Reverse Traceroute

Ethan Katz-Bassett, Harsha V. Madhyastha, Vijay K. Adhikari, Arvind Krishnamurthy, Thomas Anderson

RIPE 58, May 2009

This work partially supported by Cisco, Google, NSF

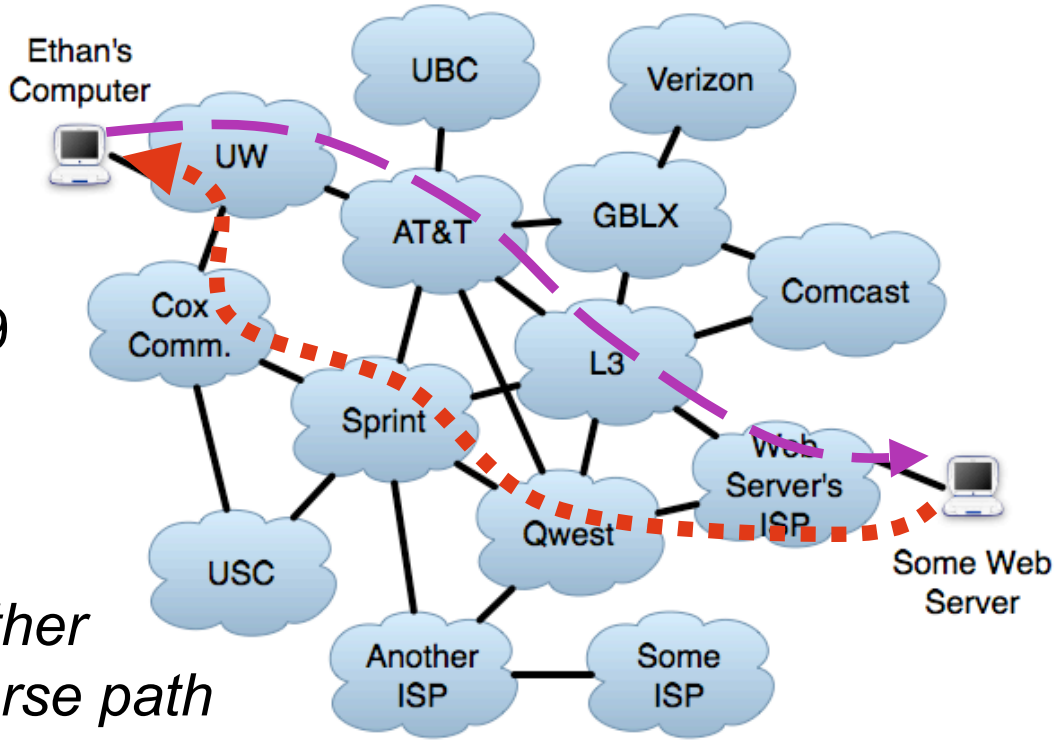
What is traceroute used for?

- Diagnosis:
  - Is a destination reachable?
    - If yes, what is the route taken?
    - If no, where does it seem to be broken?
  - Is path longer than necessary?
- Researchers from UW use traceroute to:
  - Map the Internet
  - Predict performance and compare ISPs
  - Detect black holes and reachability problems

Traceroute's Fundamental Limitation

"*The number one go-to tool is traceroute. The number one plague [is path asymmetry, as] the reverse path itself is completely invisible.*"

NANOG tutorial, 2009



"*In order to more precisely troubleshoot problems… [we need] the ability to gather information about the reverse path back from clients to [our] nodes.*"

Draft paper by large content provider, 2009

Motivation and Goal

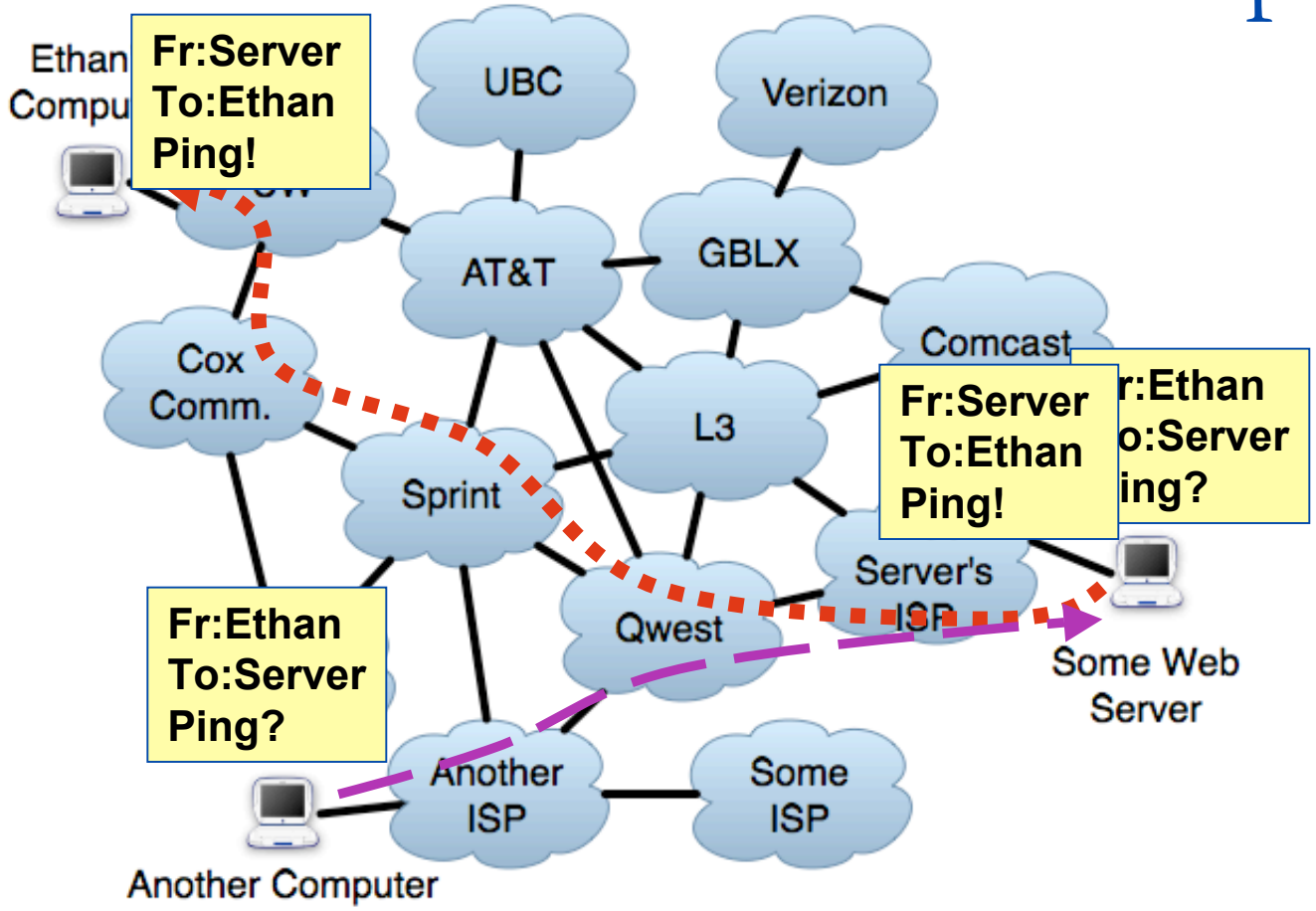
- Reverse route information useful for same reasons as traceroute
  - But ***D*** must run traceroute to get path from ***D***
  - Use public traceroute server?
  - Ask mailing list for help?
  - Assume symmetric routing?

Goal: Reverse traceroute, without control of destination

IP Options to Identify Reverse Hops

- Unlike TTL, *IP Options* reflected in reply, so work on forward and reverse path
- *Record Route (RR)* option
  - Record first 9 routers on path
  - If destination within 8, reverse hops fill rest of slots
  - … but average path is 15 hops, 30 round-trip
- *Timestamp (TS)* option
  - Specify ≤ 4 IPs, each records if traversed in order
  - Ping ***D***, ask for ***TS(D,R)***, to see if ***R*** on reverse path
  - "Guess" reverse hops using Internet maps
  - … but TS filtered, plus limited deployment

Spoof to Best Use VPs and RR Option



Spoofing? Isn't that bad?

- We use only a restricted version that is perfectly safe
  - Only spoofing as nodes we control
    - Like a "reply to" address
    - Send from a vantage point to another, through destination
  - Rate limit, restrict destinations (no broadcast IPs)
  - We've sent millions of spoofed probes to 10s of thousands of IPs, no complaints
- Lets us approximate control of destinations

Coverage of IP Options

- Of IPs in traceroutes from PlanetLab to all prefixes:

  Record route:

  - 58% within 8 hops of some PL vantage point
  - 1% dropped RR packets [*Sherwood, SIGCOMM 2008*]
  - 9% do not record [*Sherwood, SIGCOMM 2008*]

  Timestamp:

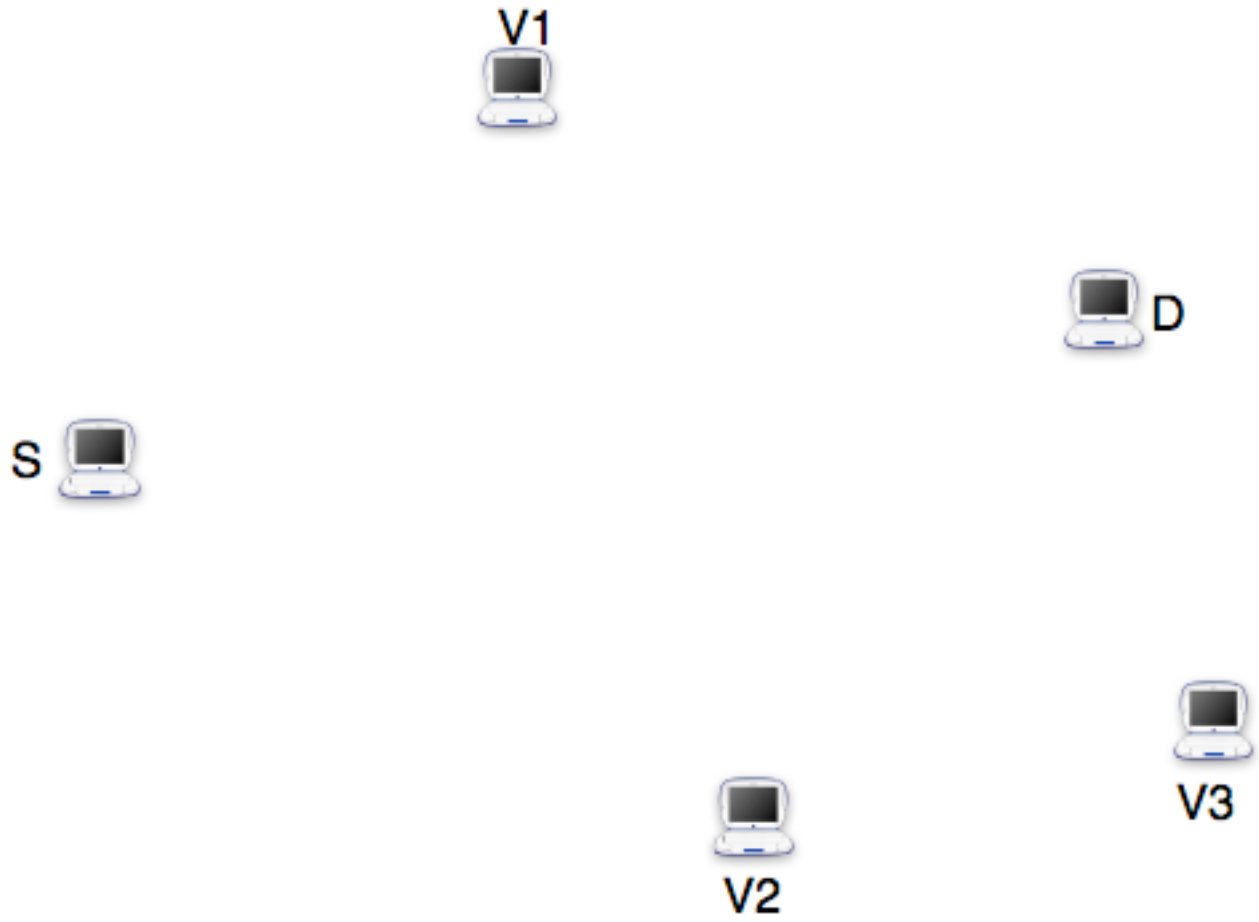
  - 37% gave valid timestamps
  - Additional 18% replied with TS=0
  - 61 of top 100 ASes timestamp from most routers

