Reliable routing

Brighten Godfrey
CS 538 March 13, 2017
The Internet is messy in practice

- Transient loops
- Persistent loops
- Asymmetry
- Instability

How to look inside a black box
Looking inside a black box

End-to-end measurement from vantage points combined with careful statistics

A standard for the field

- **End-to-End Effects of Internet Path Selection** [Savage ’99]
- **RON** [Anderson ’01]
- Related area: network tomography

Many resources now available

- PlanetLab, Seattle P2P testbed, RouteViews, DIMES, CAIDA, ...

### Related area:
- **RON** [Anderson ’01]
- **Selection** [Savage ’99]

### Table 1: Sites participating in the study

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adv</td>
<td>Advanced Network &amp; Services, Armonk, NY</td>
</tr>
<tr>
<td>austr</td>
<td>University of Melbourne, Australia</td>
</tr>
<tr>
<td>austz</td>
<td>University of Newcastle, Australia</td>
</tr>
<tr>
<td>batman</td>
<td>National Center for Atmospheric Research, Boulder, CO</td>
</tr>
<tr>
<td>bnl</td>
<td>Brookhaven National Lab, NY</td>
</tr>
<tr>
<td>bsd1</td>
<td>Berkeley Software Design, Colorado Springs, CO</td>
</tr>
<tr>
<td>connix</td>
<td>Caravela Software, Middlefield, CT</td>
</tr>
<tr>
<td>harv</td>
<td>Harvard University, Cambridge, MA</td>
</tr>
<tr>
<td>inria</td>
<td>INRIA, Sophia, France</td>
</tr>
<tr>
<td>korea</td>
<td>Pohang Institute of Science and Technology, South Korea</td>
</tr>
<tr>
<td>lbl</td>
<td>Lawrence Berkeley Lab, CA</td>
</tr>
<tr>
<td>lbl1</td>
<td>LBL computer connected via ISDN, CA</td>
</tr>
<tr>
<td>mid</td>
<td>MIDnet, Lincoln, NE</td>
</tr>
<tr>
<td>mit</td>
<td>Massachusetts Institute of Technology, Cambridge, MA</td>
</tr>
<tr>
<td>ncar</td>
<td>National Center for Atmospheric Research, Boulder, CO</td>
</tr>
<tr>
<td>near</td>
<td>NEARnet, Cambridge, Massachusetts</td>
</tr>
<tr>
<td>nrao</td>
<td>National Radio Astronomy Observatory, Charlottesville, VA</td>
</tr>
<tr>
<td>oce</td>
<td>Occ-van der Grinten, Venlo, The Netherlands</td>
</tr>
<tr>
<td>panix</td>
<td>Public Access Networks Corporation, New York, NY</td>
</tr>
<tr>
<td>pubnix</td>
<td>Pix Technologies Corp., Fairfax, VA</td>
</tr>
<tr>
<td>rain</td>
<td>RAINet, Portland, Oregon</td>
</tr>
<tr>
<td>sandia</td>
<td>Sandia National Lab, Livermore, CA</td>
</tr>
<tr>
<td>sdsc</td>
<td>San Diego Supercomputer Center, CA</td>
</tr>
<tr>
<td>sintef1</td>
<td>University of Trondheim, Norway</td>
</tr>
<tr>
<td>sintef2</td>
<td>University of Trondheim, Norway</td>
</tr>
<tr>
<td>srl</td>
<td>SRI International, Menlo Park, CA</td>
</tr>
<tr>
<td>ucl</td>
<td>University College, London, U.K.</td>
</tr>
<tr>
<td>ucla</td>
<td>University of California, Los Angeles</td>
</tr>
<tr>
<td>ucol</td>
<td>University of Colorado, Boulder</td>
</tr>
<tr>
<td>uke</td>
<td>University of Kent, Canterbury, U.K.</td>
</tr>
<tr>
<td>umann</td>
<td>University of Mannheim, Germany</td>
</tr>
<tr>
<td>umont</td>
<td>University of Montreal, Canada</td>
</tr>
<tr>
<td>unij</td>
<td>University of Nijmegen, The Netherlands</td>
</tr>
<tr>
<td>usc</td>
<td>University of Southern California, Los Angeles</td>
</tr>
<tr>
<td>ustutt</td>
<td>University of Stuttgart, Germany</td>
</tr>
<tr>
<td>wust1</td>
<td>Washington University, St. Louis, MO</td>
</tr>
<tr>
<td>xor</td>
<td>XOR Network Engineering, East Boulder, CO</td>
</tr>
</tbody>
</table>

