Present and future of IRRd

Massimo Candela
Principal Engineer
Global IP Network
massimo@ntt.net
@webrobotics

Sasha Romijn
Reliably Coded
sasha@reliablycoded.nl
@sash@hachyderm.io

IRRd 4

Internet Routing Registry daemon version 4 is an IRR database server, processing IRR objects in the RPSL format. Its main features are:

- Validating, cleaning and storing IRR data, and extracting information for indexing.
- Providing several query interfaces to query the IRR data.
- Mirroring other IRR databases using file imports and NRTM.

Massimo Candela | massimo@ntt.net | @webrobotics

A bit of history

- Commissioned in 2018 by NTT
  - Developed by Sasha Romijn
  - Open source
    - https://github.com/irrdnet/irrd
  - Support also received by: ARIN, Merit, RIPE NCC, LACNIC, Netnod and Internetstiftelsen
- Used by many:
  - NTT (NTTCOM)
  - Merit (RADb)
  - ARIN
  - LACNIC
  - and more…

IRRd Milestones

- 4.0 (May 2019)
  - Feature parity with IRRd v2/3 *with test coverage*

- 4.1 (Sept 2020)
  - Performance improvements
  - Scope filtering
  - RPKI-aware mode

- 4.2 (Sept 2021)
  - GraphQL query interface
  - API for retrieving and creating objects

- 4.3 (June 2023)
  - Source priority option

What we addressed in 4.4

- Total refactoring of authentication and authorization
  - Split authentication from maintainer objects
  - Introduced scopes
  - Introduced “superusers”
  - Introduced 2FA
  - Introduced API keys
    - for programmatic access with the API
    - possibility to assign scopes

What we addressed in 4.4

- Safer Person/Role/mntner data handling
  - They can be deleted only if not referenced anywhere (since 4.0)
  - Superusers can now delete and recreate whenever they want
  - To save an object, you need to fix all references
  - nic-handles/maintainers name cannot be reused
    - If ever used in 4.4, even if no longer visible in the data
    - If still visible in the data

What we addressed in 4.4

- Data preloading
  - Resolution of as-sets and route-objects is precalculated and stored in memory
    - including mbrs-by-ref
  - 3 to 9x times faster

4.3 vs. 4.4 performance

```
❯ time echo '!iAS2914:AS-GLOBAL,1' | nc
localhost 43|wc -c
662778
0.00s user 0.01s system 0% cpu  8.167 total
```

```
❯ time echo '!iAS2914:AS-GLOBAL,1' | nc
localhost 43|wc -c
662769
0.00s user 0.00s system 0% cpu  0.943 total
```

```
❯ time echo '!aAS-SEABONE' | nc localhost 43|wc
-c
39318385
wc -c  0.00s user 0.19s system 0% cpu  30.176
total
```

```
❯ time echo '!aAS-SEABONE' | nc localhost 43|wc
-c
39318366
wc -c  0.02s user 0.17s system 3% cpu  5.098
total
```

Next phase of IRRd 4

- SSO support
  - IRRd extended with SSO support through Keycloak, with Keycloak functioning as an intermediary between IRRd and any external OAuth/OpenID system
  - Identity providers can be PeeringDB and RIR accounts
- Even more performance improvement!

