CSCI-1680
Network Layer: Inter-domain Routing

Nick DeMarinis

Based partly on lecture notes by Rodrigo Fonseca, Rob Sherwood, David Mazières, Phil Levis, John Jannotti

Warmup

Suppose router R has the following table:

Dest.	Cost	Next Hop
A	3	S
B	4	T
C	~~5~~ 6	S
D	~~6~~ 5	~~U~~ S

NO CHANGE
TIE => SAME COST
COST INCREASED!
BETTER ROUTE!

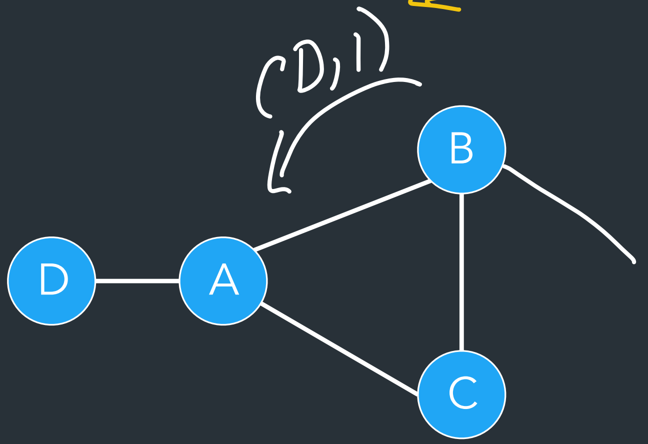
What happens when it gets this update from router S?

Dest.	Cost
A	2
B	3
C	5
D	4
E	2

← NEW!

What would happen if a route we previously saw from S was missing?
=> Link may have done down (timeout and remove after some interval)

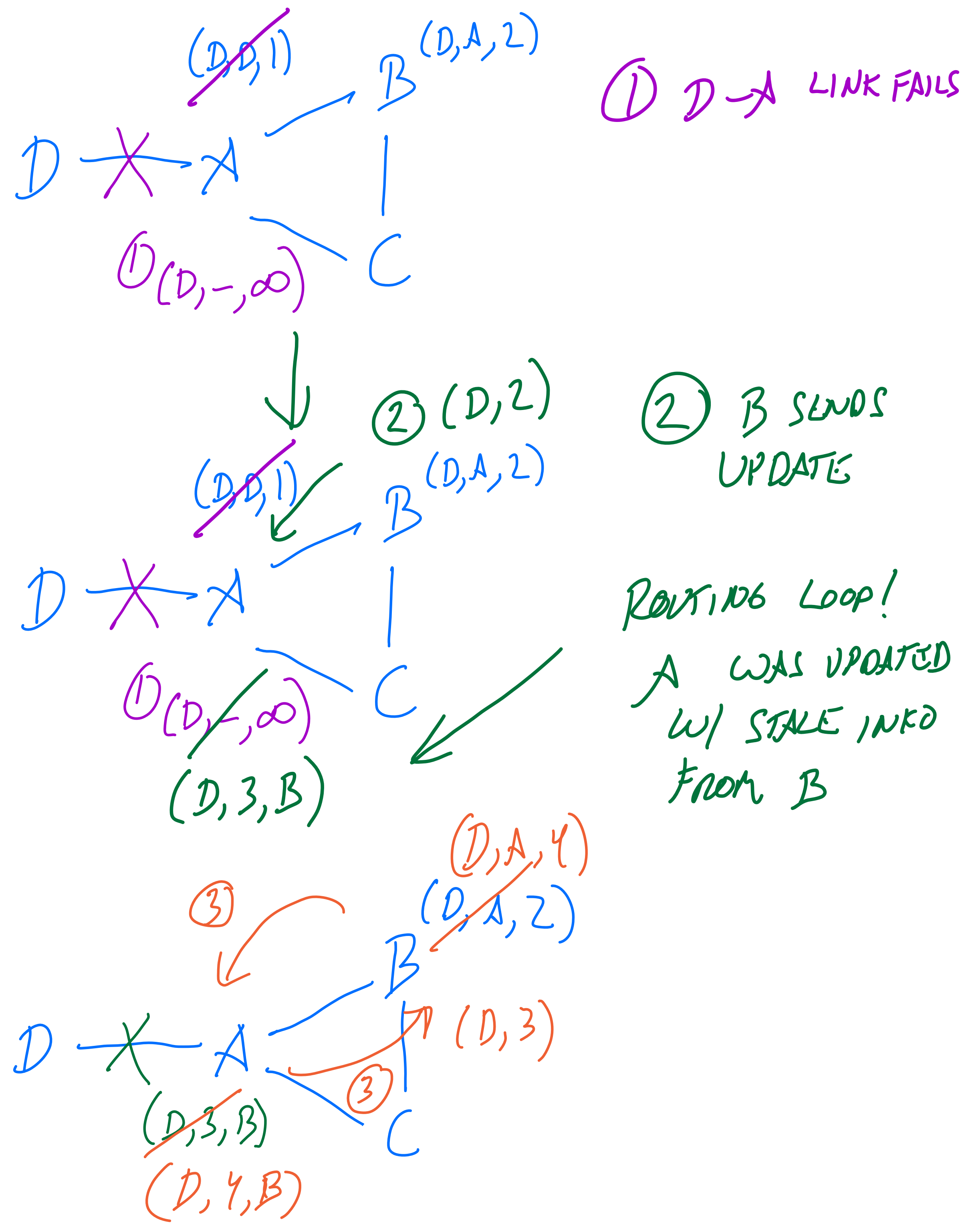
What happens when the D-A link fails?



(DEST, COST)

ONLY INFO CONTAINED IN A DISTANCE-VECTOR UPDATE!

Updates occur in a loop with increasing cost until cost reaches infinity (16)!
=> Count to infinity => long time to converge when links fail



<u>Count to infinity</u>: cost keeps increasing until it reaches infinity

=> "Bad news travels slowly"

=> In RIP: "infinity" == 16

Why does this happen? DV only based on info from neighbors, and not enough info to resolve loops, etc.

Can we avoid loops?

- Does IP TTL help? Nope.
- Simple approach: consider a small cost *n* (e.g., 16) to be infinity

Fundamental problem: distance vector only based on local information!
=> Not enough info to resolve loops, race conditions, count-to-infinity,
but there are some tricks…

RFC1058 (1988): The original RIP standard*

RFC 1058   Routing Information Protocol   June 1988

supply the information that is needed to do routing.

1.1. Limitations of the protocol

This protocol does not solve every possible routing problem. As mentioned above, it is primary intended for use as an IGP, in reasonably homogeneous networks of moderate size. In addition, the following specific limitations should be mentioned:

