use split horizon?

B likes this route!
Can counting-to-infinity happen if we use split horizon?
Can counting-to-infinity happen if we use split horizon?

B advertises this route to A

### Routing Table

<table>
<thead>
<tr>
<th>Dst</th>
<th>Nxt, Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>C, 3</td>
</tr>
</tbody>
</table>

### Next Hop Table

<table>
<thead>
<tr>
<th>Dst</th>
<th>Nxt, Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>C, 3</td>
</tr>
</tbody>
</table>
Can counting-to-infinity happen if we use split horizon?

A likes this route!
Can counting-to-infinity happen if we use split horizon?

A advertises this route to C.
Can counting-to-infinity happen if we use split horizon?

Will advertise to B and we’ll continue in circles…
Questions on Distance-Vector?
Recall: Murphy said...

- There are an endless number of possible solutions to routing

- I’m going to constrain our initial discussion to how “archetypal Internet” works

Let’s try and come up with some of these other solutions …
Things to ponder...

- Fundamentally, what information do you need to compute paths?
- What (other) information could a router advertise?
- Does computing paths have to be a distributed process?
#1: Link-State Routing

- Every router discovers the entire network graph (nodes and edges)
  - By having each router flood their local information (list of neighbors) to all other routers
#1: Link-State Routing

- Every router discovers the entire network graph (nodes and edges)
  - By having each router flood their local information (list of neighbors) to all other routers
#1: Link-State Routing

Topic for next lecture!
#2: What if we had (only) F compute paths instead?

Q: are we still limited to least-cost paths w/ SDN?
#3: Other solutions to go beyond least-cost paths?

Packet header carries (R1, R2, R3)
Recall: Two Things Routers Do

**Forwarding**
- Looks up packet's destination in table and sends packet to given neighbor
- *Inherently local*: depends only on arriving packet and local table
- Primary responsibility of router's *data plane*
- Time scale: per packet arrival (nanoseconds?)

**Routing**
- Communicates with other routers to determine how to populate tables for forwarding
- *Inherently global*: must know about all destinations, not just local ones
- Primary responsibility of router's *control plane*
- Time scale: per network event (e.g. per failure)

**Q**: Does SDN still follow this?
Questions?