RIPE Atlas probes and the User-Defined Measurements

Stéphane Bortzmeyer
AFNIC
bortzmeyer@nic.fr

RIPE Atlas probes and the User-Defined Measurements

Stéphane Bortzmeyer
AFNIC
bortzmeyer@nic.fr

afnic

RIPE 67
Athens
october 2013

What is RIPE Atlas?

1. A set of small probes installed in many houses, LAN and data centers in the world. More volunteers welcome (specially outside of Europe).

afnic

What is RIPE Atlas?

1. A set of small probes installed in many houses, LAN and data centers in the world. More volunteers welcome (specially outside of Europe).
2. Reporting to the RIPE-NCC, which can instructs them to run active measurements.

afnic

What is RIPE Atlas?

1. A set of small probes installed in many houses, LAN and data centers in the world. More volunteers welcome (specially outside of Europe).
2. Reporting to the RIPE-NCC, which can instructs them to run active measurements.
3. Today, around 3 900 probes.

afnic

What is RIPE Atlas?

1. A set of small probes installed in many houses, LAN and data centers in the world. More volunteers welcome (specially outside of Europe).
2. Reporting to the RIPE-NCC, which can instructs them to run active measurements.
3. Today, around 3 900 probes.
4. A friendly botnet :-)

afnic

What is RIPE Atlas?

1. A set of small probes installed in many houses, LAN and data centers in the world. More volunteers welcome (specially outside of Europe).
2. Reporting to the RIPE-NCC, which can instructs them to run active measurements.
3. Today, around 3 900 probes.
4. A friendly botnet :-)

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

afnic

What is its use?

- Measurements by the RIPE-NCC for its services (K-root name server. . .),

What is its use?

- Measurements by the RIPE-NCC for its services (K-root name server. . .),
- Scientific papers on the working of the Internet,

afnic

What is its use?

- Measurements by the RIPE-NCC for its services (K-root name server. . .),
- Scientific papers on the working of the Internet,
- Funny operations problems (like the Juniper bug of network 128)

afnic

What is its use?

- Measurements by the RIPE-NCC for its services (K-root name server. . .),
- Scientific papers on the working of the Internet,
- Funny operations problems (like the Juniper bug of network 128)
- But also personal practical measurements such as “Can everybody talk to `2001:db8:1:42::bad:dcaf`?”

The Internet is critical for everything but yet not well known.

afnic

Real examples of practical use

- Strange connectivity problem at AFNIC because of a autistic router at AMS-IX (lots of ping, some traceroutes, to identify **where** the problem was)

Real examples of practical use

- Strange connectivity problem at AFNIC because of a autistic router at AMS-IX (lots of ping, some traceroutes, to identify **where** the problem was)
- Which ones of your anycast servers are used? (DNS) `https://labs.ripe.net/Members/stephane_bortzmeyer/using-atlas-udm-to-find-the-popular`

Real examples of practical use

- Strange connectivity problem at AFNIC because of a autistic router at AMS-IX (lots of ping, some traceroutes, to identify **where** the problem was)
- Which ones of your anycast servers are used? (DNS) `https://labs.ripe.net/Members/stephane_bortzmeyer/using-atlas-udm-to-find-the-popular`
- Censorship when going through China (DNS) `https://labs.ripe.net/Members/pk/denic-case-study-using-ripe-atlas`

afnic

Similar projects

- The company SamKnows `http://www.samknows.eu/` distributes probes in thousands of households inside the EU. Their active measurements are the basis of the “Broadband in Europe: Consumers are not getting the internet speeds they are paying for” `http://europa.eu/rapid/press-release_IP-13-609_en.htm`.

Similar projects

- The company SamKnows `http://www.samknows.eu/` distributes probes in thousands of households inside the EU.
- The european Leone research project `http://www.leone-project.eu/` also uses the SamKnows. (There was a talk at RIPE 66.)

Similar projects

- The company SamKnows `http://www.samknows.eu/` distributes probes in thousands of households inside the EU.
- The european Leone research project `http://www.leone-project.eu/` also uses the SamKnows.
- In the USA, there is the Bismark projet `http://projectbismark.net/`. Currently ˜ 150 active probes.

Less similar projects

Many projects based on software “probes”:

- Netalyzr `http://netalyzr.icsi.berkeley.edu/`, Java code to debug the network, used a lot among gamers, P2Pers. . .

