CSC 525:
Computer Networks

Resilient Overlay Networks

David G. Andersen

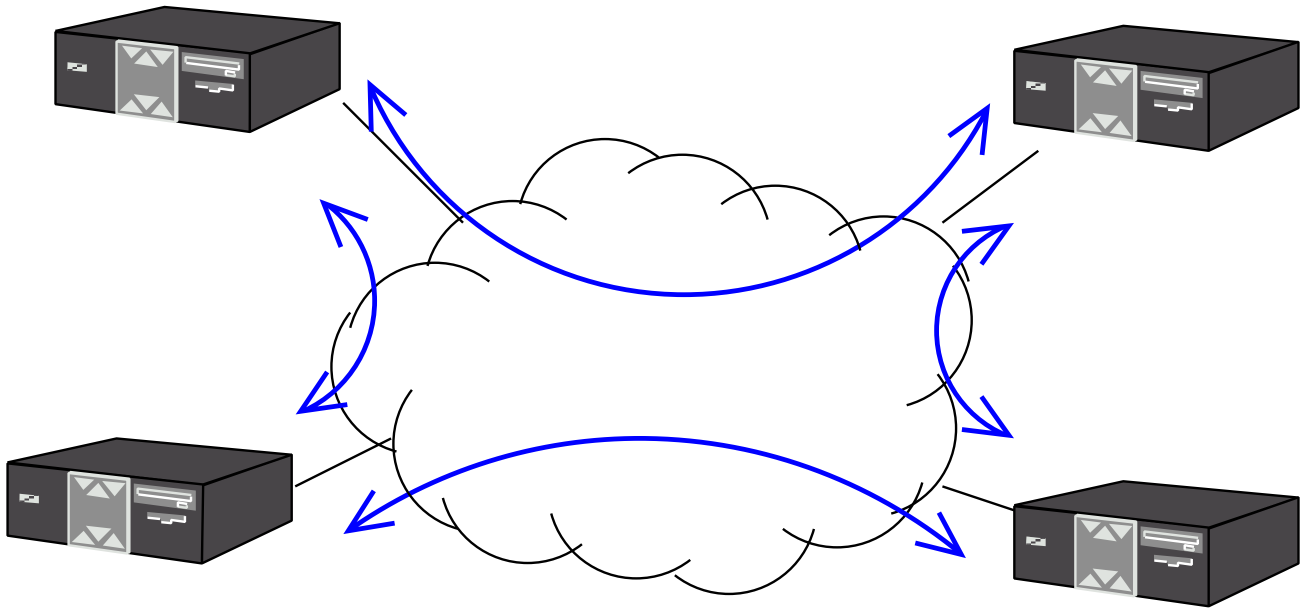
Hari Balakrishnan, M. Frans Kaashoek, Robert Morris

