RIPE Atlas Tutorial

SAFNOG-4, TZ

Amreesh Phokeer
AFRINIC
RIPE Atlas

- Introduction to RIPE Atlas
- Using RIPE Atlas as a Visitor
- Looking up Public Probes
- Finding Results of Public Measurements
- Creating a Measurement
  - Demo and Exercise A
- Network Monitoring
  - Exercise B: Using Streaming API
- Command-line Interface Toolset
  - Exercise C: Using RIPE Atlas CLI
- Use Cases
- More RIPE Atlas Features
- Take Part in the Atlas Community
Introduction to RIPE Atlas
• Learn how to use RIPE Atlas for network monitoring and troubleshooting
• Learn how to create specific tailor-made measurements that suit your exact needs, using API calls or the command line interface
• Opportunity for hands-on practice
• Get answers to your questions
RIPE Atlas

From Wikipedia, the free encyclopedia

RIPE Atlas is a global, open, distributed internet measurement platform, consisting of thousands of measurement devices that measure internet connectivity in real time.
• We assume you have already used RIPE Atlas
• Do you have a RIPE NCC Access account?
  —If not - quickly create one: access.ripe.net
• Do you have credits to spend?
  —You get a voucher from us

—amreesh@afrinic.net
…is a global, open, distributed Internet measurement platform, consisting of thousands of measurement devices that measure Internet connectivity in real time
• Goal: View Internet reachability
• Probes hosted by volunteers
• Measurements towards root name servers
  - Visualised as Internet traffic maps
• Users can also run customised measurements
  - ping, traceroute, DNS & SSL/TLS, NTP and HTTP*
• Data publicly available
RIPE Atlas is a global active measurements platform
• Goal: view Internet reachability
• Probes hosted by volunteers
• Data publicly available

atlas.ripe.net
• **Built-in** global measurements towards root nameservers
  - Visualised as Internet traffic maps

• **Built-in** regional measurements towards “anchors”

• **Users** can run customised measurements
  - ping, traceroute, DNS, SSL/TLS, NTP and HTTP
• 10,400+ probes connected (318RIPE Atlas Anchors)
• 5,700+ results collected per second
• 18,200+ measurements currently running
• v1 & v2: Lantronix XPort Pro
• v3: TP-Link TL-MR3020 powered from USB port
  - Does not work as a wireless router
  - Same functionality as the old probe
• RIPE Atlas anchor: Soekris net6501-70
• 10100 + probes
• Countries: 177 (90.3%)
• Originating ASNs:
  3552 (IPv4) = 6.1% coverage
  1347 (IPv6) = 9.7% coverage
RIPE Atlas Overview (2)

RIPE Atlas Controllers

Web Interface

User Server

Probes

User
• Six types of measurements: ping, traceroute, DNS, SSL/TLS, NTP and HTTP (to anchors)
• APIs and CLI tools to start measurements and get results
• Streaming data for real-time results
• New: “Time Travel”, LatencyMON, DomainMON, Tracemon
• Status checks (Icinga & Nagios)
- https://atlas.ripe.net
- Users mailing list: ripe-atlas@ripe.net
- Articles & updates on RIPE Labs: https://labs.ripe.net/atlas
- Questions and bugs: atlas@ripe.net
- Twitter: @RIPE_Atlas and #RIPEAtlas
Using
RIPE Atlas As a Visitor
Internet Maps

**DNS Root Instances**
Shows, for each probe, which root DNS server instance the probe ends up querying when they ask a particular root server. In other words, it shows the "gravitational radius" for root DNS server instances.

**Comparative DNS Root RTT**
Shows a comparison of response time for DNS SOA queries to all the root DNS servers. For each probe, a marker shows the "best" root server with colour identifying the related minimum response time.

**Root Server Performance**
This map shows the reply time to the SOA query of a particular root DNS server, over the selected transport protocol (UDP, TCP or comparison of the two) for each probe.

**RTT to Fixed Destinations**
Shows the colour coding for the RTT value for the particular destination for each probe. The minimum / average / maximum values are based on standard "ping" measurements.

**Reachability of Fixed Destinations**
Shows if the particular fixed destination is reachable or not from each probe. Red markers indicate that the specific destination for these probes are unreachable and green reachable.
We display measurement results from the last hour only.
Probes per ASN (in RIPEstat)
Found 37 RIPE Atlas probes in this network:
- Abandoned (17)
- Connected (17)
- Disconnected (3)

Switch to table view

Showing results for MK as of 2016-04-05 09:31:05 UTC
Where we want to place probes
Looking Up Public Probes
Filter based on ASN, country, location...
Probe #10010 (Register)

General Information
- **Id**: 10010
- **MAC Address**: 8B:D1:11:A9:F3:2C
- **Architecture**: ti-mr3020
- **Firmware Version**: 4680 (1070)
- **Router Type**: Not set
- **DNS Entry**: Off
- **Shared Publicly**: Yes
- **User Tags**: NAT, Chello 200498
- **System Tags**: IPv4 Works, Auto GEOIP City, IPv4-CRC1019

Management Sharing
Only the probe host is permitted to administer this probe.