Less similar projects

Many projects based on software “probes”:

- Netalyzr `http://netalyzr.icsi.berkeley.edu/`, Java code to debug the network, used a lot among gamers, P2Pers...
- Grenouille `http://grenouille.com/`, ISP performance measurements in France since 2000.

afnic

Less similar projects

Many projects based on software “probes”:

- Netalyzr `http://netalyzr.icsi.berkeley.edu/`, Java code to debug the network, used a lot among gamers, P2Pers...
- Grenouille `http://grenouille.com/`, ISP performance measurements in France since 2000.

Easier and cheaper to deploy, but dependency on the host software, less reliable for quantitative measures...

afnic

User-Defined Measurements

You can run your own measurements (UDM) on the RIPE Atlas probes.

1. But not your own programs: you can use only pre-defined protocols (ping, traceroute, DNS and TLS). HTTP is under study (it raises many issues).

User-Defined Measurements

You can run your own measurements (UDM) on the RIPE Atlas probes.

1. But not your own programs: you can use only pre-defined protocols (ping, traceroute, DNS and TLS). HTTP is under study (it raises many issues).
2. You can choose among a wide range of options (for DNS, you can ask TCP, NSID, DNSSEC DO, etc).

afnic

User-Defined Measurements

You can run your own measurements (UDM) on the RIPE Atlas probes.

1. But not your own programs: you can use only pre-defined protocols (ping, traceroute, DNS and TLS). HTTP is under study (it raises many issues).
2. You can choose among a wide range of options (for DNS, you can ask TCP, NSID, DNSSEC DO, etc).
3. You are limited in quantity: no way to run a dDoS.

afnic

The target

This is the machine you query during a measurement

If you want to compare N targets, you need to run N measurements

The credit system

1. You gain credits by hosting probes or buying credits through sponsorship or getting credits from other persons.

The credit system

1. You gain credits by hosting probes or buying credits through sponsorship or getting credits from other persons.
2. You spend credits by running UDM. Everything has a price. For instance, for DNS, TCP requests are twice as expensive as UDP ones.

The credit system

1. You gain credits by hosting probes or buying credits through sponsorship or getting credits from other persons.
2. You spend credits by running UDM. Everything has a price. For instance, for DNS, TCP requests are twice as expensive as UDP ones.
3. Credit consumption rate-limits the use of Atlas probes.

afnic

Start the workshop

Please check:

- Connectivity to `https://atlas.ripe.net`

Start the workshop

Please check:

- Connectivity to `https://atlas.ripe.net`
- Log in, and check the amount of credits

Start the workshop

Please check:

- Connectivity to `https://atlas.ripe.net`
- Log in, and check the amount of credits
- Create an API key and store it for future measurements

Start the workshop

Please check:

- Connectivity to `https://atlas.ripe.net`
- Log in, and check the amount of credits
- Create an API key and store it for future measurements
- Your favorite programming language.

afnic

Analyzing an UDM

Let's start with a **public** measurement, #1009150 (an IPv4 ping reachability measurement). Each measurement has an ID like #1009150

1. We can retrieve it from `https://atlas.ripe.net/api/v1/measurement/1009150/result/`. wget, curl, whatever. Download it.

Analyzing an UDM

Let's start with a **public** measurement, #1009150 (an IPv4 ping reachability measurement). Each measurement has an ID like #1009150

1. We can retrieve it from `https://atlas.ripe.net/api/v1/measurement/1009150/result/`. wget, curl, whatever. Download it.
2. The format is JSON and is documented in `https://atlas.ripe.net/doc/data_struct`. Let's examine the content.

afnic

Content of the JSON results

1. A big JSON array, one item per probe,

Content of the JSON results

1. A big JSON array, one item per probe,
2. Each item is a JSON object, one member is the array “result”, with one result per test (3 tests per probe, by default).

Important rules of Atlas analysis

1. Program cautiously: JSON members may be missing, for instance (yes, there is the documentation but not everything is in it)

Important rules of Atlas analysis

1. Program cautiously: JSON members may be missing, for instance (yes, there is the documentation but not everything is in it)
2. Use the actual data in the results, not the requested data. For instance, the number of probes reporting a result may be lower than the number you asked for. **Don’t assume**

Important rules of Atlas analysis

1. Program cautiously: JSON members may be missing, for instance (yes, there is the documentation but not everything is in it)
2. Use the actual data in the results, not the requested data. For instance, the number of probes reporting a result may be lower than the number you asked for. **Don’t assume**
3. Be prepared for things that are not yet ready when you ask for them. Response times may vary!

afnic

First analysis program

```
# Load the data in a JSON obect
results = json.loads(file.read())

# Main loop over the results
for probe in results:

    # Per-test loop
    for test in probe['result']:
        if test.has_key('x'):
            timeouts += 1
        elif test.has_key('error'):
            errors += 1
        elif test.has_key('rtt'):
            num_rtt += 1
            total_rtt += test['rtt']
```