MIT Laboratory for Computer Science

October 2001

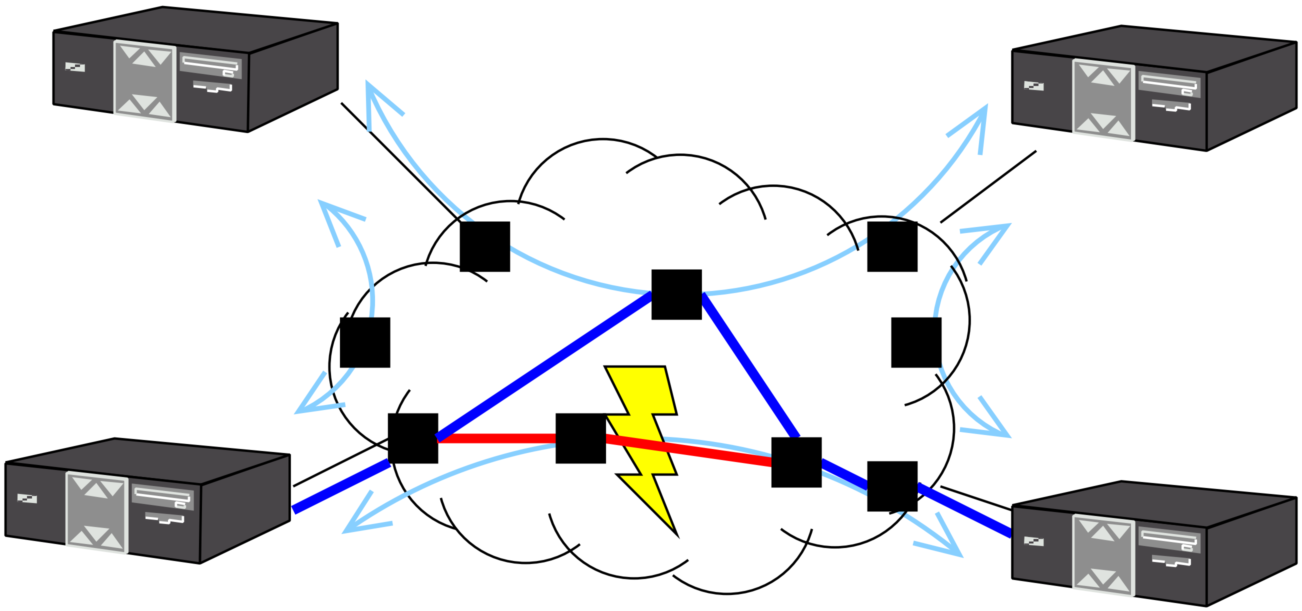
`http://nms.lcs.mit.edu/ron/`

The Internet Abstraction



- Any-to-any communication

The Internet Abstraction



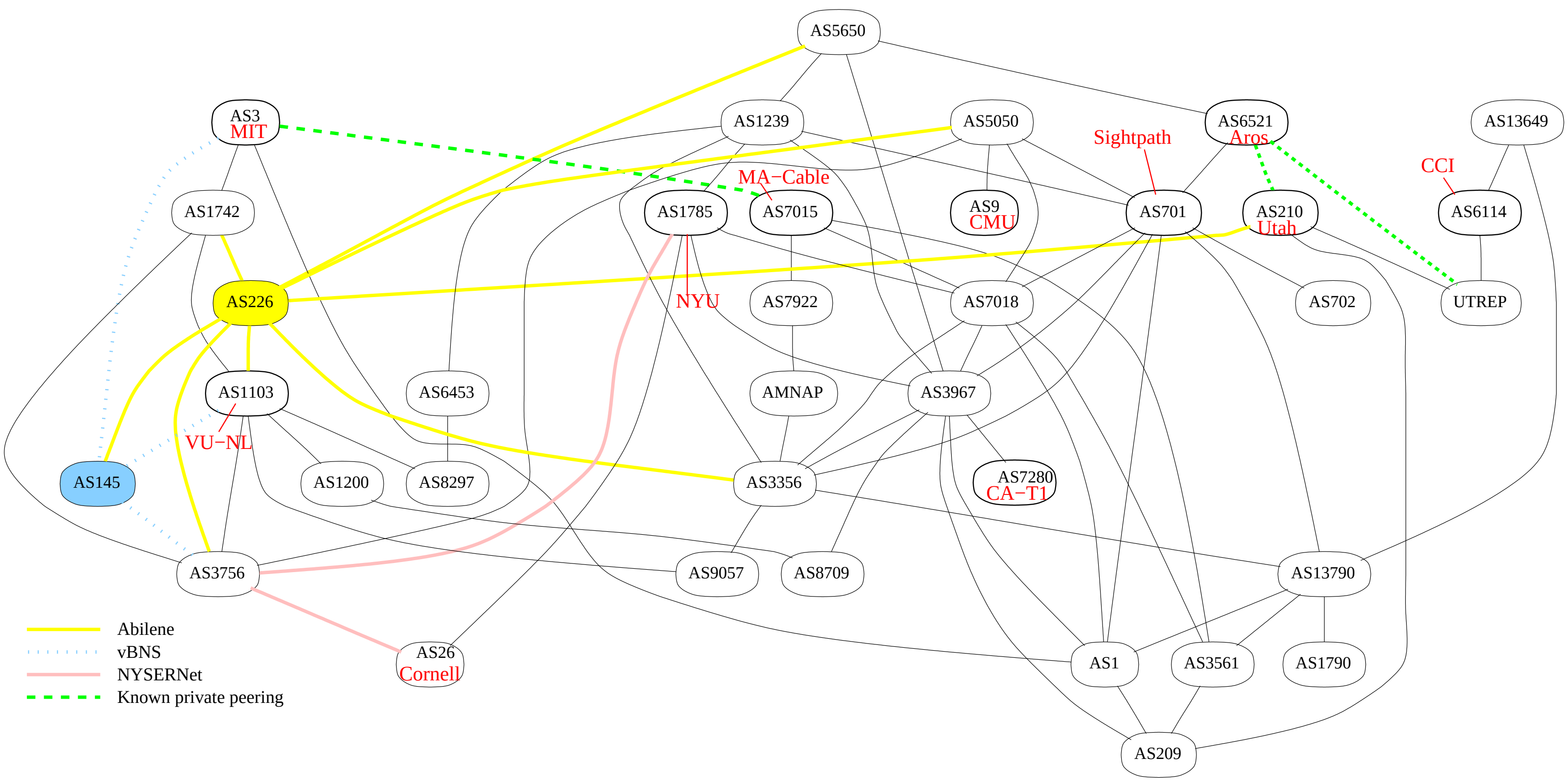
- Any-to-any communication
  transparently routing around failures

How Robust is Internet Routing?

Paxson 95-97	• 3.3% of routes had "serious problems"
Labovitz 97,00	• 10% of routes available $<$ 95% of time • 65% of routes available $<$ 99.9% of time • 3-min minimum detect+recover time; often 15 minutes • 40% of outages took 30+ mins to repair
Chandra 01	• 5% of faults $>$ 2.75 hours

The Internet *has* Redundancy

- Traceroute between 12 hosts,
  showing Autonomous Systems (AS's)



How Robust is Internet Routing?

- ✔ Scales well
- ✘ Suffers slow outage detection and recovery

Internet backbone routing also cannot:

- Detect badly performing paths
- Efficiently leverage redundant paths
- Multi-home small customers
- Express sophisticated routing policy / metrics

➔ We'd like to fix these shortcomings

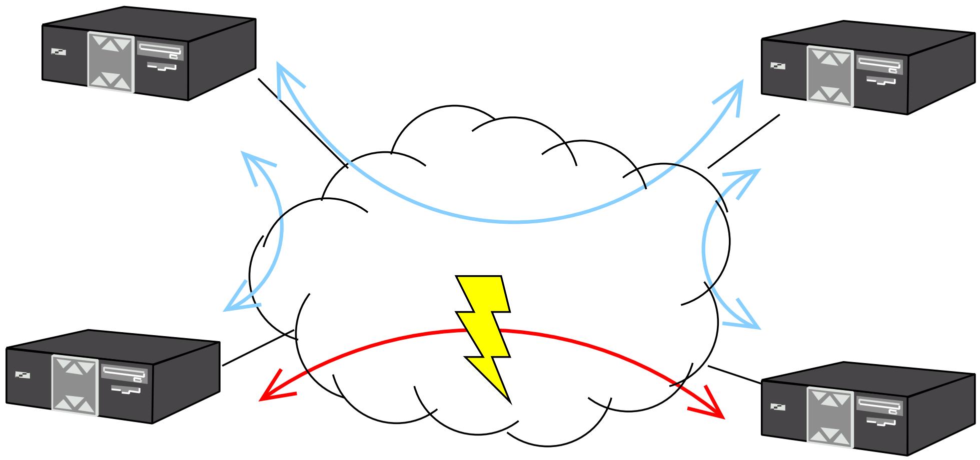
Goal

Improve communication availability
for small (3-50 node) communities:

- Collaboration and conferencing
- Virtual Private Networks (VPNs)
- 5 friends who want better service...