Connection & Traffic
- **Connected Time**: 3 days, 9 hours
- **Network Activity**: Graph showing data and packet rates over time

Location
The displayed location is an automatic best guess of the city based on IP address. By manually setting a more accurate location you can help improve the usefulness and correctness of RIPE Atlas.
- Replace multiple RRD graphs: zoom in/out in time, in the same graph
- Easier visualisation of an event’s details
- Selection of RTT class (max, min, average)
Finding Results of Public Measurements

amreesh@afrinic.net
- https://atlas.ripe.net/measurements/
• List of probes: sortable by RTT

• Map: colour-coded by RTT

• LatencyMON: compare multiple latency trends
• **TraceMON**: network topology, latency and nodes information

• **OpenIPMap**: hops geolocation on map (prototype)
- List of probes, colour-coded number of hops
• Map, colour-coded response time or diversity

• List of probes, sortable by response time
• Click on “Results”, then “Download”
• Or URL
• Or API
• Results in JSON
• Libraries for parsing on GitHub
Looking at the Result

```
{"af":6, "avg": 61.32,
"dst_addr":"2a00:1450:4004:802::1014", "dst_name":"www.google.com",
"dup":0,
"from":"2001:8a0:7f00:b201:220:4aff:fec5:5b5b",
"fw":4660,"lts":411,
"max":62.148,"min":60.372,
"msm_id":1004005, "msm_name": "Ping",
"prb_id":722,"proto":"ICMP","rcvd":10,
"result": [{"rtt":62.148},{"rtt":61.437},{"rtt":61.444},{"rtt":61.448},{"rtt":61.433},{"rtt":61.532},{"rtt":60.372},{"rtt":60.373},{"rtt":61.384},{"rtt":61.267}],
"sent":10,"size":64,
"src_addr":"2001:8a0:7f00:b201:220:4aff:fec5:5b5b",
"step":240,"timestamp":1410220847,"ttl":54,"type":"ping"},
```

**Destination (IP & name)**

**Source (probe public IP address)**

**Packet loss: difference between sent & received!**

**Reference (msm ID)**
Search for Measurements by Target in RIPEstat

Go to “RIPEstat > RIPE Atlas Activity”
• If you know the measurement ID:
  - https://atlas.ripe.net/measurements/ID
  - https://atlas.ripe.net/measurements/2340408/
• Many measurements already running!
• Search for existing public measurements first…
• Only then schedule your own measurement
Creating a Measurement
• Customer problem: cannot reach your server
  - Schedule measurements (pings or traceroutes) from up to 1,000 RIPE Atlas probes worldwide to check where the problem is

• Measuring packet loss on suspected “bad” link

• Testing anycast deployment
• RIPE NCC Access account?
  - If not, create one: ripe.net/register

• Do you have credits to spend?
  - Redeem voucher

• Redeem LIR credits monthly
• Log in to atlas.ripe.net
  - Use your RIPE NCC Access account
  - Same account for LIR Portal, RIPE Atlas, RIPEstat, RIPE Labs...
  - Create an account if you don’t already have one
• Measurements cost credits
  - ping = 10 credits, traceroute = 20, etc.

• Why? Fairness and to avoid overload
• Spending limit and max number of measurements
How can you earn credits?

- Hosting a RIPE Atlas probe
- Being a RIPE NCC member
- Hosting an anchor
- Sponsoring probes
- Being an ambassador
- Redeeming a voucher
### Credits overview

**My Atlas > Credits**

Give credits to someone

![Image of RIPE NCC website with credits section highlighted]

#### Credits Table

<table>
<thead>
<tr>
<th>Date</th>
<th>Time UTC</th>
<th>Probe ID</th>
<th>Action</th>
<th>Change</th>
<th>Old Balance</th>
<th>New Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>2016-02-02 01:00</td>
<td>00 UTC</td>
<td>Probe ID</td>
<td>Uptime extra</td>
<td>+100,000</td>
<td>152,817,561</td>
<td>153,917,561</td>
</tr>
<tr>
<td>2016-02-02 01:00</td>
<td>00 UTC</td>
<td>Probe ID</td>
<td>Uptime extra</td>
<td>+100,000</td>
<td>152,817,561</td>
<td>153,917,561</td>
</tr>
<tr>
<td>2016-02-02 01:00</td>
<td>00 UTC</td>
<td>Probe ID</td>
<td>Uptime extra</td>
<td>+100,000</td>
<td>152,817,561</td>
<td>153,917,561</td>
</tr>
<tr>
<td>2016-02-02 01:00</td>
<td>00 UTC</td>
<td>Probe ID</td>
<td>Uptime extra</td>
<td>+100,000</td>
<td>152,817,561</td>
<td>153,917,561</td>
</tr>
</tbody>
</table>
• Log in to atlas.ripe.net
• Four methods:
  1. Quick and easy
  2. Advanced GUI usage
  3. API (curl and JSON code)
  4. CLI
• Mostly used for a periodic, long-term measurement
  - Or “One-off”
• Choose type, target, frequency, start/end time, # of probes, region…
• Each measurement will have **unique ID**
• “API Compatible Specification” is generated too
• Periodic, long-term measurement
  - Single measurement? Choose “one-off”
• Choose type, target, frequency, number of probes, region…
• You will spend credits
• Each measurement: unique ID
• Using command-line and scripting: Application Programming Interface (API)
  - https://atlas.ripe.net/docs/api/v2/manual/measurements/types/
  - https://atlas.ripe.net/keys/