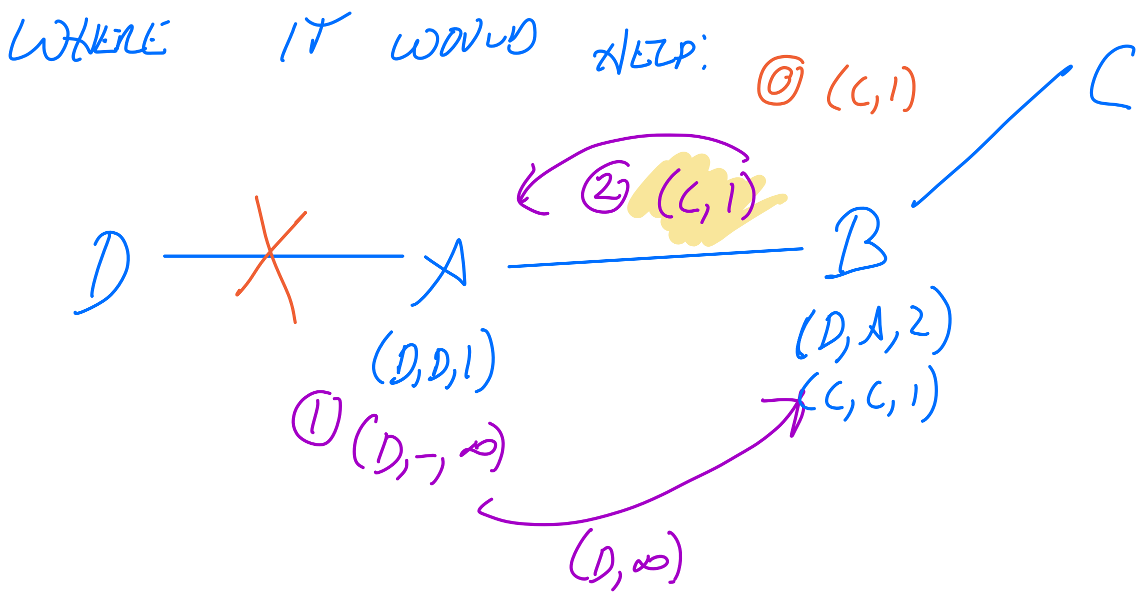
*: *Obsoleted by RFC2453 (don't use RFC 1058 for the project, Use RFC 2453 instead)*

One strategy: Split Horizon

- When sending updates to node A, don’t include routes you learned from A
- Prevents B and C from sending cost 2 to A

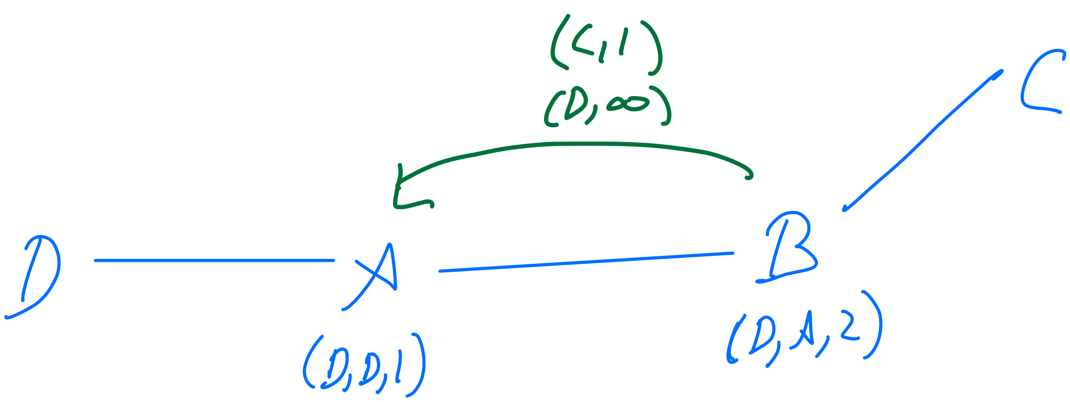
A solution (at least for RIP): **Split Horizon**

Definition: If A uses N as next hop for D, do not report to N about D
=> Prevents "linaer" routing loops, but not others



What happened?
1) D-A link fails
2) B's updates to A don't include any info about D => no change to A's table (wrt D)
3) A updates B => (D, inf)

Commonly used with: **Poison reverse**: rather than not including routes learned from A, explicitly send cost of infinity
=> Idea: may help converge in some cases (but hard to see it in practice)



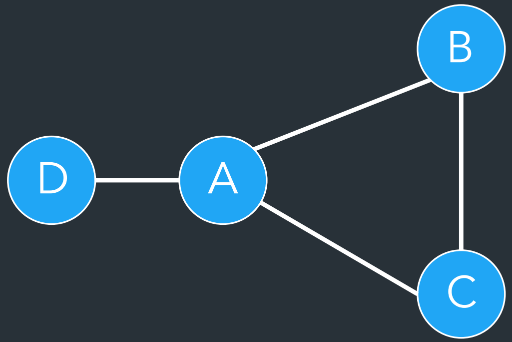
Split Horizon + Poison reverse

- Rather than not advertising routes learned from A, explicitly include cost of ∞.
- Faster to break out of loops, but increases advertisement sizes

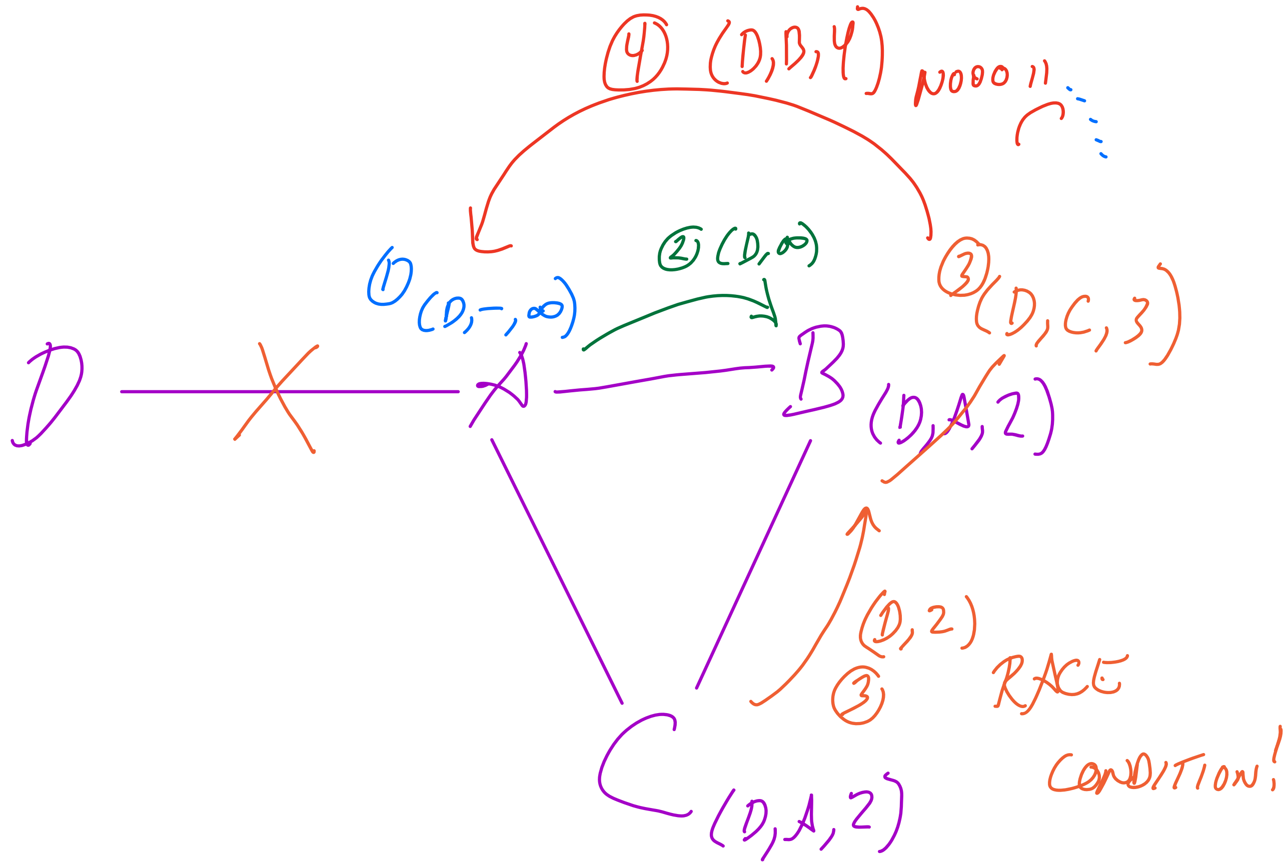
⇒ Does it help? Not completely.

=> A common convention, might reduce time to converge, but overall hard to see effect vs. split horizon

Even with split horizon + poison reverse, can still create loops with >2 nodes!



But even this can't prevent all loops!!!