- Provide feedback: https://github.com/irrdnet/irrd

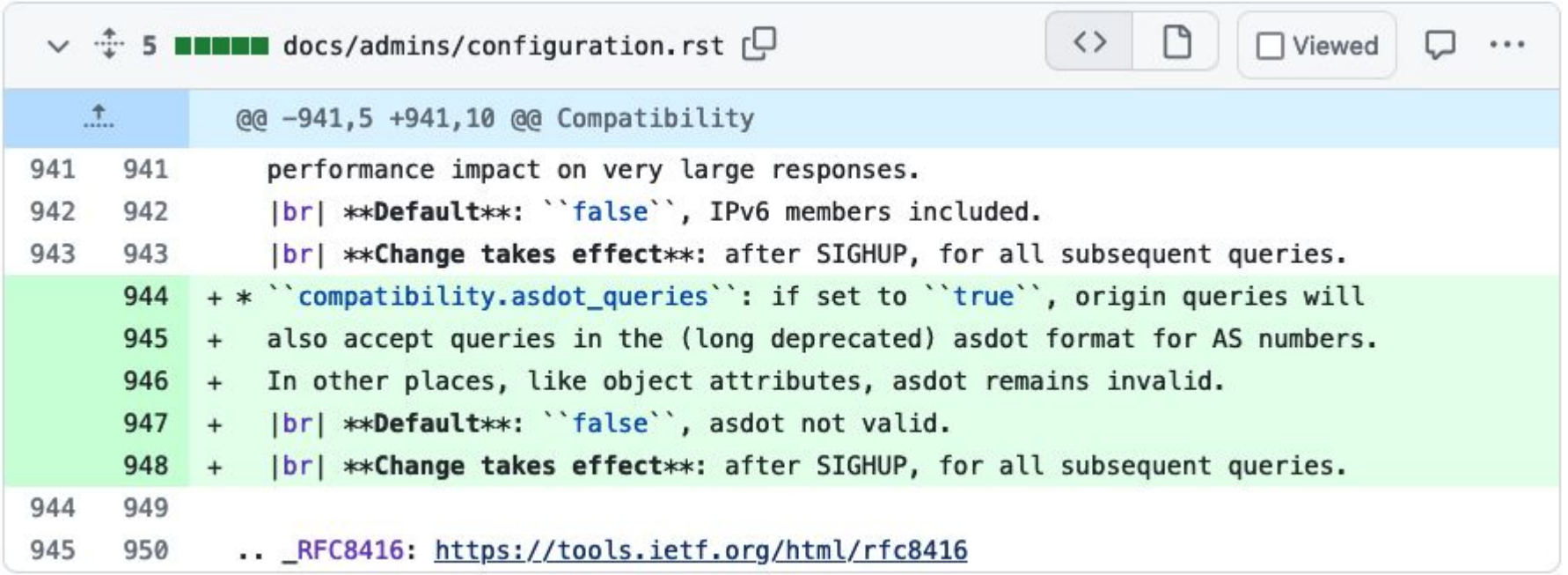
Massimo Candela | massimo@ntt.net | @webrobotics

It’s not just port 43

Sasha Romijn - @sash@hachyderm.io - sasha@reliablycoded.nl

IRRD development process

- Users include several authoritative operators and mirrors
- IRRD should generally remain compatible with all these deployments
- Reasonable defaults
- Careful design to manage complexity
- Occasional reluctant unusual features


5 docs/admins/configuration.rst
Viewed
@@ -941,5 +941,10 @@ Compatibility
941 941 performance impact on very large responses.
942 942 |br| **Default**: ``false``, IPv6 members included.
943 943 |br| **Change takes effect**: after SIGHUP, for all subsequent queries.
944 + * ``compatibility.asdot_queries``: if set to ``true``, origin queries will
945 + also accept queries in the (long deprecated) asdot format for AS numbers.
946 + In other places, like object attributes, asdot remains invalid.
947 + |br| **Default**: ``false``, asdot not valid.
948 + |br| **Change takes effect**: after SIGHUP, for all subsequent queries.
944 949
945 950 .. _RFC8416: https://tools.ietf.org/html/rfc8416

IRRD is a large project

- 17.000 lines regular code
- 15.000 lines tests
- 6.000 lines documentation

Data flows in IRRD

Loading IRR data:

- Authoritative via mail, HTTPS API, web form
- NRTM v3
- NRTM v4
- Various file imports
- Synthetic NRTM
- RPKI pseudo-IRR
- Strict and non-strict

Enrichment/filtering:

- Object suppression for RPKI, scope filtering, route object preference
- Maintainer suspension

