



**RIPE
NCC**

Webinar: Advanced RIPE Atlas Usage

Vesna Manojlovic
Massimo Candela

RIPE NCC

Amsterdam | September 2015

- Learn how to:
 - Benefit from using RIPE Atlas measurements for network monitoring and troubleshooting
 - Use API calls to create measurements
 - Integrate RIPE Atlas with existing monitoring systems
- Opportunity for hands-on practice
- Get your questions answered by developers

- We assume you have already used RIPE Atlas
- Do you have a RIPE NCC Access account?
 - If not - quickly create one: ripe.net/register
- Do you have credits to spend?
 - If not - tell us your account in the chat window

- What is your background?
 - network operator?
 - software engineer / programmer?
 - data scientist?
 - sysadmin?
 - other? please specify :-)

- Introduction to RIPE Atlas
- Creating measurements
- Integration with network monitoring systems
- Real-time performance monitoring
- Take part in the RIPE Atlas community
- Additional slides

Introduction to RIPE Atlas

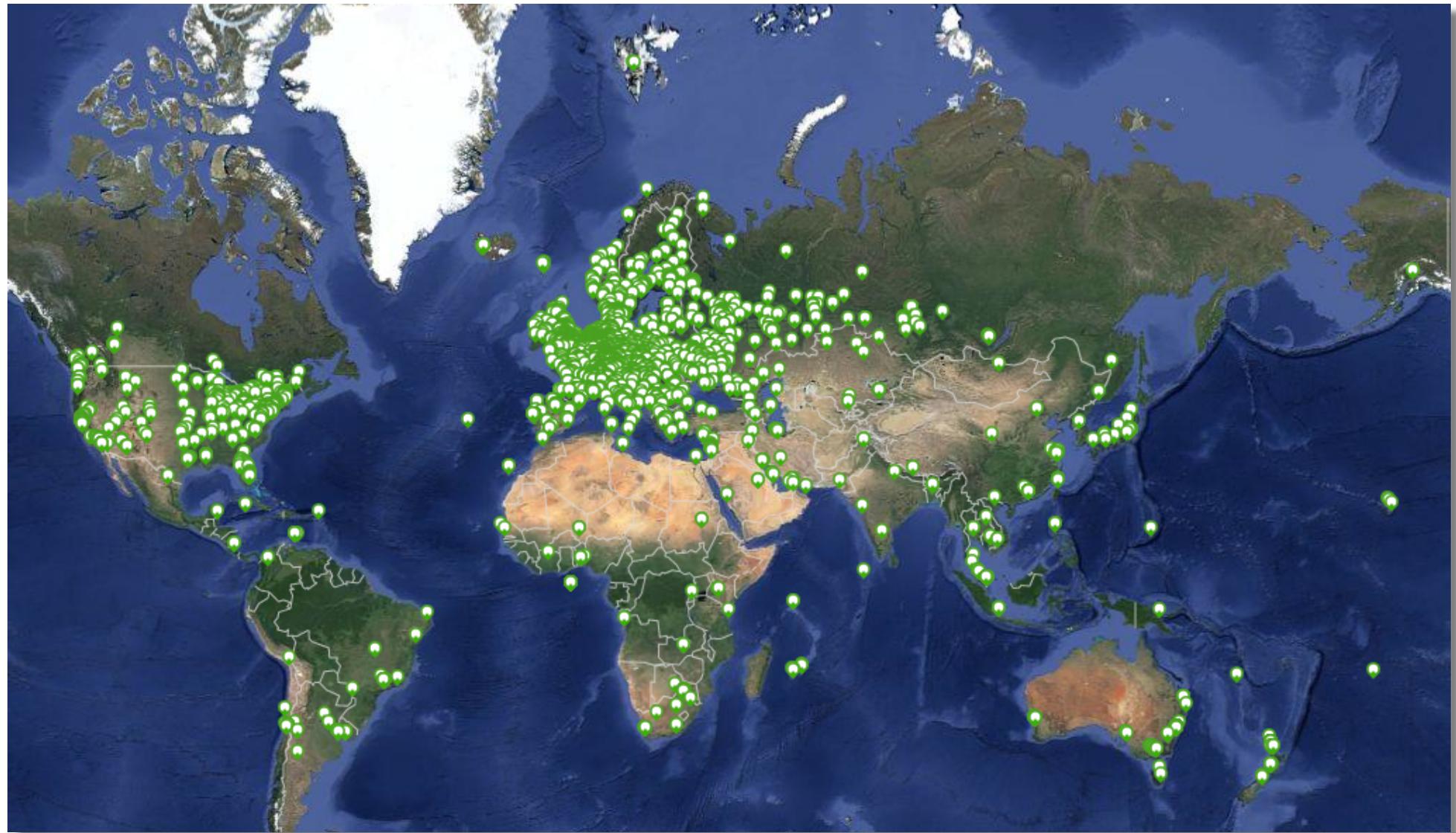


RIPE
NCC

- RIPE Atlas is a global active measurements platform
- Goal: view Internet reachability
- Probes hosted by volunteers
- Data publicly available

RIPE Atlas coverage

| 8

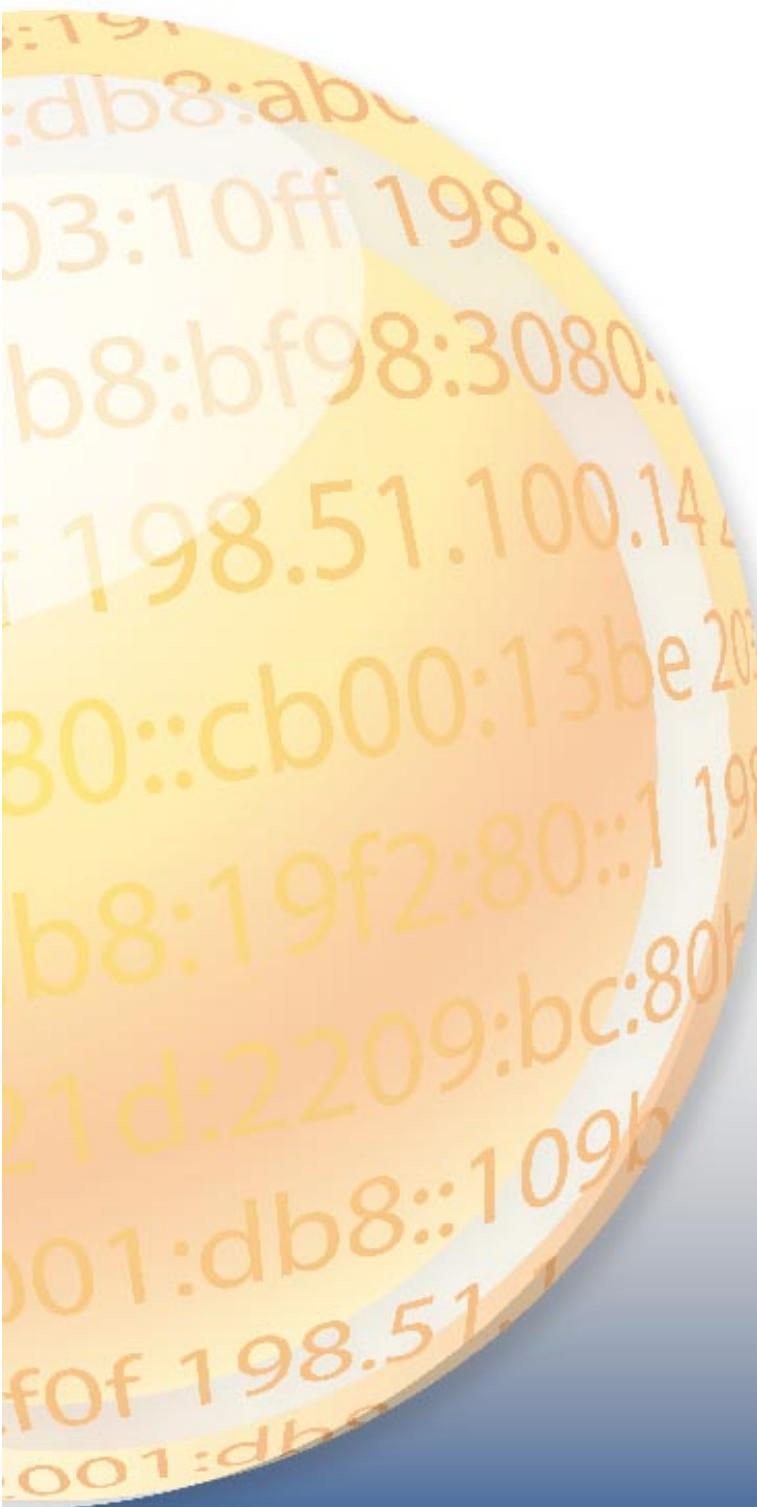


- Ongoing global measurements towards root nameservers
 - Visualised as Internet traffic maps
- Ongoing regional measurements towards “anchors”
- Users can run customised measurements
 - ping, traceroute, DNS, SSL/TLS and NTP

Numbers for mid-September 2015

| 10

- 8,600+ probes connected
- 5,000+ active users in the last quarter
- 2,500+ results collected per second
- 35,000+ customised measurements weekly



Creating a Measurement



**RIPE
NCC**

- A customer reports a problem: they cannot reach one of your servers
 - You can schedule pings or traceroutes from up to 500 RIPE Atlas probes from a particular region to check where the problem might be
- Measuring packet loss on a suspected “bad” link
- Testing anycast deployment

- Running your own measurements cost credits
 - ping = 10 credits, traceroute = 20, etc.
- Why? Fairness and to avoid overload
- Daily spending limit & max measurements user can create

- Hosting a RIPE Atlas probe earns credits
- Earn extra credits by:
 - Being a RIPE NCC member
 - Hosting an anchor
 - Sponsoring probes

RIPE Atlas <
About RIPE Atlas >
Get Involved >
Results >
My Atlas ▾
Probes
Measurements
Credits
API Keys
Messages (72 new)
Anchors
Sponsorships
Ambassador Probes
LIR Benefits
Claim 1 Million Credits
IPv6 Connectivity Test
Quick Look

This is where you're able to view the history of your credit use. There are visualisations available, and you can also transfer credits to someone else.

History **Charts & Archives** **Transfer**

My Atlas > Credits

Give credits to someone

Account Balance
Daily account balance over time

- Log in to atlas.ripe.net
- “My Atlas” > “Measurements”
- Three methods:
 1. Quick & Easy
 - Type
 - Target
 - Done!
 2. Advanced GIU usage
 3. CLI scripting using API

2: Using GUI to schedule a measurement

| 16

- Mostly used for a periodic, long time measurement
 - If just once, ASAP, choose “One-off”
- Choose type, target, frequency, # of probes, region...
 - Interactive interface helps you at each step
- Each measurement will have unique ID
- “API Compatible Specification” is generated too



- Using command-line & scripting:

Application Programming Interface (API)

- <https://atlas.ripe.net/docs/measurement-creation-api/>
- <https://atlas.ripe.net/keys/>

- You will need API keys

- To create measurements without logging in
- To securely share your measurement data