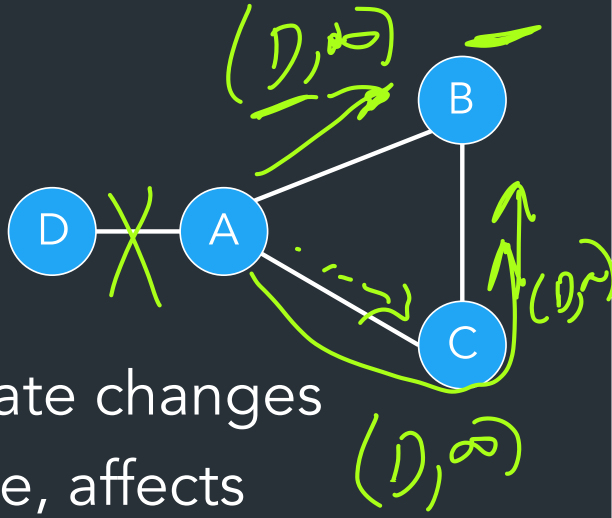



```
What happens?
 1) D-A link fails
 2) A updates B => (D, inf)
 3) Before C gets the same update, it sends (D, 2) to B
     => RACE CONDITION!!!  C might send old update to B before C
gets update from A
 4) B updates A, overwrites A's table
 5) ... count to infinity ...
```

So what can we do?

- Can't send any extra information.

Even with split horizon + poison reverse,
can still create loops with >2 nodes!

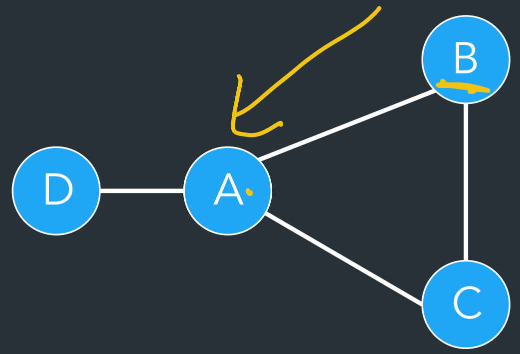


What else can we do?

- Triggered updates: send update as soon as link state changes
- Hold down: delay using new routes for certain time, affects convergence time

HOLD INFO BEFORE "COMMITTING" CHANGES

Practice



B's routing table

Dest.	Cost	Next Hop
A	1	A
C	1	C
D	2	A

Routers A,B,C,D use RIP. When B sends a periodic update to A, what does it send…

- When using standard RIP?
- When using split horizon + poison reverse?

STANDARD	SH+PR
(A, 1)	(A, ∞)
(C, 1)	(C, 1)
(D, 2)	(D, ∞)

From RFC2453, RIP v2 (1998):

3.2 Limitations of the Protocol

This protocol does not solve every possible routing problem. As mentioned above, it is primary intended for use as an IGP in networks of moderate size. In addition, the following specific limitations are be mentioned:

- The protocol is limited to networks whose longest path (the network's diameter) is 15 hops. The designers believe that the basic protocol design is inappropriate for larger networks. Note that this statement of the limit assumes that a cost of 1 is used for each network. This is the way RIP is normally configured. If the system administrator chooses to use larger costs, the upper bound of 15 can easily become a problem.

- The protocol depends upon "counting to infinity" to resolve certain unusual situations. (This will be explained in the next section.) If the system of networks has several hundred networks, and a routing loop was formed involving all of them, the resolution of the loop would require either much time (if the frequency of routing updates were limited) or bandwidth (if updates were sent whenever changes were detected). Such a loop would consume a large

Link State Routing

Link State Routing: The Alternative

Example: OSPF

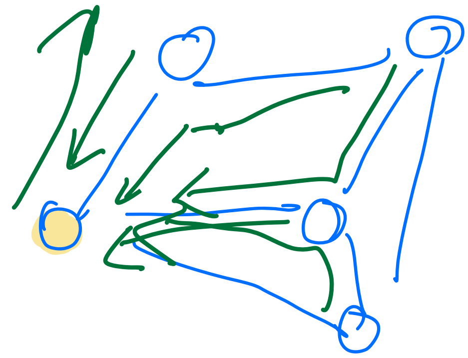
↳ INTRA-ROUTING

Strategy: each router sends information about its neighbors to *all nodes*

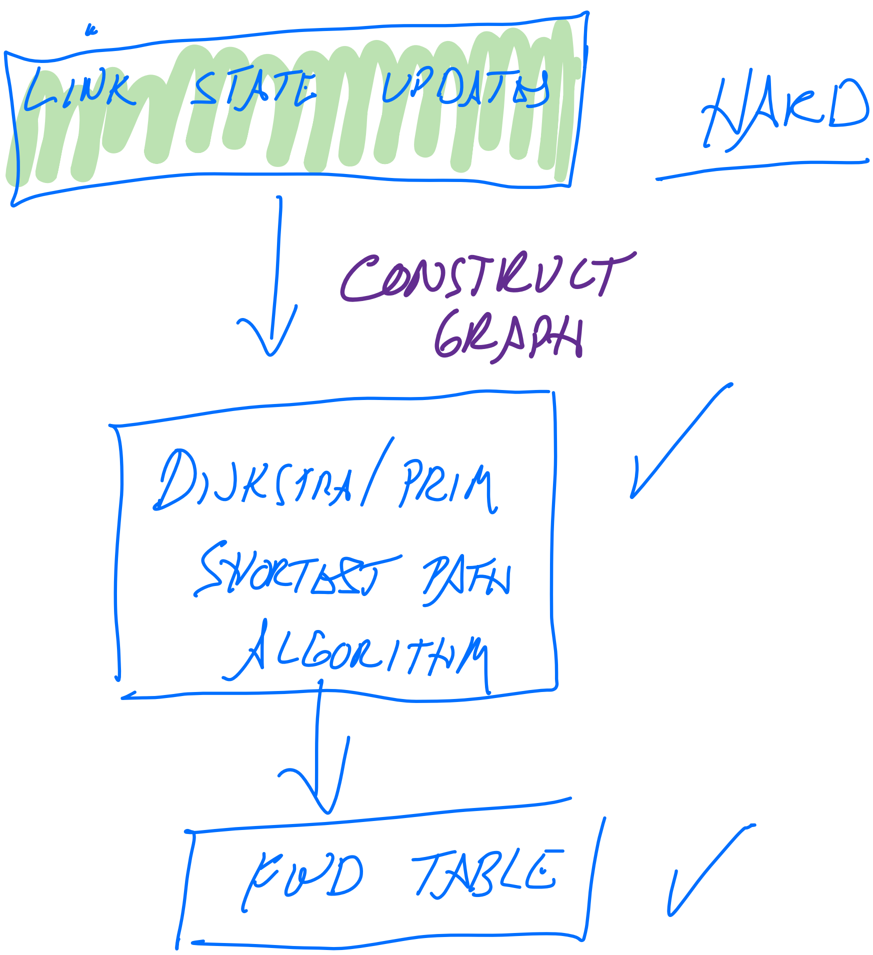
Link state routing: the idea
Strategy: Each router sends information about its neighbors to all nodes

=> Nodes build the full adjacency graph--not just neighbor info
=> Updates have a lot more state info
  => **IN RIP, WE NEVER FORWARD THE UPDATES, THEY ONLY GO TO NEIGHBORS**
   If you do:
   - How do you make sure that all nodes get them?
   - How do you make sure that they don't loop forever?
   - How do you know what information is stale?
   - How do you even name the routers?