- Good support, but not universal
- Combine both techniques to improve coverage

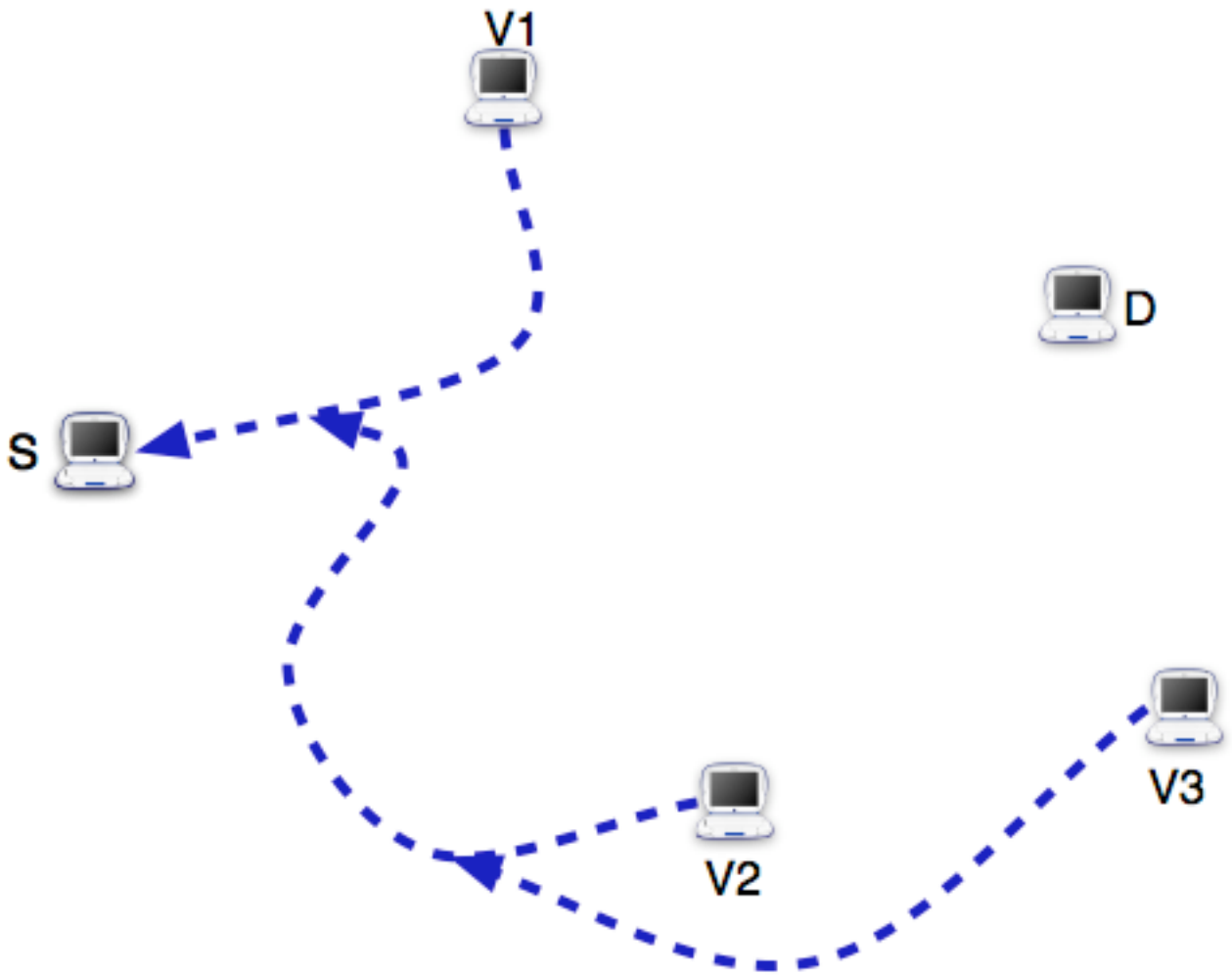
Stitching Together the Path

- Assume destination-based routing
- With Internet routing, next hop depends only on destination, not source or path so far
  - Once we know the path from ***D*** to ***R***, need only determine path from ***R*** back to ***S***
- Lets us stitch together parts of reverse path

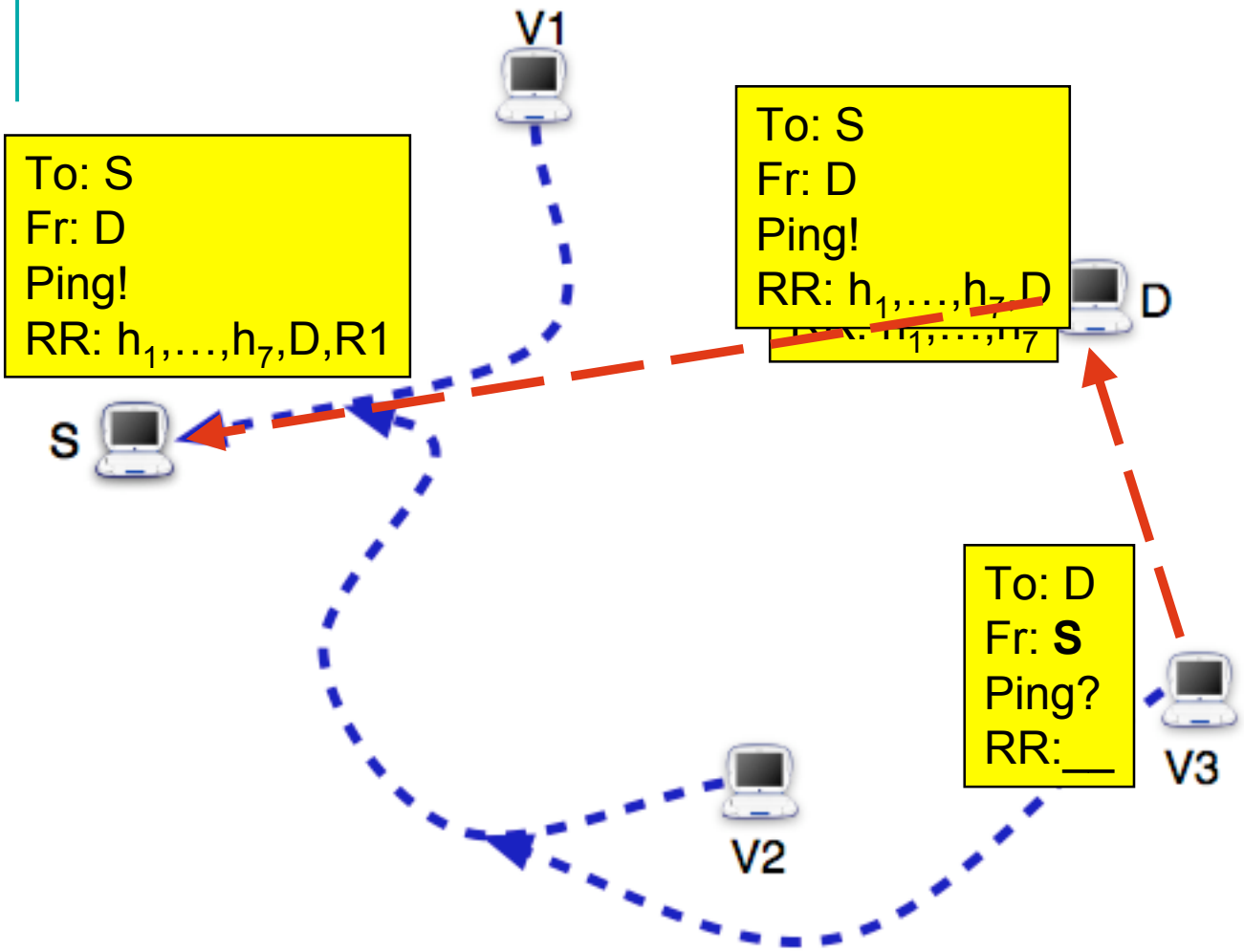
(A simplification with some caveats, but many apply to traceroute too.)



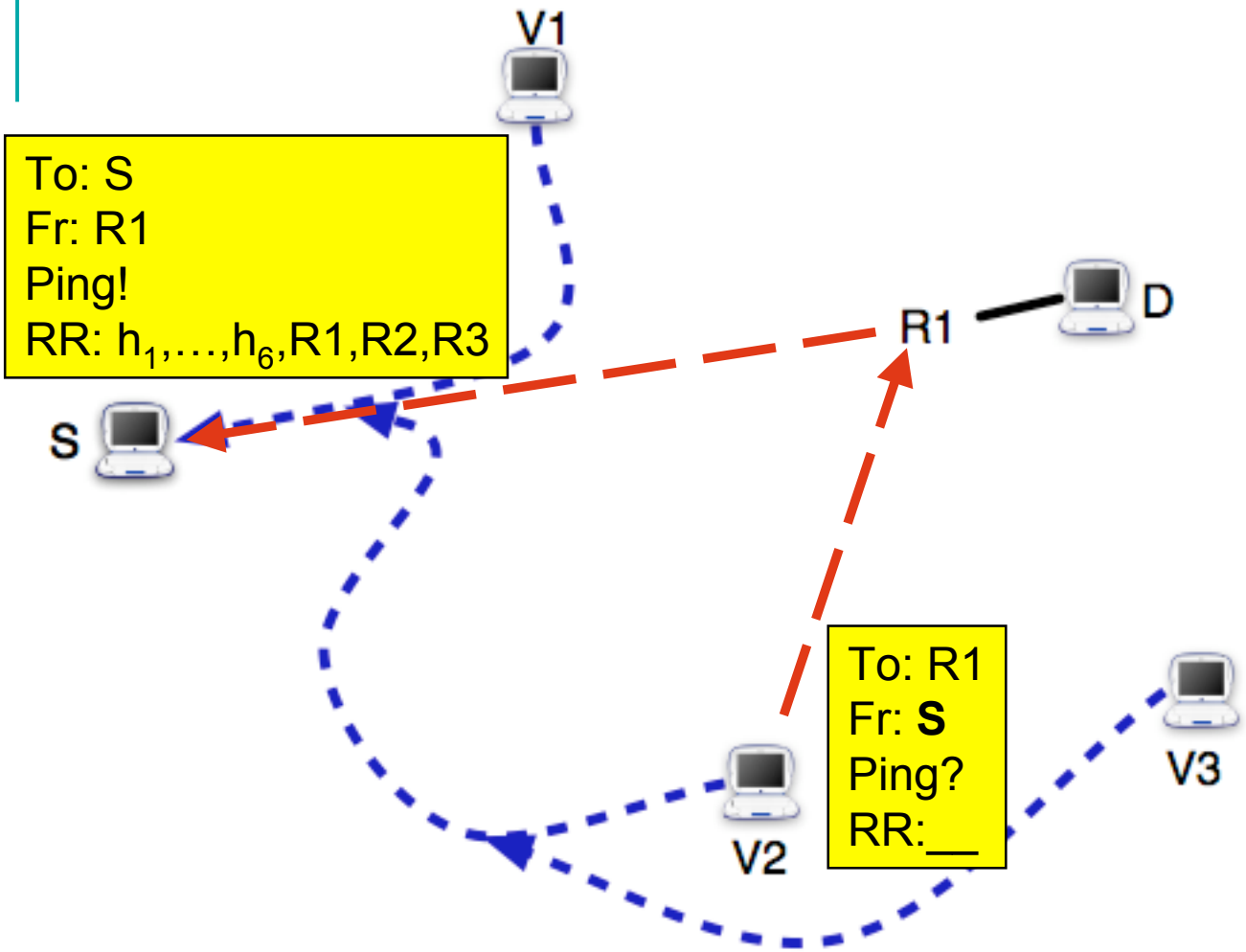
- Want reverse path from **D** back to **S**, but don't control **D**
- Set of vantage points, some of which can spoof



