Internet Routing, Measurement & Observability Walkthrough

Aaron Atac

So you want to learn about internet routing, measurement, and observability?

Cool, cool, cool…

But, I only have 25 minutes!

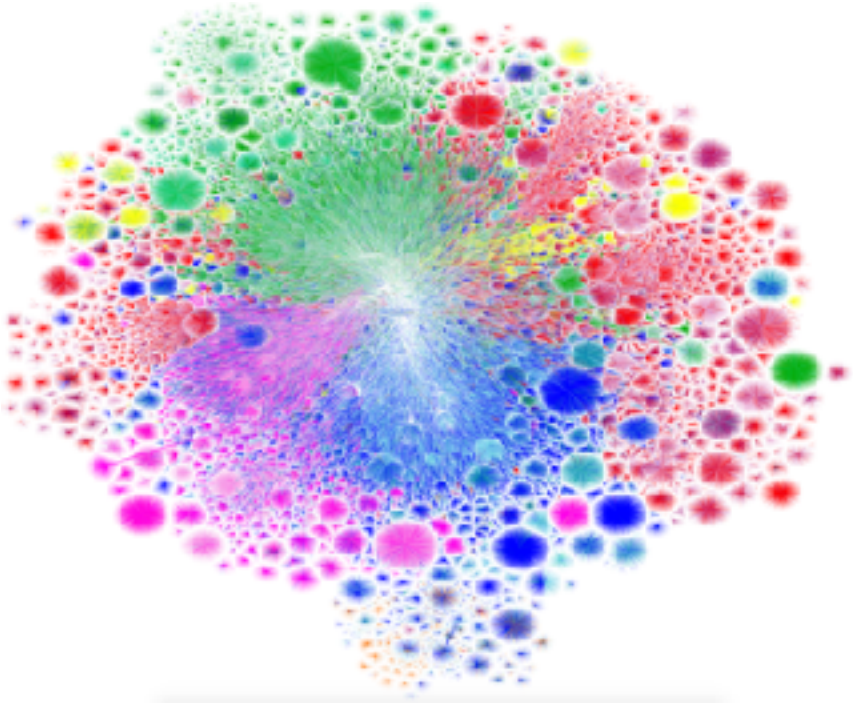
Do not fear, we’ll get through this!

I think? Ok, maybe fear a little…

What Is The Internet? Oh boy, here we go…

- Global, regional, and local networks interconnecting over some medium, predominantly **fiber optic cable**, to pass messages between machines.
- **Message -> bits -> ~ light waves ~ -> bits -> message**.
- Networks exchange paths to destinations, which are otherwise called routes, and routers route/forward messages to destinations!

Visualizing The Internet



Source: opte.org

Asia Pacific
Europe/Middle East/
Central Asia/Africa
North America
Latin American and
Caribbean
RFC1918 IP Addresses
Unknown

ASNs, IPs, and BGP

- Networks need identification to distinguish themselves from one another, and ways to distinguish their machines.
- First, network identification:
  - **ASN** = Autonomous System Number
    - Ex: AS65535 or AS4200000000
    - ASNs originally were only 16 bit, but then we ran out, and started 32 bit ASNs aka 4 byte ASNs.

ASNs, IPs, and BGP

- **IPv4/v6** = Internet Protocol…Addresses
  - Internet Destination Distinguishers
  - Aggregated or abbreviated into blocks called “prefixes”
    - Ex: 192.0.2.0/24 (v4) or 2001:db8::1/32 (v6)
  - Why v6? We ran out of numbers.
  - IPv4: ~4 billion addresses (2^{32})
  - IPv6: 340 undecillion, or 340 billion billion billion billion (2^{128})

BGP (Border Gateway Protocol)

