Monitoring IP anycast using traceroute

R.P. (Adi) Aditya
aditya@grot.org

Disclaimer

As of January 6, 2014, I work for Microsoft.

All the work described in this presentation was done prior to that date.

Neither my current, nor former employers are responsible for any of the content included in this presentation.

All mistakes and other ideas contained are my own, and I make no guarantees or warranties about them.

IP Anycast routing for High-availability

How?

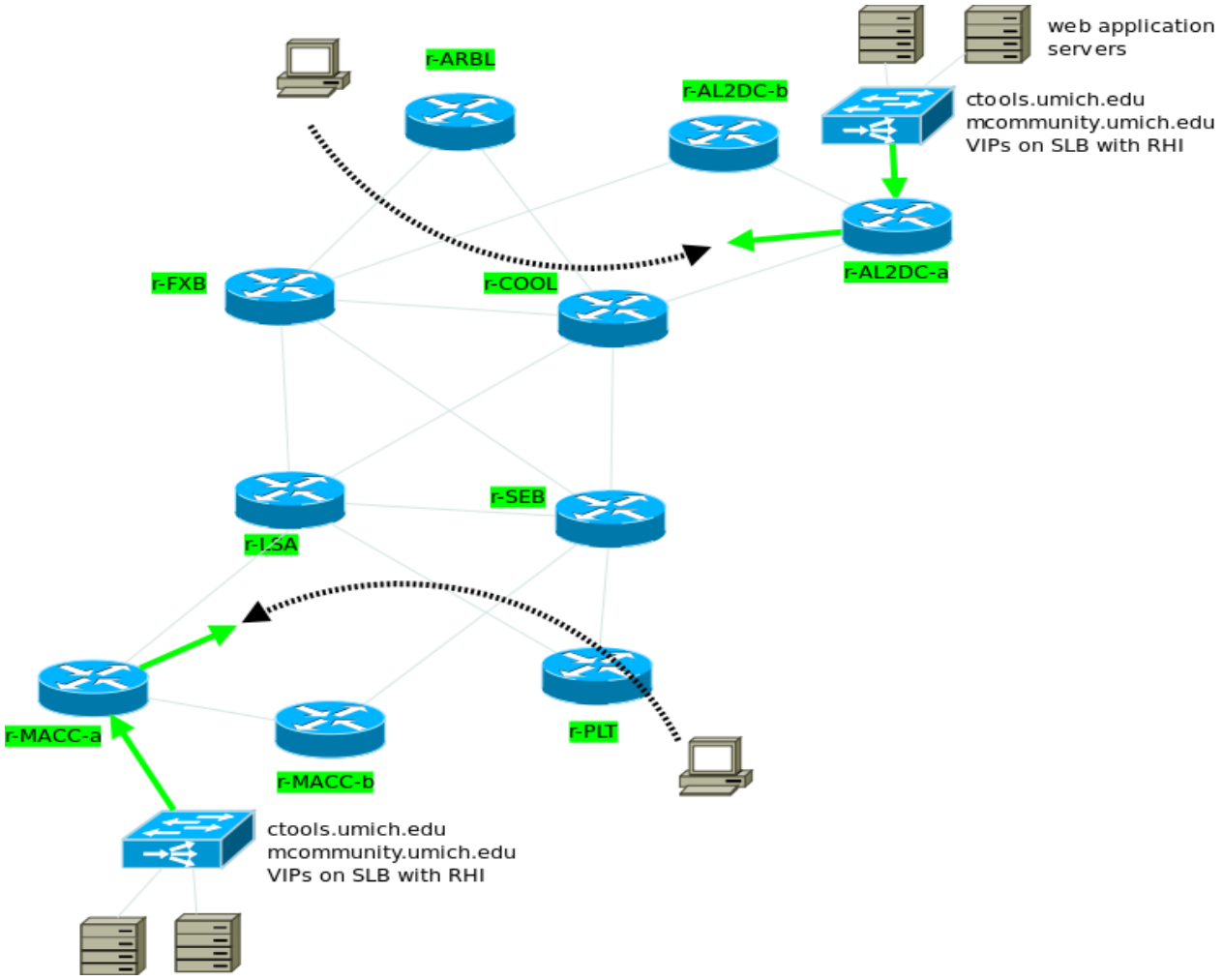
- Use routing protocol to advertise same IP address in multiple locations simultaneously.