Your turn: analyze #1009150

1. How many probes did not reach the target, even once?
2. Which are they? (field `prb_id`)

afnic

Periodic measurements

- You can also create periodic (automatically repeated) measurements

Periodic measurements

- You can also create periodic (automatically repeated) measurements
- They have a start time and a end time

afnic

Periodic measurements

- You can also create periodic (automatically repeated) measurements
- They have a start time and a end time
- They are of course much more costly (watch your credits!)

afnic

Analyze a periodic measurement

1. Download the results of #1025096 (DNS requests to `d.nic.fr`)

```
starttime = time.gmtime(result['timestamp'])
starttimes[starttime.tm_hour] += 1
rtpertime[starttime.tm_hour] += float(result['result']['
```

Analyze a periodic measurement

1. Download the results of #1025096 (DNS requests to `d.nic.fr`)
2. Plot the RTT per hour to see daily cycles

```
starttime = time.gmtime(result['timestamp'])
starttimes[starttime.tm_hour] += 1
rtpertime[starttime.tm_hour] += float(result['result']['
```

afnic

Analysis of a traceroute

The result of traceroute measurements is not very readable in JSON.

Use community-contributed `json2traceroute.py`:

```
% python json2traceroute.py 1013442.json
From:  130.79.86.251    2259    FR-U-STRASBOURG OSIRIS -
Source address:  130.79.86.251
Probe ID:  2279
1    130.79.86.253    2259    FR-U-STRASBOURG OSIRIS - U
2    193.51.183.130    2200    FR-RENATER Reseau Nationa
...
7    217.70.176.214    29169    GANDI-AS Gandi SAS    [1
8    217.70.176.250    29169    GANDI-AS Gandi SAS    [3
9    217.70.190.232    29169    GANDI-AS Gandi SAS    [1
```

Creating an UDM through the Web

1. Click “One-off measurement” and choose “ping”, one probe, and one of the Anchors as a target `https://atlas.ripe.net/anchors/list` (add `.anchors.atlas.ripe.net`)

Creating an UDM through the Web

1. Click “One-off measurement” and choose “ping”, one probe, and one of the Anchors as a target `https://atlas.ripe.net/anchors/list` (add `.anchors.atlas.ripe.net`)
2. “One-off measurement” and choose “DNS IN AAAA”, one probe, an ODVR resolver `https://www.dns-oarc.net/oarc/services/odvr` as the target and `www.afnic.fr` as the query.

Also, a command-line tool, using the API

https://github.com/astrikos/ripe-atlas-cmdline

```
% python atlas_manage.py oneoff -f ~/.atlas/auth
Specify Target:ns1.bortzmeyer.org
Specify Type:ping
Specify Start Time [Unix Timestamp\Leave blank for now]:
Specify Number of Probes (Integer):5
Specify Probes Source Type [area/country/prefix/asn/prob
Specify Probes Source [WW/West/North-Central/South-Centr
You are about to create a new oneoff RIPE Atlas UDM with
{'definitions': [{'description': 'Ping ns1.bortzmeyer.or
      'ping', 'target': 'ns1.bortzmeyer.org', 'is_oneoff
      'af': 4}], 'probes': [{'requested': 5, 'type': 'ar
[y/n]:y
A new oneoff UDM just created with id: 1026479
Seems we got more than 90% (5) of requested probes, slee
5 more secs to be sure we get the maximum probes Atlas c
```

Creating an UDM through the API

Basic rules:

1. Create a JSON object with the parameters

Creating an UDM through the API

Basic rules:

1. Create a JSON object with the parameters
2. Do a REST request to `https://atlas.ripe.net/api/v1/measurement/`

Creating an UDM through the API

Basic rules:

1. Create a JSON object with the parameters
2. Do a REST request to `https://atlas.ripe.net/api/v1/measurement/`
3. Parse the JSON result (you'll get the measurement ID)

Create the JSON parameters

Documentation in `https://atlas.ripe.net/doc/measurement-creation-api/`

```
{'definitions': [
          {'target': '192.0.2.1', 'af': 4, 'packets': 3,
           'type': 'ping', 'is_oneoff': True,
           'description': 'Ping 192.0.2.1 from GR'}],
 'probes': [
          {'requested': 5, 'type': 'country',
           'value': 'GR'}]}
```

Selection of probes

You have a wide choice of probe selection criteria:

1. By country,
2. By AS,
3. By prefix,
4. By measurement (“Use the same probes as in measurement #123456”)
5. By probe ID if you know the probes you’re interested in.