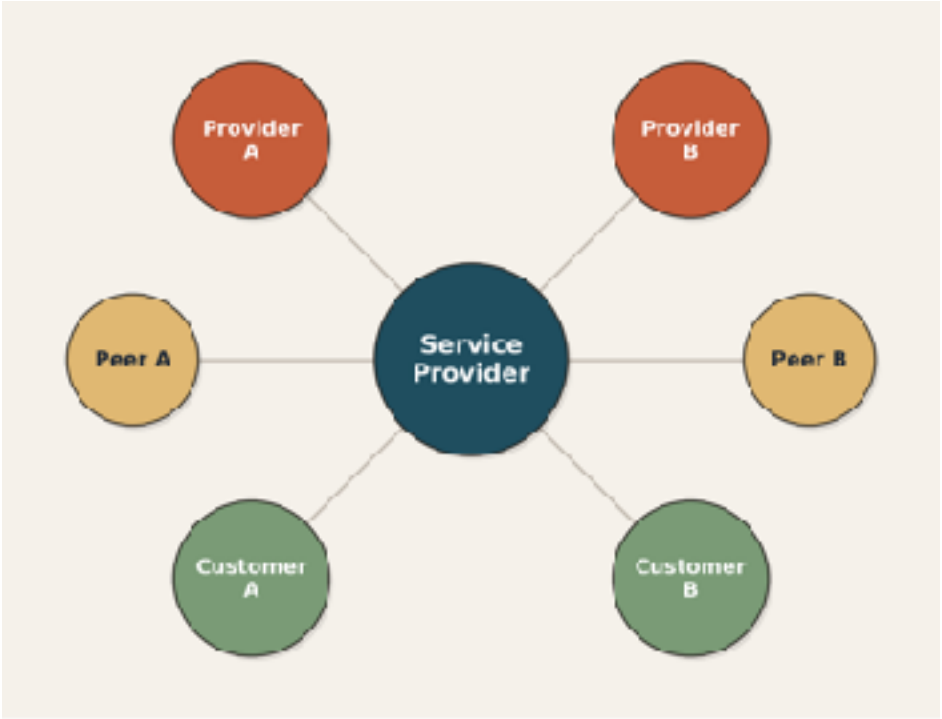
- **BGP** is how networks exchange routes to destinations.
- BGP is like Google Maps. By default, BGP is supposed to find the shortest route to your specific destination.
- However, everyone has a different map!
- Paths are unidirectional or one-way. Asymmetry of traffic flow is common.
- Two networks path length = 2, but geographically can be far!

BGP (Border Gateway Protocol)

- Connectivity to a global uniform internet does not exist!
- Some countries even censor, block, or disconnect their citizens from reaching certain destinations if not all.
- Per cidr-report, ~60% of networks are only seen via one path!
- Every network has different views of paths. Research has shown significant deviation. See Desperately Seeking Default.