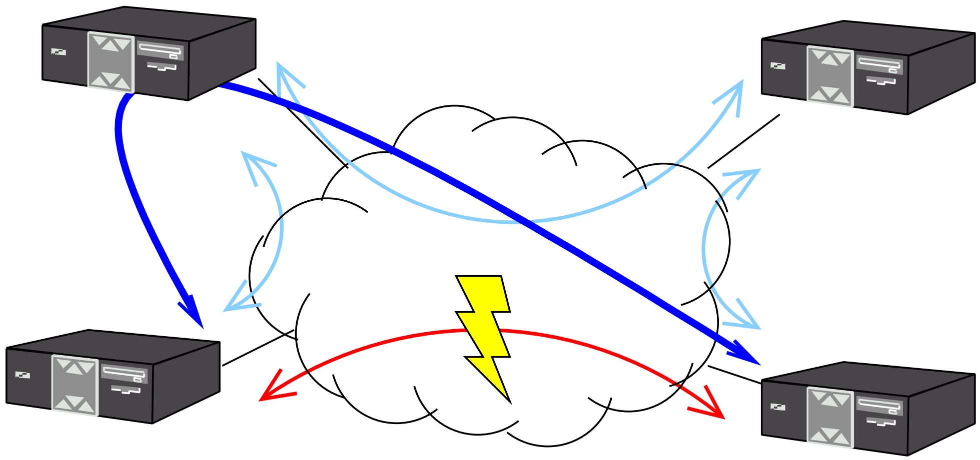
Interest in improving communication between *any* members of the community

RON: Routing around Internet Failures



The Internet takes a while to re-route

RON: Routing around Internet Failures



The Internet takes a while to re-route

... Cooperating hosts in different routing domains
can do better by re-routing through a peer node

Overlays

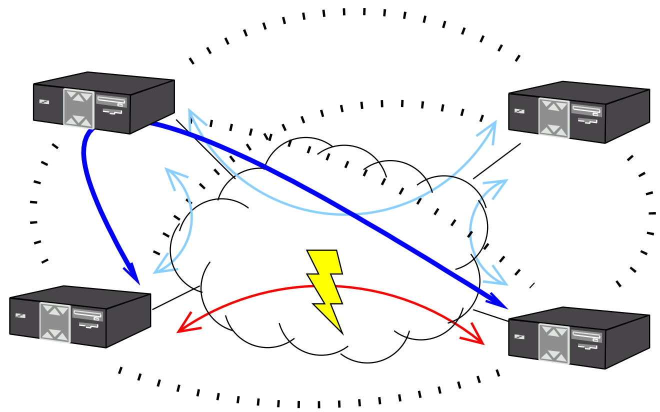
- Old idea in networks

✔ Easily deployed

✔ Lets Internet focus on scalability

✔ Keep functionality between *active* peers

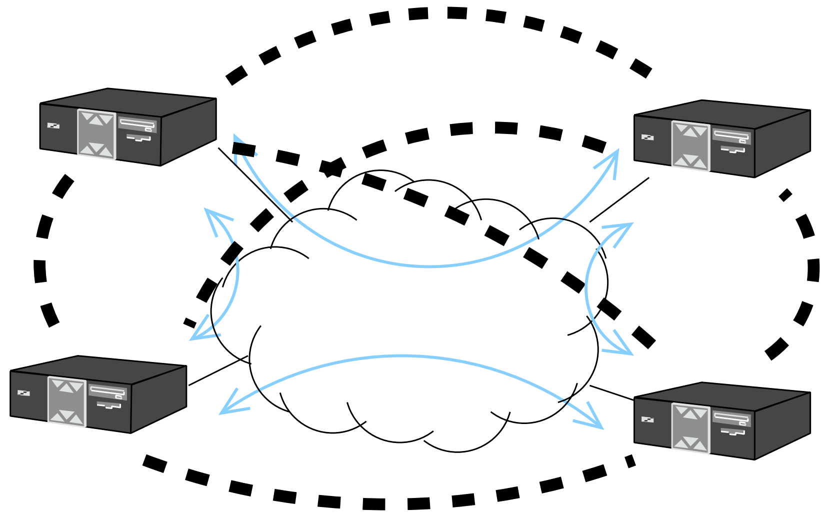
The Approach



Probes and Routing

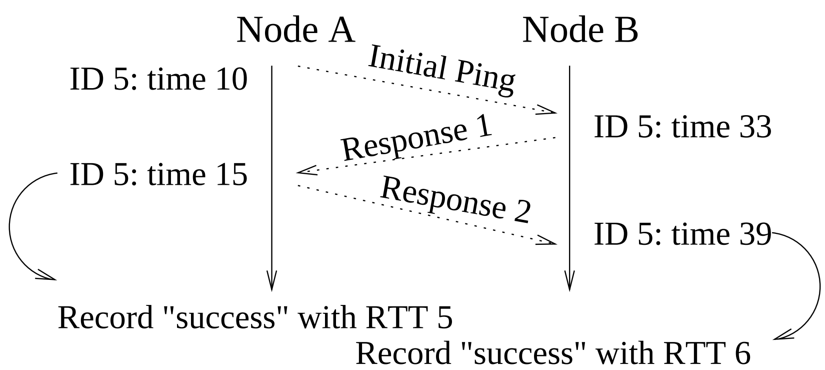
- Frequently measure *all* inter-node paths
- Exchange routing information
- Route along app-specific best path consistent with routing policy

Architecture: Probing



- Probe between nodes, determine path qualities
  - $O\left(N^2\right)$ probe traffic with active probes
  - Passive measurements

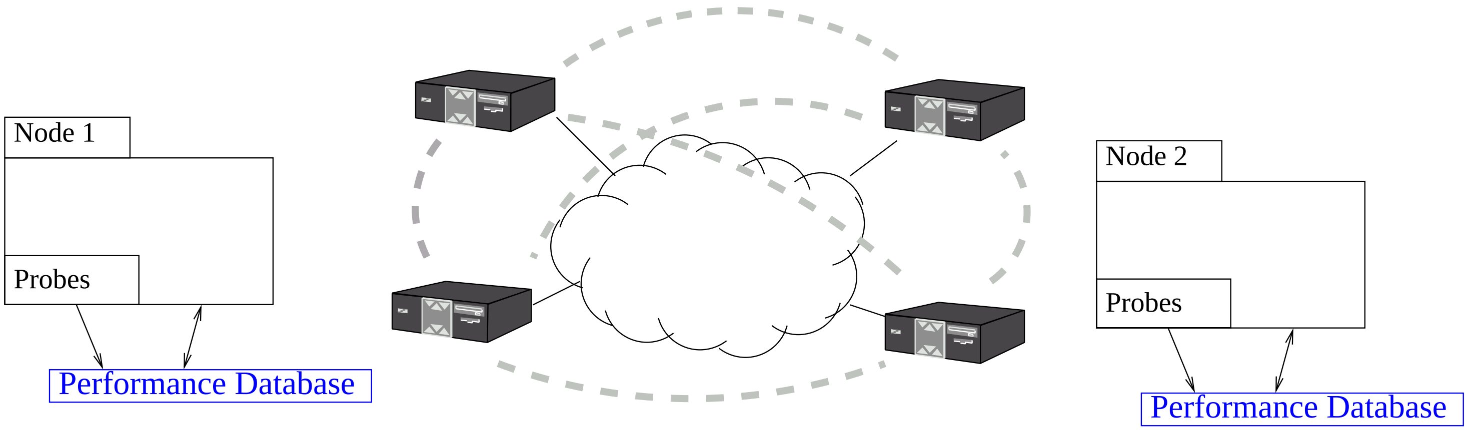
Probing and Outage Detection



- Probe every random(14) seconds
- 3 packets, both sides get RTT and reachability
- If "lost probe," send next immediately