• You will need API keys
  - To create measurements without logging in
  - To securely share your measurement data
Create a New Measurement

Step 1: Definitions

- Target: bbc.co.uk
- Description: Ping measurement to bbc.co.uk
- Interval: 240
- Address Family*: IPv4
- Packets: 3
- Size: 48

Advanced Options

Step 2: Probe Selection

- Worldwide: 10

Step 3: Timing

- This is a One-off: false
- Start time (UTC): As soon as possible
- Stop time (UTC): Never

Measurement API Compatible Specification

Create My Measurement(s)
Measurement API Compatible Specification

```bash
curl --dump-header - -H "Content-Type: application/json" -H "Accept: application/json" -X POST -d '{
  "definitions": [
    {
      "target": "nrc.nl",
      "af": 4,
      "packets": 3,
      "size": 48,
      "description": "Ping measurement to nrc.nl",
      "interval": 240,
      "resolve_on_probe": false,
      "skip_dns_check": false,
      "type": "ping"
    }
  ]
}'
```
• Go to MyAtlas
• Click on “Create an API Key”
• Choose “permission”: “schedule new measurement”
• Careful! Time is UTC!
• Give it a label
The image shows a dialogue box titled "Create API Key". It contains fields for UUID, Created, Label, Valid from, Valid to, Enabled, and a section labeled "Grants".

The "Grants" section includes a list of permissions and targets, with one selected permission highlighted. The list includes:
- credits
- Get information about your credits
- Transfer credits to another user
- keys
- List all of your API keys
- Get info about a particular API key
- Update an existing API key
- Create a new API key
- Delete an API key

Additional permissions include:
- measurements
- Get results from a non-public measurement
- List your measurements
- Stop a running measurement
- Update an existing measurement
- Schedule a new measurement
- probes
- Get restricted information about a probe

The dialogue box also contains a note: "This key won't have any effect unless you grant one or more permissions to it."
DEMO
Create a Measurement (GUI)
Explore advanced parameters
Create a Measurement

Exercise A
• Create a ping measurement:
  - Involving ten probes
  - To a target of your choice
  - Source is your country
  - Duration of two days
• 1. Warm-up: Create a measurement using the GUI
• 2. Create API Key
• 3. Schedule a measurement using the API
Selecting probes with new set wizard
• Useful hint: once you generate a measurement, copy “API Compatible Specification” to text file.
• Take note of the measurement ID!
1. Click on “Create an API Key”
2. Permission: “schedule a new measurement”
3. “Target” is not applicable (N/A) for this type
1. Give it a label
2. Give it a duration of validity (leave empty for defaults)
3. “Key” value to be passed on to the API call (next step)
• Schedule a measurement using API
  - Use the “key” you just generated
  - Hint: copy and past API call syntax from the measurement generated by the GUI

• Example:

```sh
curl -H "Content-Type: application/json" -H "Accept: application/json" -X POST -d '{ "definitions": [ { "target": "ping.xs4all.nl", "description": "My First API Measurement", "type": "ping", "af": 4 } ], "probes": [ { "requested": 10, "type": "country", "value": "RS" } ] }'
https://atlas.ripe.net/api/v1/measurement/?key=YOUR_API_KEY
```
Measurement API Compatible Specification

curl -d dump-header --H "Content-Type: application/json" --H "Accept: application/json" -X POST -d '{
  "definitions": [
    {
      "target": "nrc.nl",
      "af": 4,
      "packets": 3,
      "size": 48,
      "description": "Ping measurement to nrc.nl",
      "interval": 240,
      "resolve_on_probe": false,
      "skip_dns_check": false,
      "type": "ping"
    }
  ]
}

https://atlas.ripe.net/api/v2_measurements/?key=YOUR_KEY_HERE
becha$ curl -H "Content-Type: application/json" -H "Accept: application/json" -X POST -d '{ "definitions": [ { "target": "ping.xs4all.nl", "description": "My First Measurement", "type": "ping", "af": 4 } ], "probes": [ { "requested": 10, "type": "country", "value": "RS" } ] }' https://atlas.ripe.net/api/v1/measurement/?key=7b4c3441-4504-4d83-9ed7-fbf1a807d060

{"measurements": [2421551]} becha$
• Create a TCP traceroute measurement:
  - Involving 10 probes
  - The closest five to the training course location (Vienna)
  - To a target of your choice
  - Duration of two days
  - Repeated every 60 seconds
Network Monitoring
• Integrate “status checks” with existing monitoring tools (Icinga, Nagios)
• Using real-time data streaming
  - Server monitoring
  - Detecting and visualising outages
1. Create a RIPE Atlas ping measurement
2. Go to “status checks” URL (RESTful API call)
3. Documentation:
   - https://atlas.ripe.net/docs/api/v2/manual/measurements/status-checks.html
4. Add your alerts in Nagios or Icinga
Results:

https://atlas.ripe.net/api/v2/measurements/2340408/status-check?max_packet_loss=20
{ "total_alerts": 38, "global_alert": true, "probes": { "109": { "source": "Area: NW", "last_packet_lost": 8, "lost": 58.754, "alert": false }, "1004": { "alert": null, "last": null }, "1212": { "alert": null, "last_packet_lost": 100, "alert": true, "source": "Area: MP", "alert_response": [ "loss" ] }, "1133": { "alert": null, "last": null }, "109": { "source": "Area: NW", "alert": true, "alert_response": [ "loss" ] } } }
• Allows users to receive the measurement results as soon as they are sent by the probes in real time
  - Publish/subscribe through web sockets