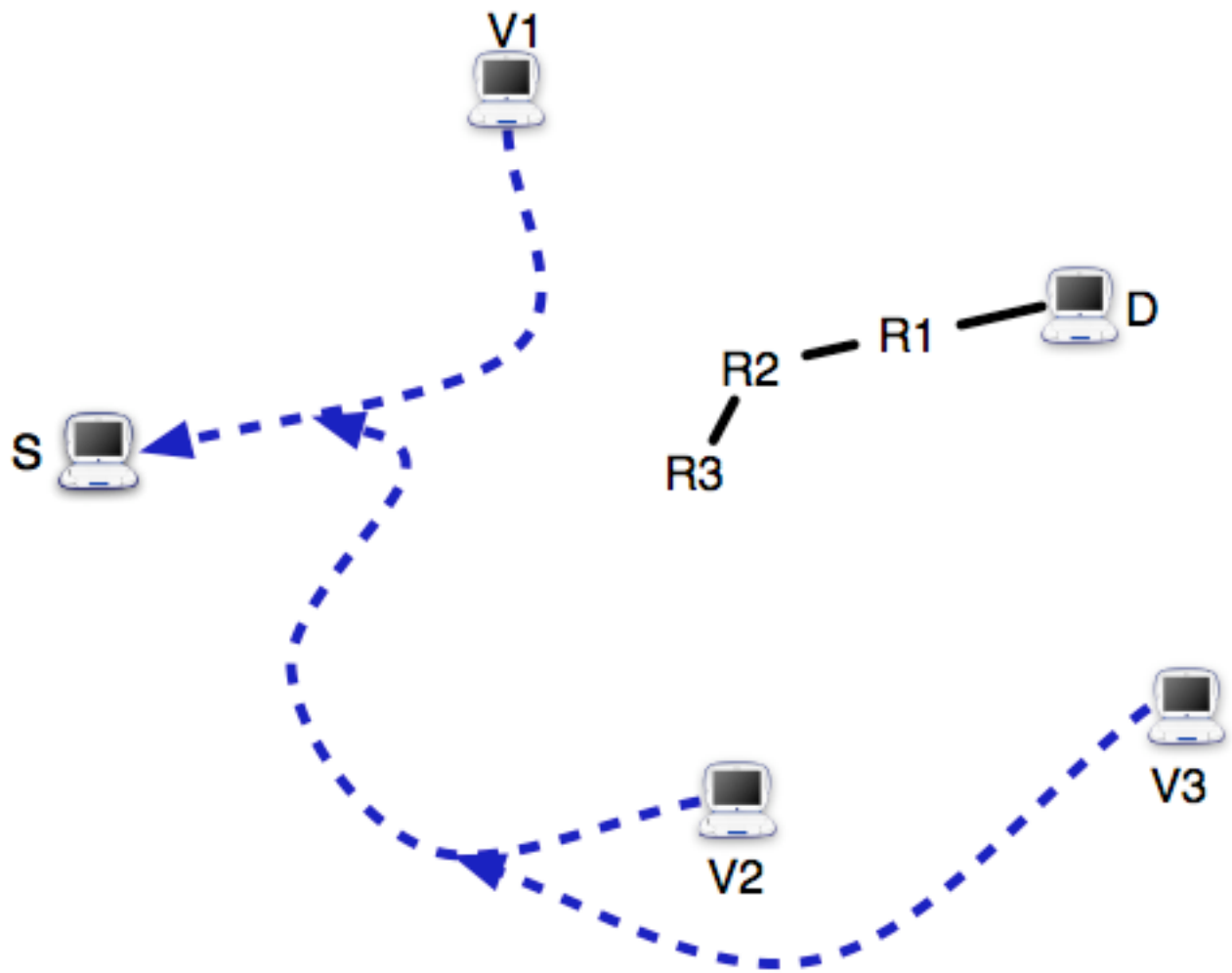
- Traceroute from all vantage points to **S**
- Gives atlas of paths to **S**; if we hit one, we know rest of path

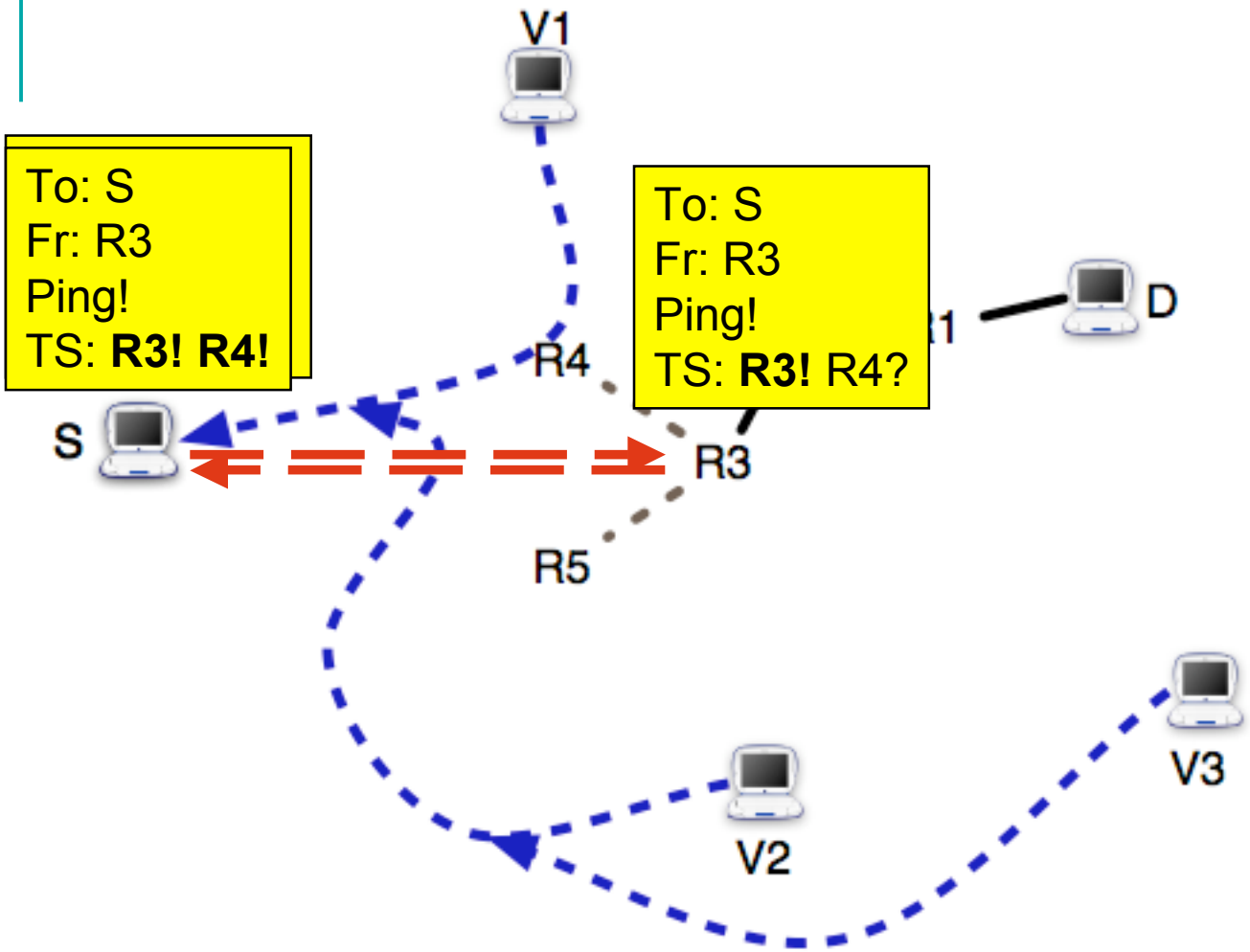


- From vantage point within 8 hops of **D**, ping **D** spoofing as **S** with record route option
- **D**'s response will contain recorded hop(s) on return path

