Learning to Extract Geographic Information from Internet Router Hostnames

Matthew Luckie
Zachary Bischof

Bradley Huffaker
Marianne Fletcher

Alexander Marder
k claffy

CoNEXT 2021

University of Waikato

CAIDA / UC San Diego
Motivation: Where are these routers located?

Router #1
- 154.54.9.6
- 173.205.55.118
- 206.111.0.201

Router #2
- 154.54.12.54
- 129.250.193.162
- 64.125.14.239

Router #3
- 109.200.218.13
- 4.14.228.118
- 168.143.105.162

Router #4
- 216.66.14.186
- 4.69.219.110
- 195.22.206.99
Motivation: Where are these routers located?

One approach: delay measurements.

System is located within distance implied by observed RTT from a known location.
Motivation: Where are these routers located?

Router #1
- 4ms from iad (Dulles, VA, US)
- 154.54.9.6
- 173.205.55.118
- 206.111.0.201
- ~400km
- Range implied by RTT

Router #2
- 3ms from iad (Dulles, VA, US)
- 154.54.12.54
- 129.250.193.162
- 64.125.14.239
- ~300km
- Range implied by RTT

Router #3
- 3ms from cgs (College Park, MD, US)
- 109.200.218.13
- 4.14.228.118
- 168.143.105.162
- ~300km
- Range implied by RTT

Router #4
- 4ms from cgs (College Park, MD, US)
- 216.66.14.186
- 4.69.219.110
- 195.22.206.99
- ~400km
- Range implied by RTT
Motivation: Where are these routers located?

<table>
<thead>
<tr>
<th>Router #1</th>
<th>4ms from iad (Dulles, VA, US)</th>
<th>154.54.9.6</th>
<th>173.205.55.118</th>
<th>206.111.0.201</th>
<th>~400km</th>
<th>No further than 300-400km from systems in the Washington D.C. area.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Router #4</td>
<td>4ms from cgs (College Park, MD, US)</td>
<td>216.66.14.186</td>
<td>4.69.219.110</td>
<td>195.22.206.99</td>
<td>~400km</td>
<td></td>
</tr>
</tbody>
</table>
Substantial work requires accurate router geolocation

Internet resilience to natural disasters, optimality of paths, etc.

**iPlane: An Information Plane for Distributed Services**
Uncovering Performance Differences among Backbone ISPs with Netdiff

**Nation-State Routing: Censorship, Wiretapping, and BGP**
Detecting Traffic Differentiation in Backbone ISPs with NetPolice

**A DISTRIBUTED SYSTEM FOR LARGE-SCALE GEOLOCATION OF INTERNET HOSTS**
Effective Diagnosis of Routing Disruptions from End Systems

**Residential Links Under the Weather**

**Solar Superstorms: Planning for an Internet Apocalypse**

**On inferring regional AS topologies**

**Measuring and Evaluating Large-Scale CDNs**
Packet Caches on Routers: The Implications of Universal Redundant Traffic Elimination

**Geography and Routing in the Internet**

**Out of Sight, Not Out of Mind - A User-View on the Criticality of the Submarine Cable Network**

**Geographic Locality of IP Prefixes**

Selection of work including SIGCOMM papers published in 2019 and 2021
Intuition: Naming Conventions

Hostnames suggest Washington D.C. area.

Goal: build a system that learns conventions that each operator uses to encode geohints.
Challenge: operators use different conventions

Operators can choose their own convention.

We need to accommodate them all.

Our method inferred conventions for
1023 suffixes w/ IPv4 routers
241 suffixes w/ IPv6 routers
Challenge: operators use different dictionaries

Router #1
- xo.iad02.atlas.cogentco.com
- as2828.was14.ip4.gtt.net
- te9-2-0d0.cir1.ashburn-va.us.xo.net

Router #2
- vodafone.iad02.atlas.cogentco.com
- ae-0.vodafone.asbnva02.us.bb.gin.ntt.net
- zayo.vodafone.er2.iad10.us.zip.zayo.com

Router #3
- interactive.edge1.washington111.level3.net
- ce-0-4-0-2.r05.asbnva02.us.ce.gin.ntt.net

Router #4
- level3-as3356.e0-51.switch2.ash1.he.net
- ae-1-3510.edge1.washington111.level3.net
- level3.ashburn2.ash.seabone.net

Most common: operators embed IATA airport code of closest airport.