BGP Peering Relationships

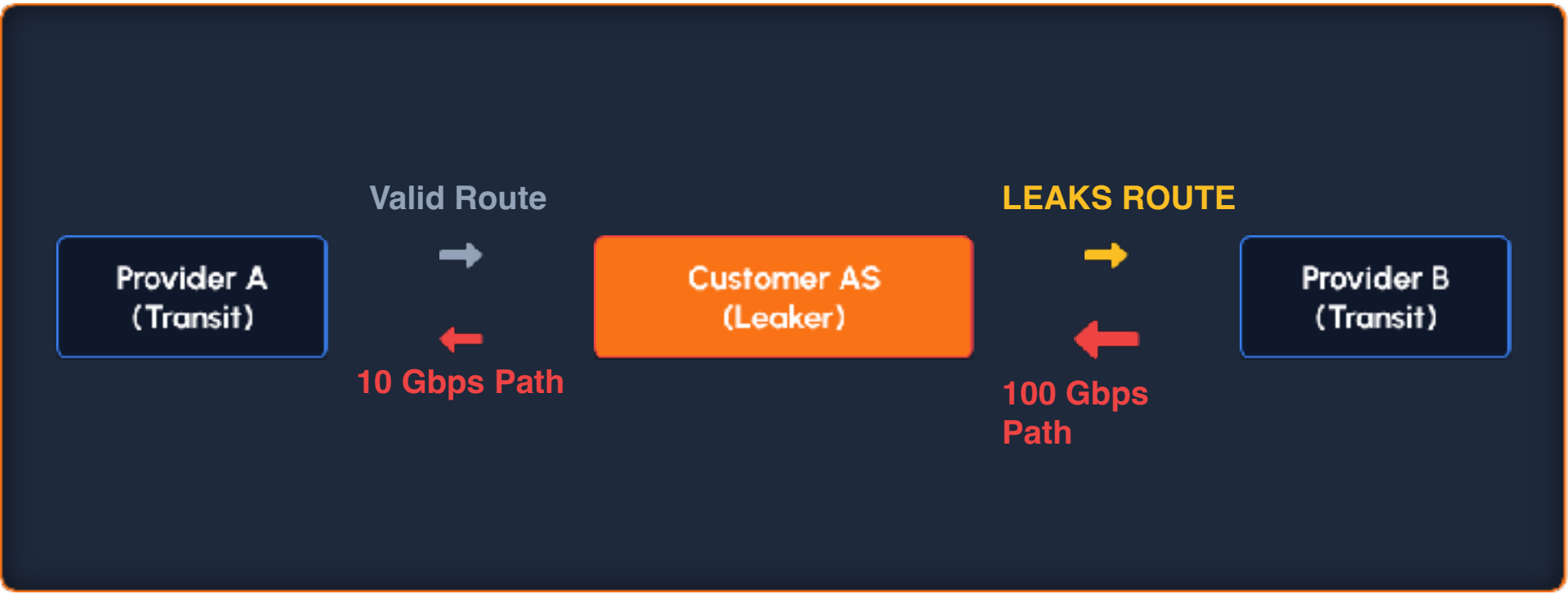
- **Provider**: You pay them for internet access.
- **Peer**: You *ideally* exchange routes/traffic “settlement free”.



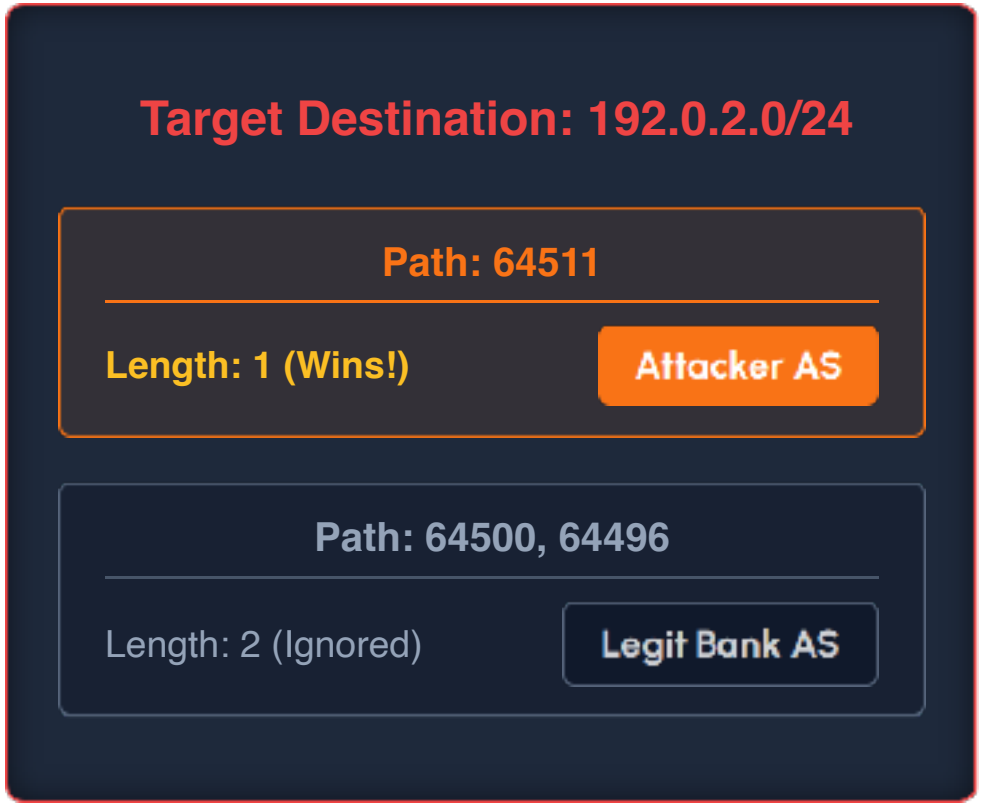
BGP Route Leaks & Hijacks

- **BGP Route Leaks**: Unintentional advertisement of a route to a network.
  - Someone/thing misconfigured their router(s).
- **BGP Route Hijacks**: Unauthorized origination of IP space.
  - Someone/thing misconfigured their routers, haven’t updated their routers (squatting), or is trying to capture your traffic as an adversary for a malicious purpose.