  Timeout based on RTT and RTT variance
- If N lost probes, notify outage

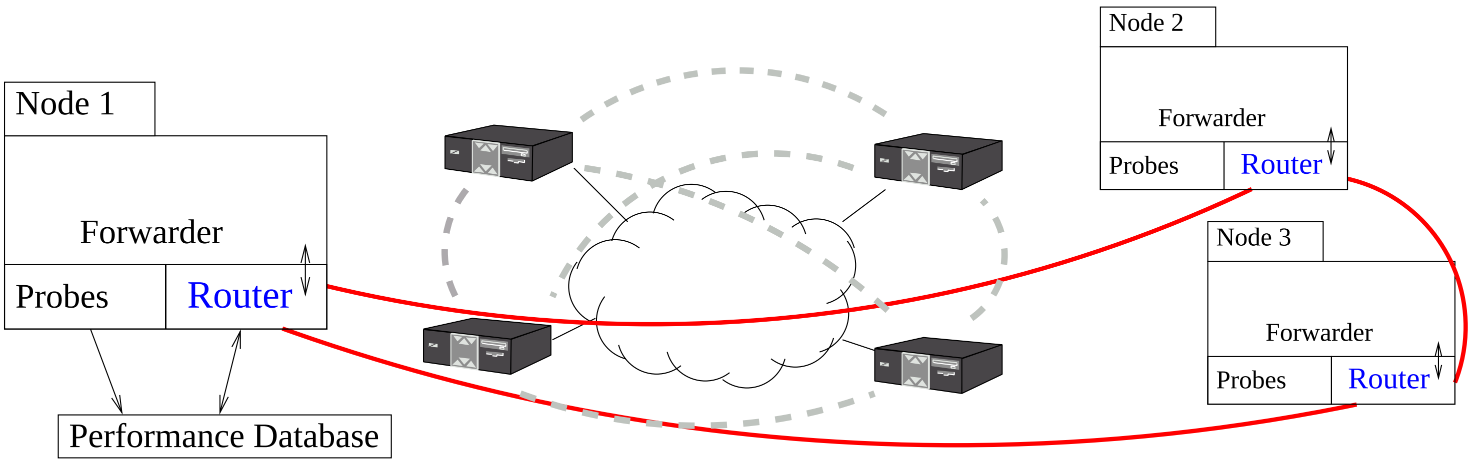
Architecture: Performance Database



- Probe between nodes, determine path qualities

➔ Store probe results in performance database

Architecture: Routing Protocol



- Probe between nodes, determine path qualities
- Store probe results in performance database

➔ Link-state routing protocol between nodes
Disseminates info using the overlay

Routing: Announcements

- Link-state announcements from perf. db
- Announce every 10-20 seconds
- Latency: EWMA with parameter .9:

$$lat_{avg} = 0.9 \times lat_{avg} + .1 \times new\_sample$$

- Loss: Average of last 100 samples
- Outage: Any success in last 4 probes

Routing: Predicting paths

Combine link metrics into a path estimate.

- Latency: $\sum L_1, L_2, ..., L_N$
- Loss: $\prod \rho_1, \rho_2, ..., \rho_N$
- Throughput: (TCP Throughput Equation)

$$score = \frac{\sqrt{1.5}}{rtt \cdot \sqrt{\rho}}$$

- Outage: Any outage anywhere?