• There are three types of data:
  - Measurement results
  - Probe connection status events
  - Measurements metadata
• Visualising network outages
  - http://sg-pub.ripe.net/demo-area/atlas-stream/conn/
• Real-time server and performance monitoring
• Filtering and reusing measurement results
• Documentation:
  - https://atlas.ripe.net/docs/result-streaming/
1. Create a socket
2. Create a callback (function)
   - for each event type
   - to be executed for each message received
3. Start listening to the channel
4. Declare what you want to receive for that event type
1. Create a RIPE Atlas ping measurement

2. Go to “Status Checks” URL

3. Add your alerts in Icinga or Nagios
• Status Checks via RIPE Atlas' RESTful API
  → https://atlas.ripe.net/api/v2/measurements/MEASUREMENT_ID/status-check
• Define alert parameters:
  - Threshold  % of probes successfully received a reply
  - How many most recent measurements to base it on
  - Acceptable maximum packet loss
• Community of operators contributed configuration code!
  - Making use of the built-in “check_http” plugin

• GitHub repo examples:

• Post on Icinga blog:
  - https://www.icinga.org/2014/03/05/monitoring-ripe-atlas-status-with-icinga-2/
Using streaming API

Exercise B
• Preconfigure web browser
• In Safari
  - Preferences > Advanced> Show Develop menu
• Chrome or Firefox needs no reconfiguration
• Scenario: customers complain it takes a long time to reach your server
• Action: ping your server from 50 probes
  - Choose acceptable latency threshold
  - Notice and react when you start receiving samples
• Task: Use the ping measurement ID 1791207
  - Choose which threshold (e.g. greater than 30ms)
  - Impose threshold on “min” (the minimum result of the three ping attempts)
1. [http://atlas.ripe.net/webinar/streaming01.html](http://atlas.ripe.net/webinar/streaming01.html)
2. Open the development console
3. Wait for results to arrive
4. Save the HTML file locally and edit the code
// The following file is needed for the streaming

<script src="https://atlas-stream.ripe.net/socket.io.js"></script>

<script>
  // Create a connection
  var socket = io("https://atlas-stream.ripe.net");
  socket.on("/stream/socket.io", function(result)
  {
    console.log("I received ", result); // Print the result in the console
  });

  // Subscribe to results coming from all the probes involved in the measurement 2340456
  socket.emit("atlas_subscribe", { stream_type: "result", msn: 1234567 });
</script>
Streaming results before editing
<!-- The following file is needed for the streaming -->
<script src="https://atlas-stream.ripe.net/socket.io.js"></script>

<script>
// Create a connection
var socket = io("https://atlas-stream.ripe.net", { path : "/stream/socket.io" });

// Declare a callback to be executed when a measurement result is received
socket.on("atlas_result", function(result){
    console.log("I received ", result); // Print the result in the console
});

// Subscribe to results coming from all the probes involved in the measurement
socket.emit("atlas_subscribe", { stream_type: "result", msm: 1791207 });

</script>
{  "af": 4,  "prb_id": 10048,  "result": [  {   "rtt": 19.212845  },  {   "rtt": 19.28432  },  {   "rtt": 18.943275  }  ],  "ttl": 55,  "avg": 19.1468133333,  "size": 48,  "from": "81.157.60.234",  "proto": "ICMP",  "timestamp": 1479310322,  "dup": 0,  "type": "ping",  "sent": 3,  "msm_id": 2340408,  "fw": 4740,  "max": 19.28432,  "step": 60,  "src_addr": "192.168.2.101",  "rcvd": 3,  "msm_name": "Ping",  "lts": 70,  "dst_name": "193.0.10.197",  "min": 18.943275,  "group_id": 2340408,  "dst_addr": "193.0.10.197"  },
Common parameters (in addition to stream_type)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>prb</td>
<td>A specific probe ID. If you don’t set this parameter, you will receive results from all the probes</td>
</tr>
<tr>
<td>acceptedFields</td>
<td>A list of accepted fields name, the messages will be pruned server side. If you don’t set this parameter you will receive all the fields</td>
</tr>
<tr>
<td>enrichProbes</td>
<td>If you want to enrich the information received with the “probestatus” stream about the probes (e.g. lat, long), set this option to true</td>
</tr>
<tr>
<td>equalsTo</td>
<td>Allows to filter by values. E.g. with {status: &quot;connected&quot;, asn: &quot;33334444&quot;} you will receive all the messages with a connected status and ASN equals to 3333 or 4444</td>
</tr>
<tr>
<td>lessThan</td>
<td>Allows to filter by values. E.g. with {valueX: 15} you will receive all the messages with a valueX less than 15</td>
</tr>
<tr>
<td>greaterThan</td>
<td>Allows to filter by values. E.g. with {valueX: 15} you will receive all the messages with a valueX greater than 15</td>
</tr>
</tbody>
</table>