Airport codes are unique.
Challenge: operators use different dictionaries

Router #1
xo.iad02.atlas.cogentco.com  
as2828.was14.ip4.gtt.net  
te9-2-0d0.cir1.ashburn-va.us.xo.net

Router #2
vodafone.iad02.atlas.cogentco.com  
ae-0.vodafone.asbnva02.us.bb.gin.ntt.net  
zayo.vodafone.er2.iad10.us.zip.zayo.com

Router #3
interactive.edge1.washington.111.level3.net  
ce-0-4-0-2.r05.asbnva02.us.ce.gin.ntt.net

Router #4
level3-as3356.e0-51.switch2.ash1.he.net  
ae-1-3510.edge1.washington.111.level3.net  
level3.ashburn2.ash.seabone.net

Also common: operators embed place names.

Challenge: at least 27 populated places named “Washington”
4 named “Ashburn”
Challenge: operators use different dictionaries

| Router #1 | xo.iad02.atlas.cogentco.com | as2828.was14.ip4.gtt.net | te9-2-0d0.cir1.ashburn-va.us.xo.net |
| Router #2 | vodafone.iad02.atlas.cogentco.com | ae-0.vodafone.asbnva02.us.bb.gin.ntt.net | zayo.vodafone.er2.iad10.us.zip.zayo.com |
| Router #3 | usqas1-rt002i.i3d.net | interactive.edge1.washington111.level3.net | ce-0-4-0-2.r05.asbnva02.us.ce.gin.ntt.net |
| Router #4 | level3-as3356.e0-51.switch2.ash1.he.net | ae-1-3510.edge1.washington111.level3.net | level3.ashburn2.ash.seabone.net |

Also common: operators embed (portions of) a CLLI code.

<table>
<thead>
<tr>
<th>Chars</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Place</td>
<td>asbn</td>
</tr>
<tr>
<td>2</td>
<td>State/country</td>
<td>va</td>
</tr>
<tr>
<td>5</td>
<td>various</td>
<td></td>
</tr>
</tbody>
</table>

asbnva = Ashburn, VA, US
Challenge: operators use different dictionaries

UN LOCODEs are less common, and not always human readable.

<table>
<thead>
<tr>
<th>Chars</th>
<th>Meaning</th>
<th>e.g.</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Country</td>
<td>us</td>
</tr>
<tr>
<td>3</td>
<td>Place</td>
<td>qas</td>
</tr>
</tbody>
</table>

qas = Ashburn, VA, US
Challenge: operators use different dictionaries

Some operators helpfully embed country or state codes in their hostnames.

i.e., we know that “ashburn” refers to the one in Virginia, US.
Challenge: operators deviate from dictionaries

| Router #1 | xo.iad02.atlas.cogentco.com  
|           | as2828.was14.ip4.gtt.net  
|           | te9-2-0d0.cir1.ashburn-va.us.xo.net |

- he.net and seabone.net both use “ash” to mean “Ashburn, VA, US”.

| Router #2 | vodafone.iad02.atlas.cogentco.com  
|           | ae-0.vodafone.asbnva02.us.bb.gin.ntt.net  
|           | zayo.vodafone.er2.iad10.us.zip.zayo.com |

- “ash” is the IATA code for “Nashua, NH, US”

| Router #3 | interactive.edge1.washington111.level3.net  
|           | ce-0-4-0-2.r05.asbnva02.us.ce.gin.ntt.net |

- Implication: must learn per-suffix dictionaries

| Router #4 | level3-as3356.e0-51.switch2.ash.he.net  
|           | ae-1-3510.edge1.washington111.level3.net  
|           | level3.ashburn2.ash.seabone.net |

- Challenge: abbreviations are lossy
Contributions of this work

• We design and implement a method that automatically
  - learns regexes that extract geohints from hostnames,
  - learns new geohints when operators deviate from the dictionary.

