FANTAIL:
Facilitating Advances in Network Topology Analysis

Young Hyun
CAIDA

Advisory Committee Meeting
Oct 8, 2020
Architecture

FANTAIL

client

frontend server
(interactive web interface and web API)

Elasticsearch
(storage & search)

Apache Spark
(data processing & analysis)
Querying

1. High-level query

2. Low-level queries

3. Matching traces (JSON)

- User specifies high-level search criteria
- FANTAIL performs low-level Elasticsearch queries against relevant indexes and traceroute fields
• User specifies high-level search criteria + desired data processing/analyses to apply to matching traces
<table>
<thead>
<tr>
<th>Query</th>
<th>Selection Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>vp</strong> $V$</td>
<td>vantage point is $V$</td>
</tr>
<tr>
<td><strong>vp_as</strong> $N$</td>
<td>vantage point is located in autonomous system (AS) $N$</td>
</tr>
<tr>
<td><strong>vp_country</strong> $C$</td>
<td>vantage point is located in country $C$</td>
</tr>
<tr>
<td><strong>vp_type</strong> $T$</td>
<td>vantage point is hosted by an organization of type $T$</td>
</tr>
<tr>
<td><strong>status</strong> $N$</td>
<td>traceroute has success/failure code $N$</td>
</tr>
<tr>
<td><strong>timestamp</strong> $op$ $N$</td>
<td>traceroute has timestamp $op$ $N$</td>
</tr>
<tr>
<td><strong>dest_rtt</strong> $op$ $N$</td>
<td>RTT of traceroute destination is $op$ $N$ ms</td>
</tr>
<tr>
<td><strong>pathlen</strong> $op$ $N$</td>
<td>length of traceroute path is $op$ $N$</td>
</tr>
<tr>
<td><strong>has_mpls</strong> $T/F$</td>
<td>whether there is ($T$) or is not ($F$) MPLS in the traceroute path</td>
</tr>
</tbody>
</table>

$op$ is $<$ or $=$ or $>$
<table>
<thead>
<tr>
<th>Query</th>
<th>Selection Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>dest</strong> $G$</td>
<td>traceroute destination is any address $x \in G$</td>
</tr>
<tr>
<td><strong>hop</strong> $G$</td>
<td>traceroutes with any address $x \in G$ appearing at any hop</td>
</tr>
<tr>
<td><strong>neigh</strong> $G_1 \ldots G_n$</td>
<td>traceroutes with $n$ distinct neighboring hop addresses $x_i \in G_i$</td>
</tr>
</tbody>
</table>

$T =$ target address/prefix/AS/country

$G =$ target group $T_1 | \ldots | T_m$

**query:** neigh $10.0.0.0/8 | 192.168.0.0/16$ $\text{AS}1 | \text{AS}2 | \text{AS}3$

matches any trace with hop addresses $x$ and $y$ such that

$$(x \in 10.0.0.0/8 \text{ or } x \in 192.168.0.0/16)$$

and $(y \in \text{AS}1 \text{ or } y \in \text{AS}2 \text{ or } y \in \text{AS}3)$
Year 1 results

- implemented nearly all queries (except has_mpls)
- implemented 4 analysis modules
  - hop-addrs, ip-links, ip-paths, ip-rtts
- imported most of 2016-2020 team-probing traces into Elasticsearch
  - 45 billion traces; 32.5 TB as stored in Elasticsearch
- implemented interactive web interface for executing queries and data processing
Year 2 deliverables

- acquire and deploy cluster to host Elasticsearch
- support annotating traces with DNS, IXP, bdrmapIT, and TNT data
- implement all remaining analysis modules
- implement web API
  - perform queries and execute data processing pipeline
- finish interactive web interface
  - execute data processing pipeline
• hop addresses
• IP links
• IP paths
• IP-RTT distributions
IP links

- extract unique IP links (direct and indirect) from matching traces
- output format:

  link count

  link:
  
  \( A = B \)  for adjacent addresses \( A \) and \( B \)
  
  \( A-n-B \)  for addresses \( A \) and \( B \) separated by \( n \) non-responding hops

  count:  number of traces where a given link appeared

- responding destination address prefixed with "D"

  129.122.31.254 = 129.122.0.5
  212.187.195.161-2-4.68.72.254
  196.201.62.221 = D 109.27.101.45
IP paths

- extract unique IP paths from matching traces
- output format:

  **path count**

  **path:** a sequence of hops separated by "=" or "-n-
  
  **count:** number of traces with the given path

- responding destination address prefixed with "D"
- multiple responding addresses at hop separated by commas

129.122.31.254=129.122.0.5=41.189.172.33,196.201.62.220
=196.201.62.221=212.187.195.161-2-4.68.72.254
=D109.27.101.45 1
represents the path:

129.122.31.254
129.122.0.5
41.189.172.33, 196.201.62.220 (two addresses at this hop)
196.201.62.221
212.187.195.161
*
* 
4.68.72.254
109.27.101.45 (responding destination)
IP-RTT distributions

- calculate min, max, avg, stddev, and percentiles (25th, 50th, 75th, and 95th) of RTTs for each IP hop/destination per monitor

- output format (CSV):

```
vp=addr, count, min, max, avg, stddev, 25th, 50th, 75th, 95th
```

- `vp`: name of vantage point
- `addr`: hop/destination address
- `count`: number of RTT samples for the given `vp=addr`

```
```
Discussion

- data processing results download page
  - show file size, count of unique objects, ... what else?

- recent vs. old data
  - are recent years of data the most useful?
  - keep older data in Elasticsearch "frozen index"; slower to access

- annotating traces with DNS, IXP, bdrmapIT, and TNT data
  - how will people use?
  - how to deal with sparseness of these auxiliary data?

- suggestions for analysis modules?
Thanks!

- will announce demo FANTAIL account for advisory committee to try out FANTAIL

- email fantail-info@caida.org if interested in personal FANTAIL account for long-term use