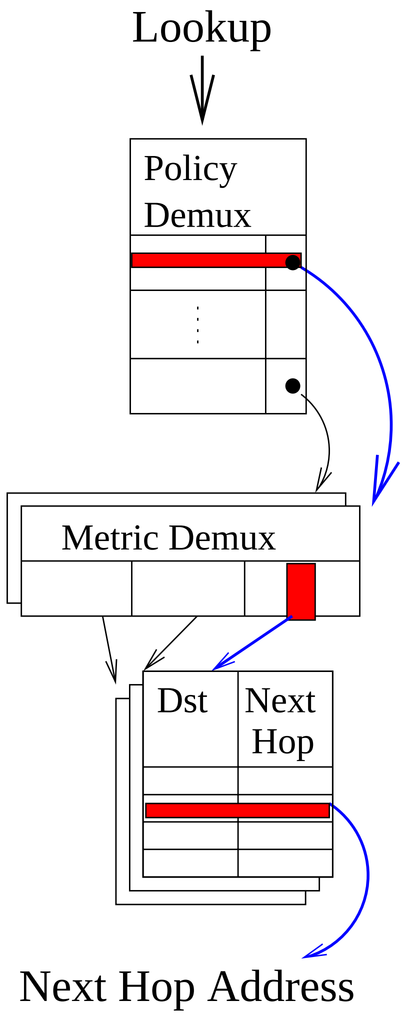
Routing: Building Forwarding Tables

Policy routing

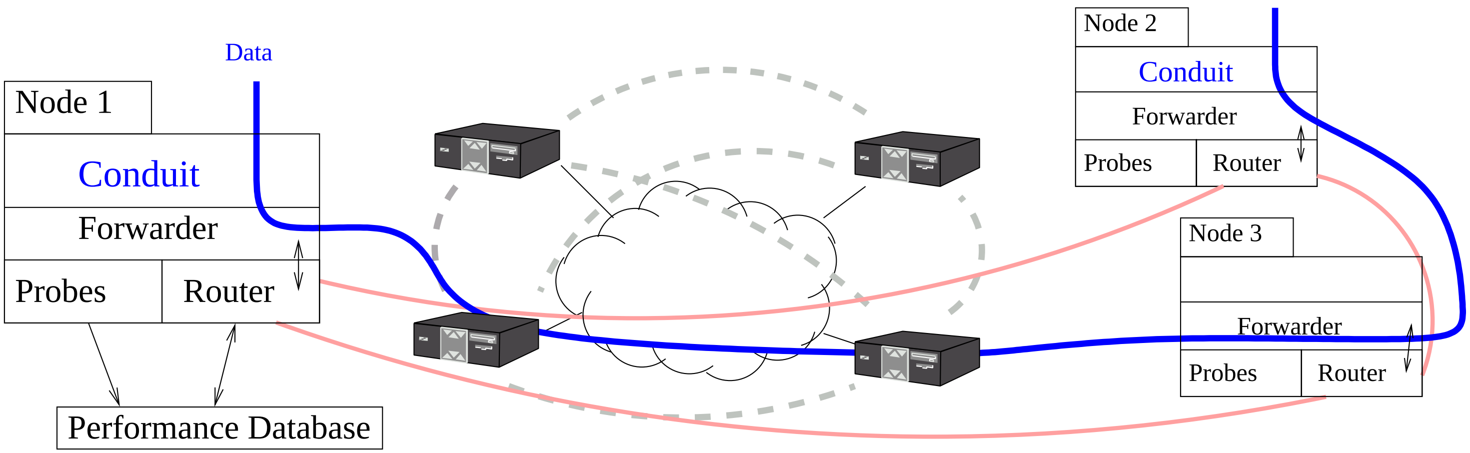
- Classify by policy
- Generate table per policy
- E.g. Internet2 Clique

Metric optimization

- App tags packets
  (e.g. "low latency")
- Generate one table per metric



Architecture: Forwarding



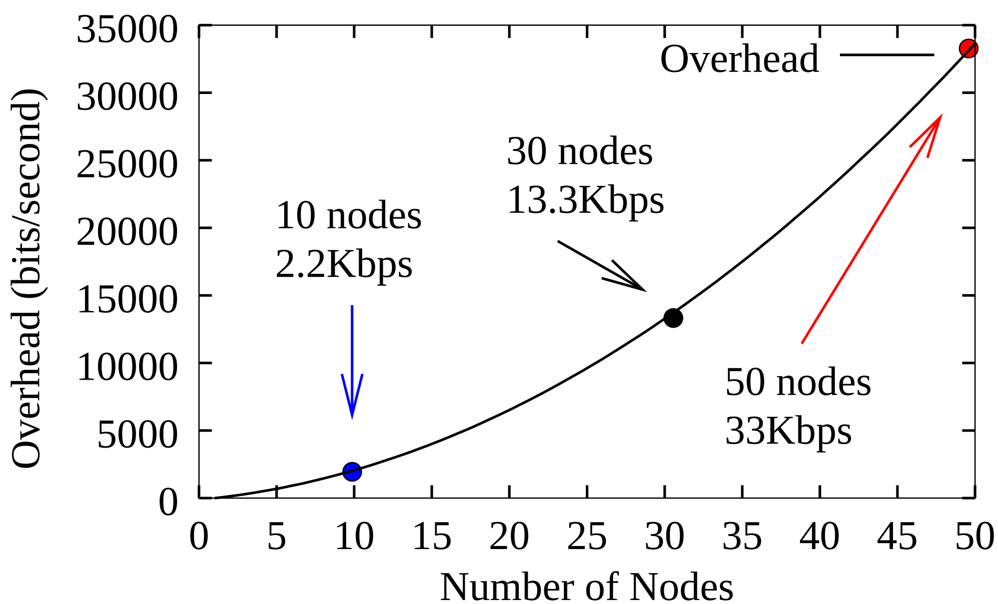
- Probe between nodes, determine path qualities
- Store probe results in performance database
- Link-state routing protocol between nodes
- ➜ Data handled by application-specific conduit

  Forwarded in UDP

Scaling

Routing and probing add packets:

Responsiveness vs. overhead vs. size



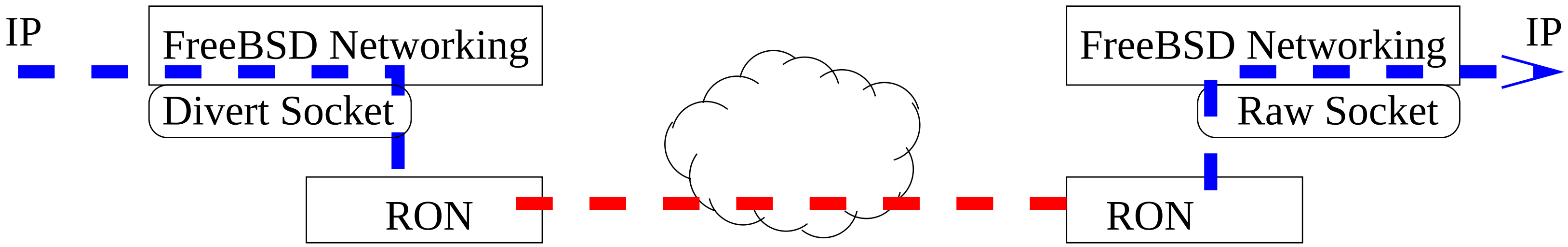
- ✘ 50 nodes is pushing the limit
- ✔ But is enough for *many* community apps

RON Clients and Applications

RON is a set of *libraries*...

... you have to build something with them.