• We publicly release
  - the source code implementation as part of Hoiho,
    (Hoiho: Holistic Orthography of Internet Hostname Observations)
  - the inferred naming conventions and a utility to apply them.
    - https://www.caida.org/tools/measurement/scamper/

Image: Brent Beaven
Department of Conservation (New Zealand)
Key Results

• For an **August 2020** set of **2.56M** routers with IPv4 addresses

  - **8.8%** had hostnames containing apparent geohints

  - Our method inferred naming conventions for **906 suffixes** that extracted geohints from **86.8%** of these routers

    147 (**38.2%**) of 461 suffixes deviated from IATA dictionary: deviation from dictionary was common

  - We evaluated our method on four sets of routers, with **IPv4** and **IPv6** routers, to infer **1023** and **241** conventions, respectively, for these routers.

<table>
<thead>
<tr>
<th>Type</th>
<th>Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>IATA</td>
<td>461</td>
</tr>
<tr>
<td>City</td>
<td>372</td>
</tr>
<tr>
<td>CLLI prefix</td>
<td>96</td>
</tr>
<tr>
<td>LOCODE</td>
<td>10</td>
</tr>
<tr>
<td>Facility</td>
<td>2</td>
</tr>
</tbody>
</table>
Selected Related Work

- undns: (SIGCOMM 2002)
- CBG: (IMC 2004)
- DRoP: (CCR 2014)
- HLOC: (TMA 2017)
- Hoiho: (IMC 2019 + 2020)
Selected Related Work: **undns**

- **undns**: (SIGCOMM 2002)
- CBG: (IMC 2004)
- DRoP: (CCR 2014)
- HLOC: (TMA 2017)
- Hoiho: (IMC 2019 + 2020)

Hand-crafted regexes built by manually interpreting hostnames. Hand-crafted rules to interpret extracted output.

Example routers

```
^be-\d+\.(cor|bdr)\d+\.([a-z]{3})\d+\.[a-z]{2,3}.vocus.net.au$
```

```
type=1 {  
  cor "backbone"
  bdr "gateway"
}
loc=2 {  
  per "Perth, Australia"
  mel "Melbourne, Australia"
  syd "Sydney, Australia"
}  
```
Selected Related Work: **CBG**

- **undns**: (SIGCOMM 2002)
- **CBG**: (IMC 2004)
- **DROp**: (CCR 2014)
- **HLOC**: (TMA 2017)
- **Hoiho**: (IMC 2019 + 2020)

CBG infers a system is located at the centroid of distance constraints built using delay measurements from vantage points with known locations.

(Figure 2 of “Constraint Based Geolocation of Internet Hosts”)
Selected Related Work: **DRoP**

- **undns**: (SIGCOMM 2002)
- **CBG**: (IMC 2004)
- **DRoP**: (CCR 2014)
- **HLOC**: (TMA 2017)
- **Hoiho**: (IMC 2019 + 2020)

**Matched**
- `den1-core-01-ae1.360.net`
- `den1-core-01-xe-1-1-0.360.net`
- `den1-core-02-ae1.360.net`
- `den1-core-02-xe-0-1-0.360.net`
- `lax1-edge-01-1-1-1.360.net`
- `sea1-edge-02-lag1.360.net`
- `pdx2-access-01-1-1-2.360.net`

**Unmatched**
- `den1-core-01-xe-1-1-0.360.net`
- `den1-core-02-xe-0-1-0.360.net`

Automatically built regexes that extract apparent geohints from router hostnames, using RTT constraints collected by traceroute.

**Limitation**: RTT constraints collected by traceroute do not provide tight constraints. Multiple works report that more DRoP-inferred locations are wrong than correct.
Selected Related Work: **HLOC**

- undns: (SIGCOMM 2002)
- CBG: (IMC 2004)
- DRoP: (CCR 2014)
- **HLOC**: (TMA 2017)
- Hoiho: (IMC 2019 + 2020)

For each hostname, identify possible geohints and collect RTT measurements from nearby VPs to confirm.