Parameters for "result" stream_type

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msm</td>
<td>A specific measurement ID</td>
</tr>
<tr>
<td>type</td>
<td>Streams all the results of the specified type, i.e. ping, traceroute, ntp, http, dns, ssl</td>
</tr>
<tr>
<td>sourceAddress</td>
<td>Streams all the results coming from a probe having the specified address</td>
</tr>
<tr>
<td>sourcePrefix</td>
<td>Streams all the results coming from a probe having an address in the specified prefix</td>
</tr>
<tr>
<td>destinationAddress</td>
<td>Streams all the results measuring the specified address</td>
</tr>
<tr>
<td>destinationPrefix</td>
<td>Streams all the results measuring an address in the specified prefix</td>
</tr>
<tr>
<td>passThroughHost</td>
<td>Streams all the traceroutes passing through the specified host. Only for traceroute measurements.</td>
</tr>
<tr>
<td>passThroughPrefix</td>
<td>Streams all the traceroutes passing through a host in the specified prefix. Only for traceroute measurements.</td>
</tr>
<tr>
<td>sendBacklog</td>
<td>Immediately fetch the last few minutes of results for a specific measurement ID. Mostly intended to cover the gap between the last available data in the REST API and the currently streamed results, and to recover results that might have been missed during brief disconnections.</td>
</tr>
<tr>
<td>buffering</td>
<td>If set to true, the samples will be sent in bundles in order to reduce the network overhead. When this option is true, the socket is not volatile. False by default.</td>
</tr>
</tbody>
</table>
<!-- The following file is needed for the streaming -->
<script src="https://atlas-stream.ripe.net/socket.io.js"></script>
<script>
  // Create a connection
  var socket = io("https://atlas-stream.ripe.net", { path : "/stream/socket.io" });

  // Declare a callback to be executed when a measurement result is received
  socket.on("atlas_result", function(result){
      console.log("I received ", result); // Print the result in the console
  });

  // Subscribe to results coming from all the probes involved in the measurement 2340808
  socket.emit("atlas_subscribe", { stream_type: "result", msm: 1791207, greaterThan: {min:30} });
</script>
<script src="https://atlas-stream.ripe.net/socket.io.js"></script>

// Create a connection
var socket = io("https://atlas-stream.ripe.net", { path : "/stream/socket.io" });

// Declare a callback to be executed when a measurement result is received
socket.on("atlas_result", function(result){
    console.log("I received ", result); // Print the result in the console
});

// Subscribe to results coming from all the probes involved in the measurement 2340408
socket.emit("atlas_subscribe", { stream_type: "result", msm: 1791207, greaterThan: {min:30} });

</script>
<table>
<thead>
<tr>
<th>Elements</th>
<th>Network</th>
<th>Resources</th>
<th>Timelines</th>
</tr>
</thead>
<tbody>
<tr>
<td>![received](af: 4, prb_id: 6893, result: [{rtt: 113.02871}, {rtt: 112.869494}, {rtt: 112.920425}], …)</td>
<td>![received](af: 4, prb_id: 16010, result: [{rtt: 131.01558}, {rtt: 130.12254}, {rtt: 131.107265}], …)</td>
<td>![received](af: 4, prb_id: 17914, result: [{rtt: 119.536135}, {rtt: 120.41047}, {rtt: 120.15117}], …)</td>
<td>![received](af: 4, prb_id: 19798, result: [{rtt: 43.87587}, {rtt: 43.370515}, {rtt: 43.310655}], …)</td>
</tr>
</tbody>
</table>
• Same situation as in the exercise before, but you didn’t schedule a measurement in advance
  - You don’t have a measurement ID

• You want to get all the measurements reaching 216.58.212.227

• Now restrict the results to just include ping measurements
socket.emit("atlas_subscribe", { 
    stream_type: "result",
    type: "ping",
    destinationAddress: "216.58.212.227"
});

Documentation:

https://atlas.ripe.net/docs/result-streaming/
• https://atlas.ripe.net/docs/rest/
• https://github.com/RIPE-NCC/ripe.atlas.sagan
• https://atlas.ripe.net/docs/measurement-creation-api/
  • https://atlas.ripe.net/doc/credits
  • https://atlas.ripe.net/doc/udm
    – https://atlas.ripe.net/keys/
    – https://atlas.ripe.net/docs/keys2/
• https://atlas.ripe.net/docs/measurement-latest-api/
  - Widget monitoring value in real time (100 probes pinging websites worldwide)
  - Alert based on average measurements per hour
  - Big network event, e.g. Internet outage in a region
  - DNS domain monitoring; configurable measurements using ten RIPE Atlas anchors

• https://labs.ripe.net/Members/suzanne_taylor_muzzin/ripe-atlas-latest-results-api-and-parsing-library
Command-line Interface (CLI) Toolset
• Familiar output (ping, dig, traceroute)
• Linux/OSX
• Windows [experimental]
• In red you can see the limited command set for the Windows installation of ripe-atlas
• Open source
  - RIPE NCC led community contribution

• Documentation
  - https://ripe-atlas-tools.readthedocs.org/

• Source:
  - https://github.com/RIPE-NCC/ripe-atlas-tools/
- Install RIPE Atlas tools
  - OSX: sudo easy_install pip
  - sudo pip install ripe-atlas-tools
  - Linux: Available from many package repositories
  - ...or same as in OSX
• Reuse the API key of the first exercise
  - Or create a new one at https://atlas.ripe.net/keys/
• Configure your CLI
  - ripe-atlas configure --set authorisation.create=MY_API_KEY
• Fetch the ping measurement 2340408
  - ripe-atlas report 2340408
• Search all probes in AS 3333
  - ripe-atlas probe-search --asn 3333