Publication:

- Whois on TCP 43
- Whois HTTPS API
- GraphQL HTTPS
- NRTM v3
- NRTM v4
- Event stream with HTTPS JSONL download and WebSockets

Querying beyond plain TCP

Sasha Romijn - @sash@hachyderm.io - sasha@reliablycoded.nl

```
$ telnet rr.ntt.net 43
Trying 2001:418:3ff:5::192:40...
Connected to rr.ntt.net.
Escape character is '^]'.
!!
!aAS-RELIABLYCODED
A52
2001:678:d44::/48 2001:678:d44:1::/64 2.57.252.0/24
C
!iAS2914:AS-GLOBAL,1
A671808
AS1 AS10 AS100 AS10000 AS10001 ...
```

16 IRRD style queries, 6 RIPE queries, 5 flags

```
❯ curl
'https://irrd.as213279.net/v1/whois/?q=!oRIPE-NCC-MNT'

inetnum:        84.205.64.0 - 84.205.95.255
netname:        RIPE-NCC-RIS-BEACON
org:            ORG-RIEN1-RIPE
country:        EU
remarks:        RIPE NCC RIS anchors and beacons for BGP
studies
admin-c:        DUMY-RIPE
...
```

Exact same query interface over HTTPS

```
query {
  rpslObjects(mntBy: "DEMO-MNT") {
   rpslPk
   mntBy
   source
   ... on RPSLAsSet {
     members
   }
   ... on RPSLRouteSet {
     members
   }
  }
}
```

GraphQL query

```
{
  "data": {
    "rpslObjects": [
      {
       "rpslPk": "AS-EXAMPLE",
       "mntBy": ["DEMO-MNT"],
       "source": "RIPE",
       "members": ["AS64500", "AS64501", "AS-EXAMPLE2"]
      }
    ]
  }
}
```


GraphQL response

```
{rpslObjects(rpslPk: "2001:7FB::/32", sources:
["RIPE"]) {
   rpslPk
   source
   mntByObjs {
      rpslPk
      adminCObjs {
         ... on RPSLPerson {
            address
         }
      }
   }
}
```