Resilient IP Forwarder Client

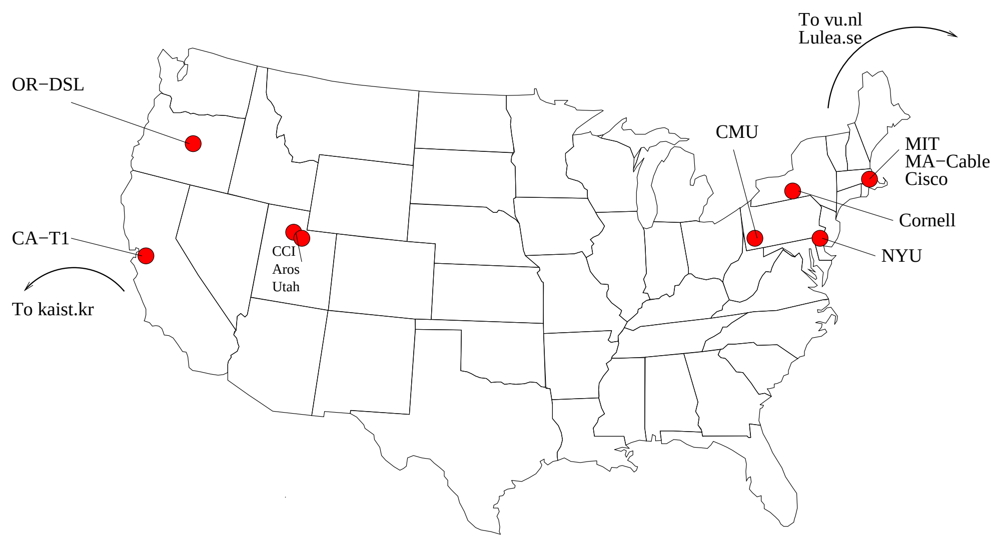


➔Transparent RON of any traffic

Evaluation

- Two datasets from Internet deployment (running for several months now)
- RON_1: 12 nodes, 64 hours, Mar 2001
- RON_2: 16 nodes, 85 hours, May 2001
- Compared RON-chosen paths to the Internet

Deployment



- 16 hosts in the US, Europe, and Asia.
  (A few more online now. Want a RON?)

- Variety of network types / bandwidths

- N^2 scaling of paths seen

Evaluation Methodology

- Loss & latency. Each node repeats:
  1. Pick random node j
  2. Pick a probe type ($direct, latency, loss$) round-robin. Send to j
  3. Delay for random interval
- Throughput: As above, but do 1M TCP bulk transfer to random host
- Record traceroutes for post-processing

Evaluation Details: Policy

- *Never, ever* use the Internet2 to improve life for a host not already connected to the Internet2

✘ Internet2 is high-speed, research-only net: atypically fast, uncongested, and reliable

➔ Implemented via RON's policy routing component

Major Results

- ✔ Probe-based outage detection effective
  - RON takes ~10s to route around failure
    Compared to BGP's several minutes
  - Many Internet outages are avoidable
  - RON often improves latency / loss / throughput [paper]
- ✔ Single-hop indirect routing works well
- ✔ Scaling is explicitly not our forte
  but big enough

RON_1 vs Internet 30 minute loss rates

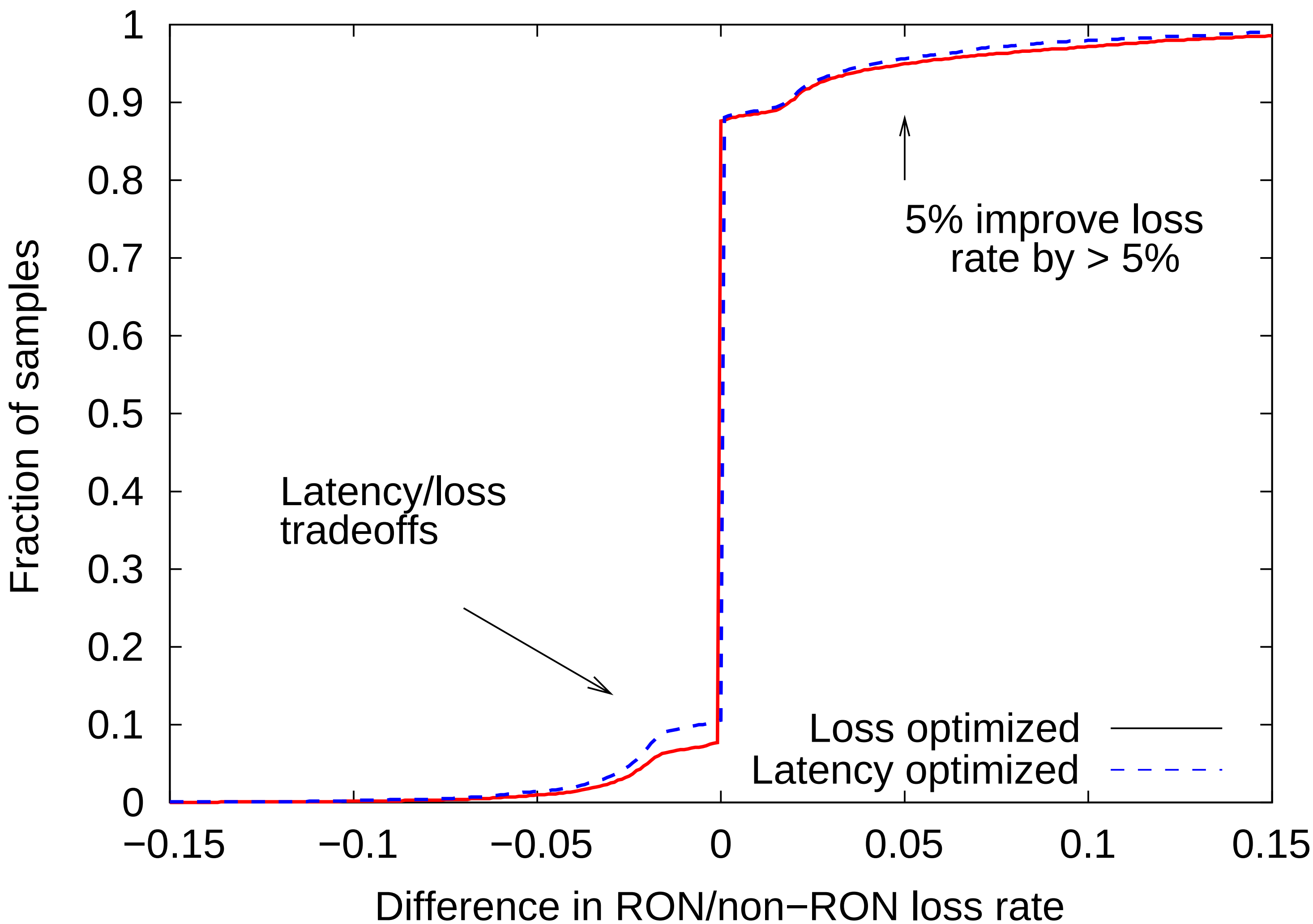
Internet Loss Rate

[90,100]	12									
[80,90)	2									
	1									
	3	1								
	1									
	3									
	8	1								
[20,30)	87	8	4							
[10,20)	362	32	12							
(0,10)	2188	44	3							
	(0,10)	[10,20)	[20,30)							