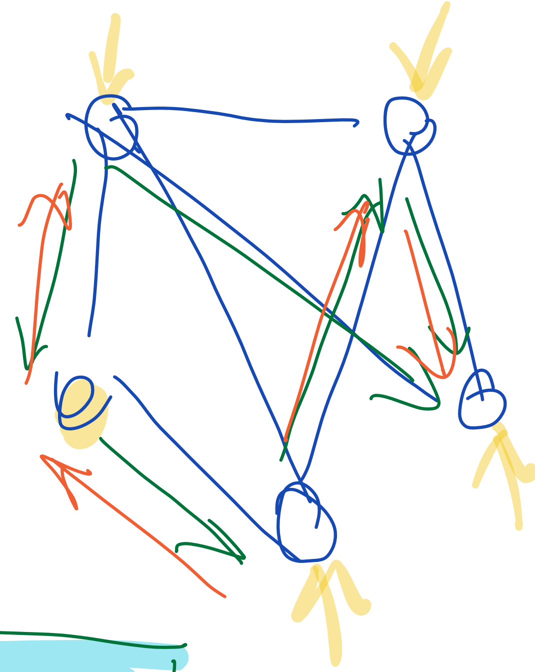
  => This is hard, and involves a lot of state info



A LINK STATE ROUTER

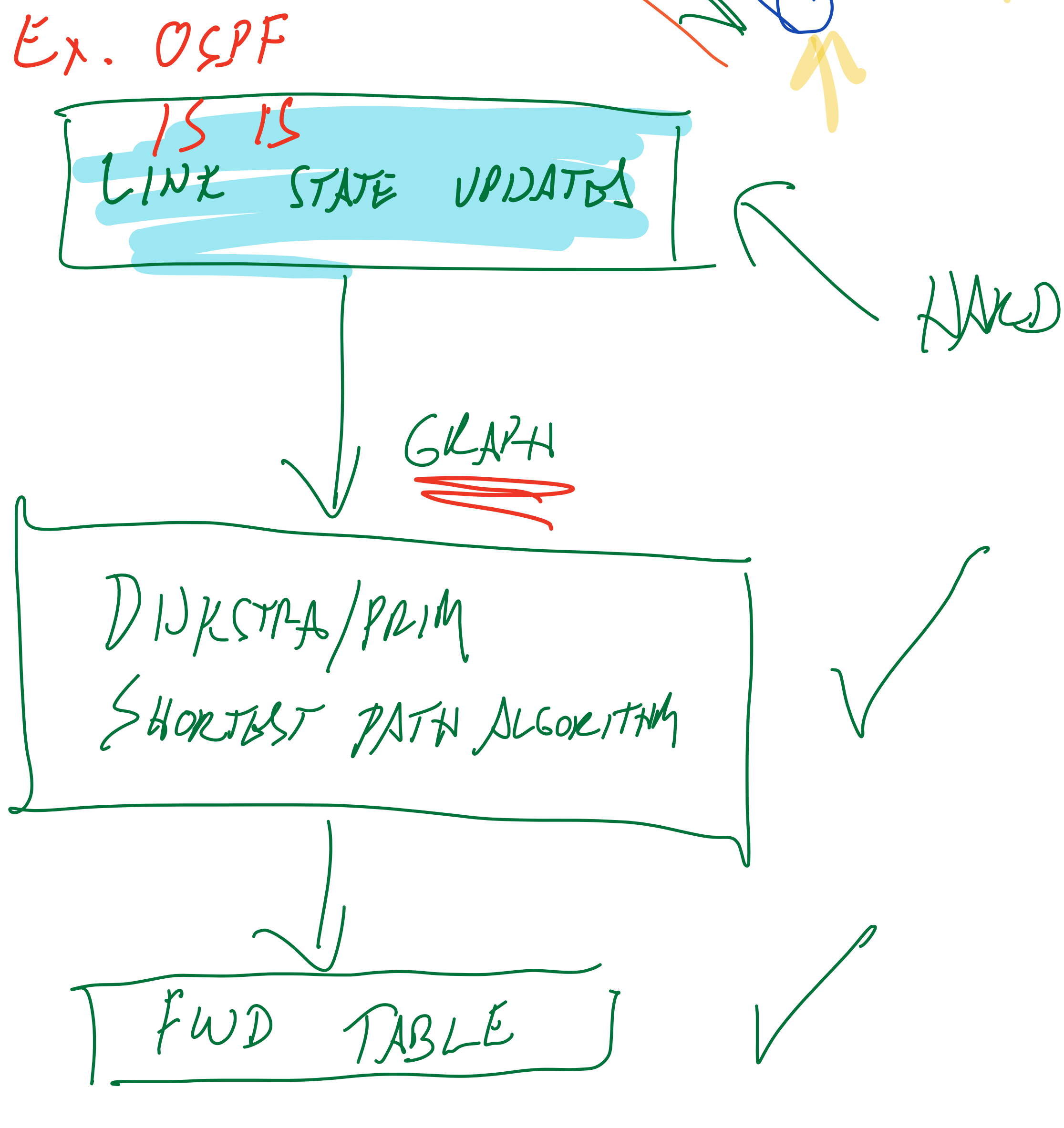


Idea: each router sends information about neighbors to all nodes



=> Nodes can build full adanacency graph--not just neighbor info

=> Link state update



Sending the updates is actually hard:
- How do you know that information is stale? Versioning/timestamps
- How can you can make sure that all nodes get the updates
  - ..... and also don't loop forever
  - How do you even name the routers?

=> LS: Updates are a lot larger, have more state info

=> But better properties for avoiding loops, no count-to-infinity, etc.

Link State Routing: The Alternative

Strategy: each router sends information about its neighbors to *all nodes*

- Nodes build the full graph, not just neighbor info

  => Can define "areas" to scale this in large networks

- Updates have more state info
  - Node IDs, version info (sequence number, TTL), …

    => Can be used to detect loops, stale info

⇒ Focuses on building a consistent view of network state

Tradeoffs: Link State (LS) vs. Distance Vector (DV)

- LS sends more messages vs. DV ⇒ MORE INFO VS. LINK-STATE
- LS requires more computation vs. DV ⇒ MORE COMPUTATION AT EACH NODE
- Convergence time
  - DV: Varies (count-to-infinity)
  - LS: Reacts to updates better
- Robustness
  - DV: Bad updates can affect whole network
  - LS: Bad updates affect a single node's update