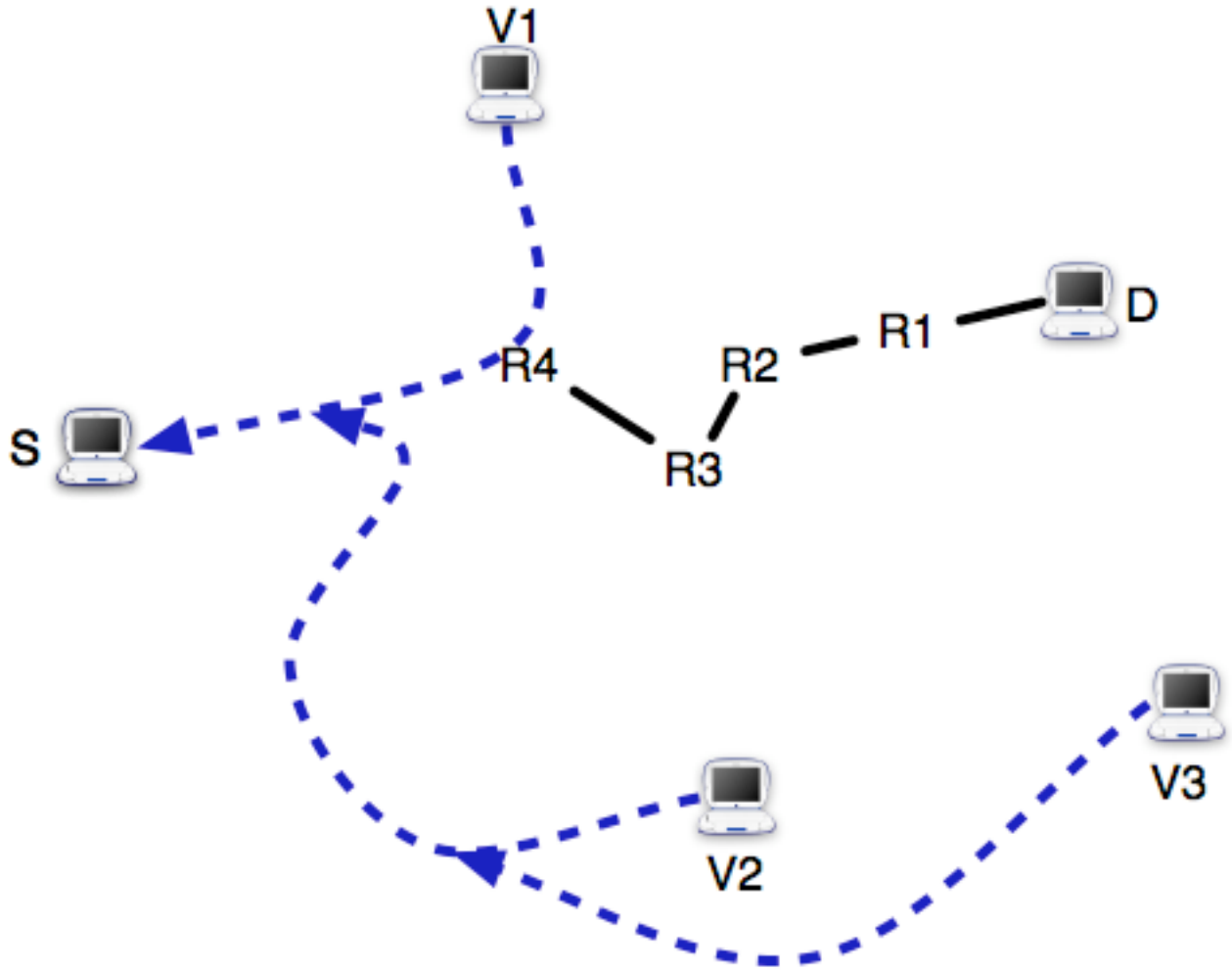


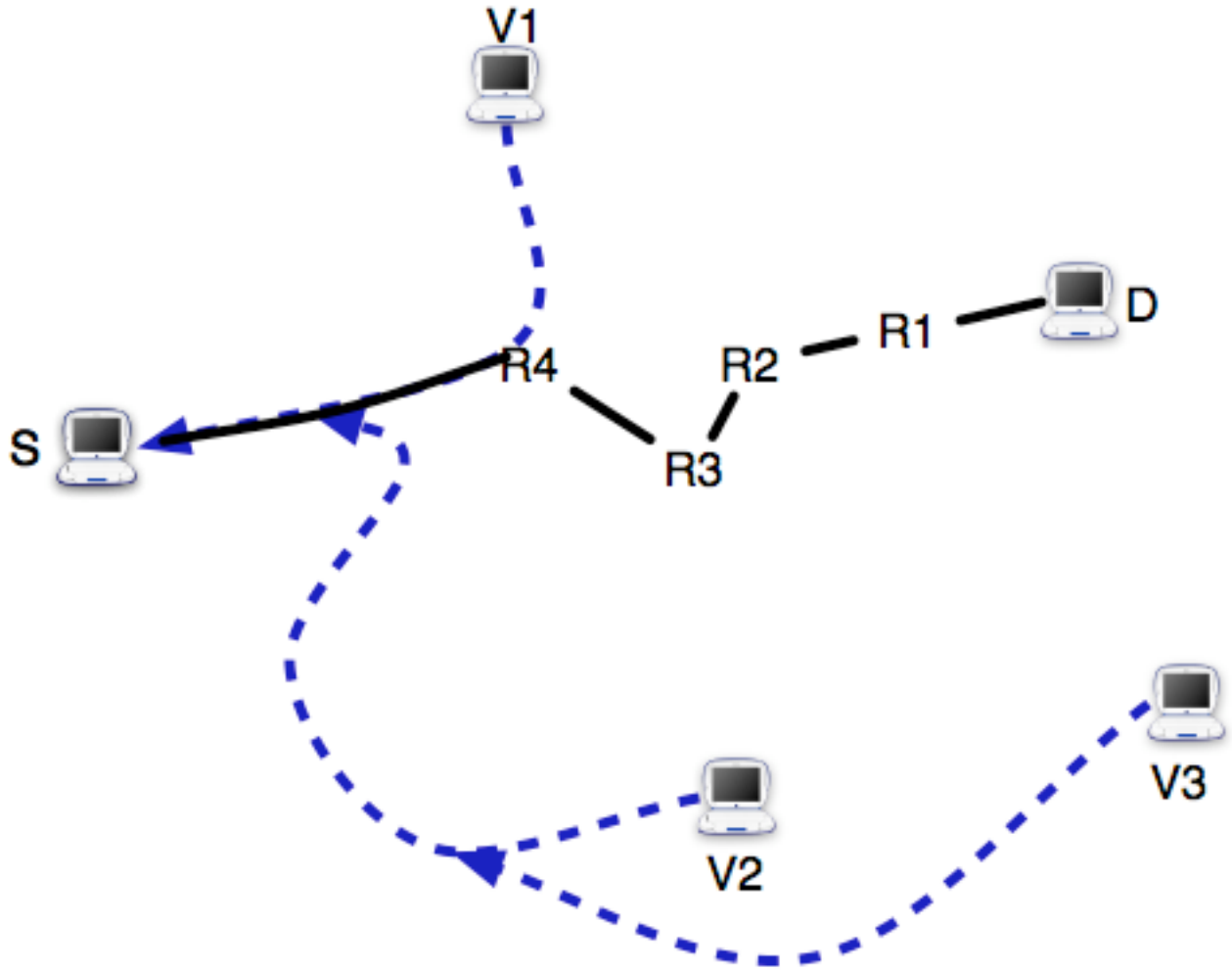
- Iterate, performing TTL=8 pings and spoofed RR pings for each router we discover on return path


V1
D
R1
R2
R3
S
V2
V3

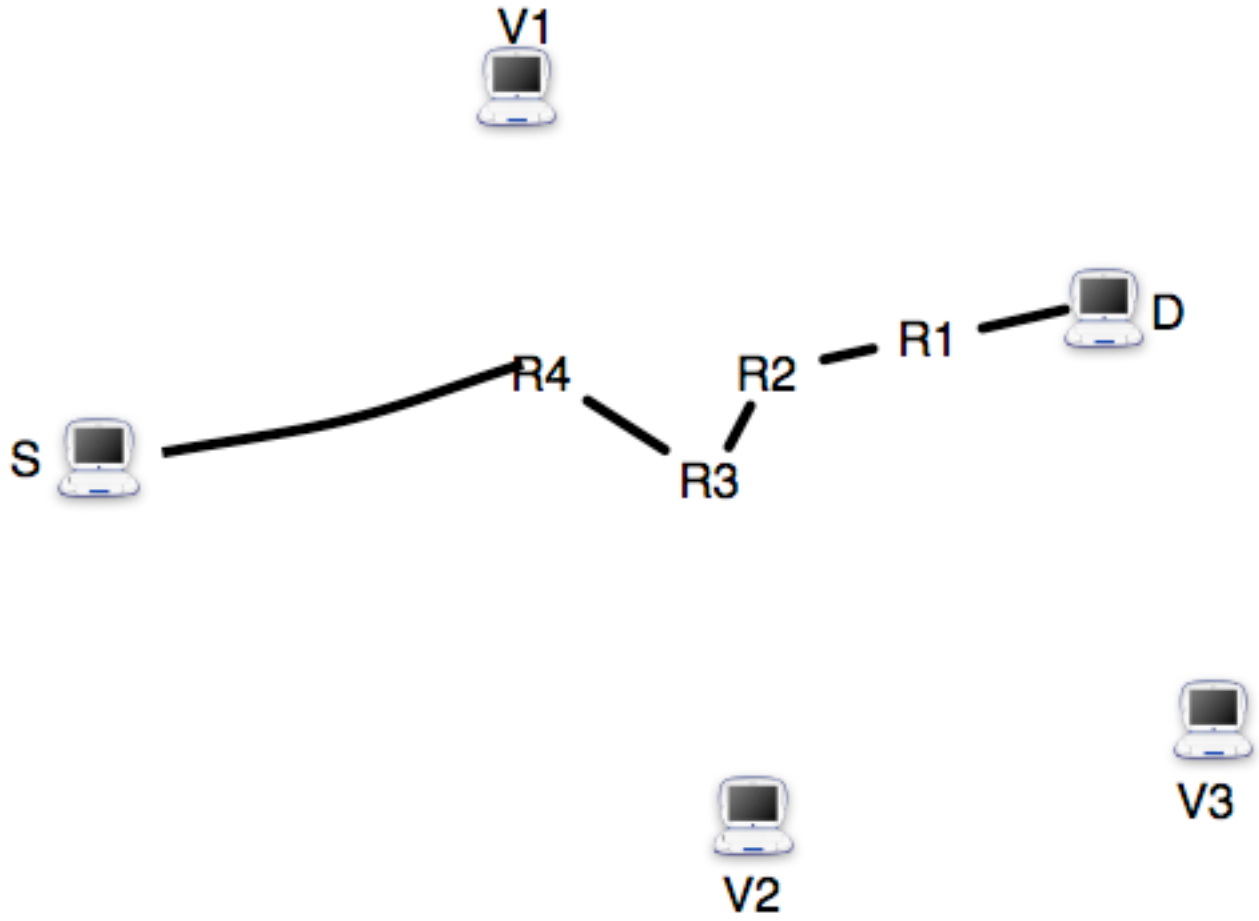


- If no spoofing vantage points within 8 hops, consider set of routers directly connected to **R3** (in pre-measured topology)
- Use timestamp option to try to verify which is on return path