BGP Route Leak Example



BGP Route Hijack Example



MANRS

- Mutually Agreed Norms for Routing Security (MANRS)
  - Filtering: Prevent propagation of incorrect routing info.
  - Anti-spoofing: Prevent traffic with spoofed source IPs.
  - Coordination: Maintain globally accessible contact info.
  - Global Validation: Facilite validation of routing info.
  - https://observatory.manrs.org/#/overview

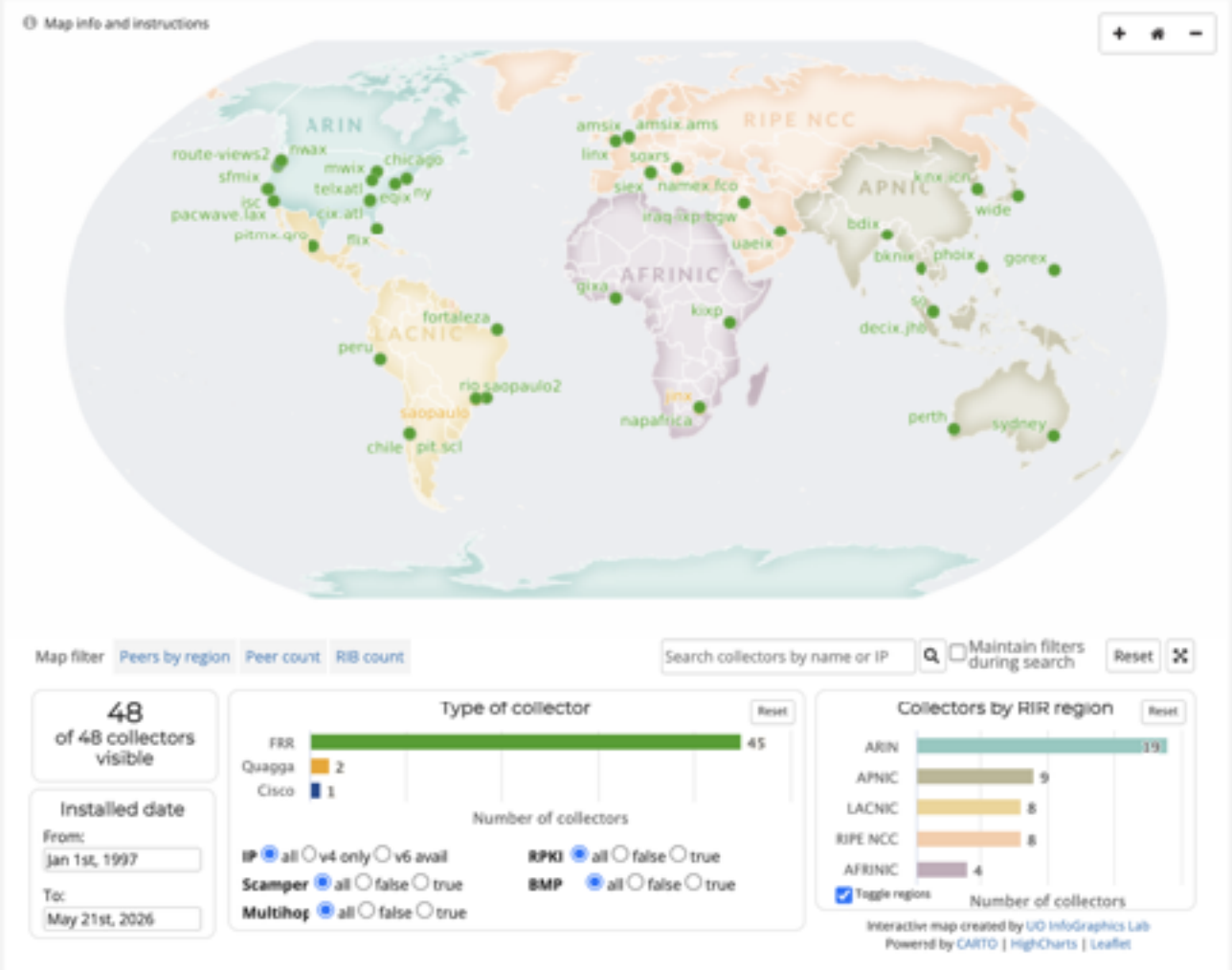
How do you observe and measure the internet? Well…