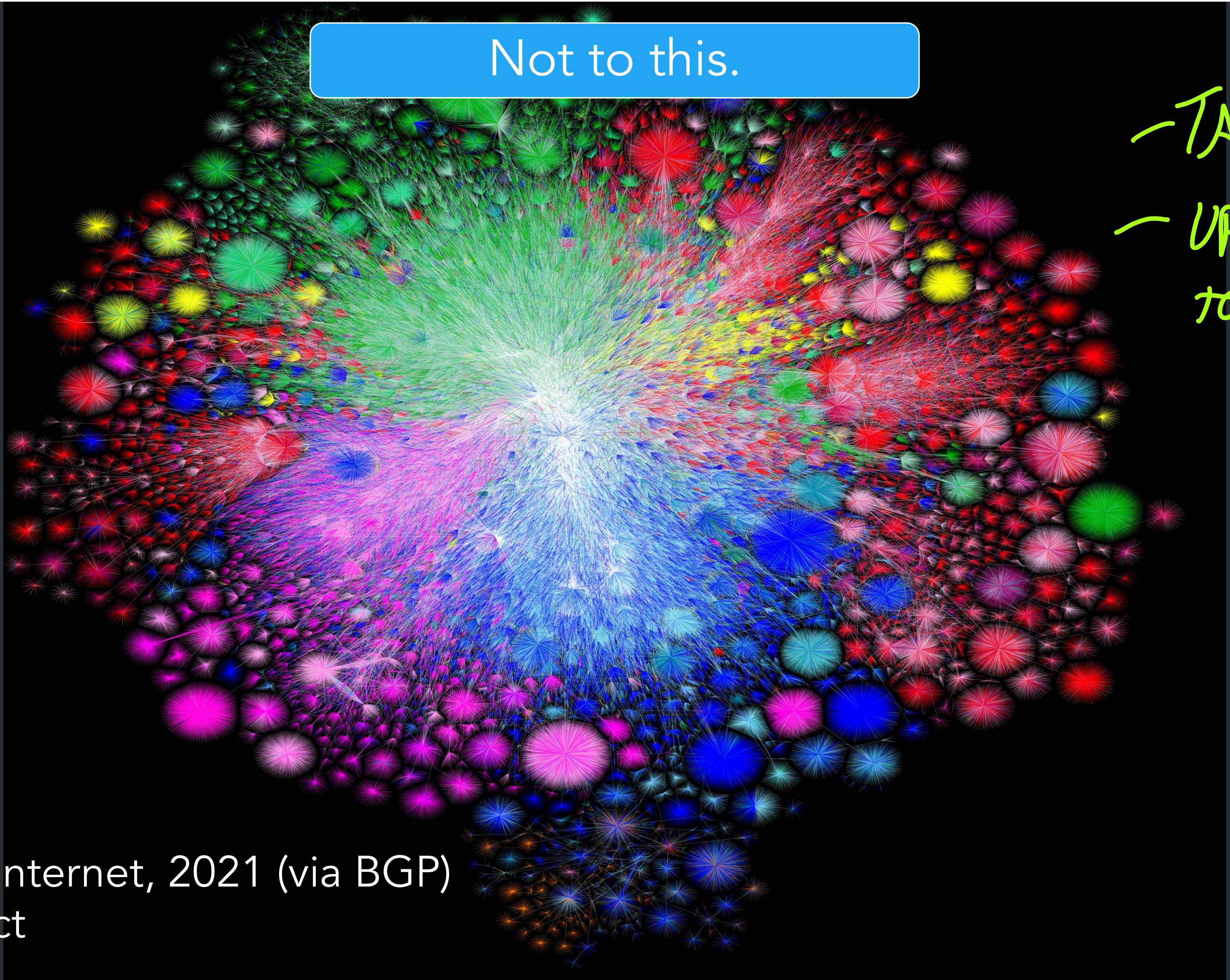
LS: HARDER TO HAVE BAD INFO PROPAGATE

So why not just use OSPF everywhere?

Does it scale?

No.

Not to this.



Map of the Internet, 2021 (via BGP)
OPTE project

- TABLE IS BIG
- UPDATES TO ALL IS HARD

Why not?

⇒ Can't build a full routing graph with the whole Internet

⇒ More a policy problem than a technical problem

- No unified way to represent cost
- No single administrator
- Networks (ASes) have different policies on what "best" routes to choose

INTER-DOMAIN ROUTING

Need a different routing mechanism for exterior routing => BGP

With BGP: we talk about routing to Autonomous Systems (ASes)

= > Generally, large networks advertise some set of IP prefixes to the Internet

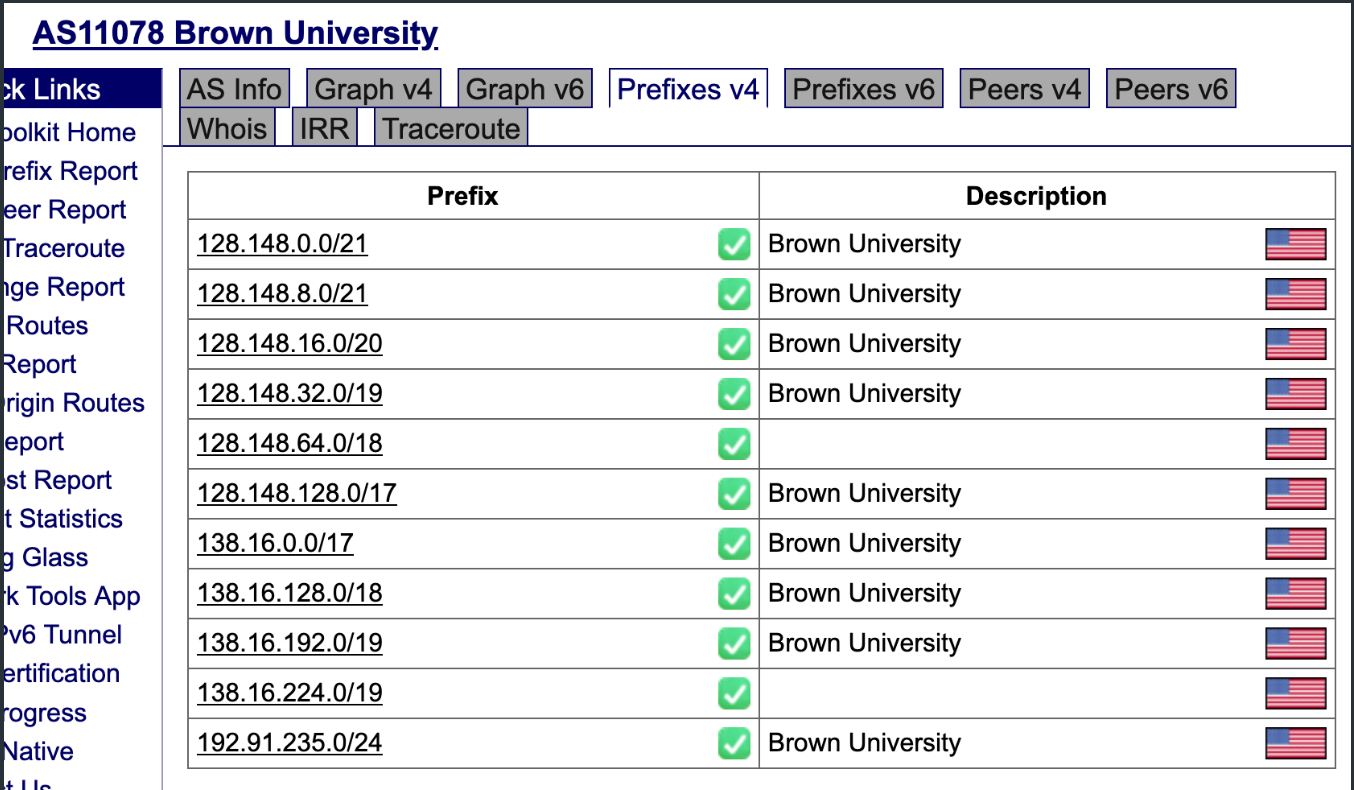
=> Each AS has its own *policy* for how it does routing

- Different goals, interests, political agendas, financial incentives,....

With BGP: we talk about routing to Autonomous Systems (ASes)

= > Generally, large networks advertise some set of IP prefixes to the Internet

=> Each AS has its own *policy* for how it does routing



AS11078 Brown University

ck Links
olkit Home
refix Report
eer Report
Traceroute
nge Report
Routes
Report
rigin Routes
eport
st Report
t Statistics
g Glass
k Tools App
v6 Tunnel
ertification
rogress
Native
t Us

AS Info | Graph v4 | Graph v6 | Prefixes v4 | Prefixes v6 | Peers v4 | Peers v6 | Whois | IRR | Traceroute

Prefix	Description
128.148.0.0/21	Brown University
128.148.8.0/21	Brown University
128.148.16.0/20	Brown University
128.148.32.0/19	Brown University
128.148.64.0/18	
128.148.128.0/17	Brown University
138.16.0.0/17	Brown University
138.16.128.0/18	Brown University
138.16.192.0/19	Brown University
138.16.224.0/19	
192.91.235.0/24	Brown University