V1
D
R1
R2
R4
S
R3
V3
V2

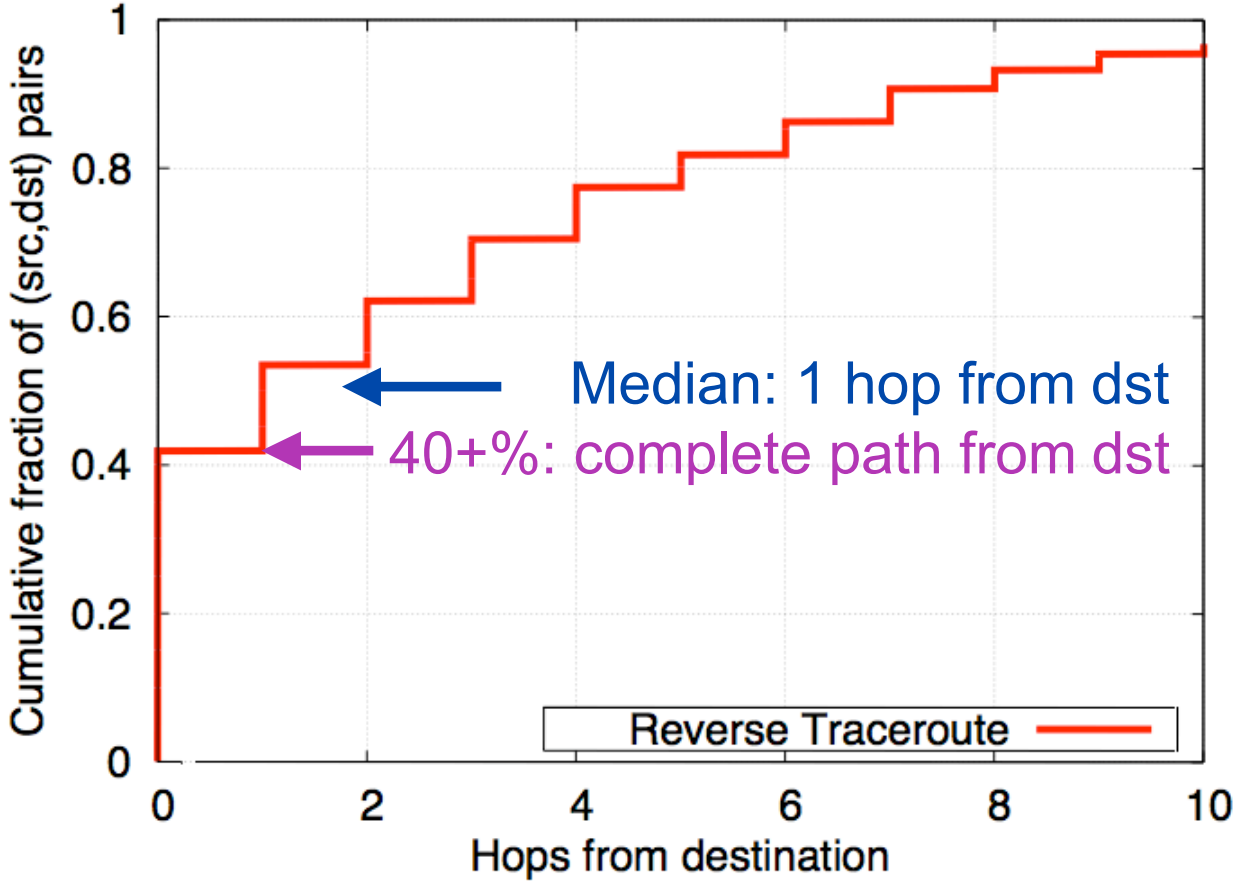


- Once we see a router on a known path, we know remainder



Techniques combine to give us complete path

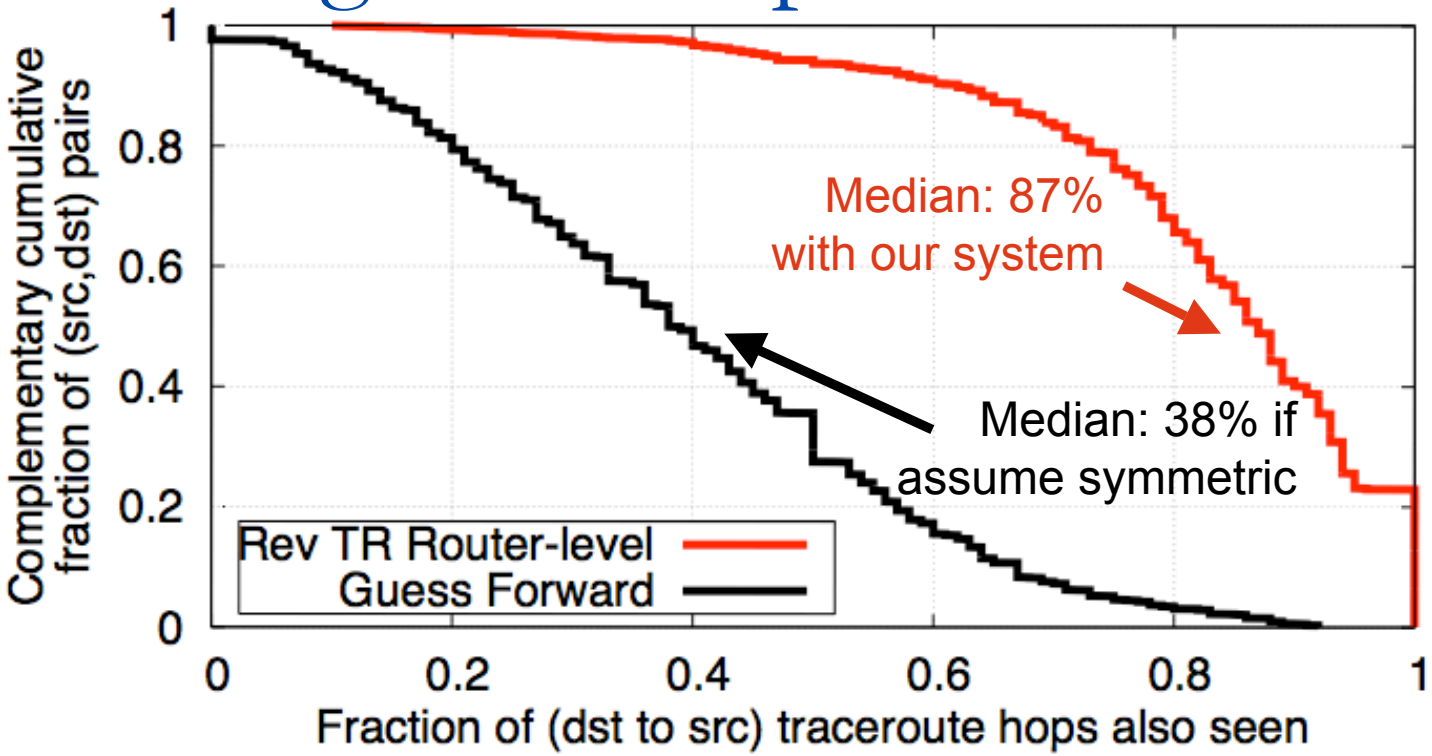
How often does it work?



Reverse paths from 200 random destinations across Internet back to 11 PlanetLab sites around the world

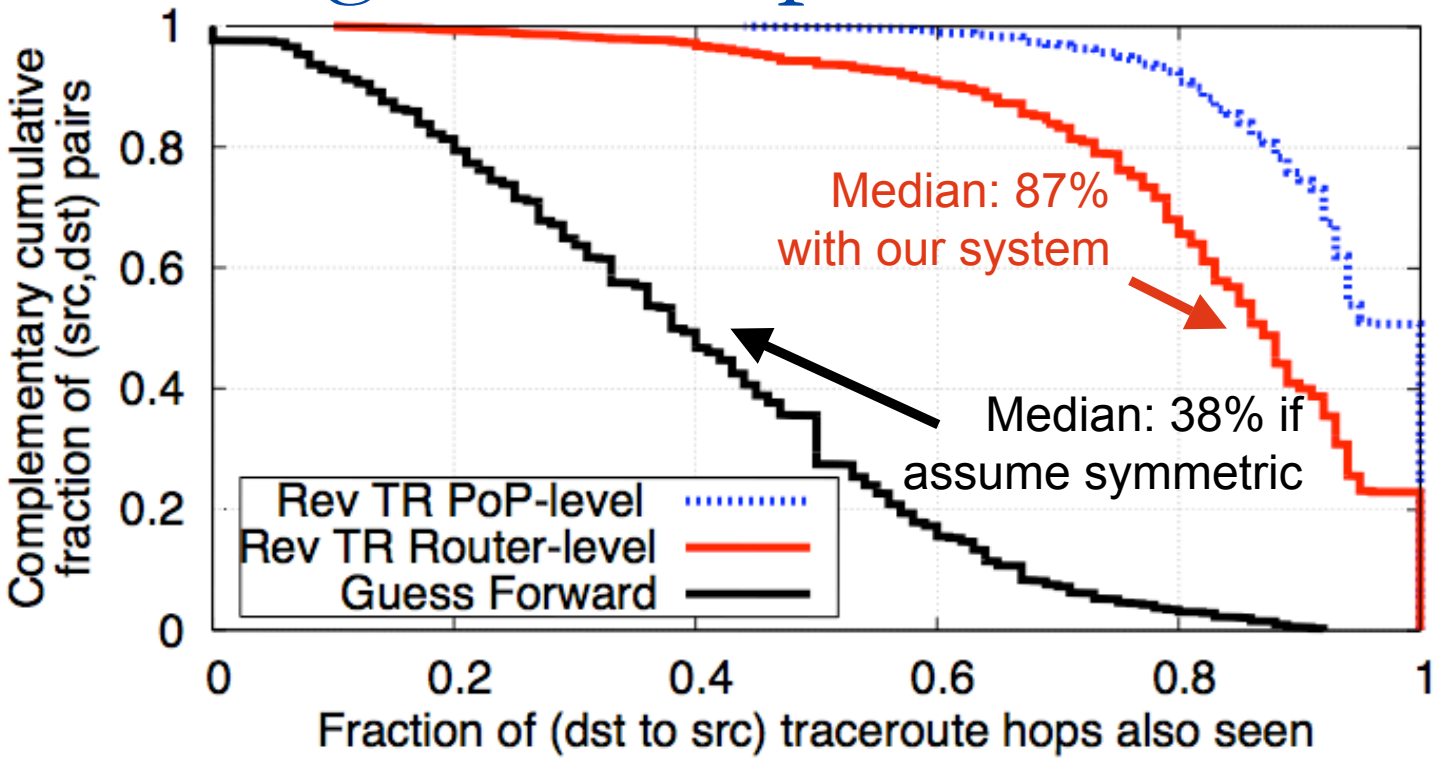
- Often able to determine complete reverse path
- When not, can often get minus last few hops
- Would improve with more spoofing vantage points

Does it give same path as traceroute?



- 200 PlanetLab destinations, where we can directly traceroute “reverse” path
- Usually identify most hops seen by traceroute
- Hard to know which interfaces are on the same router

Does it give same path as traceroute?



- 200 PlanetLab destinations, where we can directly traceroute "reverse" path
- Usually identify most hops seen by traceroute
- Hard to know which interfaces are on the same router
  - If we consider PoPs instead, median=100% accurate

Example of debugging inflated path

- 150 ms round-trip time Orlando to Seattle (3x expected)
  - E.g., Content provider detects poor client performance
- *(Current practice)* Issue traceroute, check if indirect

Hop no.	DNS name / IP address	Location	RTT
1	132.170.3.1	Orlando, FL	0ms
2	198.32.155.89	–	0ms
3	jax-flrcore-7609-1-te23-v1820-1.net.flrnet.org	Jacksonville, FL	3ms
4	atlantaix.cox.com	Atlanta, GA	9ms
5	ashbbbrj02-ae0.0.r2.as.cox.net	Ashburn, VA	116ms
6	core2.te5-1-bbnet1.wdc002.pnap.net	Washington, DC	35ms
7	cr1.wdc005.inappnet-62.core2.wdc002.internap.net	Washington, DC	26ms
8	cr2-cr1.wdc005.internap.net	Washington, DC	24ms
9	cr1.mia004.inappnet.cr2.wdc005.internap.net	Miami, FL	53ms
10	cr1.sea002.inappnet.cr1.mia004.internap.net	Seattle, WA	149ms

- Indirectness: FL→DC→FL, but does not explain huge latency jump from 9 to 10

Example of debugging inflated path

- *(Current practice)* Issue traceroute, check if indirect
  - Does not fully explain inflated latency
- *(With our tool)* Issue reverse traceroute, check rev path

Hop no.	DNS name / IP address	Location	RTT
1	cr1.sea002.inappnet.cr1.mia004.internap.net.	Seattle, WA	148ms
2	cr1.sea002.inappnet.cr2.lax009.internap.net.	Seattle, WA	141ms
3	internap-peer.lsanca01.transitrail.net.	Los Angeles, CA	118ms
4	te4-1–4016.tr01-lsanca01.transitrail.net.	Los Angeles, CA	118ms
5	te4-1–160.tr01-plalca01.transitrail.net.	Palo Alto, CA	109ms
6	te4-1.tr01-sttlwa01.transitrail.net.	Seattle, WA	92ms
7	te4-1.tr01-chcgil01.transitrail.net.	Chicago, IL	41ms
8	te2-1–583.tr01-asbnva01.transitrail.net.	Ashburn, VA	23ms
9	132.170.3.1	Orlando, FL	0ms
10	planetlab2.eecs.ucf.edu.	Orlando, FL	0ms

- Indirectness: WA→LA→WA
  Bad rev path causes inflated round-trip delay

Summary

- Traceroute is very useful tool,
  but cannot provide reverse path
- Our reverse traceroute system fixes limitation,
  provides complementary information
- Gives most hops as if you issued traceroute
  from remote site
- Useful for troubleshooting, and could give more
  complete picture during unreachability

Reverse Traceroute and RIPE

- Building downloadable tool
  - Internal testing now,
    website in June (rev TR from any destination),
    public in July (rev TR to any source)
  - Email *ethan@cs.washington.edu* to be early user
- Coverage tied to distribution of vantage points
  - Similar to hosting public traceroute server
  - Have some hosts to contribute?