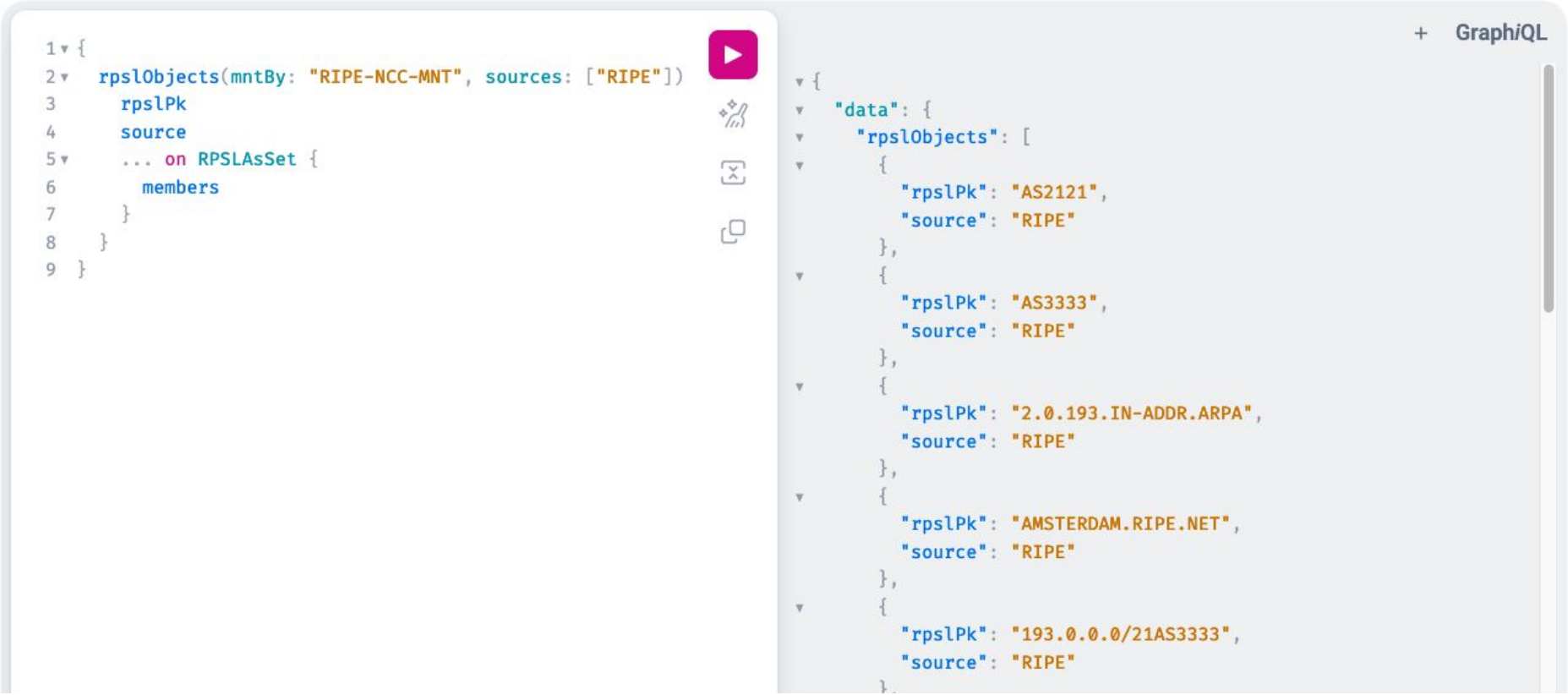
GraphQL query

```
{"data": {
   "rpslObjects": [
   {
      "rpslPk": "2001:7FB::/32",
      "source": "RIPE",
      "mntByObjs": [
         {
         "rpslPk": "RIPE-NCC-END-MNT",
         "adminCObjs": [
            {
               "address": [
               "RIPE Network Coordination Centre",
               "P.O. Box 10096",
```


GraphQL response

```
{
  q1: rpslObjects(
   mntBy: "ONE-MNT",
   sources: ["RIPE"]) {
   rpslPk
   source
  }
  q2: rpslObjects(
   ipLessSpecificOneLevel: "192.0.2.0/24",
   rpkiStatus: valid) {
   rpslPk
  }
}
```

Combine queries



GraphQL interactive playground

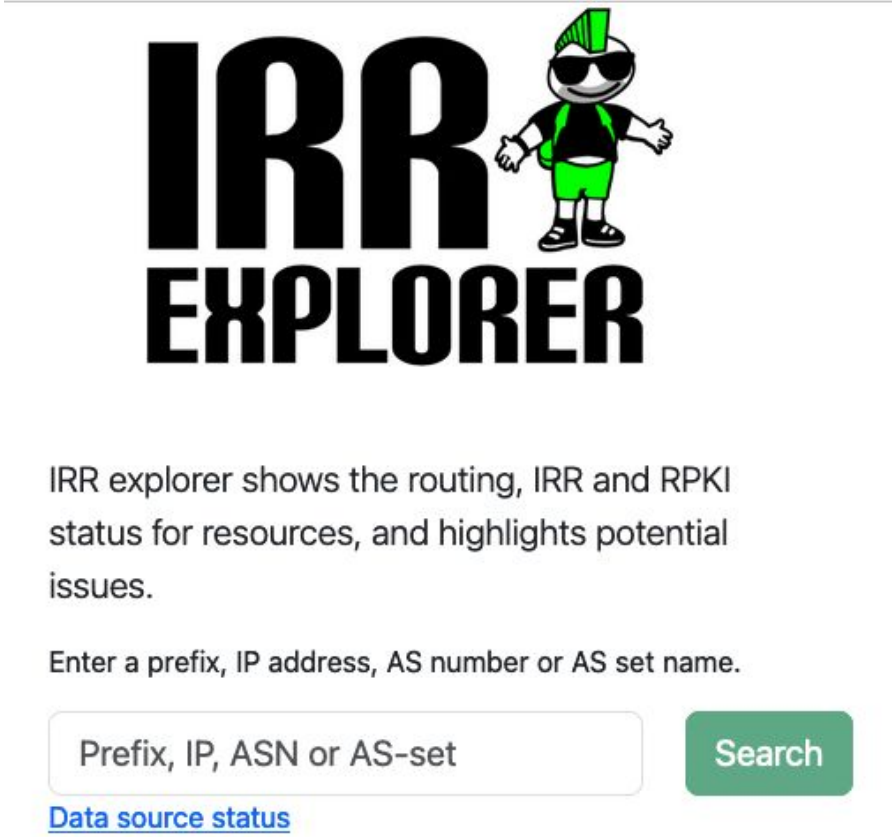
GraphQL help

```
❯ curl -d '{"query": "{asSetPrefixes(setNames:
[\"AS-RIPENCC\"]){rpslPk prefixes}}"}' -H
"Content-Type: application/json"
https://irrd.as213279.net/graphql/

{"data":{"asSetPrefixes":[{"rpslPk":"AS-RIPENCC","pref
ixes":["2001:7fb:fe14::/48","2001:7fb:fe17::/48","84.2
05.70.0/24",
...
```