BGP: A Path Vector Protocol

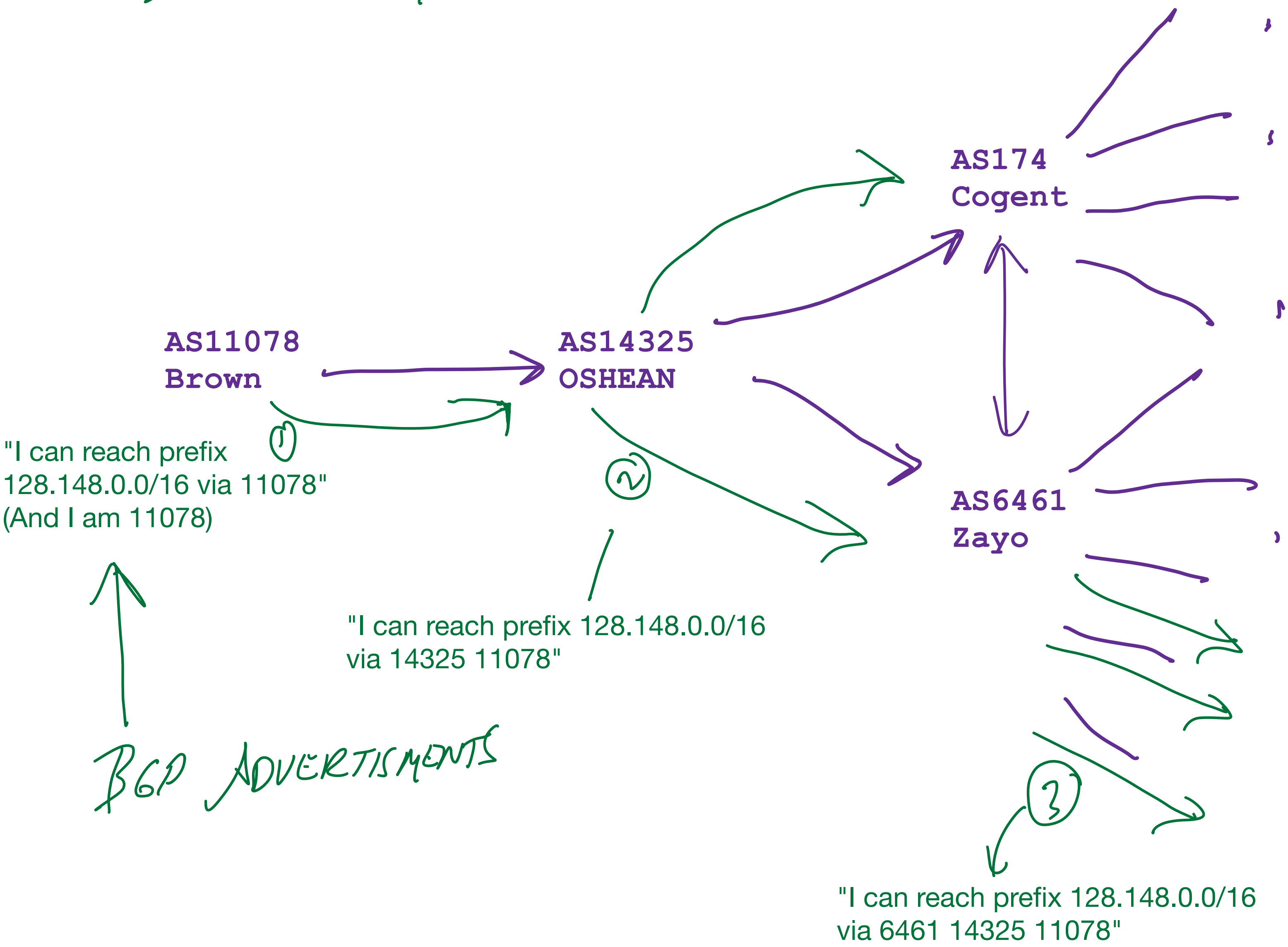
Distance vector + extra information

PREFIX

eg. *"I can reach prefix 128.148.0.0/16 through ASes 44444 3356 14325 11078"*

PATH

SKETCH: BGP



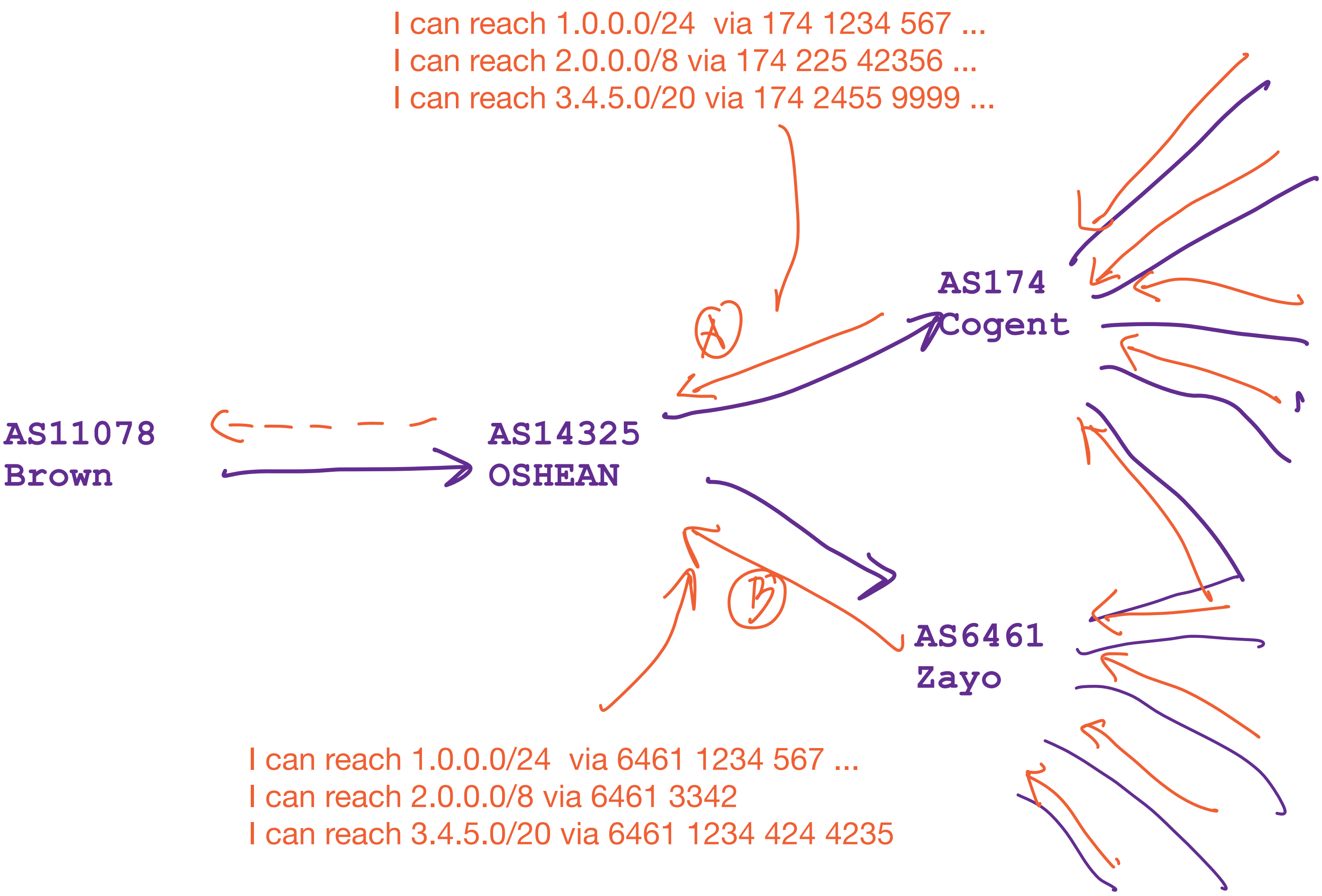
Key ideas:

- Routers send announcements which include the path to reach the AS "originating" the prefix
  - Each AS should add itself to the path

- Policy part: ASes decide which paths to propagate to their neighbors, based on their own policies
  - Examples:
    - ISP will only advertise routes for customers, if it pays them...
    - Can block access by not advertising certain routes..