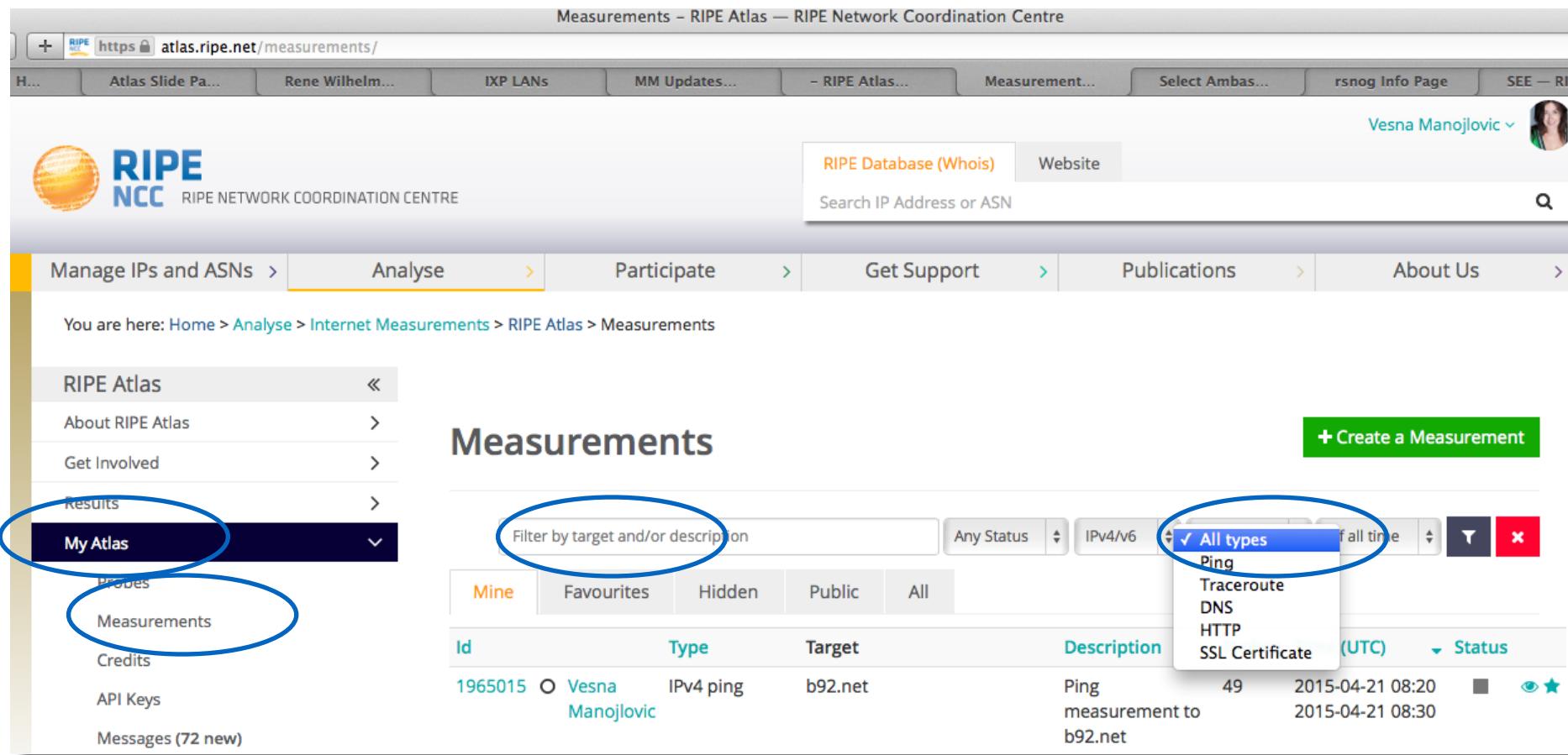
- API documentation:

- <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/>

Looking up measurements results

| 18

- Go to “My Atlas” > “Measurements”



The screenshot shows the RIPE Atlas Measurements page. On the left, there's a sidebar with a navigation menu:

- RIPE Atlas
- About RIPE Atlas
- Get Involved
- Results
 - My Atlas** (circled)
 - Probes
 - Measurements (circled)
 - Credits
 - API Keys
 - Messages (72 new)

The main content area has a search bar at the top right. Below it, there's a navigation bar with tabs: RIPE Database (Whois), Website, and a search input field. The "Analyse" tab is currently selected.

Measurements

A green button labeled "+ Create a Measurement" is visible. Below it is a table with columns: Id, Type, Target, Description, (UTC), and Status. A dropdown menu is open over the "Type" column, showing options: Any Status, IPv4/v6, All types (selected), Ping, Traceroute, DNS, HTTP, and SSL Certificate. The table contains one row of data:

ID	Type	Target	Description	(UTC)	Status
1965015	IPv4 ping	b92.net	Ping measurement to b92.net	49 2015-04-21 08:20	2015-04-21 08:30

Available visualisations: ping

| 19

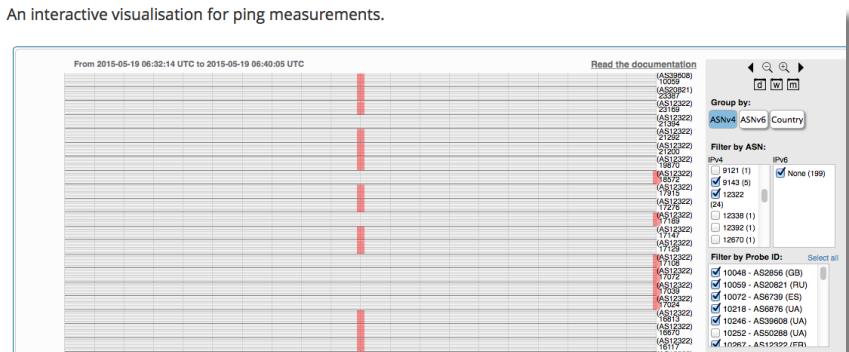
- List of probes: sortable by RTT

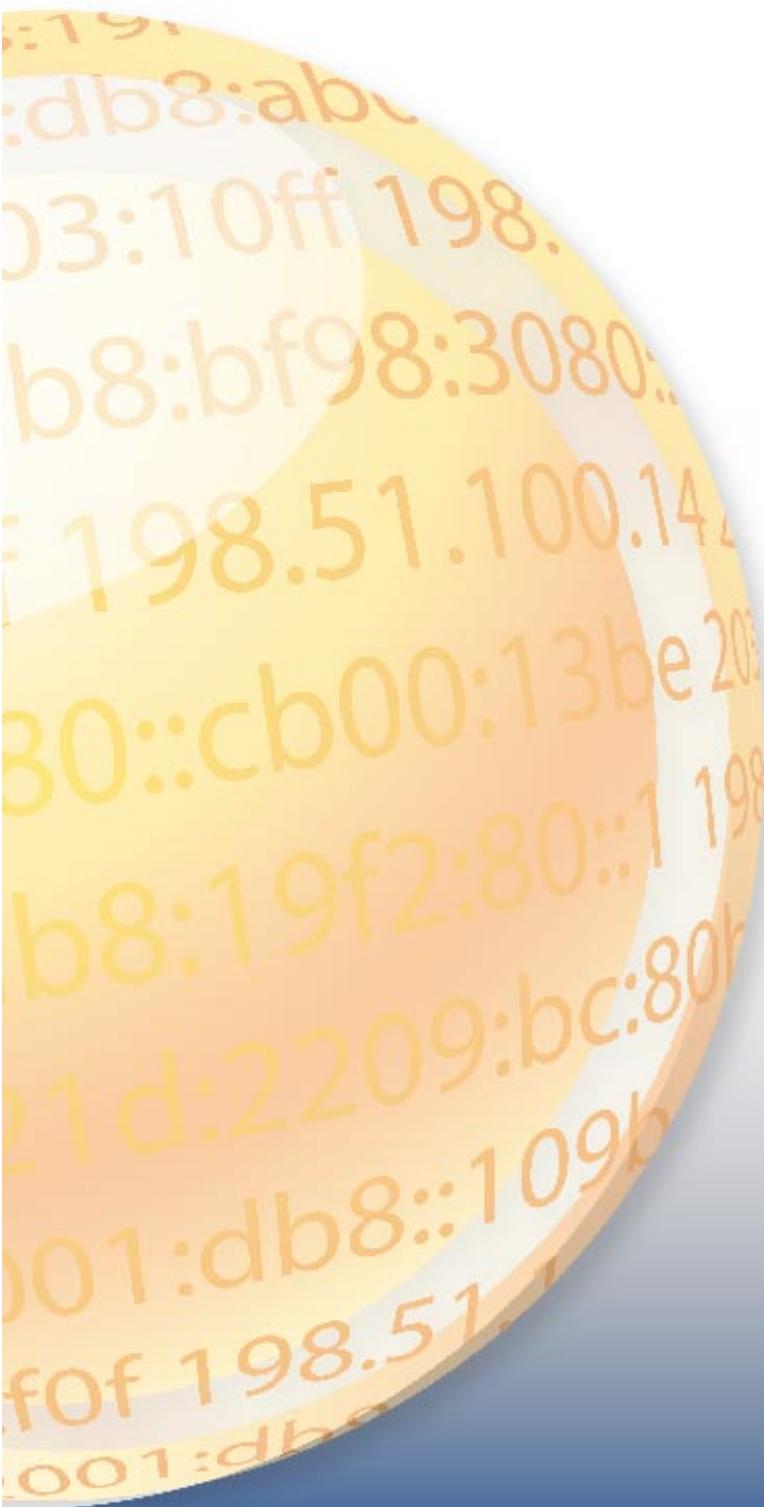
Probe	ASN (v4)	ASN (v6)	Time	RTT
6019	3333	3333	2015-05-19 09:23	1.157
6069	59469	59469	2015-05-19 09:23	15.253
6111	198068	198068	2015-05-19 09:23	37.760
6112	197216	197216	2015-05-19 09:23	35.494
10008	3851		2015-05-19 09:23	24.664
10218	6876		2015-05-19 09:23	37.952
10246	39608		2015-05-19 09:23	36.313
10252	50288		2015-05-19 09:23	62.441
10267	12322		2015-05-19 09:23	31.498
10296	51214		2015-05-19 09:23	x Unreachable

- Map: colour-coded by RTT



- Seismograph: stacked multiple pings with packet loss





Exercise: Create a Measurement



**RIPE
NCC**

- 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

Sub-task 1: Use web interface

| 22

The screenshot shows the RIPE Atlas Measurements page. On the left, there's a sidebar with links like RIPE Atlas, About RIPE Atlas, Get Involved, Results, and My Atlas (which is currently selected). The main area is titled "Measurements" and features a green button "+ Create a Measurement". Below this is a search/filter bar with fields for target and description, status, type, and time. A table lists measurements with columns for ID, Type, Target, Description, Probes, Time (UTC), and Status. Two rows are visible:

ID	Type	Target	Description	Probes	Time (UTC)	Status
1965015	IPv4 ping	b92.net	Ping measurement to b92.net	49	2015-04-21 08:20 2015-04-21 08:30	● eye star
1940389	IPv4 sslicert	twitter.com	SSL measurement to twitter.com	104	2015-04-07 09:39 2015-04-07 09:45	● eye star

- Useful hint: once you generate a measurement, copy “API Compatible Specification” to text file
- Note Measurement-ID

Create a New Measurement

The screenshot shows the "Create a New Measurement" wizard with three steps:

- Step 1 Definitions:** A section for selecting measurement types. Buttons include + Ping, + Traceroute, + DNS, + SSL, and + NTP.
- Step 2 Probe Selection:** A section for probe selection. It shows "Worldwide" and "50" probes, with buttons for + New Set - wizard, + New Set - manual, + IDs List, and + Reuse a set from a measurement.
- Step 3 Timing:** A section for setting timing. It includes a checkbox "This is a One-off:", "Start time:" (set to "As soon as possible"), and "Stop time:" (set to "Never").

At the bottom, there's a link "Measurement API Compatible Specification" and a blue button "Create My Measurement(s)".