**Limitations**: confirmation bias. Does not attempt to learn geocodes not in input dictionaries.
Selected Related Work: **HLOC**

- **undns**: (SIGCOMM 2002)
- **CBG**: (IMC 2004)
- **DRoP**: (CCR 2014)
- **HLOC**: (TMA 2017)
- **Hoiho**: (IMC 2019 + 2020)

For each hostname, identify possible geohints and collect RTT measurements from nearby VPs to confirm.

**Limitations**: confirmation bias. Does not attempt to learn geocodes not in input dictionaries.
Selected Related Work: **HLOC**

- **undns**: (SIGCOMM 2002)
- **CBG**: (IMC 2004)
- **DRoP**: (CCR 2014)
- **HLOC**: (TMA 2017)
- **Hoiho**: (IMC 2019 + 2020)

For each hostname, identify possible geohints and collect RTT measurements from nearby VPs to confirm.

**Limitations**: confirmation bias.
Does not attempt to learn geocodes not in input dictionaries.
Selected Related Work: **HLOC**

- undns: (SIGCOMM 2002)
- CBG: (IMC 2004)
- DRoP: (CCR 2014)
- **HLOC**: (TMA 2017)
- Hoiho: (IMC 2019 + 2020)

For each hostname, identify possible geohints and collect RTT measurements from nearby VPs to confirm.

**Limitations**: confirmation bias. Does not attempt to learn geocodes not in input dictionaries.
Selected Related Work: **Hoiho**

- **undns**: (SIGCOMM 2002)
- **CBG**: (IMC 2004)
- **DRoP**: (CCR 2014)
- **HLOC**: (TMA 2017)
- **Hoiho**: (IMC 2019 + 2020)

Hoiho 2019: learn regexes that extract router names (strings shared across router interface hostnames unique to each router)

(Hoiho: Holistic Orthography of Internet Hostname Observations)
Selected Related Work: **Hoiho**

- **undns**: (SIGCOMM 2002)
- **CBG**: (IMC 2004)
- **DRoP**: (CCR 2014)
- **HLOC**: (TMA 2017)
- **Hoiho**: (IMC 2019 + 2020)

Hoiho 2020: learn regexes that extract router ownership information reported by operators in ASN tags

(Hoiho: Holistic Orthography of Internet Hostname Observations)
Overview of Our Geolocation Method in Hoiho

1. Routers Hostnames
   - ITDK
   - Public Suffix List
   - Delay Constraints
   - Reference Dictionary
   - GeoNames LOCODEs
   - OurAirports
   - CLLI, PeeringDB

Legend:
- ★ New Feature
- ★☆ Modification

2. Identify apparent geolocation hints
3. Generate and evaluate regexes
4. Dictionary refinement
5. Select best convention

Holistic Orthography of Internet Hostname Observations (Hoiho)

We licensed a database mapping CLLI prefixes to location names from iconectiv; we mapped these to lat/longs.
Overview of Our Geolocation Method in Hoiho

Focus in this talk on identifying apparent geohints, and interpreting new geohints. See paper for rest of method detail.
Identify possible geohints with input dictionary

Consider sequences of alphabetical characters that are defined in the input dictionary.
Consider sequences of alphabetical characters that are defined in the input dictionary.

Identify possible geohints with input dictionary

- ae0-2.rt.irx.\textsf{vie}.at.retn.net
- ae1-1.rt.irx.\textsf{vie}.at.retn.net
- ae1-4.rt.evo.\textsf{ams}.nl.retn.net
- ae0-4.rt.wbx.\textsf{sea}.us.retn.net
- ae1-5.rt.irx.\textsf{cph}.dk.retn.net
- ae1-5.rt.srv.mep.nl.retn.net
- de-cix1.rt.\textsf{act}.fkt.de.retn.net

- vie (Vienna, AT)
- ams (Amsterdam, NL)
- sea (Seattle, WA, US)
- cph (Copenhagen, DK)
- mep (Mersing, MY)
- srv (Stony River, AK, US)
- cix (Chiclayo, PE)
- act (Waco, TX, US)
Identify apparent geohints with RTT measurements

A string is an **apparent geohint** if measured RTTs are consistent with the location implied by the geohint.
Identify apparent geohints with RTT measurements.