Still, Here’s A List Of Tools! ¯_(ツ)_/¯

- RIR's IRRs, RADB, PeeringDB, RIPEStat, CAIDA's ASRANK, Qrator.Radar, NIST RPKI Monitor, Cloudflare Radar, RIPE RPKI Validator, isbgpsafeyet.com, RouteViews, RIPE RIS, Georgia Tech's GRIP, Georgia Tech's IODA, bgproutes.io, bgp.potaroo.net, cidr-report.org, bgp.tools, RIPE Atlas, NLNOG Ring, MANRS observatory, bgp.he.net, IRRExplorer, RPKIViews.org, dataplane.org, bgpalerter, ixpdb.euro-ix.net, and ISP looking glasses…**just to name a few**.
- I’m sorry, I wish I had time to deep dive. Let’s still talk about some of these!

Route Archiving, Collection, Monitoring, & Probing

- Two big projects that dominate most research:
  - **RouteViews**
  - **RIPE RIS** (Routing Information Service)
- Both deploy collectors around the world, networks peer with them and share their routes/views for free for the public global benefit!
- RouteViews oldest archive goes back to **1997**! RIPE RIS **1999**!



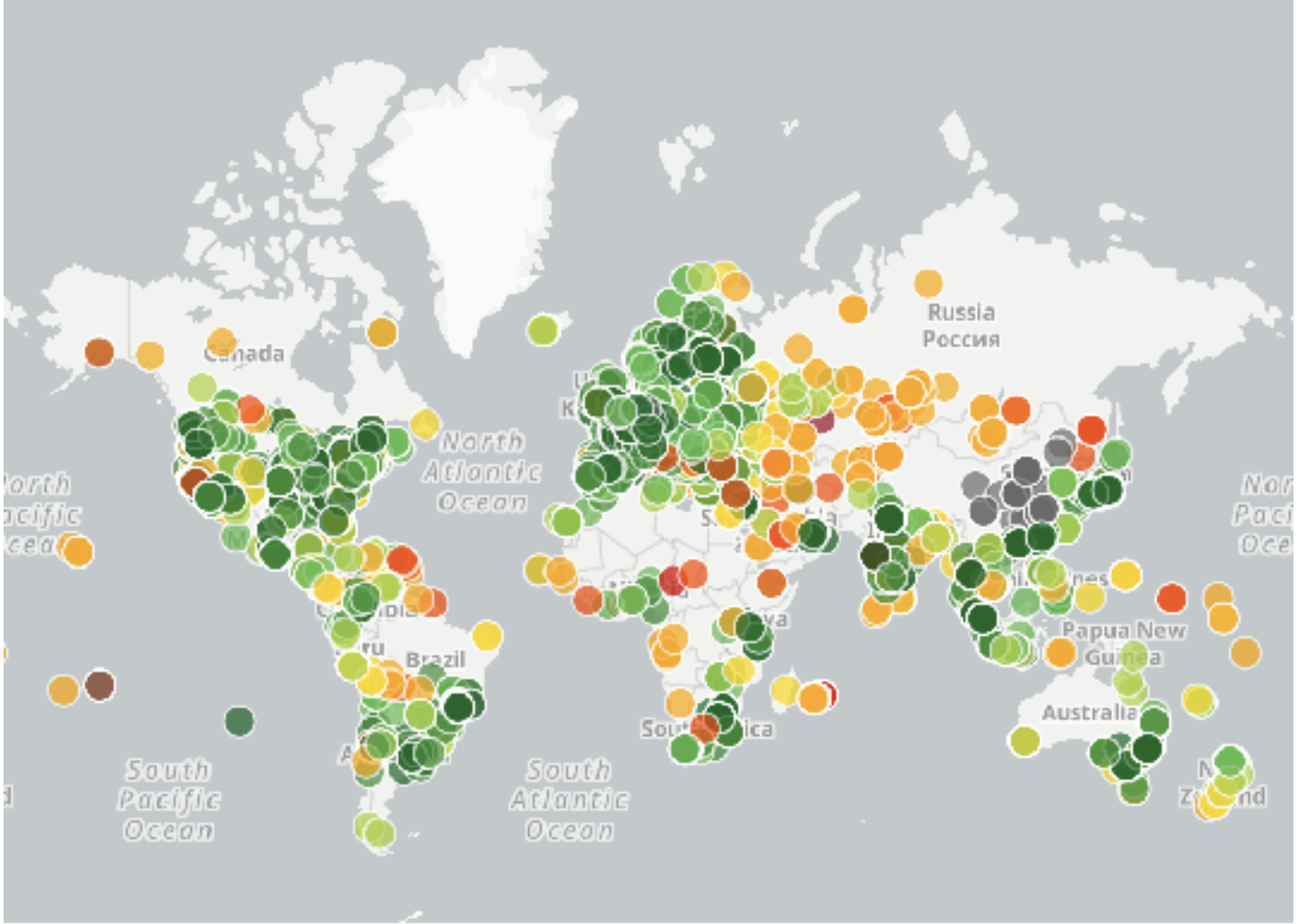
Source: https://www.routeviews.org/routeviews/map/

