OpenIPMap and Traffic Locality Checks

Robert Kisteleki

(presenting work mostly done by Emile Aben)

OpenIPMap

- It's a **prototype service** with the aim to build a geolocation database of **infrastructure** IPs
- Therefore, locating end user eyeballs is not in focus
  - This is covered by several commercial vendors and others
  - It is ok if it happens as a byproduct
- It's in a prototype stage, but it already shows potential
- Data and code are meant to be open
  - Code is on github already
  - Data is available for download

OpenIPMap

- Why? This is useful for a number of cases:
  - Knowing what infrastructure is in a particular place
    - In case of outages, disasters, …
  - Knowing what route your packets take physically
    - Understanding if traffic stays “local”
  - Optimizing your peerings
    - Help avoiding “hairpinning”
  - Other research questions

OpenIPMap

- The idea is to deal with probabilities, assembled from multiple sources:
  - Known locations of networks, e.g. IXP peering LANs
  - Harvesting reverse DNS names of traceroute hops
  - Individual user inputs — crowdsourcing
  - Triangulation from already known locations (planned)
  - Iterative approaches (planned)
- City level accuracy for now, but that could change

OpenIPMap

- Known locations of networks, e.g. IXP peering LANs
  - These are fairly well known, mostly in a single location
  - Exceptions can be handled by a “more specific” match
  - A “network” can be a prefix or a single IP address too

OpenIPMap

- Harvesting reverse DNS names of traceroute hops
  - We all know and use these:

```
...
5  juniper4.ffm.hetzner.de (213.239.245.1)  3.339 ms  3.550 ms  4.306 ms
6  as6939.fra.ecix.net (62.69.146.18)  3.516 ms  4.170 ms  3.381 ms
7  100ge5-2.core1.par2.he.net (72.52.92.13)  13.265 ms  12.609 ms  17.023 ms
8  10ge15-1.core1.ash1.he.net (184.105.213.93)  90.510 ms  90.096 ms  90.065 ms
...
```

  - But they have their own issues, far from accurate

OpenIPMap

- Individual user inputs — crowdsourcing
  - These could be highly accurate
  - If the submitter can be trusted
  - So how do we assess the confidence level in a user submitted entry?
  - It’s nice to know if an entry comes from the operator of that network or from someone else

OpenIPMap

- Triangulation from already known locations
  - Hop proximity: if a hop is <1ms away from another hop, then it's pretty much in the same place
  - The more knowledge we have, the easier it is to fill in the gaps (while staying probabilistic)

OpenIPMap

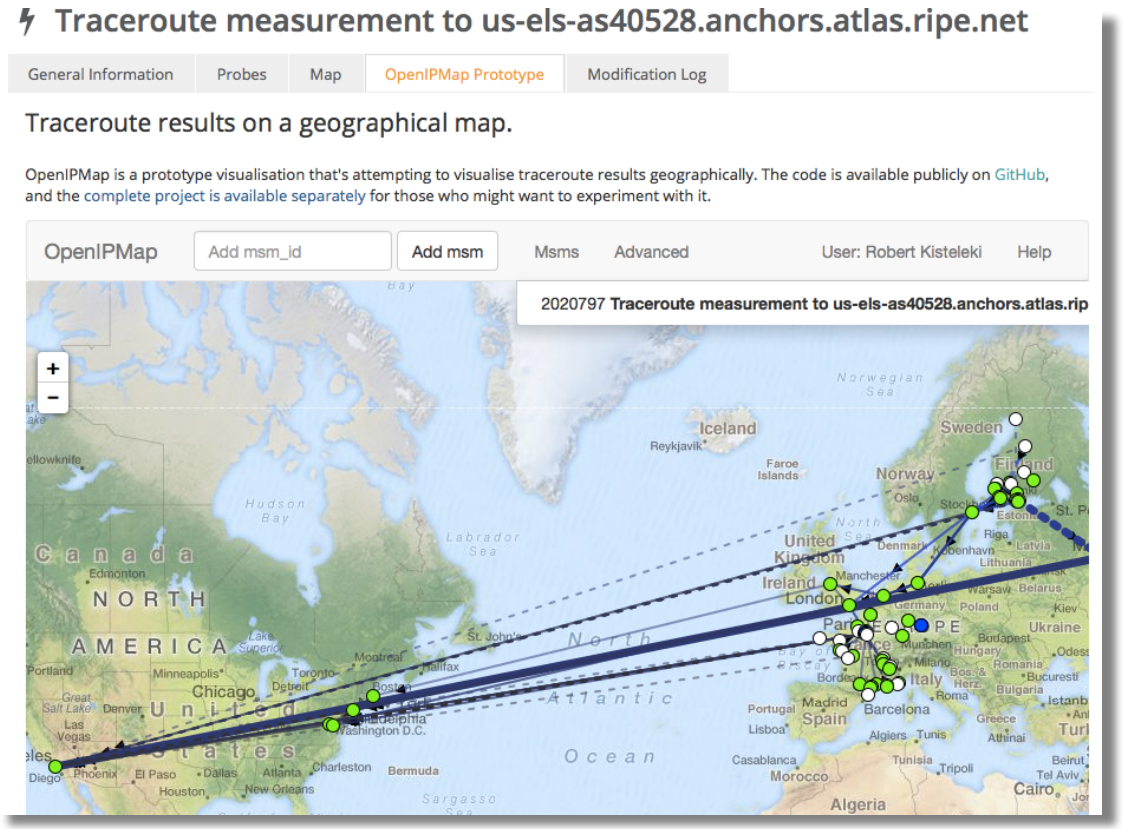
- Iterative approaches
  - Some of the functions can be re-applied to the outcome of the previous application
  - Works well with hop proximity / triangulation