A string is an **apparent geohint** if measured RTTs are consistent with the location implied by the geohint.
Identify apparent geohints with RTT measurements

A string is an **apparent geohint** if measured RTTs are consistent with the location implied by the geohint.
Identify apparent geohints with RTT measurements

Note: these hostnames appear to include an ISO-3166 country-code. Our method identifies this as something it should extract.
Build Regular Expressions to Extract Apparent Geohints

(see paper for details)

```plaintext
^\.(\[[a-z]3\])\.(\[[a-z]2\])\retn\net$
```
Build Regular Expressions to Extract Apparent Geohints

\(^.+\ المختلف ([a-z]{3}) \ Lotto ([a-z]{2}) \ Lotto .retn.net\)

IATA  CC

Our method includes a *plan* for each regex: i.e., what each extraction represents.

*see paper for details*
Build Regular Expressions to Extract Apparent Geohints

(see paper for details)

\^[a-z]\.[a-z].retn.net$

Vienna, AT

IATA: vie, CC: AT

Amsterdam, NL

IATA: ams, CC: NL

Seattle, WA, US

IATA: sea, CC: US

Copenhagen, DK

IATA: cph, CC: DK

IATA, CC
Build Regular Expressions to Extract Apparent Geohints

(see paper for details)

```
^\.+\([a-z]\{3\}\)\.[a-z]\{2\}\.retn\.net$
```

<table>
<thead>
<tr>
<th>IATA</th>
<th>CC</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>vie</td>
<td>at</td>
<td>Vienna, AT</td>
</tr>
<tr>
<td>vie</td>
<td>at</td>
<td>Amsterdam, NL</td>
</tr>
<tr>
<td>ams</td>
<td>nl</td>
<td>Seattle, WA, US</td>
</tr>
<tr>
<td>sea</td>
<td>us</td>
<td></td>
</tr>
<tr>
<td>cph</td>
<td>dk</td>
<td>Copenhagen, DK</td>
</tr>
<tr>
<td>mep</td>
<td>nl</td>
<td>???</td>
</tr>
<tr>
<td>fkt</td>
<td>de</td>
<td>???</td>
</tr>
</tbody>
</table>

ae0-2.rt.irx.vie.at.retn.net  ➙  iata: vie, cc: at
ae1-1.rt.irx.vie.at.retn.net  ➙  iata: vie, cc: at
ae1-4.rt.evo.ams.nl.retn.net  ➙  iata: ams, cc: nl
ae0-4.rt.wbx.sea.us.retn.net  ➙  iata: sea, cc: us
ae1-5.rt.irx.cph.dk.retn.net  ➙  iata: cph, cc: dk
ae1-5.rt.srv.mep.nl.retn.net  ➙  iata: mep, cc: nl
de-cix1.rt.act.fkt.de.retn.net ➙  iata: fkt, cc: de
Learn Geohints not in Dictionary

Consider abbreviations of RTT-consistent populated places

<table>
<thead>
<tr>
<th>Place</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meppe, DR, NL</td>
<td>30,697</td>
</tr>
<tr>
<td>Meppen, DR, NL</td>
<td>305</td>
</tr>
<tr>
<td>Middelkoop, UT, NL</td>
<td>370</td>
</tr>
<tr>
<td>Frankfurt am Main, HE, DE</td>
<td>650,000</td>
</tr>
<tr>
<td>Frankenthal, RP, DE</td>
<td>47,438</td>
</tr>
<tr>
<td>Falkenstein, DE</td>
<td>9,528</td>
</tr>
</tbody>
</table>

Name of candidate populated place must match first letter in abbreviation. Prefer places with known facilities, then places with higher population.