Route Archiving, Collection, Monitoring, & Probing

- **You can get real time feeds from both**, or browse their stored archives of table dumps and bgp updates.
- RouteViews & RIPE RIS **only see what they are sent**.
- **You should also monitor your own views**!
- **BMP** allows you to monitor your own feeds pre-policy, meaning before you filter out routes you'd otherwise reject for a variety of reasons. You can also BMP monitor post-policy and local rib.

Route Archiving, Collection, Monitoring, & Probing

- **RIPE Atlas**: A global network of devices, called probes and anchors, that actively measure internet connectivity. You can even run your own! Requires credits to run traces/pings/dig.
- **RIPE BGPlay**: Play back what occurred to a BGP route over a period of time visually and over API!
- **RIPEStat**: Aggregates RIPE information on IP address space and ASNs in a single web tool and API!



Source: https://atlas.ripe.net/maps/

Route Archiving, Collection, Monitoring, & Probing

- **globaltraceoute.com**: Easy frontend for RIPE Atlas probes.
- **CAIDA's ASRank** and **Qrator.Radar**: BGP relationships, events, and network size.
  - Relationships between ASNs like provider, peer, customer.
- **bgproutes.io**: Aggregated route collecting databases together.
- **bgp.tools**: Network metadata aggregator and looking glasses.
- **bgp.potaroo.net** / **cidr-report.org**: Research, stats, graphs!

- **CloudFlare Radar**: Web dashboard and API that shows global Internet traffic, attack, trends and insights.
  - Free, but you need to sign up for an account to get an API token. Sources from **CloudFlare's own data, RIPE RIS, RouteViews, and more**!
- **NIST RPKI Monitor**: RPKI stats dashboard leveraging RIPE RIS and RouteViews

Route Archiving, Collection, Monitoring, & Probing

- **NLNOG RING**: Community of networks providing shell access to internal servers for debugging.
- **Georgia Tech’s GRIP**: Monitors for BGP hijacks
- **Georgia Tech’s IODA**: Monitors internet outages