Why IP Anycast routing for HA services?

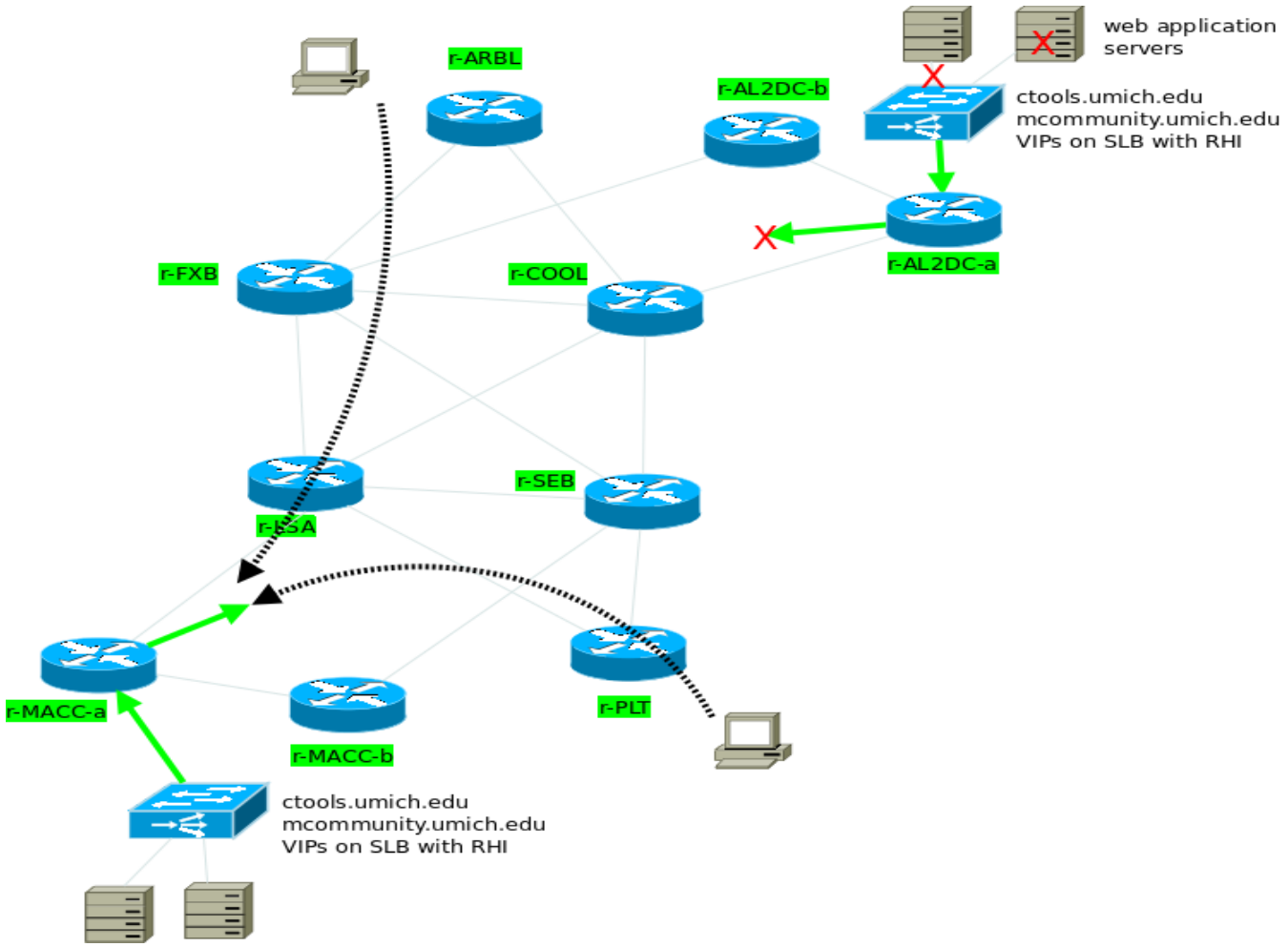
- Provides "shortest"-path routing (lowers latency)
- "transparent" failover in case of instance failure
- network partition resilience (if app/service is also partition tolerant)
- load-distribution

if done right!