At the same time, "upstream" providers send announcements to "downstream" networks, telling them about prefixes they know about

=> This is how networks connect to the entire internet



Similarly, ASes needs to decide....

- Which routes they install in their own tables
- Which routes they "propagate" to "downstream" ASes

=> We'll define more about what this means, and what "upstream" and "downstream" mean next lecture!

BGP: A Path Vector Protocol

Distance vector + extra information

eg. "I can reach prefix 128.148.0.0/16 through ASes 44444 3356 14325 11078"

- For each route, router store the complete path (ASs)
- No extra computation, just extra storage (and traffic)
- BGP gets to decide what path to *advertise* to neighbors

Fun fact: loops are easy to avoid…

eg. *"I can reach prefix 128.148.0.0/16 through ASes 44444 3356 14325 11078"*

What would a loop look like?

BGP: A Path Vector Protocol

Distance vector + extra information

eg. *"I can reach prefix 128.148.0.0/16 through ASes 44444 3356 14325 11078"*

- For each route, router store the complete path (ASs)
- No extra computation, just extra storage (and traffic)
- BGP gets to decide what path to *advertise* to neighbors

⇒ BGP routers look at path to decide how to "propagate" route, based on policy
⇒ Can easily avoid loops!

BGP Implications

- No loops!
- Not all ASs know all paths
- Reachability not guaranteed
  - Decentralized combination of policies

POLICY DECISIONS ON HOW TO PROPAGATE

- Scaling
  - 74K ASs
  - 959K+ prefixes
  - ASs with one prefix: 25K
  - Most prefixes by one AS: 10008 (Uninet S.A. de C.V., MX)

Source: cidr-report 18Oct2022

Why study BGP?

BGP is what makes the Internet run.

Lots of problems…

Explainer

Facebook outage: what went wrong and why did it take so long to fix after social platform went down?

RYAN SINGEL SECURITY FEB 25, 2008 10:37 AM

Pakistan's Accidental YouTube Re-Routing Exposes Trust Flaw in Net

TECHNOLOGY

How Was Egypt's Internet Access Shut Off?

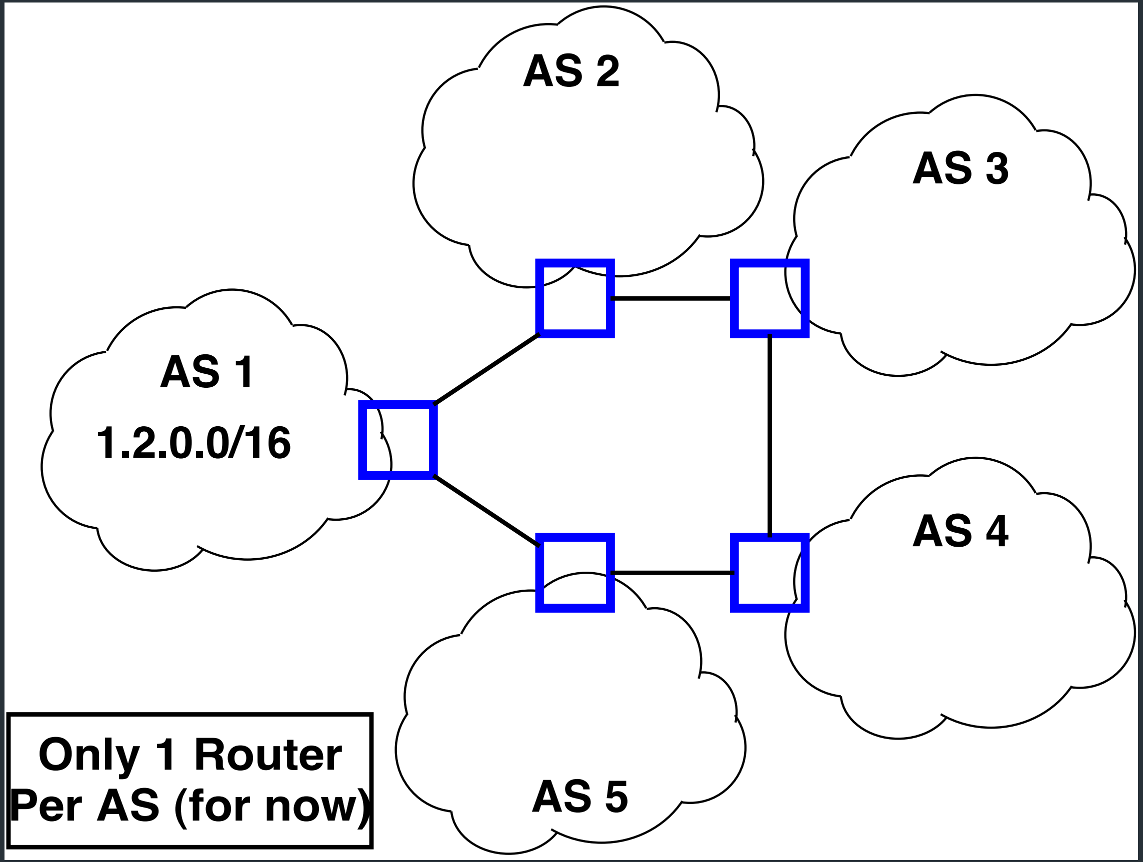
How Russia Took Over Ukraine's Internet in Occupied Territories

By Adam Satariano and
Graphics by Scott Reinhard
Aug. 9, 2022

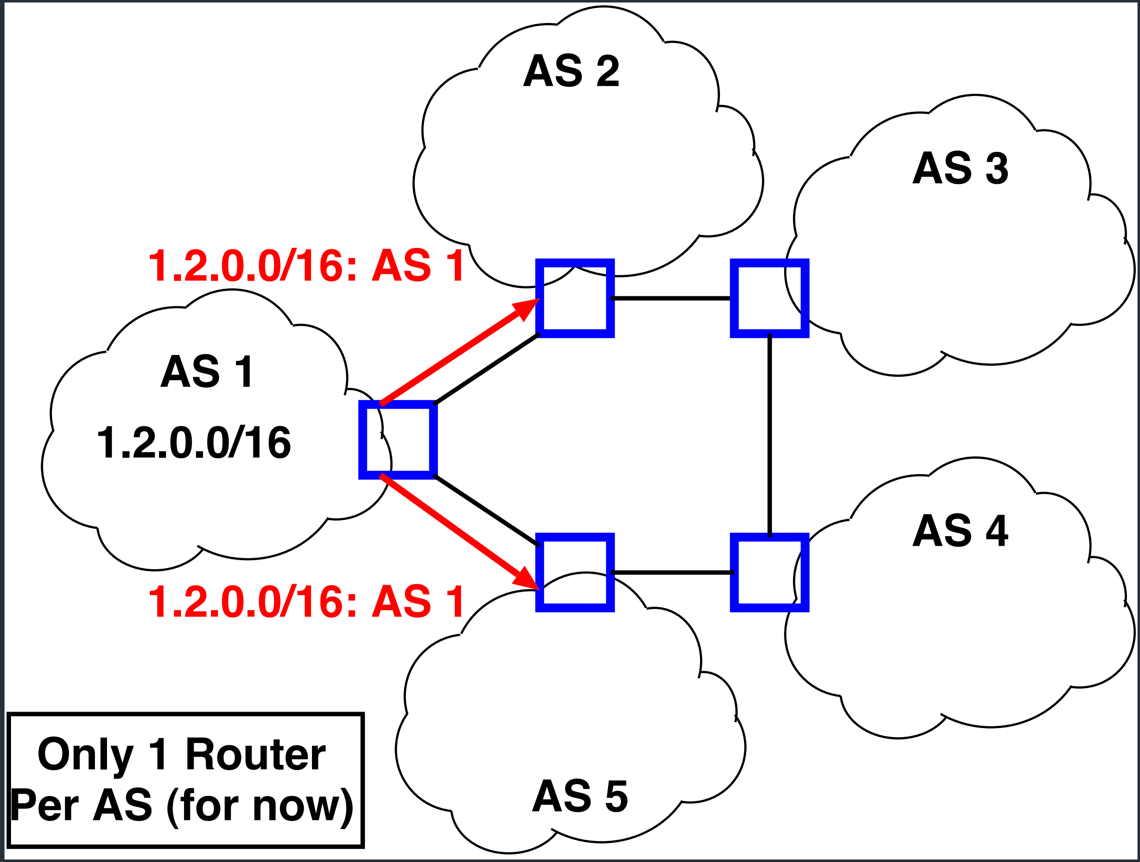
A Network Operations Center (NOC)