★ Place has a facility listed in PeeringDB

ae1-5.rt.srv.mep.nl.retn.net
4ms from Amsterdam, NL

de-cix1.rt.act.fkt.de.retn.net
8ms from Zurich, CH
Validation of learned geohints against ground truth

<table>
<thead>
<tr>
<th>aorta.net</th>
<th>as8218.eu</th>
<th>geant.net</th>
<th>gtt.net</th>
<th>he.net</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/4</td>
<td>3/3</td>
<td>8/8</td>
<td>12/12</td>
<td>4/4</td>
</tr>
<tr>
<td>(75%)</td>
<td>(100%)</td>
<td>(100%)</td>
<td>(100%)</td>
<td>(100%)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ntt.net</th>
<th>retn.net</th>
<th>seabone.net</th>
<th>tfbnw.net</th>
<th>zayo.net</th>
</tr>
</thead>
<tbody>
<tr>
<td>17/18</td>
<td>25/34</td>
<td>14/15</td>
<td>2/14</td>
<td>4/4</td>
</tr>
<tr>
<td>(94.4%)</td>
<td>(73.5%)</td>
<td>(93.3%)</td>
<td>(14.3%)</td>
<td>(100%)</td>
</tr>
</tbody>
</table>

Overall: 92/117 (78.6%)

We obtained ground truth for the learned geohints from operators at 10 different networks.
Validation of learned geohints against ground truth

<table>
<thead>
<tr>
<th></th>
<th>aorta.net</th>
<th>as8218.eu</th>
<th>geant.net</th>
<th>gtt.net</th>
<th>he.net</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3/4</td>
<td>3/3</td>
<td>8/8</td>
<td>12/12</td>
<td>4/4</td>
</tr>
<tr>
<td></td>
<td>(75%)</td>
<td>(100%)</td>
<td>(100%)</td>
<td>(100%)</td>
<td>(100%)</td>
</tr>
<tr>
<td></td>
<td>ntt.net</td>
<td>retn.net</td>
<td>seabone.net</td>
<td>tfbnw.net</td>
<td>zayo.net</td>
</tr>
<tr>
<td></td>
<td>17/18</td>
<td>25/34</td>
<td>14/15</td>
<td>2/14</td>
<td>4/4</td>
</tr>
<tr>
<td></td>
<td>(94.4%)</td>
<td>(73.5%)</td>
<td>(93.3%)</td>
<td>(14.3%)</td>
<td>(100%)</td>
</tr>
</tbody>
</table>

**Overall**

|        | 92/117    | (78.6%) |

Overall, 78.6% of the learned geohints we validated identified the correct place.
Validation of learned geohints against ground truth

<table>
<thead>
<tr>
<th></th>
<th>aorta.net</th>
<th>as8218.eu</th>
<th>geant.net</th>
<th>gtt.net</th>
<th>he.net</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/4</td>
<td>3/3</td>
<td>8/8</td>
<td>12/12</td>
<td>4/4</td>
<td></td>
</tr>
<tr>
<td>(75%)</td>
<td>(100%)</td>
<td>(100%)</td>
<td>(100%)</td>
<td>(100%)</td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>92/117</td>
<td>(78.6%)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>ntt.net</th>
<th>retn.net</th>
<th>seabone.net</th>
<th>tfbnw.net</th>
<th>zayo.net</th>
</tr>
</thead>
<tbody>
<tr>
<td>17/18</td>
<td>25/34</td>
<td>14/15</td>
<td></td>
<td>2/14</td>
<td>4/4</td>
</tr>
<tr>
<td>(94.4%)</td>
<td>(73.5%)</td>
<td>(93.3%)</td>
<td>(14.3%)</td>
<td>(100%)</td>
<td></td>
</tr>
</tbody>
</table>

Outlier: Facebook, which places datacenter facilities in low-population locations that are not used as peering facilities (not in PeeringDB)
Validation of conventions against ground truth

- **Our method in Hoiho** inferred the correct location for 94.0% of hostnames across 14 suffixes.
- **DRoP and undns** coverage is lower as their conventions are old.
- **HLOC and DRoP** FPs are because they don’t learn custom geohints.
- **undns** also missing location mappings.