Thanks!

Questions? Ideas? Applications?

Questions?

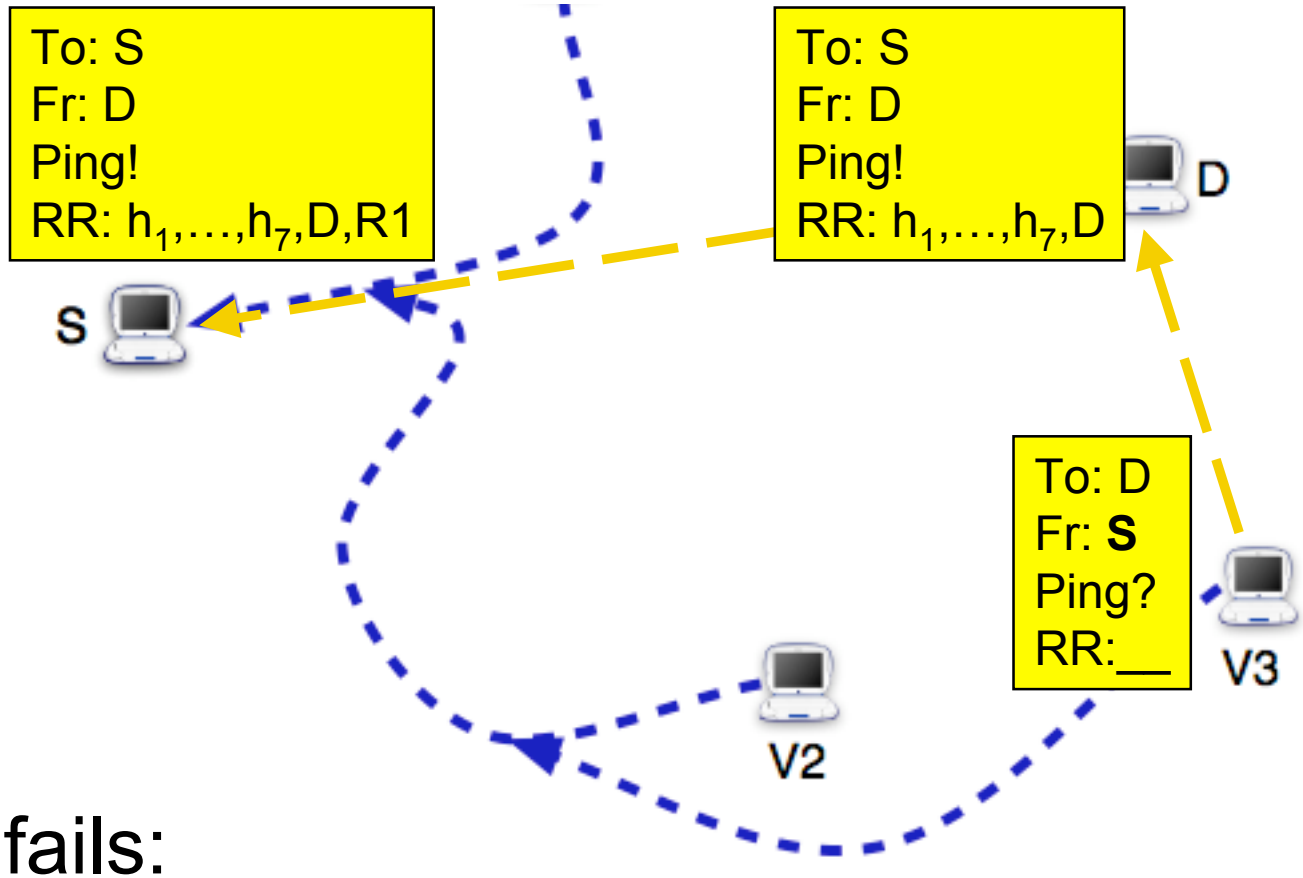
Please contact *ethan@cs.washington.edu* if you want to be an early user of our tool or to host vantage points

Techniques Applied to Unreachability

```
traceroute to 18.0.0.1 (18.0.0.1), 64 hops max, 40 byte packets
 1  128.208.3.102   0.710 ms 0.291 ms 0.275 ms
 2  205.175.108.21 0.489 ms 0.648 ms 0.273 ms
...
 9  216.24.186.33   74.425 ms  73.705 ms  73.820 ms
10  216.24.184.102 73.218 ms  73.274 ms  73.228 ms
11  * * *
12  * * *
13  * * *
```

- With traceroute, forward and reverse path failures look the same
- With **Hubble**
  - 68% of black holes were partial
  - Able to isolate direction of failure in 68% of these
- With new reverse traceroute techniques?

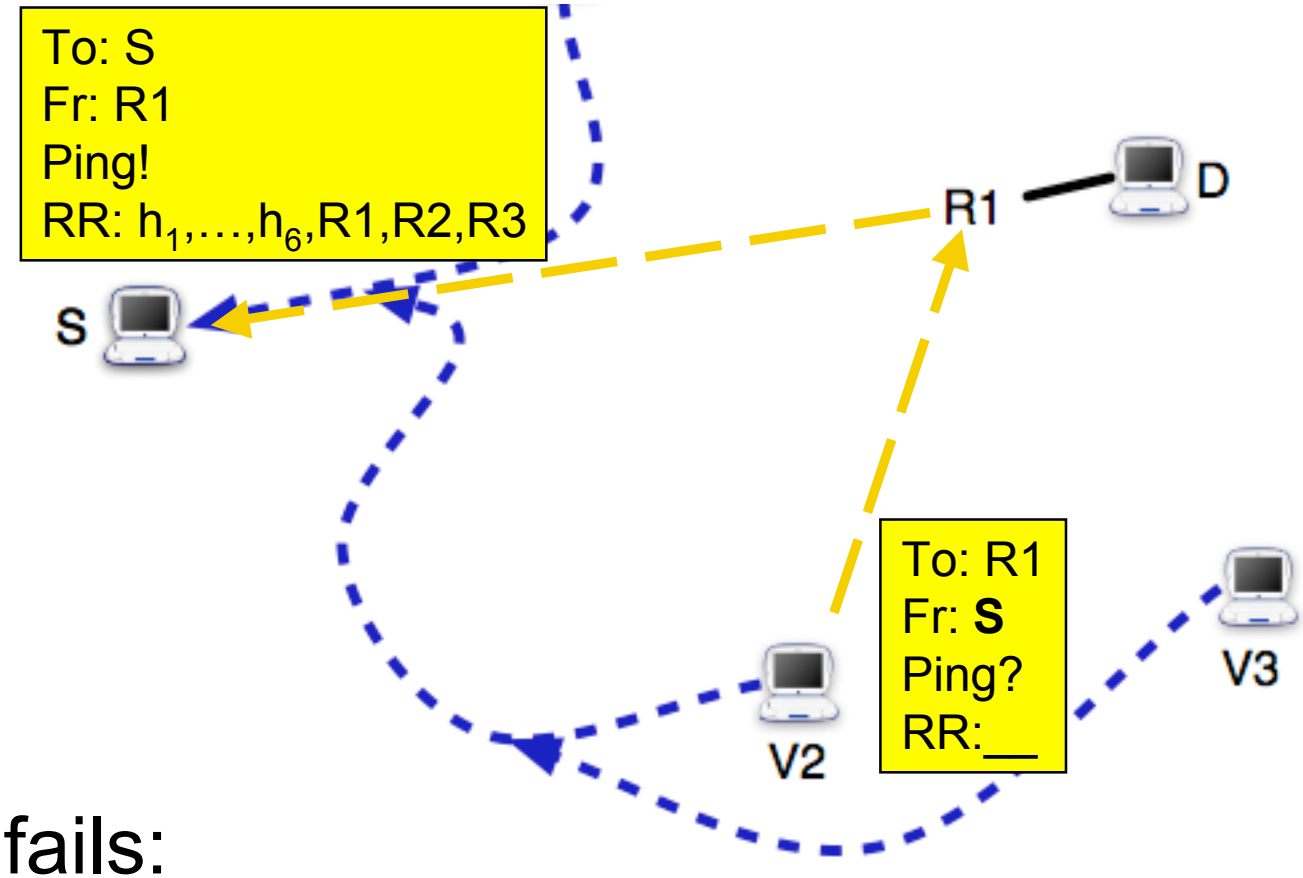
Techniques Applied to Unreachability



If ***S→D*** fails:

- Perform reverse traceroute, spoofing every probe as **S**

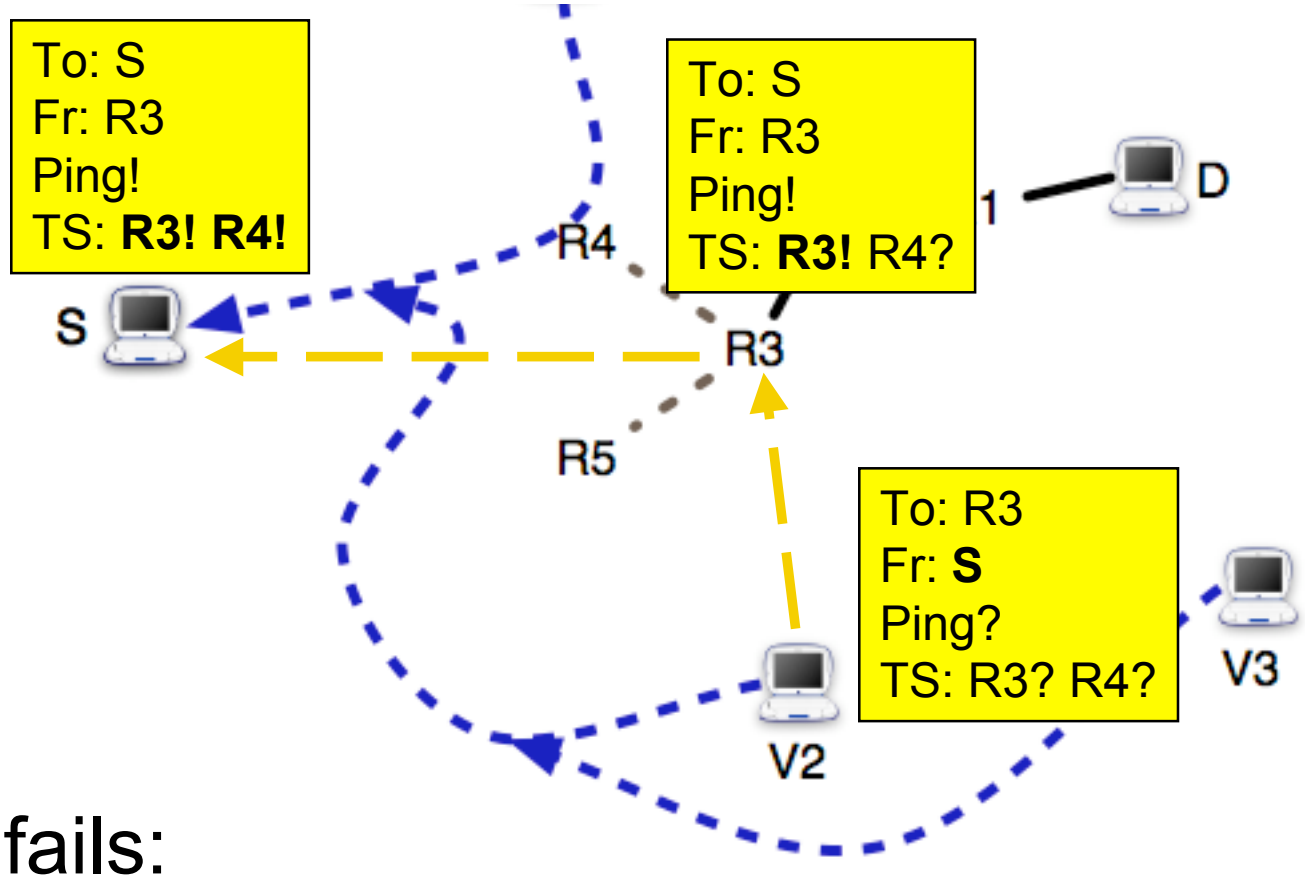
Techniques Applied to Unreachability



If ***S→D*** fails:

- Perform reverse traceroute, spoofing every probe as **S**

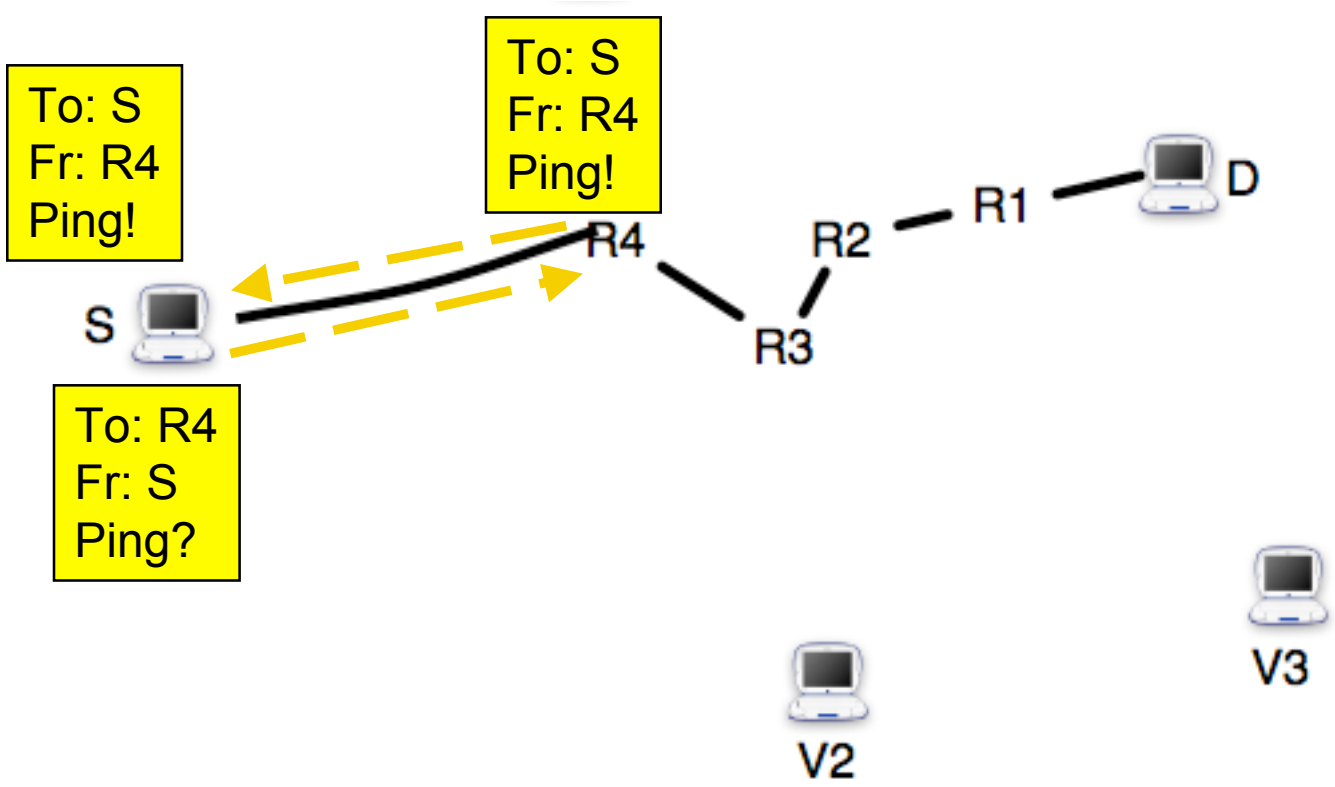
Techniques Applied to Unreachability



If ***S→D*** fails:

- Perform reverse traceroute, spoofing every probe as **S**

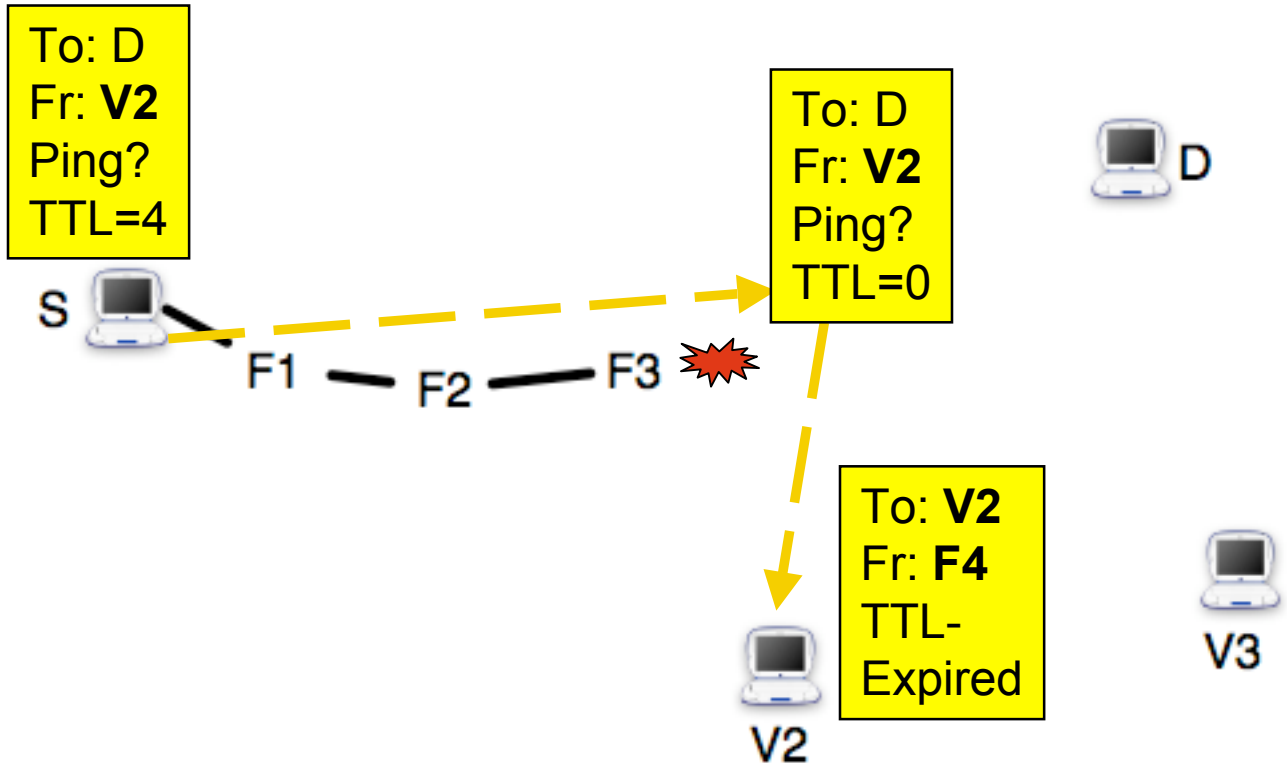
Techniques Applied to Unreachability



If ***S→D*** fails:

- Perform reverse traceroute, but spoofing every probe as **S**
- **S** pings each hop to check reachability, traceroutes to compare paths to partial forward path to **D**

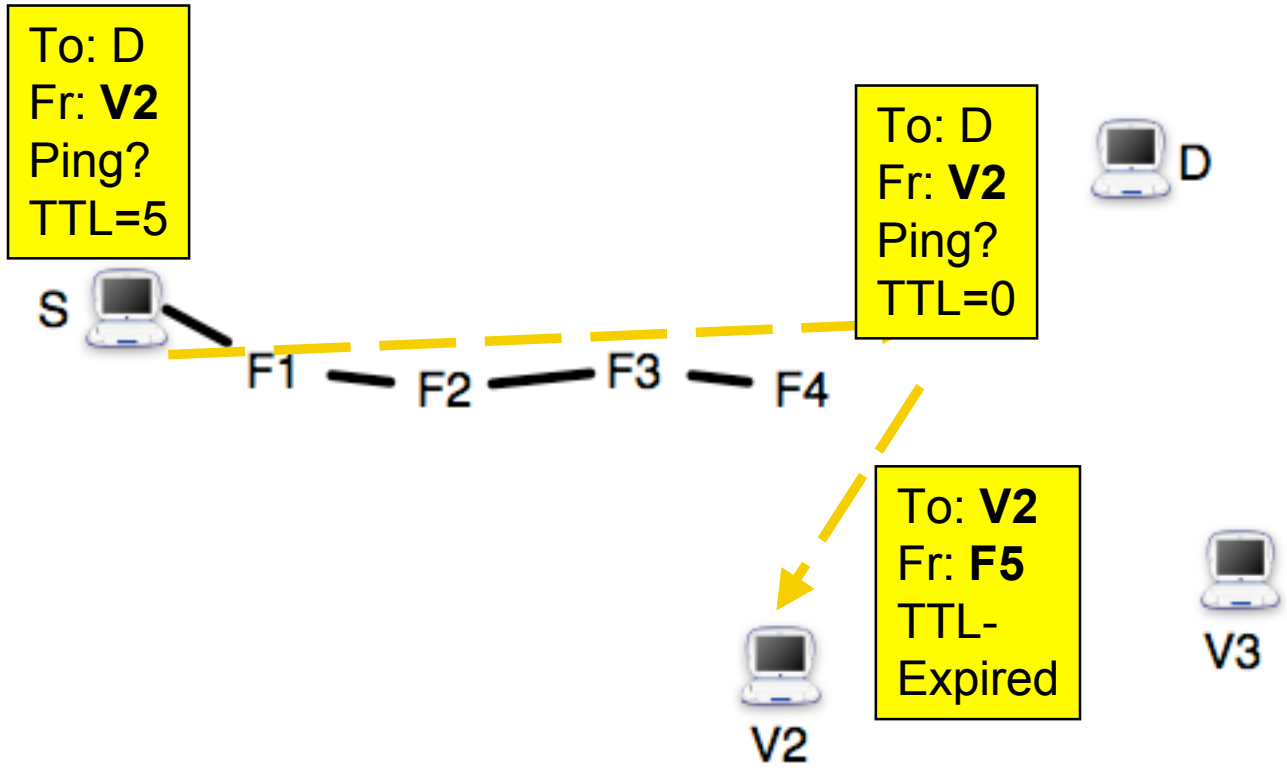
Techniques Applied to Unreachability



If ***D*→*S*** fails:

- **S** traceroutes, spoofing as vantage point that **D** can reach

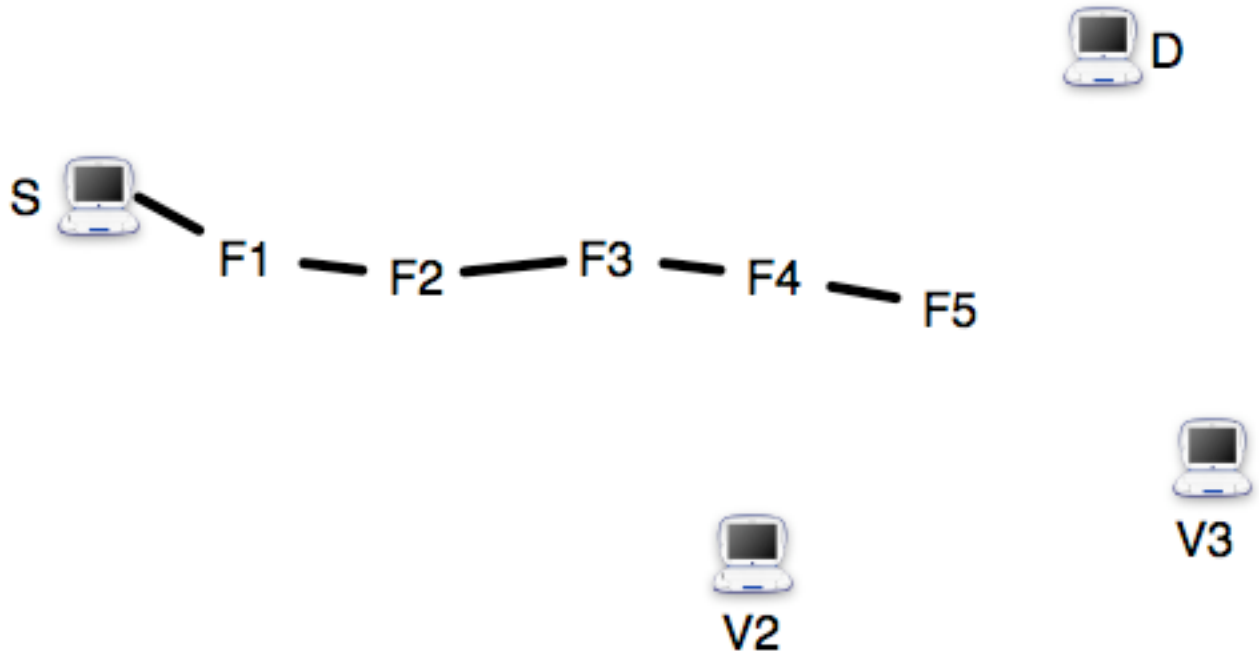
Techniques Applied to Unreachability



If ***D*→*S*** fails:

- **S** traceroutes, spoofing as vantage point that **D** can reach

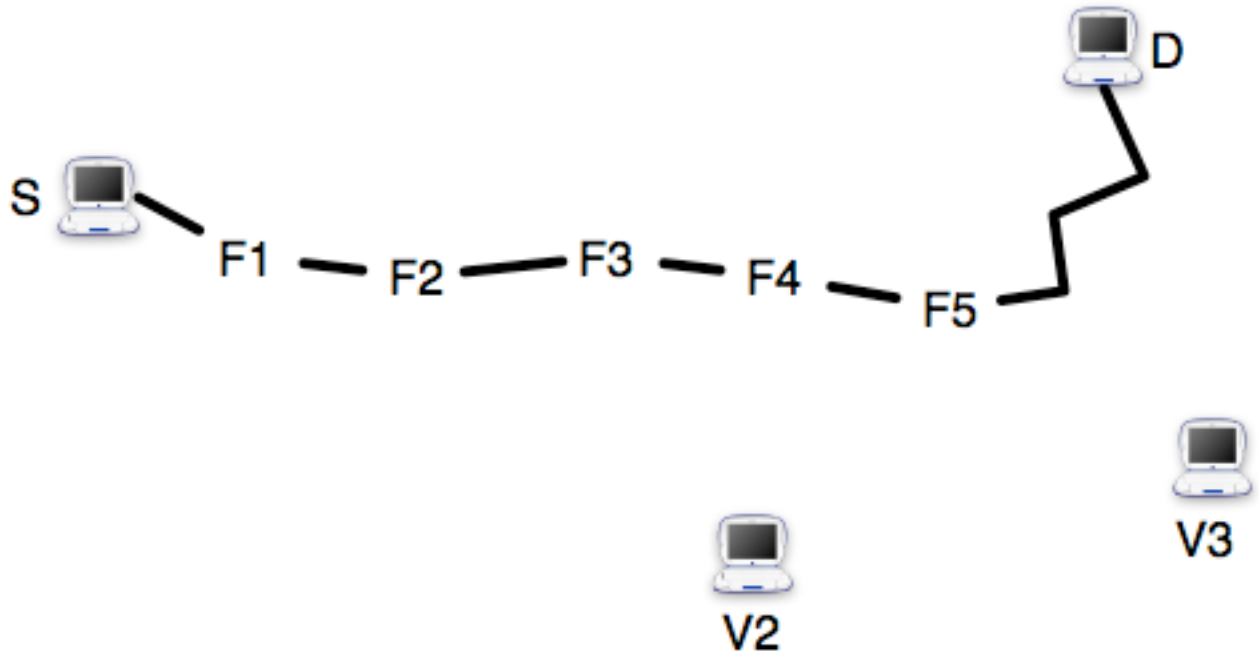
Techniques Applied to Unreachability



If ***D*→*S*** fails:

- **S** traceroutes, spoofing as vantage point that **D** can reach

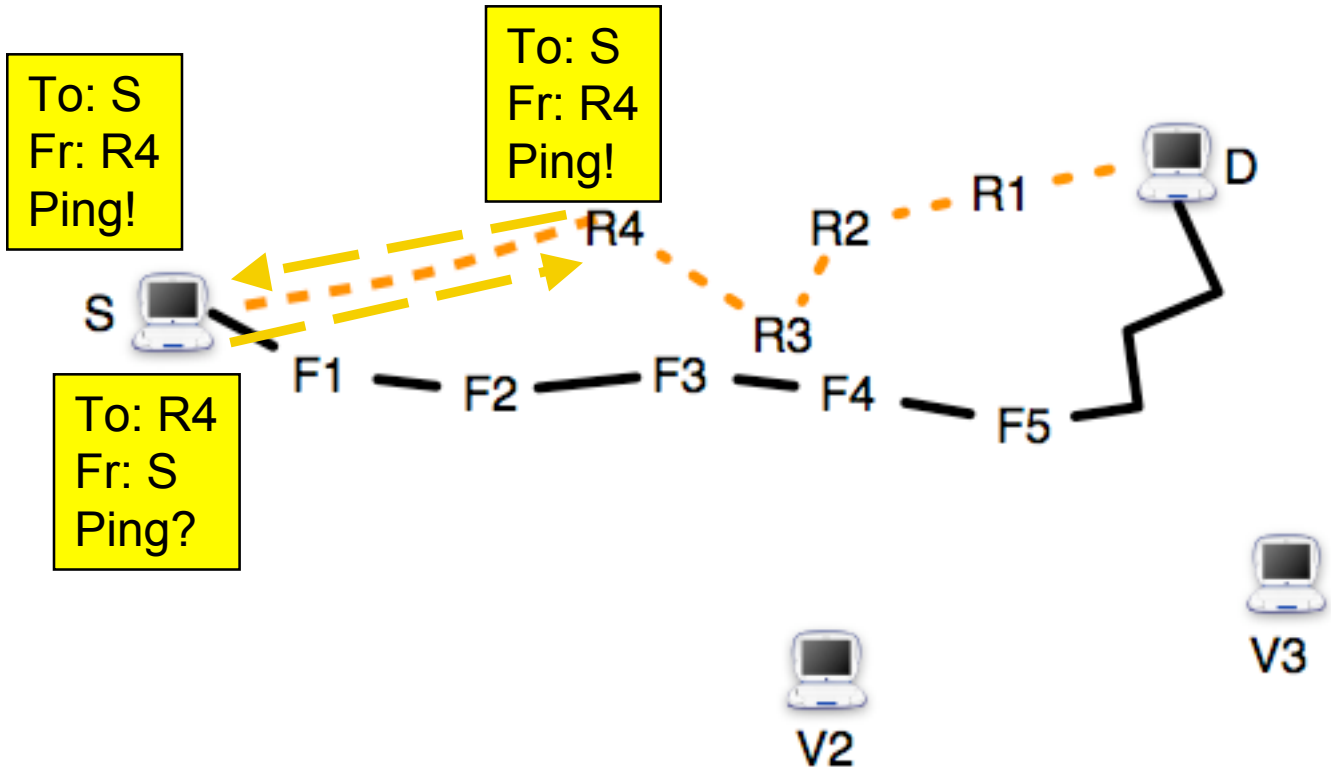
Techniques Applied to Unreachability



If ***D→S*** fails:

- **S** traceroutes, spoofing as vantage point that **D** can reach; ping/ rev traceroute fwd hops to check paths to **S**

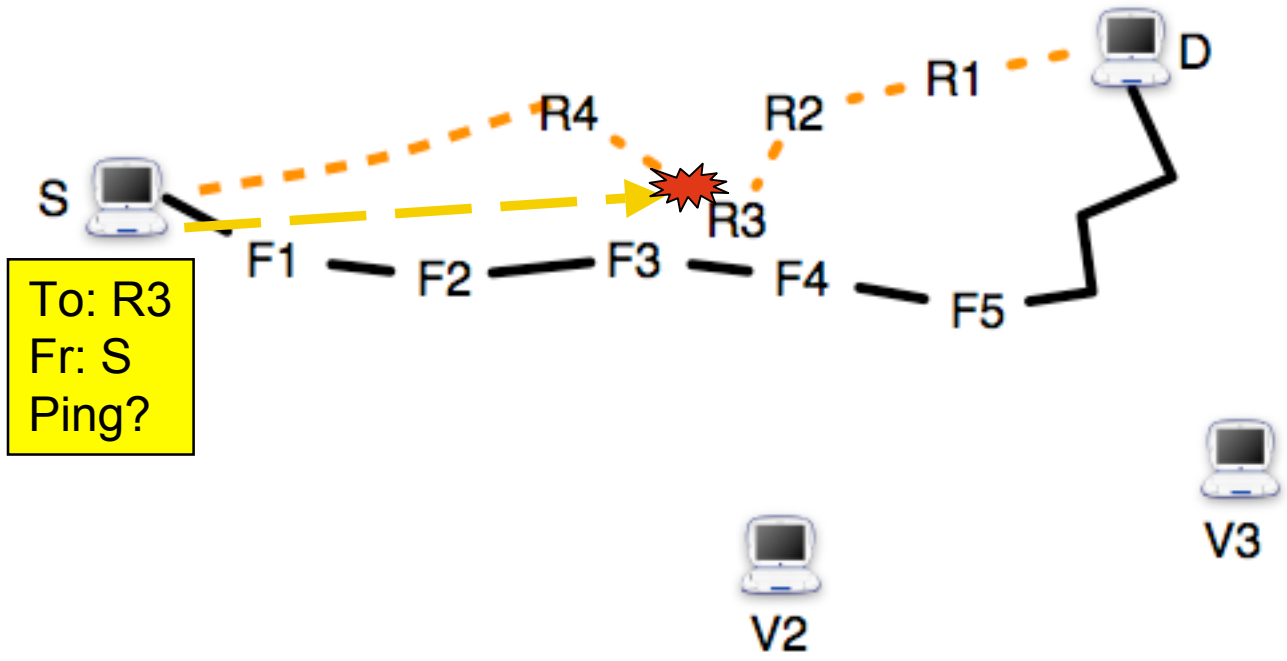
Techniques Applied to Unreachability



If ***D→S*** fails:

- **S** traceroutes, spoofing as vantage point that **D** can reach
- If pre-measured reverse traceroute predates failure, find farthest hop that can reach **S** and first that can't

Techniques Applied to Unreachability



If ***D*→*S*** fails:

- **S** traceroutes, spoofing as vantage point that **D** can reach
- If pre-measured reverse traceroute predates failure, find farthest hop that can reach **S** and first that can't