Demo: AS11078

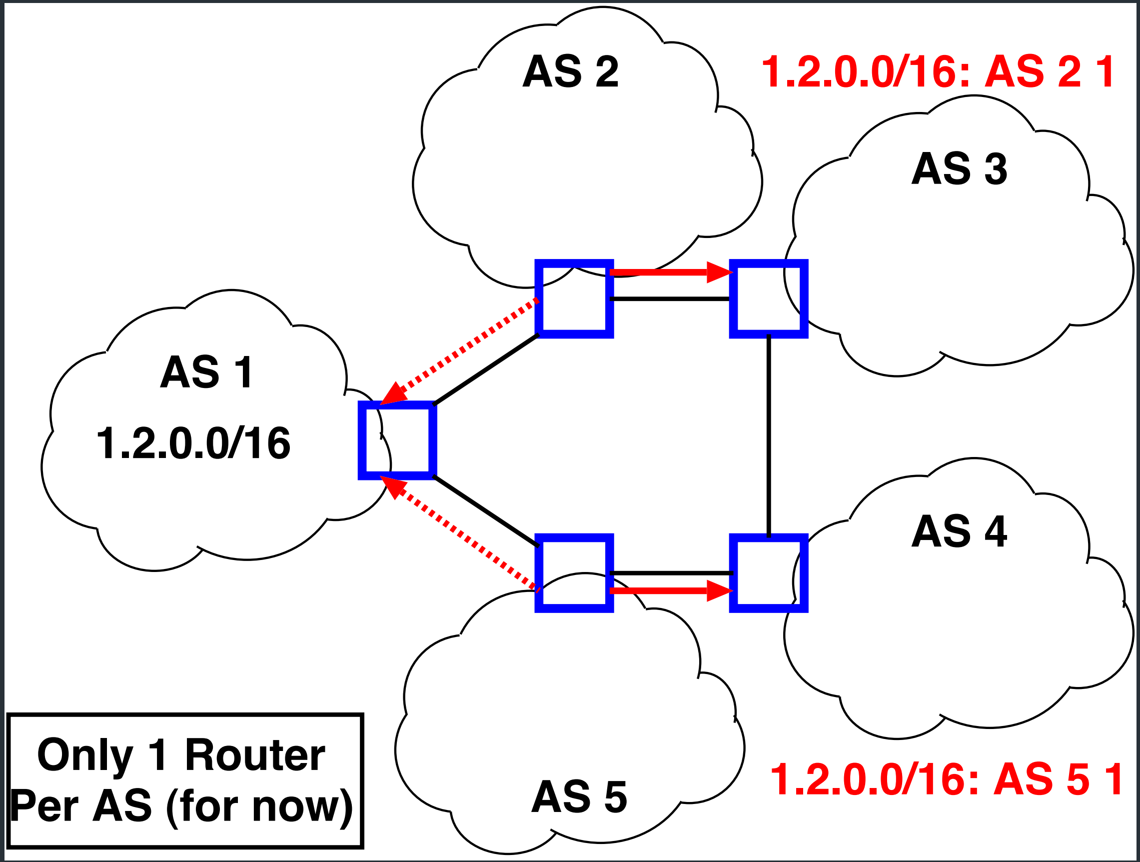
BGP Example



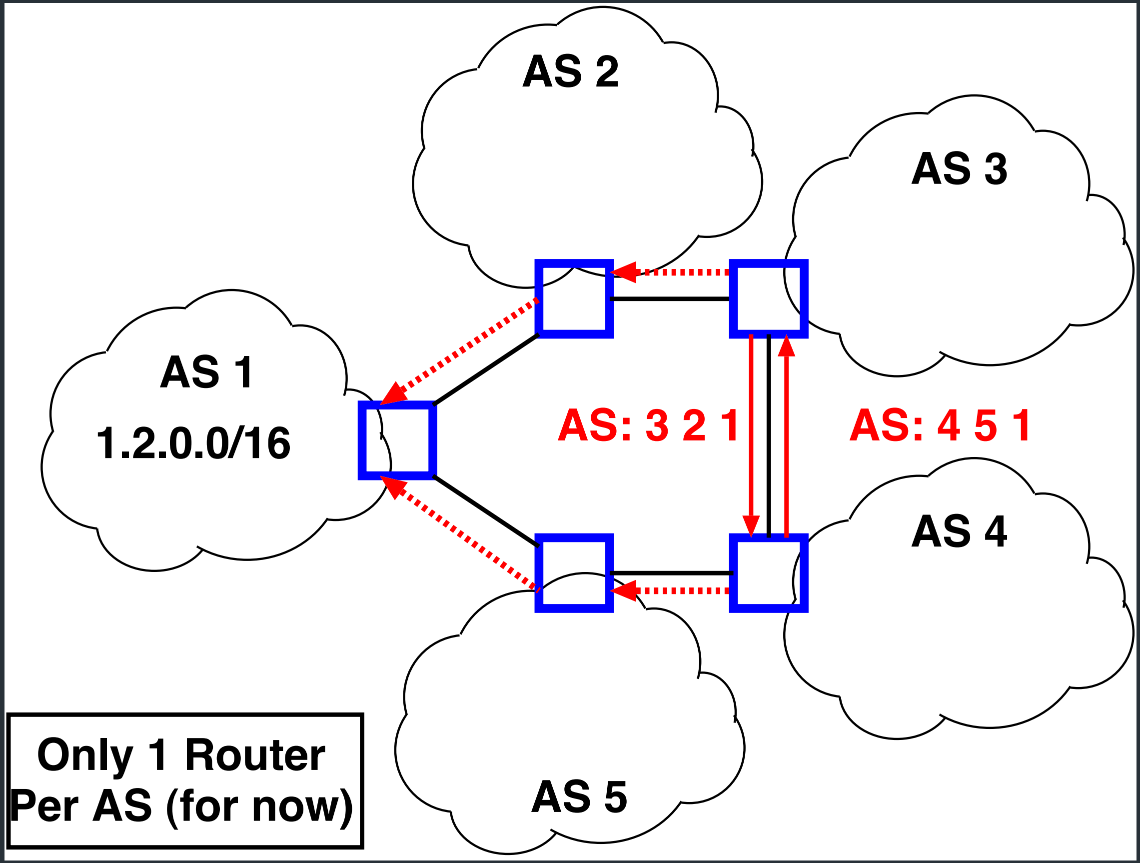
BGP Example


AS 2
AS 3
1.2.0.0/16: AS 1
AS 1
1.2.0.0/16
AS 4
1.2.0.0/16: AS 1
Only 1 Router
Per AS (for now)
AS 5

BGP Example


AS 2
1.2.0.0/16: AS 2 1
AS 3
AS 1
1.2.0.0/16
AS 4
Only 1 Router
Per AS (for now)
AS 5
1.2.0.0/16: AS 5 1

BGP Example


AS 2
AS 3
AS 1
1.2.0.0/16
AS: 3 2 1
AS: 4 5 1
AS 4
AS 5
Only 1 Router
Per AS (for now)

BGP Example