Anycast auto-routes to "nearest" instance of app



Anycast auto-reroutes on failure of an app instances



- network anycast routing is worth deploying for your service if your network and routing are more stable than the server infrastructure and application
- typically used for short-lived, session-less services
- used to great effect for DNS recursive and auth service, both intra- and inter- domain
- used occasionally for inter-domain CDNs for HTTP, over TCP
- use it intra-domain for HTTP/S web applications for all the advantages it provides

Anycast Problems

- harder to debug if there are routing oscillations
- Coordinate data on multiple nodes/instances -- client might hit multiple versions
- generally more complexity, needs better operator training
- Need better tools to monitor, catch and fix problems - one example follows…

Tool Requirements

1. Tools to alert when service is unavailable are well-known (Nagios etc.)
2. Tool to monitor route changes are complex and expensive (mostly) and usually provide more noise than signal
3. Multiple tools needed to monitor anycast routing, including one that alerts when there is a forwarding path change to a node
4. Need a decent way to visualize the change and see behavior over time

One (simplistic) method to monitor forwarding path changes

This is a working tool, but with rough edges

- use periodic traceroute (well understood, platform and service independent)
- store results of periodic traceroutes in a stripped-down consistent format
- store the path (as in returned IP of each hop) in a text file
- store the text file in a VCS (I used RCS for simplicity)
- do a diff on each commit, and if there are any, then the forward path changed! alert/record that

```
1|198.111.227.33|177
2|198.111.225.10|177
3|198.111.225.4|177
4|192.122.183.77|237
5|198.108.22.137|237
6|198.108.23.12|237
7|206.223.119.64|0
8|64.50.232.69|4181
9|*|-1
10|134.215.208.90|4181
11|69.21.64.135|4181
12|*|-1
```

The simplified traceroute output is stored in a text file whose columns are:

1. hop
2. ip from which ttl exceeded msg sourced
3. AS (from traceroute -a or -A)

the per hop time is stored separately in a RRD

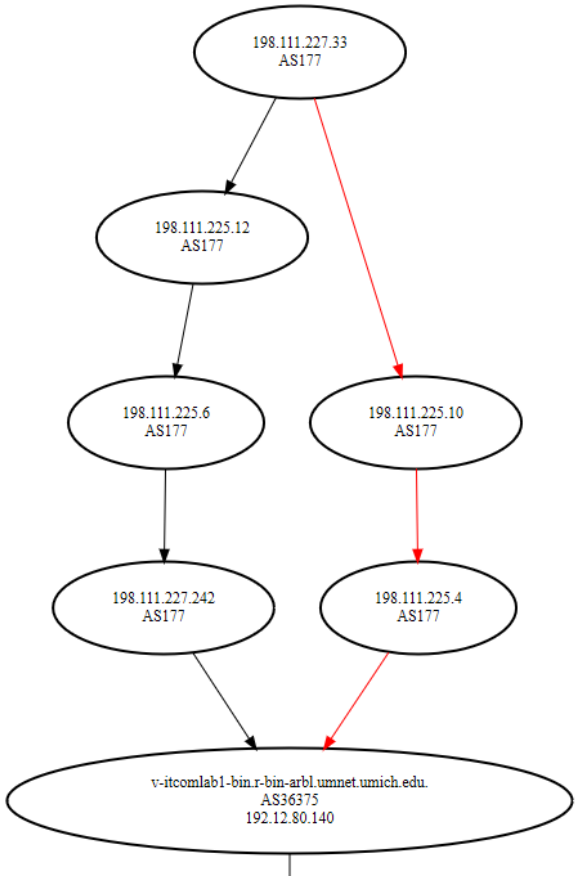
Diffs between committed versions simply show which hops have changed, been added or removed -- this textual format for the diff is hard to understand unless the path change is minimal

```
diff -r1.1928 -r1.1927
2,4c2,4
< 2|198.111.225.10|177
< 3|198.111.225.4|177
< 4|192.122.183.77|237
---
> 2|198.111.225.12|177
> 3|198.111.225.6|177
> 4|192.122.183.81|237
9c9
< 9|*|-1
---
> 9|64.50.232.102|4181
```

host traceroute path time

diff ctools.umich.edu---mu.ilab.umnet.umich.edu-141.211.48.10.txt

new	old	rev	date	
◉	○	1.468	04/29/2014 17:17:34	1398806254
○	◉	1.467	04/28/2014 19:17:36	1398727056
○	○	1.466	03/23/2014 07:17:36	1395573456
○	○	1.465	03/23/2014 06:17:33	1395569853