Note: gap between top of bar and 100% are false negatives: geohints missed by a method.
Validation of conventions against ground truth

Though undns has lowest coverage, it has highest PPV: TP/(TP+FP). The locations it has in its dictionary are generally correct.
Limitation: not all operators use an easily parsed convention

<table>
<thead>
<tr>
<th>VP, US</th>
<th>RTT</th>
<th>Hostname</th>
</tr>
</thead>
<tbody>
<tr>
<td>atl, us</td>
<td>7ms</td>
<td>atnga00002cce9-irb-2.infra.cdn.att.net</td>
</tr>
<tr>
<td>ord, us</td>
<td>9ms</td>
<td>bcvoht00002ccej9-irb-2.infra.cdn.att.net</td>
</tr>
<tr>
<td>dal, us</td>
<td>5ms</td>
<td>dlltx00001ccej9-irb-2.infra.cdn.att.net</td>
</tr>
<tr>
<td>jfk, us</td>
<td>1ms</td>
<td>nycny00002ccej9-irb-2.infra.cdn.att.net</td>
</tr>
<tr>
<td>dal, us</td>
<td>4ms</td>
<td>rd3tx00001ccej9-ae120-100.infra.cdn.att.net</td>
</tr>
<tr>
<td>sjc, us</td>
<td>4ms</td>
<td>scaca00002ccej9-ae120-200.infra.cdn.att.net</td>
</tr>
</tbody>
</table>

**Dictionary:**

- **atnga**: Atlanta, GA
- **bcvoht**: Brecksville, OH
- **dlltx**: Dallas, TX
- **nycny**: New York City, NY
- **rd3tx**: Richardson, TX
- **scaca**: Sacramento, CA

**AT&T uses a convention with no punctuation between 3-letter abbreviation of place and 2-letter state code. 3-letter abbreviations are not based on airport codes and difficult even for a human to decipher.**
Summary

• We designed and implemented a method that automatically
  - learns regexes that extract geohints from hostnames,
  - learns new geohints when operators deviate from the dictionary.

• We publicly release
  - the source code implementation as part of Hoiho,
    (Hoiho: Holistic Orthography of Internet Hostname Observations)
  - the inferred naming conventions and a utility to apply them.

- https://www.caida.org/tools/measurement/scamper/

<table>
<thead>
<tr>
<th>Method</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hoiho</td>
<td>94.0%</td>
</tr>
<tr>
<td>HLOC</td>
<td>73.1%</td>
</tr>
<tr>
<td>DRoP</td>
<td>56.6%</td>
</tr>
<tr>
<td>undns</td>
<td>21.8%</td>
</tr>
</tbody>
</table>
BACKUP SLIDES
High-level Approach

- Infer if an operator embeds information identifying the location of the router in PTR hostname records for router interfaces

**Input:**
- Mozilla public suffix list to identify where domains can be registered (.net, .org, .co.nz)
- Hostnames for router interfaces observed by traceroute (PTR records)
- Router alias inferences from MIDAR, mercator
- RTT measurements using ICMP, UDP, and TCP pings
- Geohint dictionary with IATA, ICAO, CLLI prefixes, LOCODEs, Towns, States, Countries

**Output:** regular expressions that extract router geolocation, and a dictionary to interpret the geohints.
### Results: Coverage of Inferred Naming Conventions

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>total</td>
<td>2.56M</td>
<td>2.57M</td>
<td>559K</td>
<td>525K</td>
</tr>
<tr>
<td>with hostname</td>
<td>1.41M (55.0%)</td>
<td>1.39M (54.1%)</td>
<td>84K (15.1%)</td>
<td>84K (16.0%)</td>
</tr>
<tr>
<td>with apparent geohint</td>
<td>225K (8.8%)</td>
<td>220K (8.5%)</td>
<td>29K (5.3%)</td>
<td>31K (5.8%)</td>
</tr>
<tr>
<td>geolocated</td>
<td>195K (7.6%)</td>
<td>183K (7.1%)</td>
<td>26K (4.7%)</td>
<td>27K (5.2%)</td>
</tr>
</tbody>
</table>

We used CAIDA ITDKs where we simultaneously collected RTT samples from available CAIDA Archipelago Vantage Points. Our conventions extracted 83.4% - 89.6% of apparent geohints.
Validation of conventions with ground truth