• Show specific fields
  - ripe-atlas probe-search --asn 3333 --field asn_v6 --field country --field description --field status

• Search for probes in and around Paris
  - ripe-atlas probe-search --location "Paris, France" --radius 15
ripe-atlas probes --asn 3333

<table>
<thead>
<tr>
<th>ID</th>
<th>asn.v4</th>
<th>asn.v6</th>
<th>country</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Connected</td>
</tr>
<tr>
<td>2</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Connected</td>
</tr>
<tr>
<td>3</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Connected</td>
</tr>
<tr>
<td>4</td>
<td>3333</td>
<td>3333</td>
<td>br</td>
<td>Abandoned</td>
</tr>
<tr>
<td>5</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Connected</td>
</tr>
<tr>
<td>6</td>
<td>3333</td>
<td>3333</td>
<td>ch</td>
<td>Disconnected</td>
</tr>
<tr>
<td>7</td>
<td>3333</td>
<td>3333</td>
<td>at</td>
<td>Disconnected</td>
</tr>
<tr>
<td>8</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Disconnected</td>
</tr>
<tr>
<td>9</td>
<td>3333</td>
<td>3333</td>
<td>nz</td>
<td>Disconnected</td>
</tr>
<tr>
<td>10</td>
<td>3333</td>
<td>3333</td>
<td>us</td>
<td>Disconnected</td>
</tr>
<tr>
<td>11</td>
<td>3333</td>
<td>3333</td>
<td>se</td>
<td>Disconnected</td>
</tr>
<tr>
<td>12</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Disconnected</td>
</tr>
<tr>
<td>13</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Disconnected</td>
</tr>
<tr>
<td>14</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Disconnected</td>
</tr>
<tr>
<td>15</td>
<td>3333</td>
<td>3333</td>
<td>fr</td>
<td>Disconnected</td>
</tr>
<tr>
<td>16</td>
<td>3333</td>
<td>3333</td>
<td>de</td>
<td>Disconnected</td>
</tr>
<tr>
<td>17</td>
<td>3333</td>
<td>3333</td>
<td>de</td>
<td>Disconnected</td>
</tr>
<tr>
<td>18</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Disconnected</td>
</tr>
<tr>
<td>19</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Disconnected</td>
</tr>
<tr>
<td>20</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Disconnected</td>
</tr>
<tr>
<td>21</td>
<td>3333</td>
<td>3333</td>
<td>nl</td>
<td>Disconnected</td>
</tr>
<tr>
<td>22</td>
<td>3333</td>
<td>3333</td>
<td>us</td>
<td>Disconnected</td>
</tr>
<tr>
<td>23</td>
<td>3333</td>
<td>3333</td>
<td>ru</td>
<td>Disconnected</td>
</tr>
<tr>
<td>24</td>
<td>3333</td>
<td>3333</td>
<td>no</td>
<td>Disconnected</td>
</tr>
<tr>
<td>25</td>
<td>3333</td>
<td>3333</td>
<td>cy</td>
<td>Disconnected</td>
</tr>
<tr>
<td>26</td>
<td>3333</td>
<td>3333</td>
<td>gb</td>
<td>Disconnected</td>
</tr>
</tbody>
</table>
ripe-atlas probe-search --asn 3333 --field asn_v6 --field country --field description --field status

<table>
<thead>
<tr>
<th>ASN_v6</th>
<th>Country</th>
<th>Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>3333</td>
<td>ni</td>
<td>SG office 1</td>
<td>Connected</td>
</tr>
<tr>
<td>3333</td>
<td>ni</td>
<td>SG office 2</td>
<td>Connected</td>
</tr>
<tr>
<td>3333</td>
<td>br</td>
<td>Big BR</td>
<td>Abandoned</td>
</tr>
<tr>
<td>3333</td>
<td>us</td>
<td>The WTRC</td>
<td>Connected</td>
</tr>
<tr>
<td>3333</td>
<td>us</td>
<td>CHUV</td>
<td>Connected</td>
</tr>
<tr>
<td>3333</td>
<td>us</td>
<td>REPLACED BY PROBE 3129</td>
<td>Disconnected</td>
</tr>
<tr>
<td>3333</td>
<td>us</td>
<td>Oulu</td>
<td>Disconnected</td>
</tr>
<tr>
<td>3333</td>
<td>us</td>
<td>Provo 1</td>
<td>Abandoned</td>
</tr>
<tr>
<td>3333</td>
<td>us</td>
<td>Zetup AB</td>
<td>Disconnected</td>
</tr>
<tr>
<td>3333</td>
<td>ni</td>
<td>Disconnected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>ni</td>
<td>Disconnected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>ni</td>
<td>Disconnected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>ni</td>
<td>Disconnected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>ni</td>
<td>Disconnected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>fr</td>
<td>Connected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>de</td>
<td>Connected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>de</td>
<td>Disconnected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>de</td>
<td>Disconnected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>ru</td>
<td>Connected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>no</td>
<td>Disconnected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>cy</td>
<td>Connected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>gb</td>
<td>Connected</td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>gb</td>
<td>Connected</td>
<td></td>
</tr>
</tbody>
</table>

Showing 25 of 128 total probes.
ripe-atlas probe-search --location "Paris, France" --radius 15
• Create a ping measurement to wikipedia.org
  - One-off, default parameters
  - ripe-atlas measure ping --target wikipedia.org

Looking good! Your measurement was created and details about it can be found here:

https://atlas.ripe.net/measurements/3499718/

Connecting to stream...