Sub-task 2: Create API key

| 23

The screenshot shows the RIPE Atlas API Keys management interface. On the left, a sidebar menu under 'My Atlas' includes 'Probes', 'Measurements', 'Credits', 'API Keys' (which is selected and highlighted in blue), 'Messages (81 new)', and 'Anchors'. The main content area is titled 'API Keys' and displays a table of existing keys. The table columns are: Key, Created, Permission, Object, Label, Valid From, Valid To, and Enabled. Three keys are listed:

Key	Created	Permission	Object	Label	Valid From	Valid To	Enabled
984a774c-33ce-4b97-9767-fb48efda6c12	2013-01-31 13:05 UTC	Download results of a user defined measurement	1002953 I b.hosteddnsservice.com				✓
e5ba646b-abf1-4f01-8bf1-5267a9dd56ce	2013-01-31 12:52 UTC	Download results collected by a specific probe	13: k13				✓
9788b7e0-9d4b-4787-8a42-fce8f2f2e929	2013-01-11 14:53 UTC	Download results of a user defined measurement	1002676 I www.google.com				✓

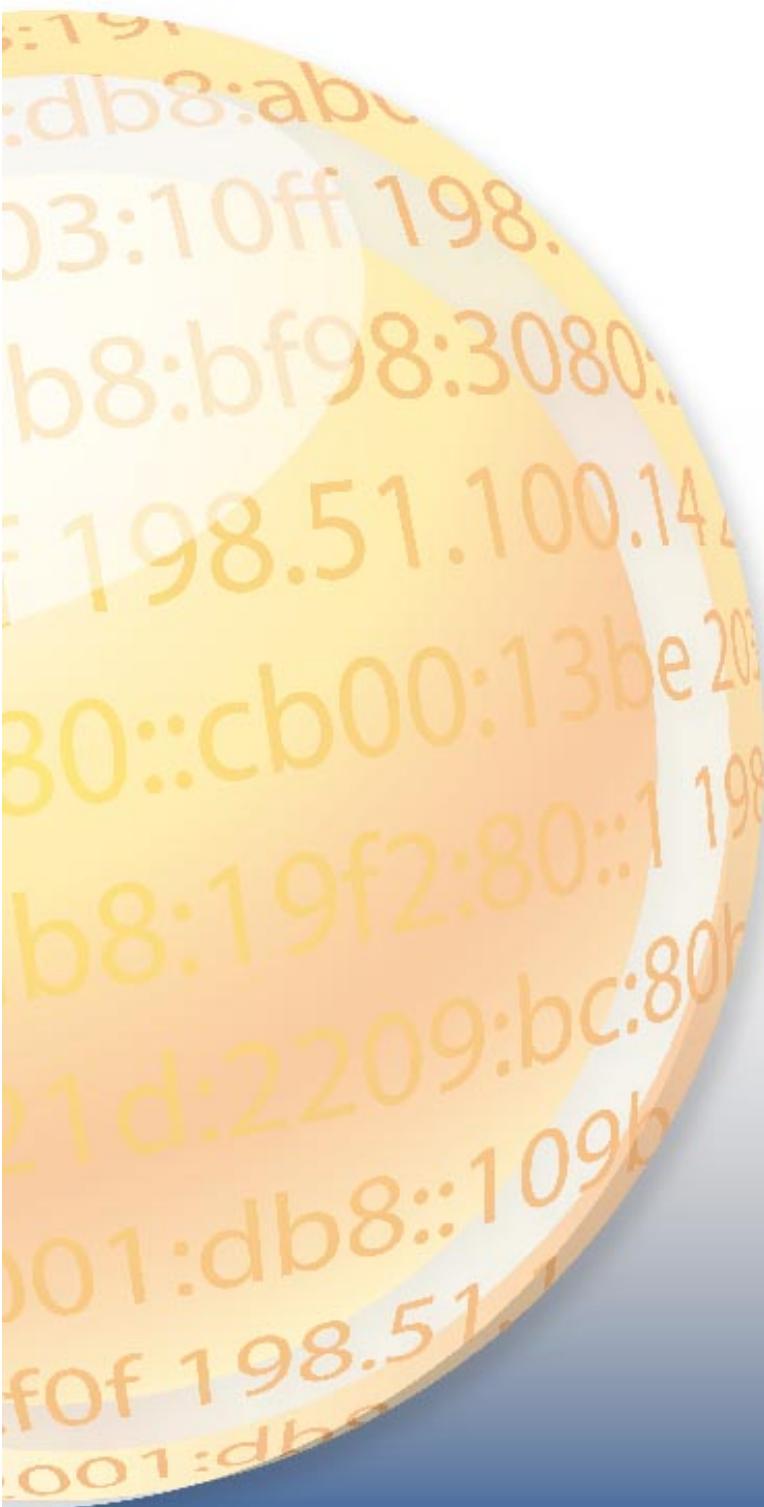
A green button labeled '+ Create an API key' is located in the top right corner of the main content area.

- Click on “Create an API Key”
- Choose type: “create a new user-defined measurement”
- “Object” is not applicable (N/A) for this type
- Give it a label
- Give it a duration of validity (leave empty for defaults)
- “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:

```
$ 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
```

```
Terminal Shell Edit View Window Help 0 b/s 0 b/s 100% wo 12:  
air-becha:~ 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-fbf1a007d060  
{"measurements": [2421551]}air-becha:~ becha$
```



Integration of RIPE Atlas with Network Monitoring Systems



RIPE
NCC

- Network operators use tools for monitoring network health (e.g. Nagios and Icinga)
- These tools can receive input from RIPE Atlas via the API
- Benefits:
 - pings from 500 out of 8,000+ probes around the world
 - See your network from the outside
 - Plug into your existing practices

1. Create a RIPE Atlas ping measurement
2. Go to “Status Checks” URL
3. Add your alerts in Nagios or Icinga



- Status checks work via RIPE Atlas' RESTful API
 - <https://atlas.ripe.net/api/v1/status-checks/>
MEASUREMENT ID/
- You define the alert parameters, for example:
 - Threshold for percentage of probes that successfully received a reply
 - How many of the most recent measurements to base it on
 - The maximum packet loss acceptable
- Documentation:
 - <https://atlas.ripe.net/docs/status-checks/>

- Community of operators contributed configuration code!
 - Making use of the built-in “check_http” plugin
- GitHub examples:
 - https://github.com/RIPE-Atlas-Community/ripe-atlas-community-contrib/blob/master/scripts_for_nagios_icinga_alerts
- Post on Icinga blog:
 - <https://www.icinga.org/2014/03/05/monitoring-ripe-atlas-status-with-icinga-2/>

Exercise: Setting up “Status Checks”

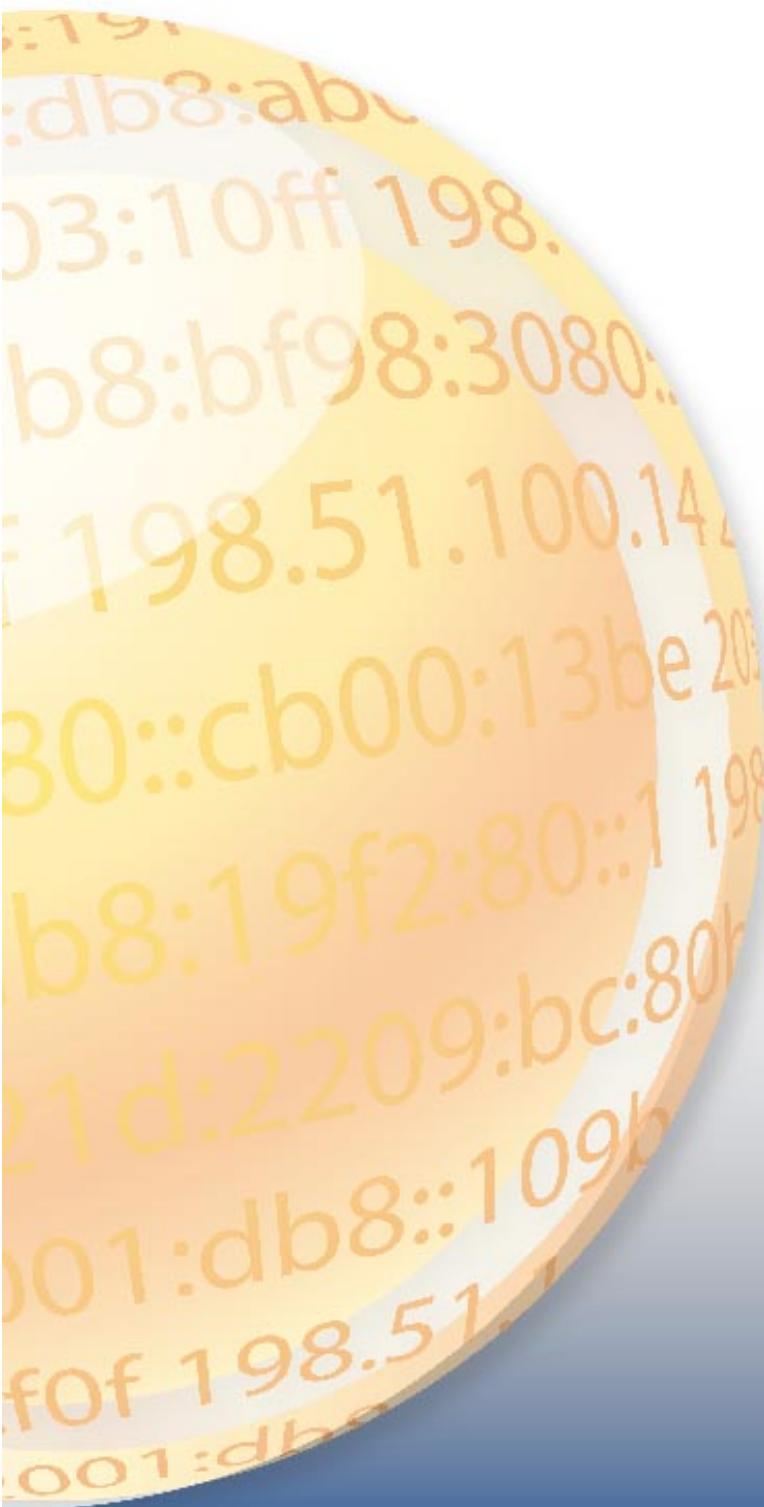


**RIPE
NCC**

- Set up and configure a “status check”
 - For an existing ping measurement <https://atlas.ripe.net/measurements/2340408/>
 - Hint: <https://atlas.ripe.net/api/v1/status-checks/2340408/>
- Configure the status check in such a way that you will trigger an alert for this measurement
- Optional: set-up status check for your own ping measurement!

- One possible solution:
 - Set the median RTT to a lower level:
 - https://atlas.ripe.net/api/v1/status-checks/2340408/?median_rtt_threshold=10
- Example of the alerts

```
{"total_alerts":32,"global_alert":true,  
"probes":{  
"18433": {"all": [null,null,null],"last":null,"last_packet_loss":100.0,"alert":true,"source": "Area: WW","alert_reasons": ["loss"]},  
"15041": {"source": "Area: WW","last_packet_loss":0.0,"last":19.928,"alert":false},  
"18696": {"all": [null,null,null],"last":null,"last_packet_loss":100.0,"alert":true,"source": "Area: WW","alert_reasons": ["loss"]},  
"16265": {"source": "Area: WW","last_packet_loss":0.0,"last":22.72,"alert":false},  
"20236": {"all": [null,null,null],"last":null,"last_packet_loss":100.0,"alert":true,"source": "Area: WW","alert_reasons": ["loss"]},  
"12944": {"all": [null,null,null],"last":null,"last_packet_loss":100.0,"alert":true,"source": "Area: WW","alert_reasons": ["loss"]},  
"2195": {"all": [null,null,null],"last":null,"last_packet_loss":100.0,"alert":true,"source": "Area: WW","alert_reasons": ["loss"]},
```