<table>
<thead>
<tr>
<th>Hostnames:</th>
<th>100%</th>
<th>99.3%</th>
<th>99.6%</th>
<th>93.0%</th>
<th>95.5%</th>
<th>98.5%</th>
<th>95.5%</th>
<th>84.3%</th>
<th>78.7%</th>
<th>88.6%</th>
<th>90.3%</th>
<th>94.7%</th>
<th>96.7%</th>
<th>94.0%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hoiho:</td>
<td>71.7%</td>
<td>99.3%</td>
<td>75.6%</td>
<td>99.6%</td>
<td>93.0%</td>
<td>95.5%</td>
<td>98.5%</td>
<td>95.5%</td>
<td>84.3%</td>
<td>78.7%</td>
<td>88.6%</td>
<td>90.3%</td>
<td>94.7%</td>
<td>96.7%</td>
</tr>
<tr>
<td>HLOC:</td>
<td>63.3%</td>
<td>79.8%</td>
<td>47.3%</td>
<td>81.1%</td>
<td>63.7%</td>
<td>82.8%</td>
<td>77.2%</td>
<td>68.4%</td>
<td>55.7%</td>
<td>78.9%</td>
<td>68.3%</td>
<td>5.3%</td>
<td>77.7%</td>
<td>73.1%</td>
</tr>
<tr>
<td>DRoP:</td>
<td>5.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>Undns:</td>
<td>61.7%</td>
<td>0.0%</td>
<td>56.5%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>2403</td>
<td>131</td>
<td>472</td>
<td>270</td>
<td>1697</td>
<td>2121</td>
<td>3397</td>
<td>479</td>
<td>1238</td>
<td>4128</td>
<td>19</td>
<td>766</td>
<td>17232</td>
</tr>
</tbody>
</table>

Percentage of hostnames with geohints

- **Hoiho:**
  - FP: 71.7%
  - TP: 99.3%

- **HLOC:**
  - FP: 63.3%
  - TP: 79.8%

- **DRoP:**
  - FP: 5.0%
  - TP: 0.0%

- **Undns:**
  - FP: 61.7%
  - TP: 0.0%
Selected Related Work: **TBG**

(Figure 6 of “Towards IP Geolocation Using Delay and Topology Measurements”)

- **undns**: (SIGCOMM 2002)
- **CBG**: (IMC 2004)
- **TBG**: (IMC 2006)
- **DRoP**: (CCR 2014)
- **HLOC**: (TMA 2017)
- **Hoiho**: (IMC 2019 + 2020)

**TBG** adds topological constraints (U) to **CBG** — i.e., intermediate routers observed using traceroute that reduce the distance that **Z** could be from **X/Y**.
Selected Related Work: **DRoP**

- undns: (SIGCOMM 2002)
- **CBG**: (IMC 2004)
- **DRoP**: (CCR 2014)
- **HLOC**: (TMA 2017)
- Hoiho: (IMC 2019 + 2020)

**Limitation**: RTT constraints collected by traceroute do not provide tight constraints. Multiple works report that more DRoP-inferred locations are wrong than correct.
Key Results: Validation

• We compared our geolocation inferences and those made by other approaches with ground truth for hostnames in 14 suffixes.
  
  - Our method has the highest coverage (94.0%) and a PPV of 95.6%

• We compared our learned geohints against ground truth from 10 suffixes with 117 suffix-specific geohints
  
  - 92/117 (78.6%) correctly identified the corresponding location

<table>
<thead>
<tr>
<th>Method</th>
<th>Coverage</th>
<th>PPV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Method</td>
<td>94.0%</td>
<td>95.6%</td>
</tr>
<tr>
<td>HLOC</td>
<td>73.1%</td>
<td>85.1%</td>
</tr>
<tr>
<td>DRoP</td>
<td>56.6%</td>
<td>87.2%</td>
</tr>
<tr>
<td>undns</td>
<td>21.8%</td>
<td>98.3%</td>
</tr>
</tbody>
</table>