Tiny layer on top of HTTPS POST

Can I query X or Y from IRRD?



90% of IRRexplorer is just an IRRD GraphQL frontend

NRTM v4

Near Real Time Mirroring v4

Sasha Romijn - @sash@hachyderm.io - sasha@reliablycoded.nl

Mirroring / replication

- One or two dozen IRRs
- Mirroring / replication to access data from different IRRs in one place
- Allows a single source for queries
- Some run local mirrors for performance
- All based on NRTM v3
- RFC2769: Routing Policy System Replication - no active implementations

NRTM v3

- “Protocol” is a big word
- Zero integrity or authenticity checks
- Poor scaling, tied to port 43
- Potential inconsistency between FTP dump and NRTM
- No consistent charset
- Silent desynchronisation
- No way to distinguish “in sync” from “everything is broken”
- Many very exciting, silent and undetectable ways to lose synchronisation

draft-ietf-grow-nrtm-v4

- Authored together with Job Snijders, Ed Shryane and Stavros Konstantaras
- Some inspiration from RRDP
- JSON-ish files on any HTTPS endpoint
- Signature and hashes for authenticity
- Single publication point and session IDs for consistency
- UTF-8 support
- Object format out of scope

“In practice, there is no uniformly implemented standard for RPSL, but merely rough outlines partially documented in different places.”

– draft-ietf-grow-nrtm-v4

draft-ietf-grow-nrtm-v4

- Small Update Notification File as a kind of index pointing to a snapshot and (usually) deltas
- Snapshot is a full dump of all the data in an IRR database
- Deltas contain changes, batched into one minute timeframes
- Snapshots and deltas completely cacheable
- JSON / JSON sequences on HTTPS endpoint

Goals and impact

- Improved reliability, security and scalability in mirroring
- No (fewer?) silent errors
- Scalability may lead to more open access to NRTM?

- If you process dumps or NRTM with your own code, may need updates
- Probably NRTMv3 will still be available for quite some time

Status and plans

- RIPE NCC has a mirror server implementation in production
- IRRD has mirror client in testing
- Interoperability achieved for most features
- Some still to be implemented
- Reverse direction to be developed
- Draft adopted by IETF GROW
- v03 published
- My work supported by LACNIC and RIPE NCC CPF

Thank you!

Massimo Candela
Principal Engineer
Global IP Network
massimo@ntt.net
@webrobotics

Sasha Romijn
Reliably Coded
sasha@reliablycoded.nl
@sash@hachyderm.io