Real-time performance monitoring



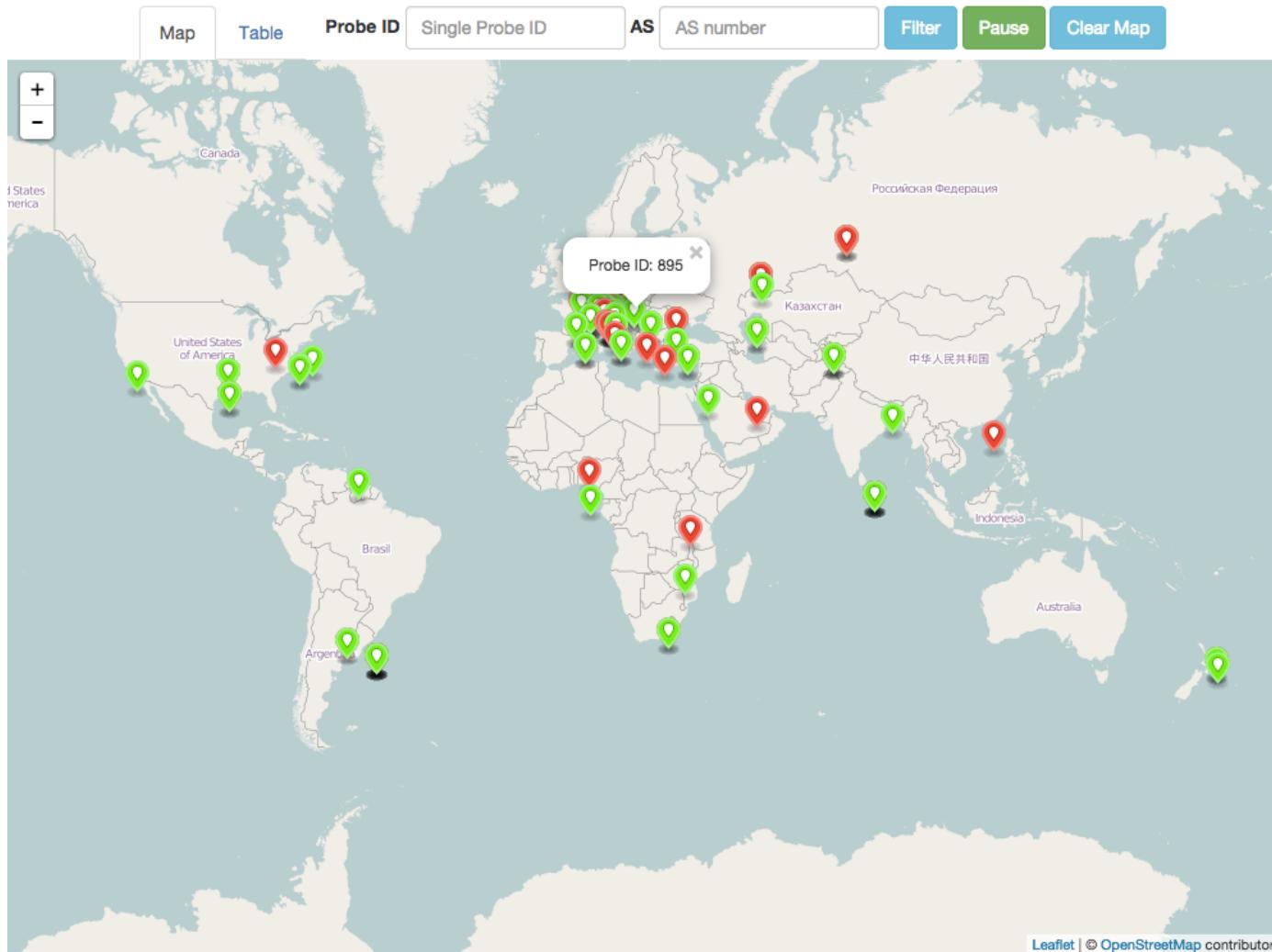
**RIPE
NCC**

- RIPE Atlas streaming is an architecture that 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 two types of data:
 - Measurement results
 - Probe connection status events

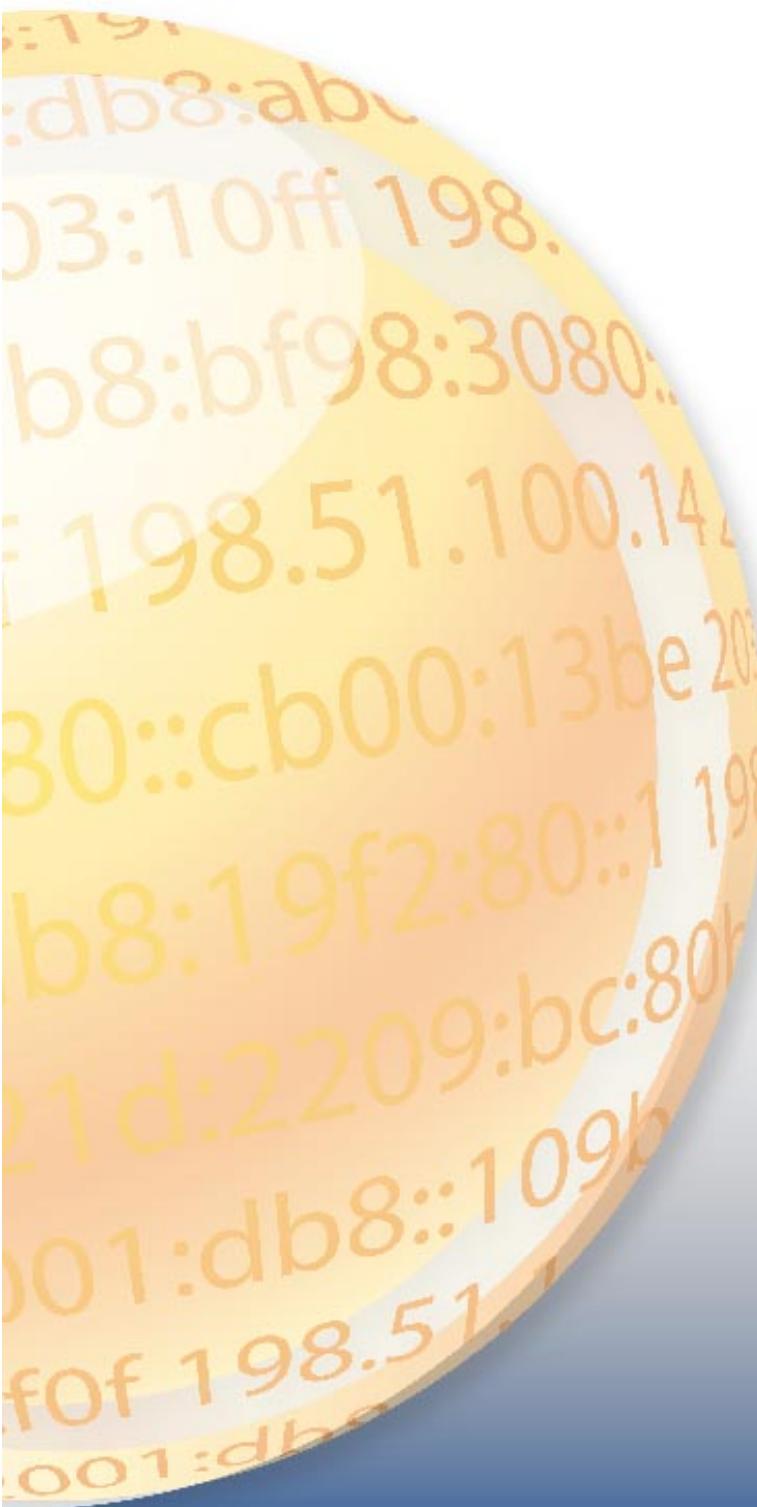
- Visualising network outages
- Server and performance monitoring
- In March 2015: used by almost all hackathon teams:
<https://labs.ripe.net/Members/becha/ripe-atlas-hackathon-results>
- Documentation:
 - <https://atlas.ripe.net/docs/result-streaming/>
 - https://labs.ripe.net/Members/suzanne_taylor_muzzin/data-streaming-in-ripe-atlas

Probe (dis)connection events

| 37



https://labs.ripe.net/Members/andreas_strikos/amsterdam-power-outage-as-seen-by-ripe-atlas



Exercise: Using streaming API



**RIPE
NCC**

- Scenario: customers are complaining that it occasionally takes a long time to reach your service or server
- Action: ping your server from 500 probes
 - Decide what is acceptable latency threshold to apply
 - Notice and react when you start receiving samples
- Task: Use the ping measurement ID 2340408
 - Choose which threshold (e.g. greater than 30ms)
 - Imposes the threshold on “min” (the minimum result of the three ping attempts)

1. Go to

<http://atlas.ripe.net/webinar/streaming01.html>

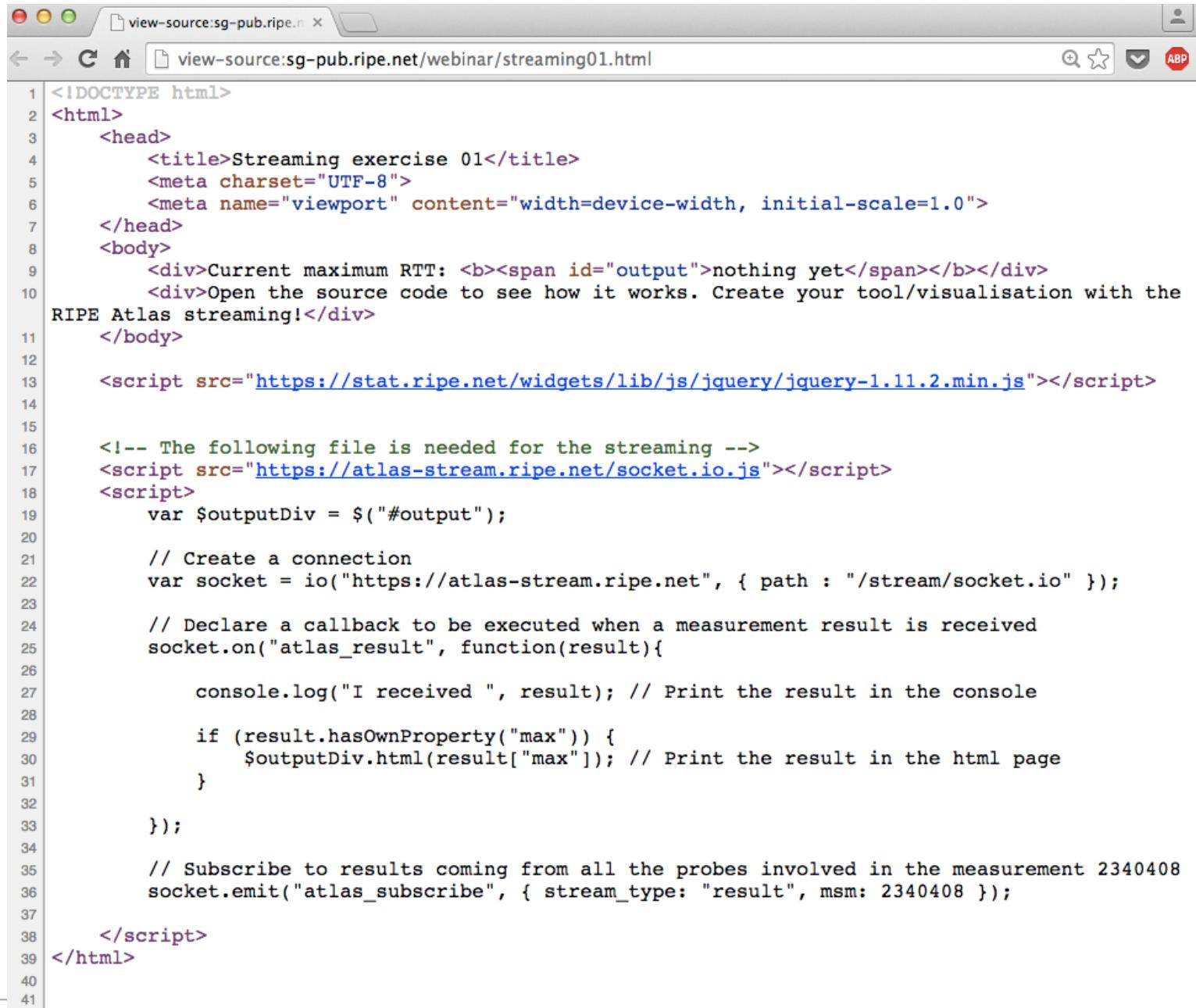
2. Open the development console

3. Wait for results to arrive

4. Optional: Save the HTML file locally and edit
the code to your liking

Page Source

| 41



The screenshot shows a web browser window with the title "view-source:sg-pub.ripe.net". The address bar also displays "view-source:sg-pub.ripe.net/webinar/streaming01.html". The page content is the source code of a web page titled "Streaming exercise 01". The code includes HTML structure, meta tags, and a script that uses jQuery and socket.io to stream measurement results from the RIPE Atlas network.

```
1 <!DOCTYPE html>
2 <html>
3     <head>
4         <title>Streaming exercise 01</title>
5         <meta charset="UTF-8">
6         <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     </head>
8     <body>
9         <div>Current maximum RTT: <b><span id="output">nothing yet</span></b></div>
10        <div>Open the source code to see how it works. Create your tool/visualisation with the
RIPE Atlas streaming!</div>
11    </body>
12
13    <script src="https://stat.ripe.net/widgets/lib/js/jquery/jquery-1.11.2.min.js"></script>
14
15
16    <!-- The following file is needed for the streaming -->
17    <script src="https://atlas-stream.ripe.net/socket.io.js"></script>
18    <script>
19        var $outputDiv = $("#output");
20
21        // Create a connection
22        var socket = io("https://atlas-stream.ripe.net", { path : "/stream/socket.io" });
23
24        // Declare a callback to be executed when a measurement result is received
25        socket.on("atlas_result", function(result){
26
27            console.log("I received ", result); // Print the result in the console
28
29            if (result.hasOwnProperty("max")) {
30                $outputDiv.html(result["max"]); // Print the result in the html page
31            }
32
33        });
34
35        // Subscribe to results coming from all the probes involved in the measurement 2340408
36        socket.emit("atlas_subscribe", { stream_type: "result", msm: 2340408 });
37
38    </script>
39 </html>
```