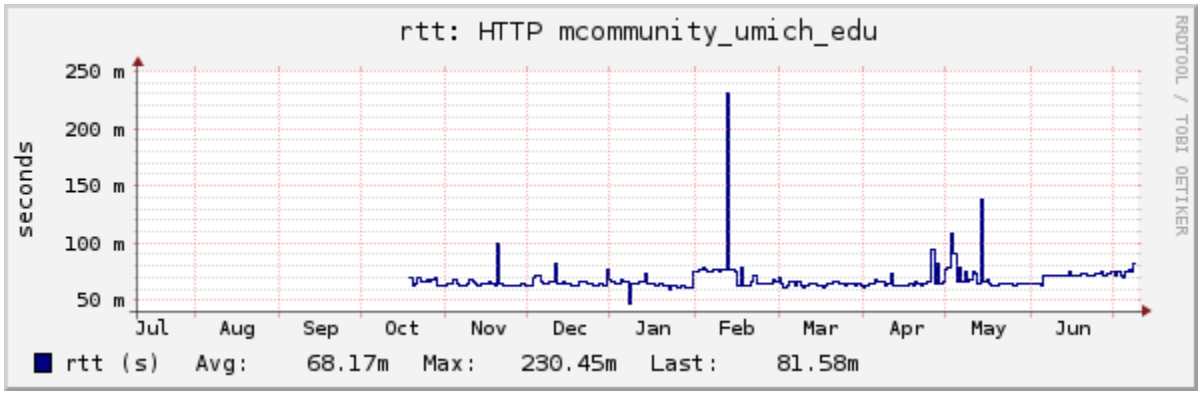
Display the VCS “commit log” of traceroute changes visually

- Much easier to see the route change “visually” when comparing traceroutes at two different times
- use a simple perl CGI to display commit dates/revisions
- use Graphviz to show route changes (red is old path)
- works for both IPv4 and IPv6
- gets confused if no ttl exceeded msg is returned resulting in “*” in traceroute
- also ECMP (depending on hashing algo) will confuse and yield false positives

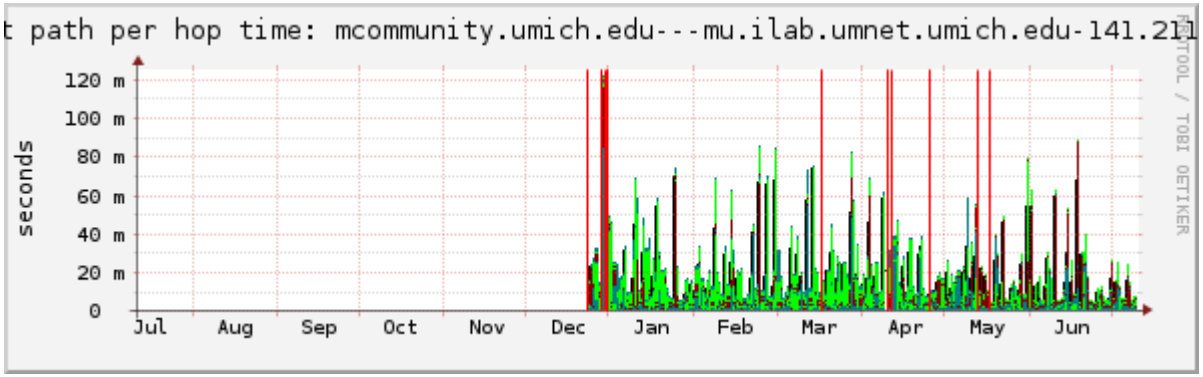
Use common tools to visualize changes over time