Reputation & Geofeeds

- **Akamai**
- **Proofpoint**
- **Spamhaus**
- **Team Cymru**
- **SPUR**
- **geolocatemuch.com**

```
medill:~ aatac$ asnrel 20130
provider: 6939
provider: 6461
provider: 23352
provider: 22335
```

```
medill:~ aatac$ lgripe 140.192.0.0/16
Prefix          | AS Path                              | Communities
------          + -------                              + -----------
140.192.0.0/16  | 15692 2914 3257 23352 20130 20130    | 2914:420 2914:1001 2914:2000 2914:3000 3257:3257 15692:2000 15692:2100
140.192.0.0/16  | 8218 6461 20130 20130 20130 20130    | 8218:102 8218:20000 8218:20110
140.192.0.0/16  | 6908 6939 20130 20130 20130 20130    | 6908:0 6908:8330 65100:1 65101:44 65102:4401 65103:4408 65104:3 65105:4
140.192.0.0/16  | 36924 6939 20130 20130 20130 20130   |
140.192.0.0/16  | 48070 6939 20130 20130 20130 20130   |
140.192.0.0/16  | 6894 6939 20130 20130 20130 20130    |
140.192.0.0/16  | 31742 6939 20130 20130 20130 20130   |
```

```
medill:~ aatac$ lgripeinfo 140.192.0.0/16
Prefix           Peers   Upstream                   Origin               AS-Len   Prepends
------           -----   --------                   ------               ------   --------
140.192.0.0/16   16      22335 (MREN)               20130 (ASN-DEPAUL)   4        0
140.192.0.0/16   13      22335 (MREN)               20130 (ASN-DEPAUL)   5        0
140.192.0.0/16   4       22335 (MREN)               20130 (ASN-DEPAUL)   6        0
140.192.0.0/16   2       22335 (MREN)               20130 (ASN-DEPAUL)   3        0
140.192.0.0/16   33      23352 (SERVERCENTRAL)      20130 (ASN-DEPAUL)   5        1
140.192.0.0/16   32      23352 (SERVERCENTRAL)      20130 (ASN-DEPAUL)   6        1
140.192.0.0/16   11      23352 (SERVERCENTRAL)      20130 (ASN-DEPAUL)   7        1
140.192.0.0/16   1       23352 (SERVERCENTRAL)      20130 (ASN-DEPAUL)   4        1
140.192.0.0/16   1       23352 (SERVERCENTRAL)      20130 (ASN-DEPAUL)   8        1
140.192.0.0/16   189     6939 (HURRICANE)           20130 (ASN-DEPAUL)   6        3
140.192.0.0/16   35      6461 (MFNX)                20130 (ASN-DEPAUL)   6        3
140.192.0.0/16   23      6939 (HURRICANE)           20130 (ASN-DEPAUL)   5        3
140.192.0.0/16   15      6939 (HURRICANE)           20130 (ASN-DEPAUL)   7        3
140.192.0.0/16   5       6461 (MFNX)                20130 (ASN-DEPAUL)   5        3
140.192.0.0/16   2       6939 (HURRICANE)           20130 (ASN-DEPAUL)   8        3
140.192.0.0/16   1       6461 (MFNX)                20130 (ASN-DEPAUL)   7        3
140.192.0.0/16   1       6461 (MFNX)                20130 (ASN-DEPAUL)   8        3
140.192.0.0/16   1       6939 (HURRICANE)           20130 (ASN-DEPAUL)   10       3
140.192.0.0/16   1       6939 (HURRICANE)           20130 (ASN-DEPAUL)   9        3
```