Example of results

| 42

```
Elements Network Sources Timeline Profiles Resources Audits | Console | AngularJS
✖ ✖ <top frame> ▾  Preserve log
Filter  Regex All | Errors Warnings Info Logs Debug  Hide network messages
XHR finished loading: GET "http://atlas-stream.ripe.net/stream/socket.io/?EIO=2&transport=polling&t=1431095373684-0".
XHR finished loading: GET "http://atlas-stream.ripe.net/stream/socket.io/?EIO=2&transport=polling&t=1431095373739-1&sid=eB0kM7zfWFT2c-ScAAaH".
I received ▶ Object {af: 4, prb_id: 16669, result: Array[3], ttl: 42, avg: 326.841...}
I received ▶ Object {af: 4, prb_id: 16669, result: Array[3], ttl: 42, avg: 325.7933333333...}
I received ▶ Object {af: 4, prb_id: 16669, result: Array[3], ttl: 42, avg: 326.048...}
I received ▶ Object {af: 4, prb_id: 16669, result: Array[3], ttl: 42, avg: 327.3253333333...}
I received ▶ Object {af: 4, prb_id: 15965, result: Array[3], ttl: 45, avg: 47.6313333333...}
I received ▶ Object {af: 4, prb_id: 15965, result: Array[3], ttl: 45, avg: 47.6996666667...}
I received ▶ Object {af: 4, prb_id: 15965, result: Array[3], ttl: 45, avg: 47.4816666667...}
I received ▶ Object {af: 4, prb_id: 19566, result: Array[3], ttl: 40, avg: 47.054...}
I received ▶ Object {af: 4, prb_id: 19566, result: Array[3], ttl: 40, avg: 47.8626666667...}
I received ▶ Object {af: 4, prb_id: 19566, result: Array[3], ttl: 40, avg: 47.5946666667...}
I received ▶ Object {af: 4, prb_id: 19566, result: Array[3], ttl: 40, avg: 47.5003333333...}
I received ▶ Object {af: 4, prb_id: 18311, result: Array[3], ttl: 49, avg: 32.577...}
I received ▶ Object {af: 4, prb_id: 18311, result: Array[3], ttl: 49, avg: 34.0843333333...}
I received ▶ Object {af: 4, prb_id: 18311, result: Array[3], ttl: 49, avg: 32.7513333333...}
I received ▶ Object {af: 4, prb_id: 16010, result: Array[3], ttl: 46, avg: 182.4463333333...}
I received ▶ Object {af: 4, prb_id: 16010, result: Array[3], ttl: 46, avg: 193.9953333333...}
I received ▶ Object {af: 4, prb_id: 16010, result: Array[3], ttl: 46, avg: 182.2913333333...}
I received ▶ Object {af: 4, prb_id: 16010, result: Array[3], ttl: 46, avg: 191.6103333333...}
I received ▶ Object {af: 4, prb_id: 14918, result: Array[3], ttl: 49, avg: 34.817...}
I received ▶ Object {af: 4, prb_id: 14918, result: Array[3], ttl: 49, avg: 35.0093333333...}
I received ▶ Object {af: 4, prb_id: 14918, result: Array[3], ttl: 49, avg: 35.0843333333...}
I received ▶ Object {af: 4, prb_id: 20668, result: Array[3], ttl: 45, avg: 38.8846666667...}
I received ▶ Object {af: 4, prb_id: 20668, result: Array[3], ttl: 45, avg: 38.8626666667...}
I received ▶ Object {af: 4, prb_id: 20668, result: Array[3], ttl: 45, avg: 38.8806666667...}
I received ▶ Object {af: 4, prb_id: 6093, result: Array[3], ttl: 49, avg: 128.7273333333...}
I received ▶ Object {af: 4, prb_id: 6093, result: Array[3], ttl: 49, avg: 128.7373333333...}
I received ▶ Object {af: 4, prb_id: 6093, result: Array[3], ttl: 49, avg: 128.8883333333...}
```

Task 2: View (dis)connect events (optional)

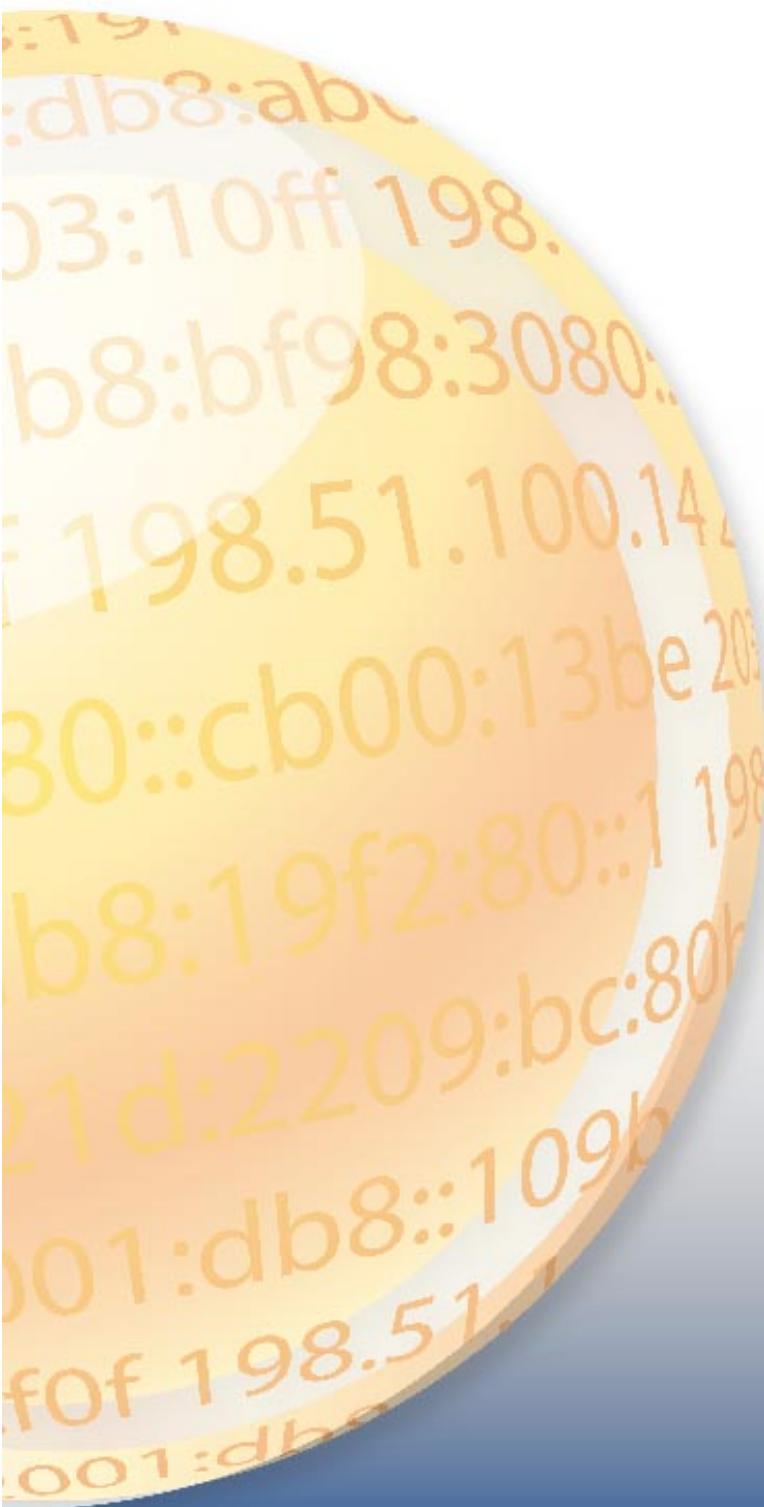
| x

- See in the console of your browser the connection and disconnection events of all the RIPE Atlas probes
- Steps:
 - Create your empty HTML page
 - Connect to the streaming
 - Subscribe to stream_type: “probestatus”

Solution

| x

```
<script src="http://atlas-stream.ripe.net/socket.io.js"></script>  
<script>  
    var socket = io("http://atlas-stream.ripe.net:80", { path : "/stream/socket.io" });  
  
    socket.on("atlas_probestatus", function(status){  
        console.log("I received ", status);  
    });  
  
    socket.emit("atlas_subscribe", { stream_type: "probestatus" });  
</script>
```



Take part in the RIPE Atlas community



RIPE
NCC

- Individual 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



- <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

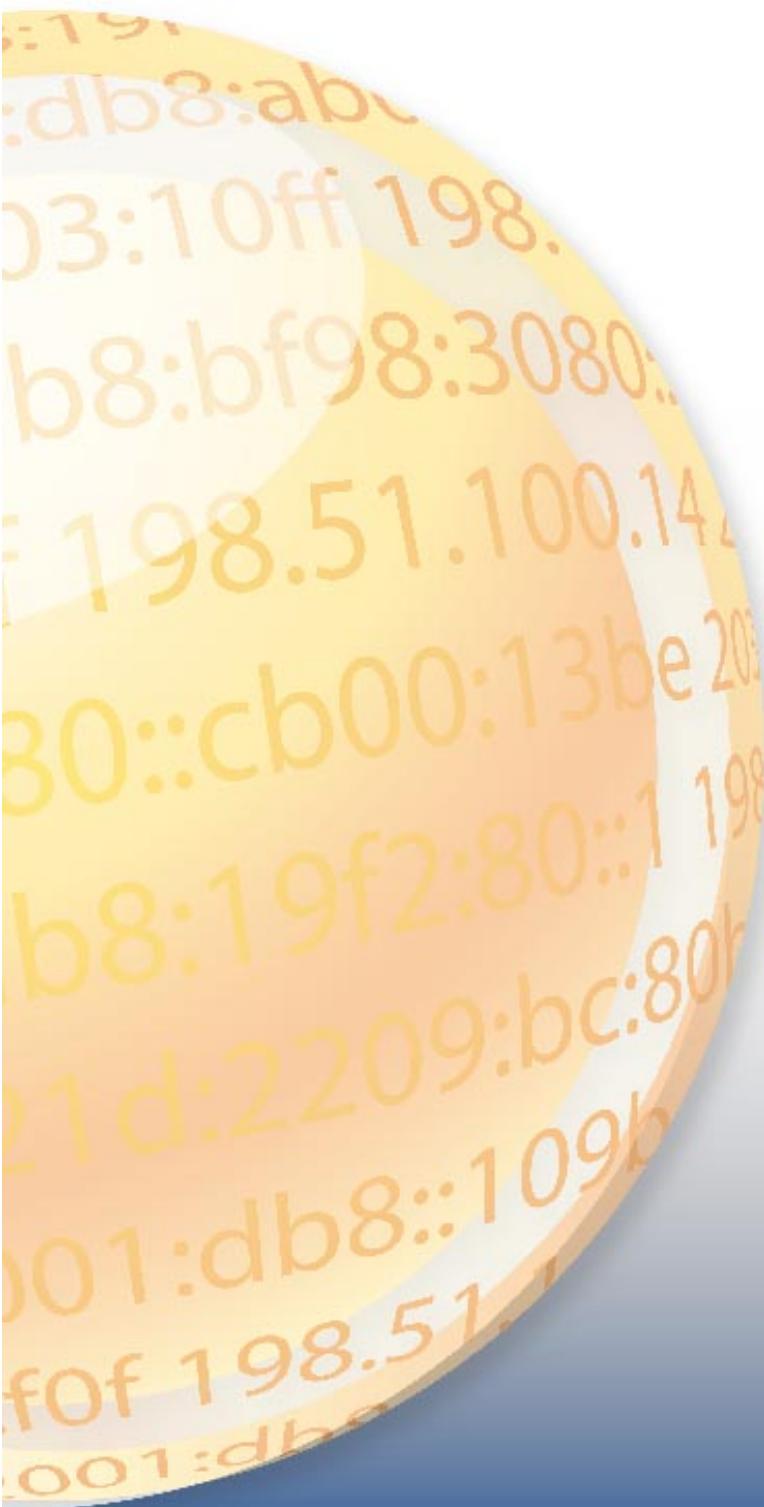
- <https://atlas.ripe.net/docs/rest/>
- <https://github.com/RIPE-NCC/ripe.atlas.sagan>
- <https://atlas.ripe.net/docs/measurement-creation-api/>
- <https://atlas.ripe.net/doc/credits>
- <https://atlas.ripe.net/doc/udm>
- <https://atlas.ripe.net/keys/>
- <https://atlas.ripe.net/docs/keys2/>