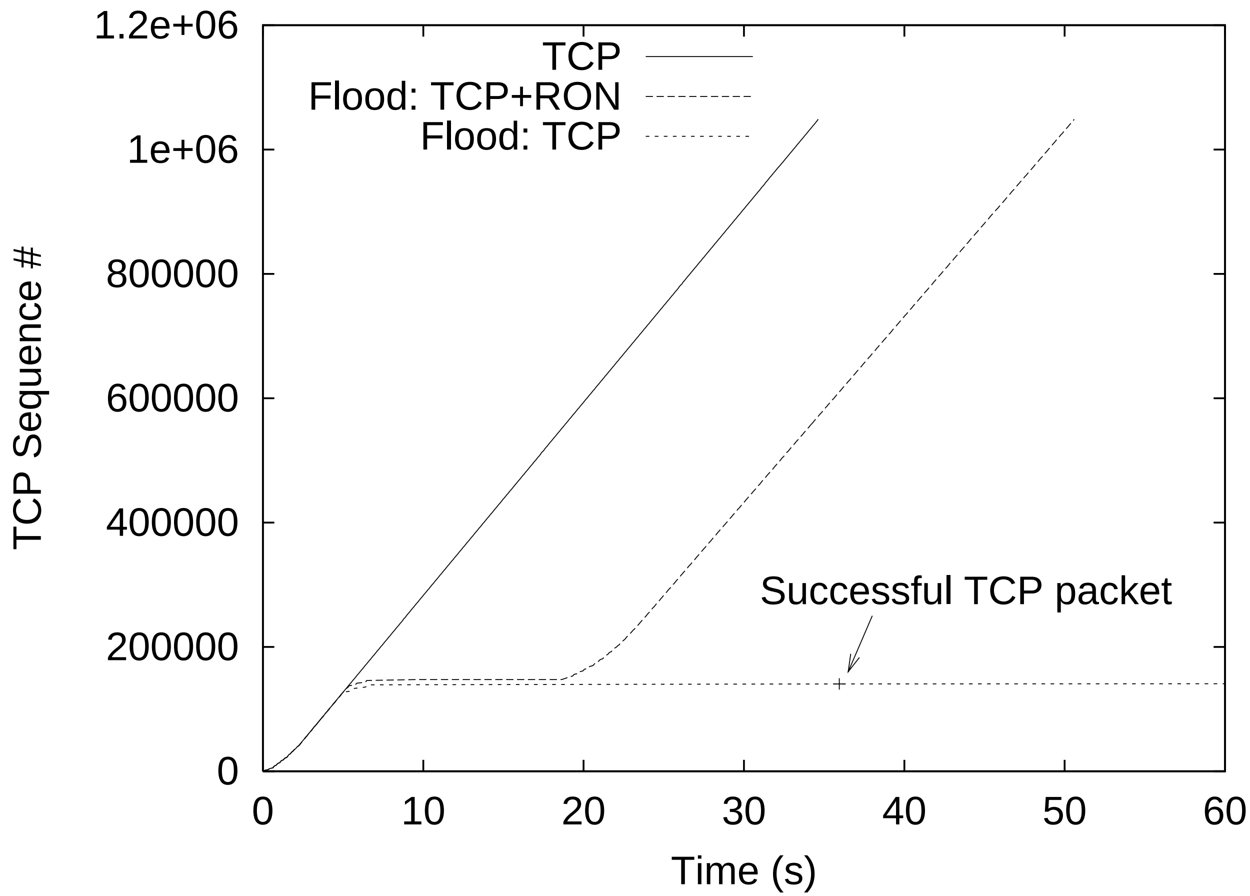
RON loss rate

- 6,825 “path hours” (13,650 samples)