WE'RE OUT OF TIME
NETFLIX


EngVid
PLEASE ASK
ME QUESTIONS


THANK YOU

FIN

Appendix

```
function lgripeinfo {
    # 1. Colors & Data Fetch
    local c1=$(tput setaf 6) c2=$(tput setaf 33) c3=$(tput setaf 208) c4=$(tput setaf 5) c5=$(tput setaf 246) cr=$(tput sgr0)
    local raw=$(curl -s "https://stat.ripe.net/data/looking-glass/data.json?resource=$1" | \
        jq -r '.data.rrcs[].peers[] | .prefix as $pfx | .as_path | split(" ") | select(length>0) | .[-1] as $o | (reduce .[] as $i (0; if $i==$o then .+1 else . end)) as $h | ($h-1) as $prep | ([.[] | select(.!=$o)] | if length>0 then .[-1] else "Direct"
end) as $up | length as $len | [$pfx, $up, $o, $len, $prep] | @tsv' | sort | uniq -c | sort -k 6n -k 1nr)

    # 2. Extract ASNs & Resolve Names (Parallelized)
    local map=$(echo "$raw" | awk '{print $3"\n"$4}' | grep -E "^[0-9]+$" | sort -u | xargs -P 20 -I {} sh -c '
        n=$(whois -h rr.ntt.net "AS$1" 2>/dev/null | grep -i "^as-name:" | head -n1 | sed "s/^as-name:[[:space:]]*//I" | cut -c1-25)
        echo "$1|${n:-Unknown}"
    ' _ {})

    # 3. Join & Print
    awk -F '|' -v p="$c1" -v u="$c2" -v o="$c3" -v c="$c4" -v f="$c5" -v r="$cr" '
    BEGIN {
        fmt = "%s%-18s %s%-8s %s%-35s %s%-35s %s%-8s %s%-8s%s\n"
        printf fmt, p, "Prefix", f, "Peers", u, "Upstream", o, "Origin", f, "AS-Len", c, "Prepends", r
        printf fmt, p, "------", f, "-----", u, "--------", o, "------", f, "------", c, "--------", r
    }
    NR==FNR && $1!="" { map[$1]=$2; next }
    {
        split($0, a, /[ \t]+/)
        un = (a[4] in map) ? " ("map[a[4]]")" : ""
        on = (a[5] in map) ? " ("map[a[5]]")" : ""
        printf fmt, p, a[3], f, a[2], u, a[4] un, o, a[5] on, f, a[6], c, a[7], r
    }' <(echo "$map") <(echo "$raw")
}
```

Appendix

```
function lgripe {
    # 1. Variables & Fetch
    local cp=$(tput setaf 33) ca=$(tput setaf 208) cc=$(tput setaf 5) cs=$(tput setaf 240) ch=$(tput setaf 196) cr=$(tput sgr0)

    # 2. Query, Align & Highlight
    curl -s "https://stat.ripe.net/data/looking-glass/data.json?resource=$1" | \
    jq -r --arg a "$2" --arg c "$3" '
        ["Prefix","|","AS Path","|","Communities"], ["------","+","-------","+","-----------"],
        (.data.rrcs[].peers[] | select(($a=="" or (.as_path|split(" ")|index($a))) and ($c=="" or (.community|tostring|contains($c))))) | [.prefix, "|", .as_path, "|", (.community // "-")]) | @tsv' | \
    column -t -s $'\t' | \
    awk -F '|' -v p="$cp" -v a="$ca" -v c="$cc" -v s="$cs" -v h="$ch" -v r="$cr" -v ta="$2" -v tc="$3" '
    NR==2 { print s $0 r; next }
    {
        ps=$2; cs_str=$3
        if (ta!="") { gsub(" "ta" ", " "h ta r a" ", ps); gsub("^"ta" ", h ta r a" ", ps); gsub(" "ta"$", " "h ta r a, ps) }
        if (tc!="") { gsub(tc, h tc r c, cs_str) }
        printf "%s%s%s%s|%s%s%s%s%s|%s%s%s%s\n", p, $1, r, s, r, a, ps, r, s, r, c, cs_str, r
    }'
}
```

Appendix

```
function asnrel() {
    local asn="${1:?ASN argument required}"

    # Construct the GraphQL JSON payload safely
    local payload
    payload=$(jq -n --arg q "{ asn(asn:\"${asn}\") { asnLinks(first:40000) { edges { node { relationship asn1 { asn } } } } } }" '{query: $q}')

    # Fetch data and format output as "relationship: asn"
    curl -s -H "Content-Type: application/json" \
        -X POST \
        --data "${payload}" \
        "https://api.asrank.caida.org/v2/graphql" | \
    jq -r '.data.asn.asnLinks.edges[].node? // empty | "\(.relationship): \(.asn1.asn)"'
}
```