- Basics: <http://www.ripe.net/lir-services/training/courses/tailor-made-workshops/#tools>
- Webinar material: <https://www.ripe.net/support/training/learn-online/webinars/advanced-ripe-atlas-usage-webinar>
- More tools:
 - <https://github.com/RIPE-Atlas-Community>
 - <https://github.com/RIPE-Atlas-Community/ripe-atlas-community-contrib/blob/master/README.md>

Additional slides



**RIPE
NCC**



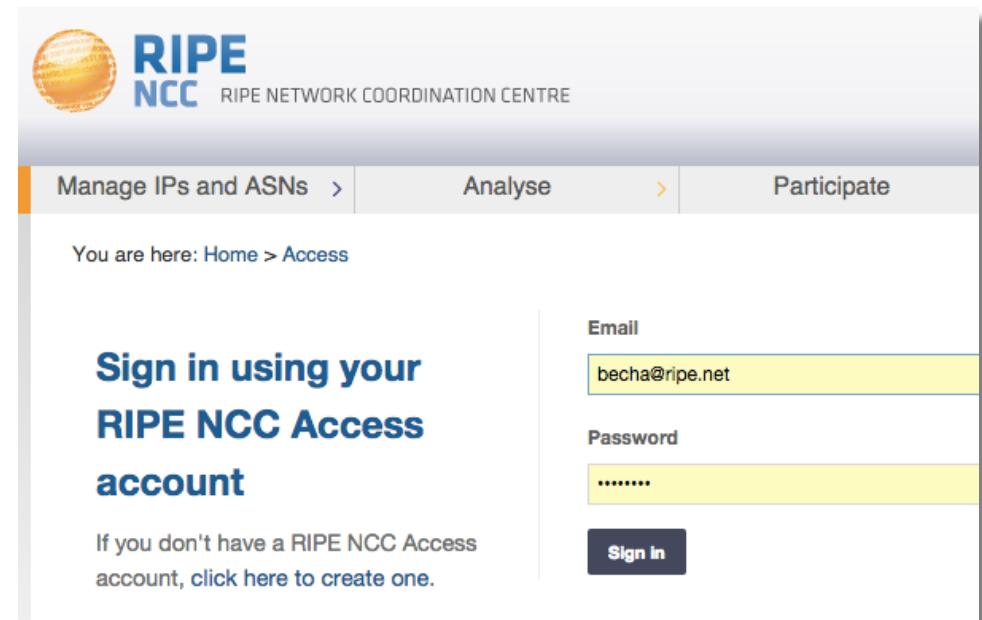
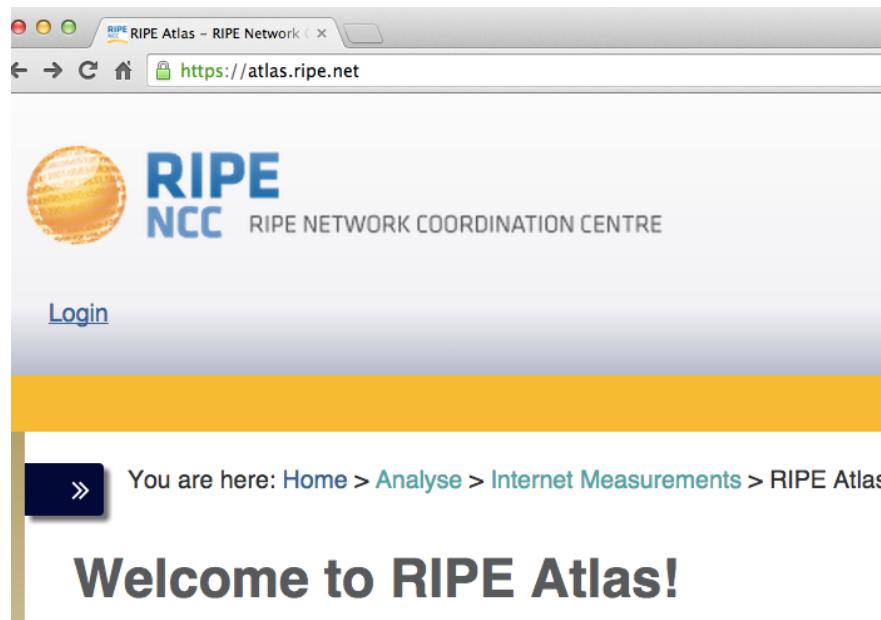
Finding Results of Public Measurements



RIPE
NCC

- There are many measurements already running!
- Search for existing public measurements first
- Schedule your own measurement if you don't find what you're looking for

- 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 have one already



Looking up measurements results

| 51

- Go to “My Atlas” > “Measurements”

The screenshot shows the RIPE Atlas Measurements page. On the left, there's a sidebar with a navigation menu:

- RIPE Atlas
- About RIPE Atlas
- Get Involved
- Results
 - My Atlas** (circled)
 - Probes
 - Measurements (circled)
 - Credits
 - API Keys
 - Messages (72 new)

The main content area has a search bar at the top right. Below it, there's a navigation bar with tabs: Analyse (selected), Participate, Get Support, Publications, and About Us. A green button labeled "+ Create a Measurement" is also present.

The main content area is titled "Measurements". It features a table with the following columns: Id, Type, Target, Description, (UTC), and Status. One row is shown in the table:

ID	Type	Target	Description	(UTC)	Status
1965015	IPv4 ping	b92.net	Ping measurement to b92.net	49 2015-04-21 08:20	2015-04-21 08:30

A dropdown menu is open over the "Type" column, showing options: All types (selected), Ping, Traceroute, DNS, HTTP, and SSL Certificate. The "All types" option is highlighted with a blue circle.

Available visualisations: ping

| 52

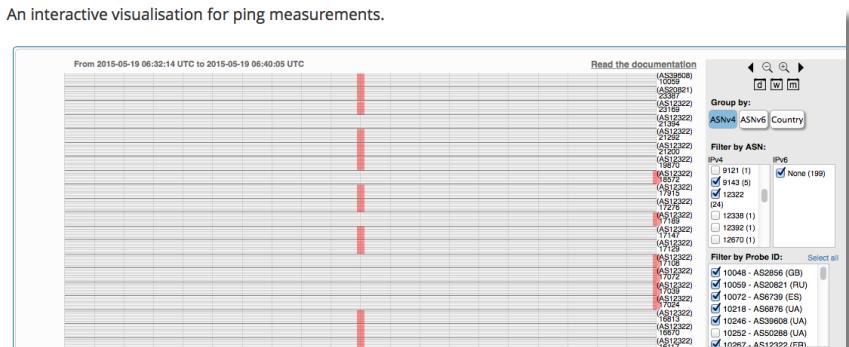
- List of probes: sortable by RTT

Probe	ASN (v4)	ASN (v6)	Time	RTT
6019	3333	3333	2015-05-19 09:23	1.157
6069	59469	59469	2015-05-19 09:23	15.253
6111	198068	198068	2015-05-19 09:23	37.760
6112	197216	197216	2015-05-19 09:23	35.494
10008	3851		2015-05-19 09:23	24.664
10218	6876		2015-05-19 09:23	37.952
10246	39608		2015-05-19 09:23	36.313
10252	50288		2015-05-19 09:23	62.441
10267	12322		2015-05-19 09:23	31.498
10296	51214		2015-05-19 09:23	x Unreachable

- Map: colour-coded by RTT



- Seismograph: stacked multiple pings with packet loss



Seismograph tips

| 53

Red = packet loss

An interesting pattern of red measurements.

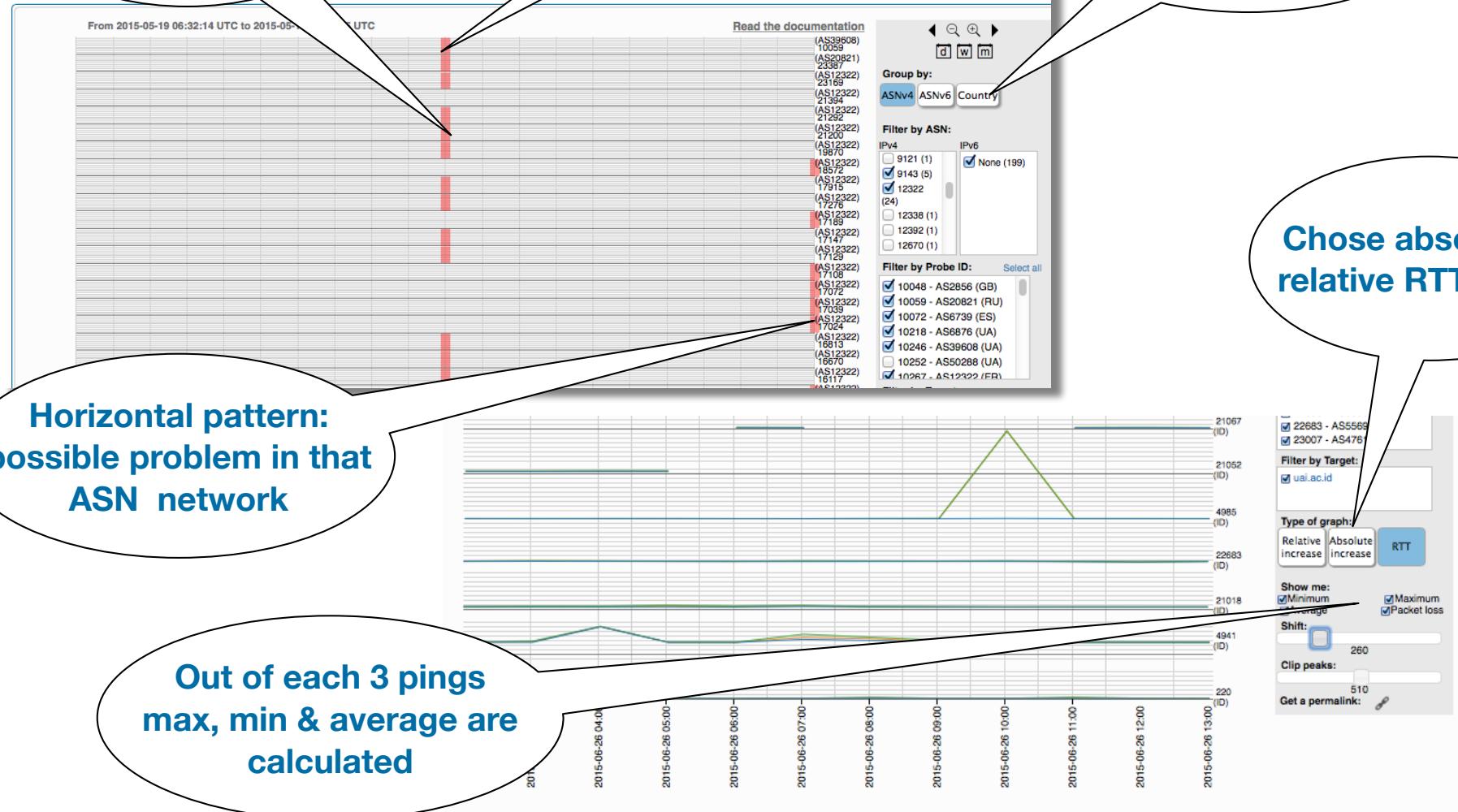
Vertical pattern:
possible problem in
your network