OpenIPMap

- IPs looked at:
  - Based on what we see in Atlas, but could be more
  - It’s not that big of a dataset

OpenIPMap in Practice

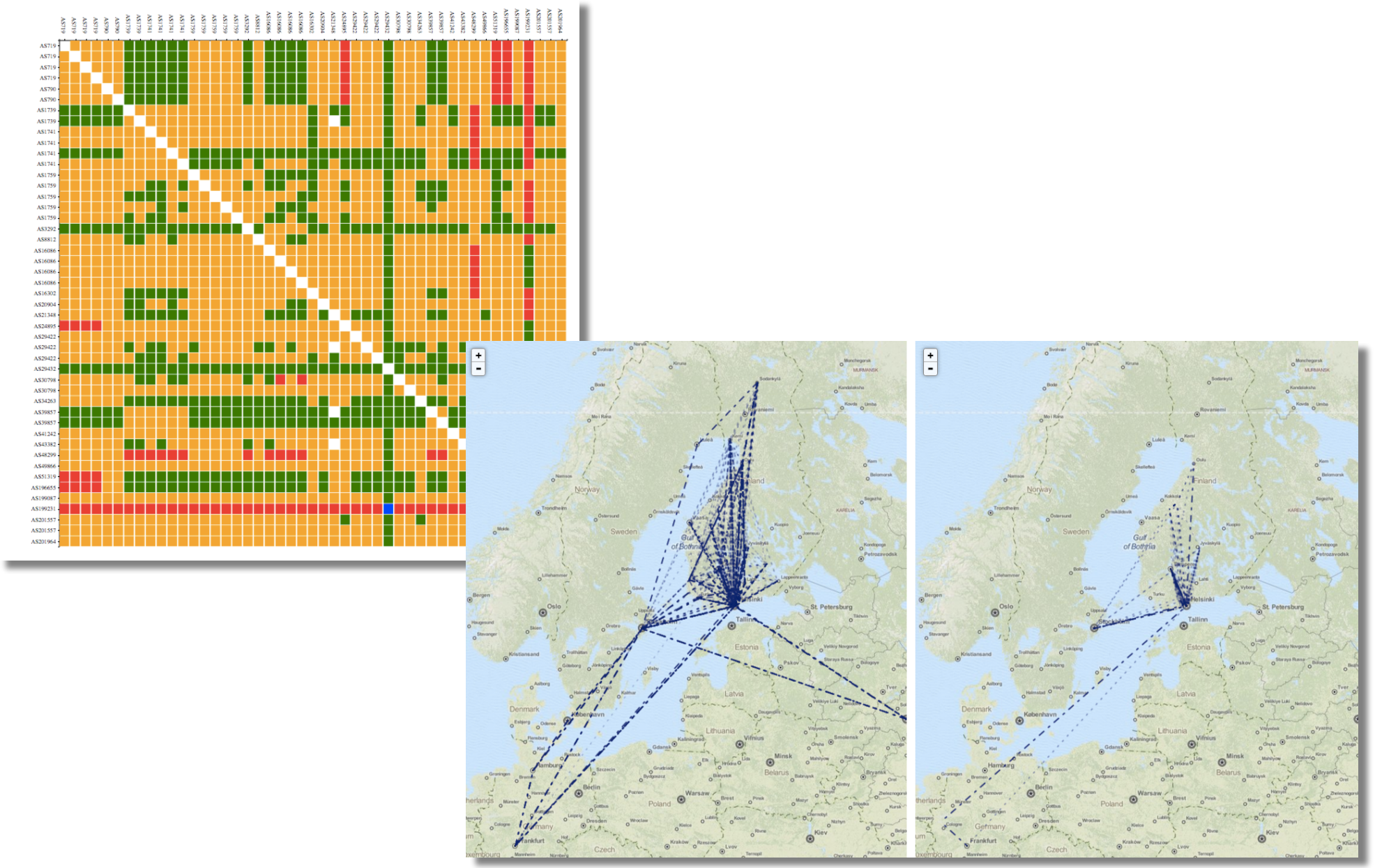
- The prototype is integrated into RIPE Atlas: it shows up as a new tab for traceroute measurements



OpenIPMap -> Locality Checks

- Combining RIPE Atlas with OpenIPMap:
  - Select a set of probes in the observed country / region
  - Execute traceroutes in a mesh (probe-to-probe) to measure how packets flow
  - Map **all** the observed IPs to their locations (if known)
  - Check which paths stay within the country / region
- Visualise this:
  - On a map
  - Highlighting in-country and out-of country traffic
  - Highlighting IXP use

Locality Checks



Locality Checks

- This tool is also on github
- By using this interactive tool you can…
  - … select the path that is going out of country
    - talk to your upstream(s)
  - … select the path that is not going via a local IXP
    - make a new peering agreement
- And if you see geolocation errors, then please contribute and fix them in OpenIPMap! :-)
  - https://marmot.ripe.net/openipmap/

Questions?