[Traceroute and retrieval of its output) Almost all of the traceroutes exhibited persistent routing loops. See [Pa96] for details. When conducting the traceroute between 27 sites. We refer to this collection of the site, and its location. In the two experiments, between 5–8% of the measurement reports the same sequence of routers multiple times. For our analysis we distinguish between three types of loops: a pathloop in which packets forwarded by a router eventually return over which it has traversed. Thus, loops are apt to form when it is possible for a measurement to show the same sequence of routers multiple times. For instance, when tracerouting from University of Trondheim, Norway to University of California, Los Angeles only 5% of the measurements show the loop. The second experiment, between 33 sites. Both measurements show an even higher percentage of loops, provided all of the routers in the network share a consistent view of the present connectivity. Thus, loops are apt to form when there is a bias towards underestimating the virtual path. When conducting the traceroute between 27 sites. We refer to this collection of the site, and its location. In the two experiments, between 5–8% of the measurement reports the same sequence of routers multiple times. For our analysis we distinguish between three types of loops: a pathloop in which packets forwarded by a router eventually return over which it has traversed. Thus, loops are apt to form when it is possible for a measurement to show the same sequence of routers multiple times. For instance, when tracerouting from University of Trondheim, Norway to University of California, Los Angeles only 5% of the measurements show the loop. The second experiment, between 33 sites. Both measurements show an even higher percentage of loops, provided all of the routers in the network share a consistent view of the present connectivity. Thus, loops are apt to form when there is a bias towards underestimating the virtual path.
Network reliability in context

Physical layer reliability

Performance reliability / quality of service

Congestion and capacity planning

Management issues

• “The presence of persistent loops of durations on the order of hours is quite surprising, and suggests a lack of good tools for diagnosing network problems.” – Paxson

Basic routing reachability

• Does the routing protocol get packets from A to B?
Reliability problems in Internet routing

- Basic issue: controlling a distributed system => inconsistent state across routers => loops, black holes
- Also in link state, distance vector

Problem: control plane is slow...

- Control plane routing does eventually converge!
- But may take 100s of milliseconds; milliseconds possible after careful tuning of protocol timers

...and data plane is fast

- Sending 50 byte packet at 40 Gbps = 10 nanoseconds
Reliability in the data plane

Fast path (data plane) needs failure reaction!

Rest of this lecture: building a solution
Equal Cost Multipath (ECMP)

- Control plane produces not one next-hop, but many
- Next hops must be closer to destination (so no loops)
- Data plane sends packet to any next-hop that’s working
Provenance and Request Certificates and using them to es-
vice their provenance without a handshake.

4.2.1 Choosing a Provenance Verifier

The obvious implementation of the above exchange is the single UDP messagel This is sue
the PV's internal maximum time, perhaps a day, see further
discussion in [396].

How many next-hops in ECMP?
Equal Cost Multipath (ECMP)

- Control plane produces not one next-hop, but many
- Next hops must be closer to destination (so no loops)
- Data plane sends packet to any next-hop that’s working
- Defeated by even a single link failure in some cases

MPLS Fast Re-Route link protection

- Explicit backup path for each failure case (link or node failure)
Equal Cost Multipath (ECMP)

- Control plane produces not one next-hop, but many
- Next hops must be closer to destination (so no loops)
- Data plane sends packet to any next-hop that’s working
- Defeated by even a single link failure

MPLS Fast Re-Route link protection

- Explicit backup path for each link
- Protects against single failure scenario (shared risk link group)
- Uses more FIB entries
- Not shortest alternate path
Holy Grail: “Ideal connectivity”

- Data plane always correctly forwards packets towards destination, even with arbitrary link failures

Is it possible?

- Yes!
- BGP, OSPF, RIP, ISIS, ..., all have loops & black holes during convergence, ultimately causing packet loss
- But that is not fundamentally necessary!