Filtering
& grouping by:
country, IPv4 ASN,
IPv6 ASN

Chose absolute or
relative RTT values

Horizontal pattern:
possible problem in that
ASN network

Out of each 3 pings
max, min & average are
calculated



Available visualisations: traceroute

| 54

- List of probes, colour-coded number of hops

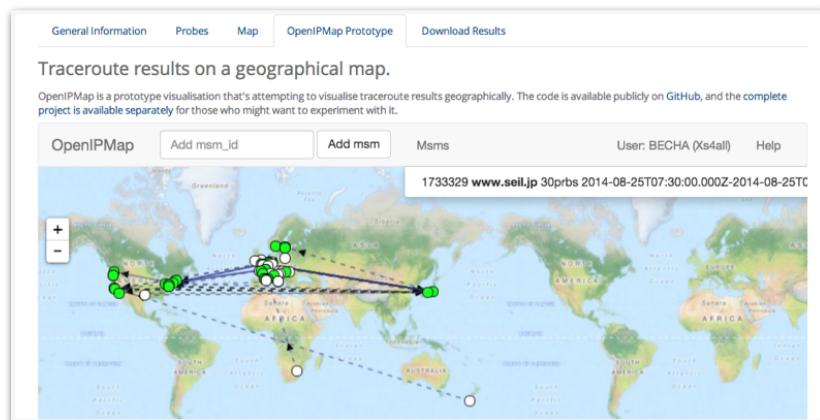
- Map

⚡ www.seil.jp

General Information Probes Map OpenIPMap Prototype Download Results

Probe	ASN (v4)	ASN (v6)	Time	RTT	Hops
2043	3313		2014-08-25 07:44	308.018	21
3246	41135		2014-08-25 07:41	259.912	12
3389	3302		2014-08-25 07:43	285.608	17
4092	37497		2014-08-25 07:40	452.889	19
4228	3269		2014-08-25 07:41	329.862	20
10024	42353		2014-08-25 07:44	x	1

- Traceroute paths map, geolocation using OpenIPMap:
<https://github.com/RIPE-Atlas-Community/openipmap>



Available visualisations: DNS

| 55

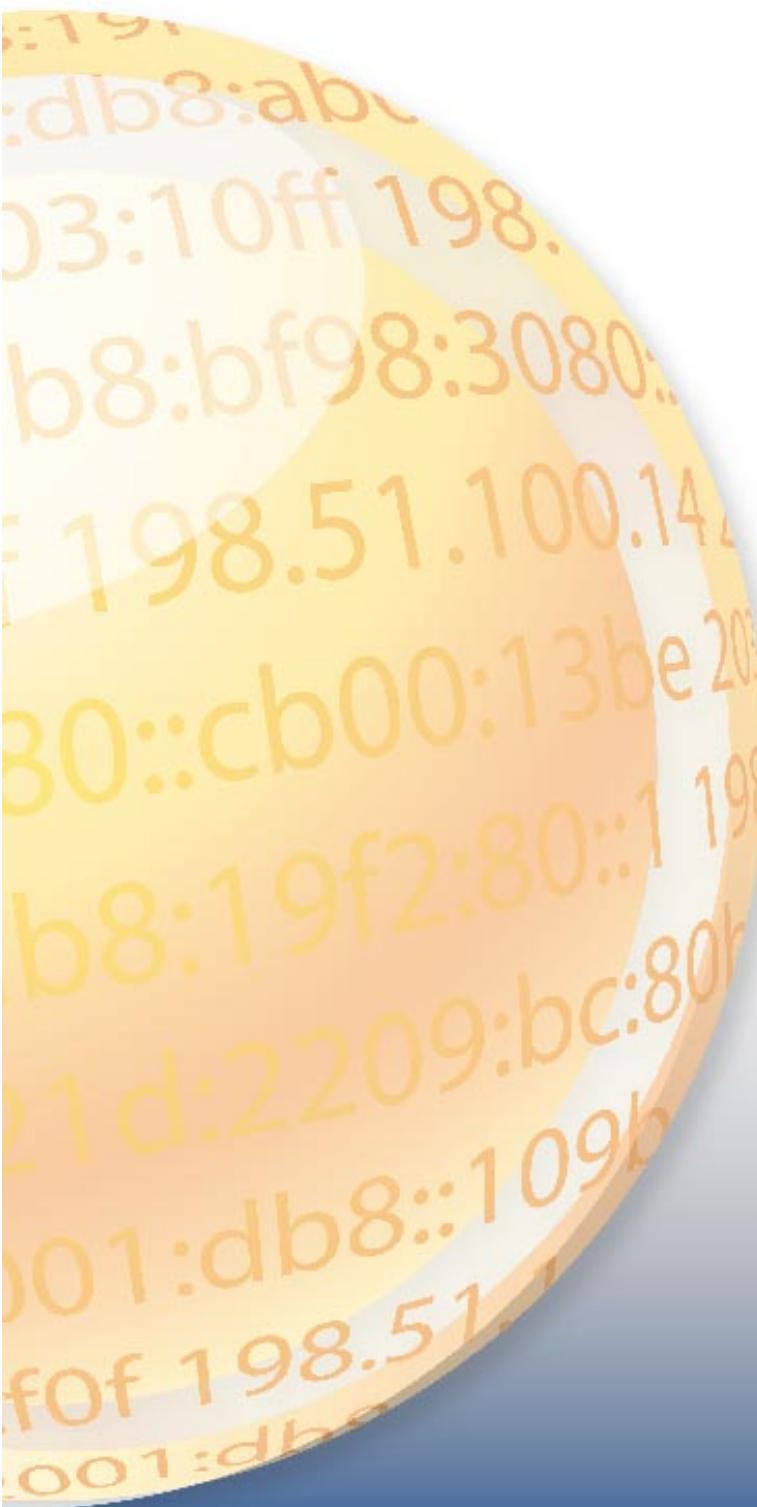
- Map, colour-coded response time or diversity
- List of probes, sortable by response time



DNS measurement to ns1.optteamax.de

General Information		Probes	Map	Download Results	Modification Log
Probe	ASN (v4)	ASN (v6)	Time	Name	Response Time
17840	6327		2015-05-19 09:38	null	<div style="width: 362.009px;"></div> 362.009
18035	43030		2015-05-19 09:50	null	<div style="width: 347.39px;"></div> 347.39
18129	327805		2015-05-19 09:49	null	<div style="width: 207.743px;"></div> 207.743
15844	32098		2015-05-19 09:48	null	<div style="width: 184.237px;"></div> 184.237
17857	852		2015-05-19 09:37	null	<div style="width: 177.694px;"></div> 177.694
19894	6327		2015-05-19 09:36	null	<div style="width: 168.689px;"></div> 168.689
19204	21513		2015-05-19 09:50	null	<div style="width: 141.199px;"></div> 141.199
15922	30036		2015-05-19 09:47	null	<div style="width: 133.309px;"></div> 133.309

- Documentation for analysing measurements results:
 - <https://atlas.ripe.net/docs/rest/>
 - <https://github.com/RIPE-NCC/ripe.atlas.sagan>
- More tools:
 - <https://github.com/RIPE-Atlas-Community>
 - <https://github.com/RIPE-Atlas-Community/ripe-atlas-community-contrib/blob/master/README.md>



Exercise: Analyse Measurement Results



**RIPE
NCC**

- Download results of a specific public measurement
- Read the text of the result, to understand structure

Task 1: Download measurement results

| 59

- Find the measurement
 - ping, IPv6 to google.com
 - msm-ID 1004005
- Click on measurement, then “Download”
 - Specify the time period
 - (for example, YESTERDAY)
- Results in JSON

- Solution URL:

- <https://atlas.ripe.net/api/v1/measurement/1004005/result/?start=1435104000&stop=1435276799&format=json>

- Save the measurement(s) locally

```
$ curl https://atlas.ripe.net/api/v1/measurement/1004005/result/?start=1435104000&stop=1435276799&format=json > measurement-test.json
```

Task 2: Look at the result

| 61

Reference
(msm ID)

```
[ {"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,
```

Destination
(IP & name)

Source (probe
public IP address)

Packet
loss: difference
between sent &
received!

```
"result":[{"rtt":62.148},{"rtt":61.437},{"rtt":61.444},{"rtt":61.448}  
 {"rtt":61.794},{"rtt":61.533},{"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"},
```

Task 3: analyse results (optional)

| 62

- Find out how many times RTT was above 60ms
 - Use Python o Javascript or something else
- For the Javascript solution, you can use this as a starting point:
 - https://stat.ripe.net/widgets/demo/script_me.html

Task 3: Examples of code

| 63

Python:

Parse json and find total avg:

```
import json
f = open("measurement.json", "r")
measurements = json.load(f)
for m in measurements:
    for r in m["result"]:
        rtt = r["rtt"]
    if rtt > 60: i += 1
i must be > than 14563.
```

Javascript:

```
<script>
var dataAPIUrl = "https://atlas.ripe.net/api/v1/
measurement/1004005/result/?start=1410220800";
jQuery.ajax({
url: dataAPIUrl, error: function() {
alert("error");
},
success: function( response ) { var i = 0;
for ( var i = 0, n = response.length; i < n; i++ ) { var
measurement = response[i];
for ( var j = 0, m = measurement.result.length; j < m; j++ ) {
var rtt = measurement.result[j].rtt;
console.log(rtt);
if (rtt > 60)
i++;
}
}
jQuery("p").html("The RTT has been above 60ms for " + i
+ " times");
},
dataType: "jsonp"
});
</script>
```

RIPE Atlas Anchors



**RIPE
NCC**

- Well-known targets and powerful probes
 - Regional baseline and “future history”
- Anchoring measurements
 - Measurements between anchors
 - 200 probes targeting each anchor with measurements
 - Each probe measures 4-5 anchors
 - Vantage points for DNSMON service
- 130+ RIPE Atlas anchors

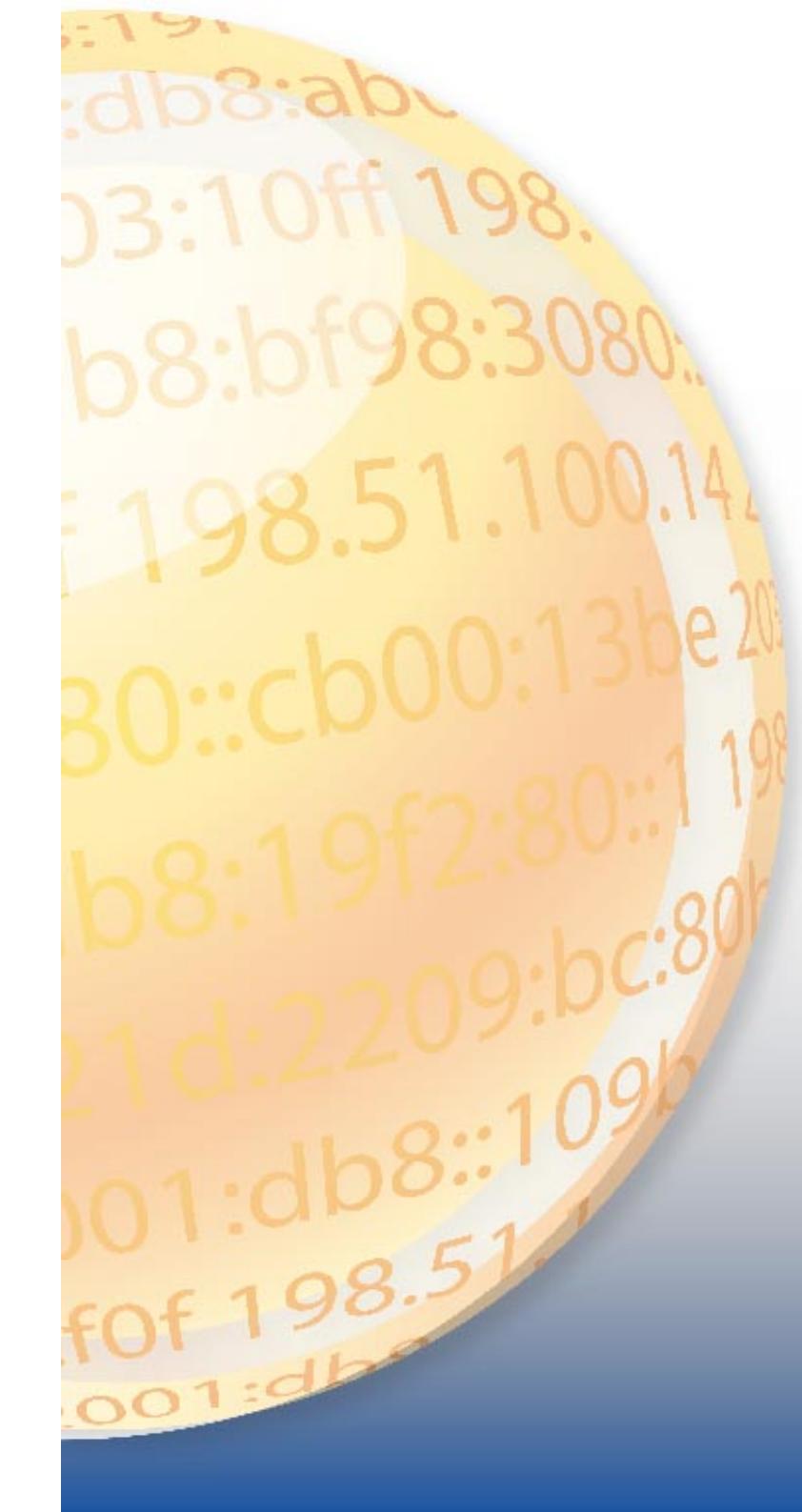


Locations of anchors

| 66



[https://atlas.ripe.net/results/maps/network-coverage/
#anchors](https://atlas.ripe.net/results/maps/network-coverage/#anchors)