RON_1 30 minute loss rate changes

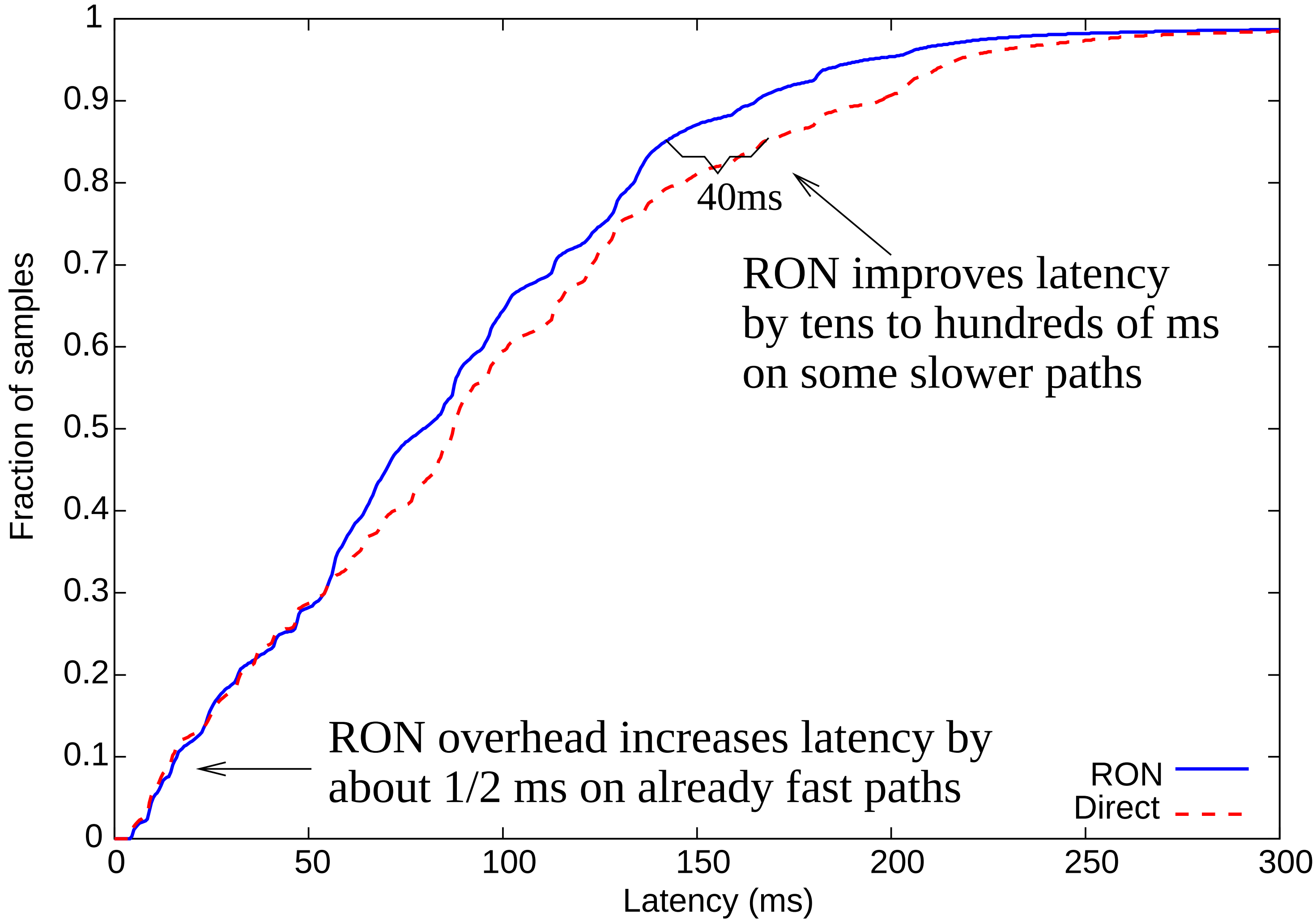


Outage Detection: Flooding Attack



BGP can’t handle this kind of problem...

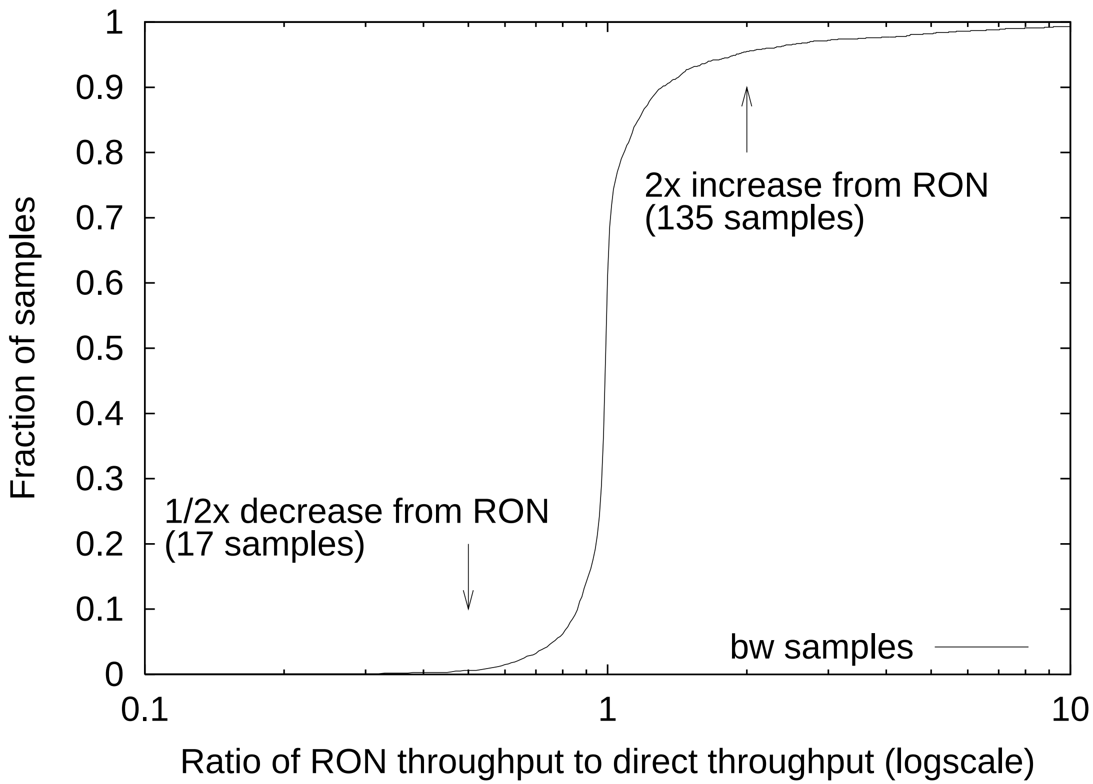
Performance: Latency



Performance: Throughput

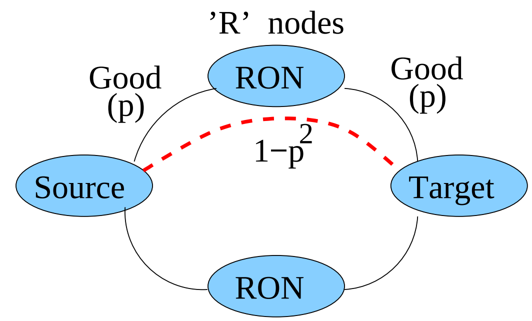
- 1% were 50% worse with RON
- 5% doubled throughput with RON
- Median unchanged: RON’s throughput optimizer looks only for big wins.

Performance: Throughput in RON_2



Single-hop Indirect Routing

$$\begin{aligned} P[\text{path good}] &= p \\ P[\text{indirect good}] &= p^2 \\ P[\text{indirect bad}] &= 1 - p^2 \end{aligned}$$



$$P[\text{At least one good}] = 1 - (1 - p^2)^{R+1}$$

Latency shortest paths from routing table dump:

- 48.8% direct paths best
- 99% best paths had 0 or 1 intermediate nodes

Policy

- RON supports flexible policies
  - Exclusive cliques (e.g. Internet2)
  - General policies (classifier + link set)
- RONs deployed between cooperating entities

  No “involuntary backdoors”
- Policy violations remain at the human level

How do users know *what* policy is?

More interesting future work.

Conclusions

- ✔ RONs improve packet delivery reliability
- ✔ Overlays attractive spot for resiliency: development, fewer nodes, simple substrate
- ✔ Single-hop indirection works well
- ✔ Small confederations respond quickly
- ➔ RON libraries are good platform for development, research

`http://nms.lcs.mit.edu/ron/`