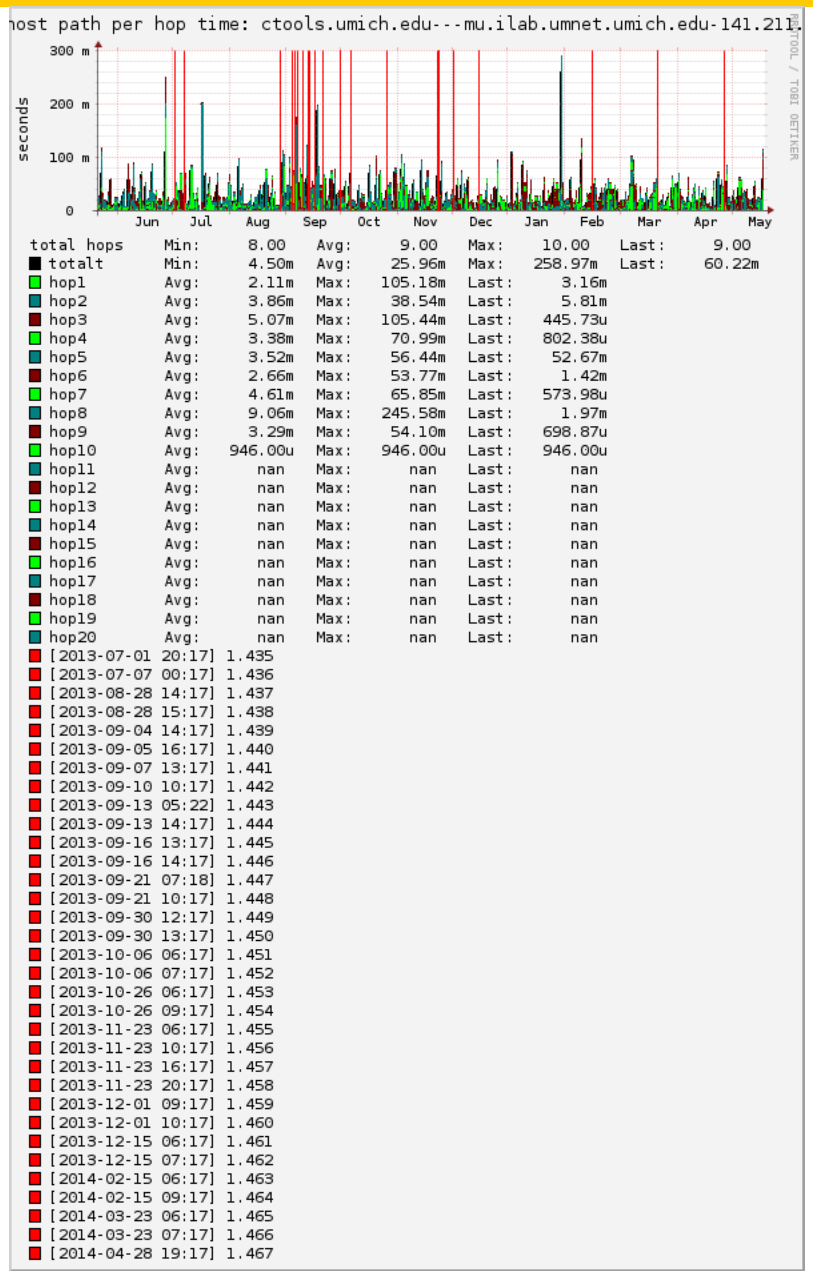
- depends on drraw -- http://web.taranis.org/drraw/
- drraw allows display of RRDs in flexible ways
- store route changes in “event files” to record when a diff was seen
- also store timing for each hop in RRDs
- show it all together for visual reference

An Enterprise web directory HTTP rtt on average from service monitoring tool (checks HTTP response time and content correctness)



and compare to traceroute per-hop RRDs + event files: red vertical lines are forwarding path switches, almost all due to planned maintenance





- this graph is for a year's worth of path switches -- pretty useful to see when the tool noticed (based on hourly samples), especially if correlated to known maint and/or outages
- if routing tables are recorded over the same period, the effect of routing changes on forwarding path can be seen
- and most useful if correlated to service (DNS, HTTP probes etc.) monitoring time series

Next steps?

- distributed monitoring
- use RIPE Atlas probes?
- more granular sampling periods (1 hour now)
- trigger traceroute based on service check
- Example installation http://test-http.ilab.umnet.umich.edu/drraw/tr/
- Example code repository at https://github.com/rpaditya/trmon

Acknowledgements

- Thanks to the University of Michigan, Ann Arbor for providing the network on which this was tested and the server on which the demo is hosted