5 minutes in small group: Devise a correct solution
Ideal connectivity

1. Every packet is eventually forwarded to destination correctly
   - Assume: arbitrary failures, but a path exists
   - Assume: no congestion or physical layer problems

2. Simple technique implementable in data plane
   - Feel free to play with packet header formats, protocols, etc.

5 minutes in small group: Devise a correct solution
Achieving ideal connectivity
The random walk

- If failure encountered, set a “random walk” bit in packet
- Whenever packet has random walk bit, send to random neighbor
- Slightly silly solution
Failure-carrying packets (FCP)

Approach

- Link state routing + link failure info carried inside packet
- Router recomputes shortest paths on the fly given new information inside packet

Key points

- Separate two functions: long-term topology distribution, handling transient changes
- Trick: carry topology updates in packet
- Demonstrates feasibility of ideal connectivity
Failure-carrying packets (FCP)

Approach
- Link state routing + link failure info carried inside packet
- Router recomputes shortest paths on the fly given new information inside packet

Key points
- Separate two functions: long-term topology distribution, handling transient changes
- Trick: carry topology updates in packet
- Demonstrates feasibility of ideal connectivity
Link reversal algorithms

Distributed algorithms for generating loop-free routes in networks with frequently changing topology

Gafni and Bertsekas
IEEE Trans. on Communications, 1981
Link reversal algorithms

Distributed algorithms for generating loop-free routes in networks with frequently changing topology

Gafni and Bertsekas
IEEE Trans. on Communications, 1981
For each destination: begin with a directed acyclic graph (DAG) where destination is the sole sink.
At each node:

- If ever all links point inward,
  - Reverse all links
At each node:

- If ever all links point inward,
  - Reverse all links
At each node:

- If ever all links point inward,
  - Reverse all links
At each node:

- If ever all links point inward,
  - Reverse all links
At each node:

- If ever all links point inward,
  - Reverse all links

Link reversal algorithms
At each node:

- If ever all links point inward,
  - Reverse all links
At each node:

- If ever all links point inward,
  - Reverse all links
At each node:

- If ever all links point inward,
  - Reverse all links

Dest.
At each node:

- If ever all links point inward, reverse all links.
At each node:

- If ever all links point inward, reverse all links

Whew! Done!

In the end, only one link flipped!
Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all

Let’s return to the beginning before convergence...
Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable.
- Thus the stable set eventually expands to include all.
Guaranteed to converge

Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Guaranteed to converge

Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Guaranteed to converge

Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all

![Diagram showing stable nodes and reversals](image-url)
Guaranteed to converge

Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Guaranteed to converge

Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Guaranteed to converge

Define stable node: no more reversals

- If node $x$ reverses adjacent to stable node $y$, then $x$ also becomes stable
- Thus the stable set eventually expands to include all
Guaranteed to converge

Define stable node: no more reversals

- If node \( x \) reverses adjacent to stable node \( y \), then \( x \) also becomes stable
- Thus the stable set eventually expands to include all

Whew! Done!
No: Protocol not yet suitable for the data plane

To reverse:

- Router must create new messages for each link
- Assumed these control messages arrive instantly and reliably
- Always have perfect information about distributed state!

Back where we started?
DDC: LR in the data plane

Key architectural point

- Make connectivity the job of the data plane
- Optimality (e.g. shortest paths) is still the job of the control plane

Problem

- LR requires sending control messages & distributed agreement on link direction – too slow for the data plane
DDC: LR in the data plane

Key algorithmic idea

- Allow stale info about link directions
- Each node can unilaterally reverse; notify neighbors later!
- Notification piggybacked on data packets using one-bit version number

Properties

- Strangely, this works...
- All events triggered by pkt arrival; no extra pkts created
- Simple bit manipulation operations
No guarantees on stretch, but empirically it’s good

Figure 3: CDF of steady state stretch for MPLS FRR and DDC in AS2914.
Take-aways

The Internet is messy and opaque

- Empirically, unreliability is common
- End-to-end measurements provide a view “inside the black box”

Highly reliable routing is possible

- requires failure response in the data plane
- single-failure protection practical, with backup paths
- surprisingly, ideal connectivity is achievable
Announcements

Wednesday

• Selfish agents

Watch for survey this week

• Select special topics
• General feedback

After spring break

• Project midterm presentations
• Course staff is here to help – book time with one of us!