48 bytes from probe #18433 94.112.176.45 to 91.198.174.192 (91.198.174.192); ttl=50 times:41.979, 41.492, 40.769,
48 bytes from probe #20111 37.151.238.188 to 91.198.174.192 (91.198.174.192); ttl=57 times:100.511, 100.136, 100.325,
48 bytes from probe #25803 176.193.48.211 to 91.198.174.192 (91.198.174.192); ttl=59 times:47.967, 47.476, 47.463,
48 bytes from probe #28313 5.199.160.0 to 91.198.174.192 (91.198.174.192); ttl=58 times:36.581, 36.245, 36.285,
48 bytes from probe #22573 89.176.43.44 to 91.198.174.192 (91.198.174.192); ttl=52 times:28.747, 27.712, 28.446,
48 bytes from probe #19413 89.71.47.56 to 91.198.174.192 (91.198.174.192); ttl=51 times:49.89, 49.779, 58.277,
48 bytes from probe #18635 78.52.132.137 to 91.198.174.192 (91.198.174.192); ttl=57 times:37.462, 38.895, 37.73,
48 bytes from probe #23223 62.65.126.46 to 91.198.174.192 (91.198.174.192); ttl=53 times:23.169, 23.432, 33.067,
48 bytes from probe #17511 87.81.148.2 to 91.198.174.192 (91.198.174.192); ttl=56 times:13.281, 12.885, 13.039,
48 bytes from probe #12584 46.175.22.202 to 91.198.174.192 (91.198.174.192); ttl=59 times:36.873, 35.788, 35.883,
- ripe-atlas measure ping --target wikipedia.org

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Geo-specific from 20 probes from Canada:
  - ripe-atlas measure ping --target example.com --probes 20 --from-country ca

• 20 Canadian probes that definitely support IPv6:
  - ripe-atlas measure ping --target example.com --probes 20 --from-country ca --include-tag system-ipv6-works

• Create a recurring measurement:
  - ripe-atlas measure ping --target example.com --interval 3600
Using RIPE Atlas CLI

Exercise C
• UNIX/LINUX/OSX:

• Terminal:
  - `sudo easy_install pip`
  - `sudo pip install ripe-atlas-tools`
  - choose “Install” in pop-up
  - `ripe-atlas configure --set authorisation.create=MY_API_KEY`
• Use the traceroute command to test the reachability of:
  - wikipedia.org
  - on TCP port 443
  - from 20 probes
  - in France
• Use the traceroute command to test the reachability of:
  - wikipedia.org
  - on TCP port 443
  - from 20 probes in
  - France

  - ripe-atlas measure traceroute --protocol TCP --target wikipedia.org --port 443 --probes 20 --from-country fr
ripe-atlas measure traceroute --protocol TCP --target wikipedia.org --port 443 --probes 20 --from

```
Looking good! Your measurement was created and details about it can be found here:

https://atlas.ripe.net/measurement/963341/
```

```
Connecting to stream...
The event name 'result' will soon be deprecated. Use the real event name 'atlas_result' instead.
```

Probes:

<table>
<thead>
<tr>
<th>Probe</th>
<th>Result</th>
<th>Time</th>
<th>Time</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>#02448</td>
<td>1 192.168.1.1</td>
<td>1.815 ms</td>
<td>0.050 ms</td>
<td>8.377 ms</td>
</tr>
<tr>
<td></td>
<td>1 213.41.216.254</td>
<td>0.878 ms</td>
<td>0.82 ms</td>
<td>0.861 ms</td>
</tr>
<tr>
<td></td>
<td>3 213.41.193.221</td>
<td>0.899 ms</td>
<td>0.929 ms</td>
<td>2.115 ms</td>
</tr>
<tr>
<td></td>
<td>4 213.41.193.282</td>
<td>1.64 ms</td>
<td>1.59 ms</td>
<td>1.481 ms</td>
</tr>
<tr>
<td></td>
<td>5 105.42.144.118</td>
<td>0.792 ms</td>
<td>1.536 ms</td>
<td>4.899 ms</td>
</tr>
<tr>
<td></td>
<td>6 77.100.149.94</td>
<td>12.31 ms</td>
<td>12.610 ms</td>
<td>11.633 ms</td>
</tr>
<tr>
<td></td>
<td>7 77.100.136.144</td>
<td>11.671 ms</td>
<td>11.240 ms</td>
<td>11.374 ms</td>
</tr>
<tr>
<td></td>
<td>8 91.190.174.254</td>
<td>11.566 ms</td>
<td>11.745 ms</td>
<td>11.16 ms</td>
</tr>
<tr>
<td></td>
<td>9 91.190.174.192</td>
<td>11.545 ms</td>
<td>11.883 ms</td>
<td>11.442 ms</td>
</tr>
</tbody>
</table>

Probes:

<table>
<thead>
<tr>
<th>Probe</th>
<th>Result</th>
<th>Time</th>
<th>Time</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>#0148</td>
<td>1 31.216.38.1</td>
<td>3.84 ms</td>
<td>4.064 ms</td>
<td>2.936 ms</td>
</tr>
<tr>
<td></td>
<td>2 40.216.89.39</td>
<td>16.166 ms</td>
<td>26.504 ms</td>
<td>16.315 ms</td>
</tr>
<tr>
<td></td>
<td>3 40.211.116.3</td>
<td>16.563 ms</td>
<td>15.093 ms</td>
<td>15.552 ms</td>
</tr>
<tr>
<td></td>
<td>4 80.209.198.176</td>
<td>27.281 ms</td>
<td>27.28 ms</td>
<td>27.288 ms</td>
</tr>
<tr>
<td></td>
<td>5 91.186.174.182</td>
<td>36.764 ms</td>
<td>26.547 ms</td>
<td>36.137 ms</td>
</tr>
</tbody>
</table>

Probes:

<table>
<thead>
<tr>
<th>Probe</th>
<th>Result</th>
<th>Time</th>
<th>Time</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>#08429</td>
<td>1 79.99.164.253</td>
<td>15.514 ms</td>
<td>2.699 ms</td>
<td>2.168 ms</td>
</tr>
<tr>
<td></td>
<td>2 79.99.167.131</td>
<td>3.049 ms</td>
<td>2.627 ms</td>
<td>2.278 ms</td>
</tr>
<tr>
<td></td>
<td>3 92.115.145.78</td>
<td>4.328 ms</td>
<td>2.399 ms</td>
<td>1.443 ms</td>
</tr>
</tbody>
</table>
Use Cases
Using RIPE Atlas to Validate International Routing Detours
Anant Shah — 30 Jan 2017

A Quick Look at the Attack on Dyn
Massimo Candela — 24 Oct 2016
Contributors: Emile Aben

Using RIPE Atlas to Monitor Game Service Connectivity
Annika Wickert — 14 Sep 2016

Using RIPE Atlas to Measure Cloud Connectivity
Jason Read — 06 Sep 2016

Using RIPE Atlas to Debug Network Connectivity Problems
Stéphane Bonczmeyer — 10 May 2016
• Do paths between ASes stay in country?
• Any difference between IPv4 and IPv6?
• How many paths go via local IXP?
• Could adding peers improve reachability?

https://www.ripe.net/ixp-country-jedi

• Experimental tool
  - Feature requests welcome!
  - Depends on probe distribution in country
• Methodology
  - **Trace route mesh** between RIPE Atlas probes
  - Identifying ASNs in country using RIPEstat
  - Identifying IXP and IXP LANs in PeeringDB
• DDoS Attack on Dyn DNS Servers (Oct. 2016)
  - 10s millions devices - Mirai botnet
  - Legitimate requests
• Monitor Game Service Connectivity (Sept. 2016)
• Requirements:
  - Check General Reachability, Latency, Historical data
  - Supported by an active and helpful community
  - Integrate with their existing logging system
• Track down an outage in one upstream
• Became sponsors
• Amsterdam Power Outage (March 2015)
• When and where the outage was happening
More RIPE Atlas Features
• Use API keys to:
  - Create measurements without logging in
  - Securely share your measurement data with others

• To create, manage and delete API keys:
  - https://atlas.ripe.net/keys/
  - https://atlas.ripe.net/docs/keys2/

• Examples:
  - https://atlas.ripe.net/docs/rest/
• **Probes:**
  - Hardware trust material (regular server address, keys)
  - No open ports; initiate connection; NAT is okay
  - Don’t listen to local traffic
  - No passive measurements

• **Measurements triggered by “command servers”**
  - SSH connections from probe to server
  - initiated by probe

• **Measurement code published**
• RIPE Atlas:
  - Guaranteed to host a probe
  - Do NOT have to host probe to perform customised measurements
  - 1,000,000 extra credits monthly via LIR Portal
  - “Quick Look” measurements via LIR Portal
  - IPv6 reachability testing (free)
  - Share probe management with LIR colleagues

• RIPEstat:
  - Historical view of RIPE Database objects
• Make comparative measurements between probes
• Check for IXPs in the path
• Where is the traffic going?
• And other cool stuff!

https://github.com/emileaben/ixp-country-jedi
Take Part in the RIPE Atlas Community
• Volunteers host probes in homes or offices
• Organisations host RIPE Atlas anchors
• Sponsor organisations give financial support or host multiple probes in their own networks
• Ambassadors help distribute probes at conferences, give presentations, etc.
• Developers contribute free and open software
• Network operators create measurements to monitor and troubleshoot
• Researchers and students write papers
• Create a RIPE NCC Access account
• Go to https://atlas.ripe.net/apply
• You will receive a probe by post
• Register your probe
• Plug in your probe
• If you receive a probe from an ambassador (trainer, sponsor, someone at a conference), just register it and plug it in!
• https://atlas.ripe.net
• http://roadmap.ripe.net/ripe-atlas/

• Users’ mailing list: ripe-atlas@ripe.net
• Articles and updates: https://labs.ripe.net/atlas
• Questions and bugs: atlas@ripe.net
• Twitter: @RIPE_Atlas and #RIPEAtlas
• AFRINIC distributes probes, sponsors Anchors
• atlas@afrinic.net to apply:
  – AS Number
  – IPv4 address
  – IPv6 address
  – No filtering