Measuring Impact of IXPs on Keeping Traffic Local

“IXP Country Jedi”



RIPE
NCC

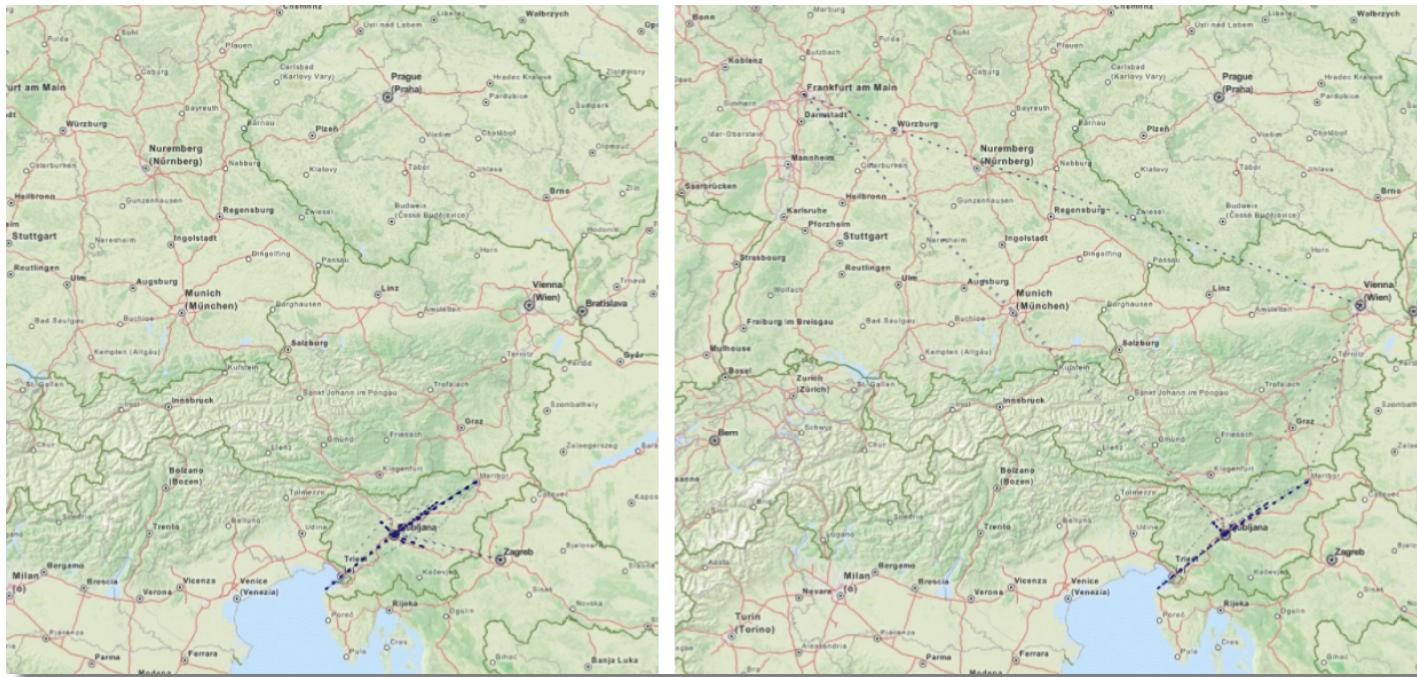
- Operators
 - Routing and traffic optimisation
- IXP operators
 - Shows how IXPs help keep traffic local and regional
- IPv6 advocates
 - Comparing IPv4 and IPv6 paths
- Country level: regulators, politicians, cyber-security...
 - How much traffic stays within the country? Where do the paths go?
 - Comparing countries with each other

- RIPE Atlas community
 - More probes in more networks = higher quality of measurements data
- Geolocation data community
 - Use case for improving data quality
- Examples:
 - <https://labs.ripe.net/Members/emileaben/measuring-ixps-with-ripe-atlas>
 - <https://labs.ripe.net/Members/emileaben/measuring-countries-and-ixps-in-the-see-region>
 - <http://sg-pub.ripe.net/emile/ixp-country-jedi/CL+AR-2015-04/geopath/>

Paths staying in the country?

| 70

- Difference between IPv4 and IPv6 paths

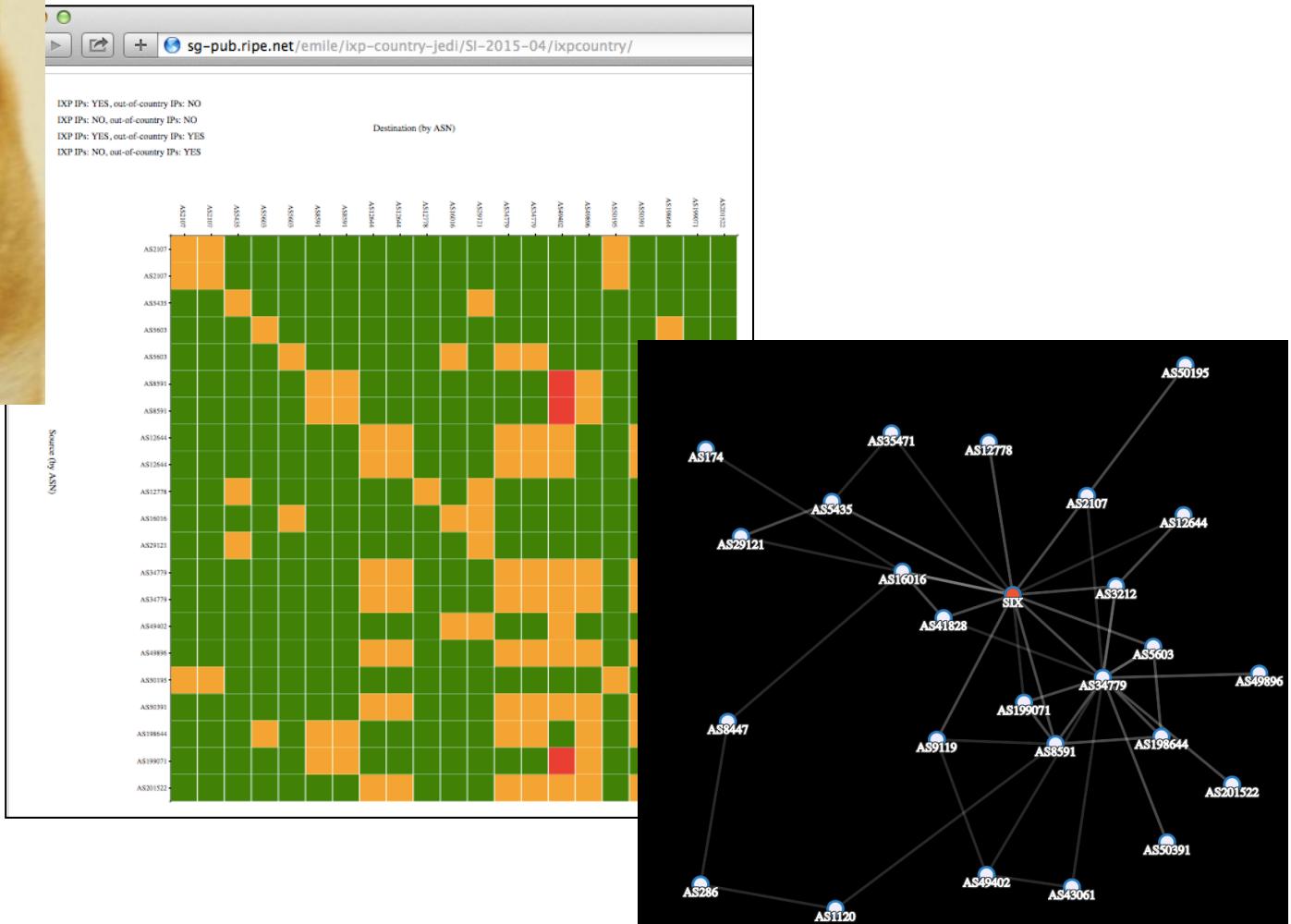


[http://sg-pub.ripe.net/emile/ixp-country-jedi/SL-2015-04/
geopath/s/SL/{RO, BG, HR, BA, ME, AL, GR}/](http://sg-pub.ripe.net/emile/ixp-country-jedi/SL-2015-04/geopath/s/SL/{RO, BG, HR, BA, ME, AL, GR}/)

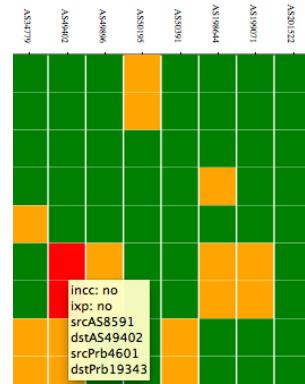
Paths going via an IXP?

| 71

<http://sg-pub.ripe.net/emile/ixp-country-jedi/SI-2015-04/>



<http://sg-pub.ripe.net/emile/ixp-country-jedi/SI-2015-04/ixpcountry/>



```
## msm_id:1962254 prb_id:4601 dst:193.169.48.40 ts:2015-04-16 09:01:06 -00:00
1 (AS8591) maribor10-ge-2-20-v987.amis.net [1.593, 1.602, 2.292] |Maribor,Maribor,SI
2 (AS8591) mx-mbl-te-1-2-0.amis.net [1.619, 1.697, 1.944] ||
3 (AS8591) mx-lj1-te-2-3-1.amis.net [3.599, 3.865, 5.148] ||
4 (AS8591) mx-zgl-xe-2-0-1.amis.net [5.568, 5.576, 5.69] ||
5 () 75.64-127.15.192.193.in-addr.arpa [5.955, 5.98, 5.985] |Zagreb,Grad Zagreb,HR|
6 (AS9119) 212.13.240.249 [5.778, 5.83, 5.935] ||
7 (AS9119) 212.103.133.4 [7.099, 7.84, 7.926] ||
8 (AS9119) 212.13.240.62 [6.597, 7.674, 7.696] ||
9 (AS9119) hsl.gw0.hsl.eu [5.833, 6.079, 6.368] ||
10 (AS49402) ntp.hsl.eu [6.657, 7.273, 8.155] ||
11 (AS49402) 193.169.48.40 [6.661, 6.691, 6.872] ||
```

- Green: “good”, as far as we can see it
 - Not a judgment, only one way of visualising data
- Red or blue: path is going out of country
 - If this is a surprise: talk to your upstream(s)
- Yellow: path is not going via a local IXP
 - If this is undesired: make a new peering agreement

- traceroute measurements using RIPE Atlas probes
- Steps:
 - Identify ASNs in the country using RIPEstat
 - Identify IXPs and IXP LANs using PeeringDB
 - Construct mesh: from all (*) country's probes to each other
 - *Maximum of two probes per ASN and only “public” probes with “good” geolocation
- Hops geolocated using “OpenIPMap” database

- Use this tool to find possible suboptimal routing and fix it
 - Find your ASN in the mesh
 - Find the person from another ASN
 - Take them out for tea :)
- To improve accuracy of this diagnostic tool
 - If your ASN is not on the graph, apply for a RIPE Atlas probe
 - Add more probes to your country to increase “resolution”
 - If you move, remember to update your probe’s geolocation

- Re-use and rewrite the code: it is free and open source software
 - <https://github.com/emileaben/ixp-country-jedi>
- Improve infrastructure geolocation: contribute data to OpenIPMap!
 - <https://marmot.ripe.net/openipmap/>
 - <https://github.com/RIPE-Atlas-Community/openipmap>