Do a REST request

Do not forget the media type `application/json`

```
# ``data'' is a Python object with the parameters
request = urllib2.Request(url)
request.add_header("Content-Type", "application/json
request.add_header("Accept", "application/json")
json_data = json.dumps(data)
conn = urllib2.urlopen(request, json_data)
```

Parse the result

```
results = json.load(conn)
print("Measurement #%s started" % (results["measurem

    except urllib2.HTTPError as e:
        print >>sys.stderr, ("Fatal error %s: %s" %
        sys.exit(1)
```

Parse the result

```
results = json.load(conn)
print("Measurement #%s started" % (results["measurem
```

Pay attention to the return code **and** the reason. 400 = wrong parameters, check the doc, 401 = wrong API key, . . .

```
except urllib2.HTTPError as e:
    print >>sys.stderr, ("Fatal error %s: %s" %
    sys.exit(1)
```

Start a periodic measurement

Start and end times have to be in number of seconds since the Unix Epoch. To measure during one week:

```
start_time_obj = datetime.datetime.utcnow()
end_time_obj = datetime.datetime.utcnow() + datetime.tim
# The next two integers will be send in the JSON data
start_time_unix = int((start_time_obj - \
         datetime.datetime(1970,1,1)).total_seconds())
end_time_unix = int((end_time_obj - \
         datetime.datetime(1970,1,1)).total_seconds())
```

Your turn: reachability test

1. Ping your employer's Web server with a one-off measurement

Your turn: reachability test

1. Ping your employer's Web server with a one-off measurement
2. 5 probes are more than enough

Retrieving the results

- When the REST request returns, the measurement is **not** over. The probes did not even receive the order.

Retrieving the results

- When the REST request returns, the measurement is **not** over. The probes did not even receive the order.
- You need to wait and to poll. Unfortunately, there is no easy way to know when it's over. (It will be improved soon.) For Python programmers, the package `RIPEAtlas` does it for you.

afnic

The algorithm

Remember the network of probes is a distributed one. Things are not synchronous.

The algorithm

1. Ask how many probes were allocated `https://atlas.ripe.net/api/v1/measurement/1009150/?fields=probes,status` It requires testing `data["status"]["name"]` to check that the measurement actually started (wait otherwise).

The algorithm

1. Ask how many probes were allocated `https://atlas.ripe.net/api/v1/measurement/1009150/?fields=probes,status`
2. Query `https://atlas.ripe.net/api/v1/measurement/1009150/result` and check that enough probes reported (wait otherwise). You may not get 100 % of probes reporting so you also need to request `https://atlas.ripe.net/api/v1/measurement/1009150/?fields=status` and test `data["status"]["name"]`.

afnic

Use of the RIPEAtlas module

Code at `https://github.com/RIPE-Atlas-Community/ripe-atlas-community-contrib/blob/master/RIPEAtlas.py`

```
import RIPEAtlas
data = { "definitions": [
            { "target": "2001:db8::f00:ba4",
              "description": "Ping my Web server" ...
measurement = RIPEAtlas.Measurement(data)
rdata = measurement.results(wait=True, percentage_requir
```

For periodic measurements, use `wait=True` for the constructor, too.

DNS analysis

1. DNS results are sent as a blob (the wire format of the response)

DNS analysis

1. DNS results are sent as a blob (the wire format of the response)
2. You need some DNS library to analyze it

DNS analysis

1. DNS results are sent as a blob (the wire format of the response)
2. You need some DNS library to analyze it
3. We use dnspython
   `http://www.dnspython.org/`

DNS

Example: analyze the result of AAAA requests

```
results = json.loads(open(filename).read())
for result in results:
    answer = result['result']['abuf'] + "=="
    content = base64.b64decode(answer)
    msg = dns.message.from_wire(content)
    for rrset in msg.answer:
            for rdata in rrset:
                if rdata.rdtype == dns.rdatatype.AAAA:
                    # Do something with rdata.address
```

Limits and problems of Internet measurements

- Ground truth: a problem that you detect can be in fact the normal state. Example: ping failure, because ICMP is filtered. Or IPv6 connectivity. Always compare with the ground truth.

Limits and problems of Internet measurements

- Ground truth
- Rate-limiting: many targets have rate-limiters, specially for ICMP echo requests. Many Atlas probes can have less success than a few.

afnic

Limits and problems of Internet measurements

- Ground truth
- Rate-limiting
- Network glitches: networks come and go. If you do three one-off tests at different moments, one may fall in a big network problem.

afnic

Conclusion

Expecting interesting papers from you at `https://labs.ripe.net/` or at RIPE meetings :-)

Merci !

www.afnic.fr
contact@afnic.fr