

ICANN's Command Line Interface RDAP Client

Perspectives on RDAP When Building a Client

Andy Newton

Registry Operations Workshop 12

20 June 2023



About Me (and RDAP)

Currently:

- Principal Engineer, GDS Technical Services, ICANN
- Author of the ICANN RDAP Command Line Interface client.



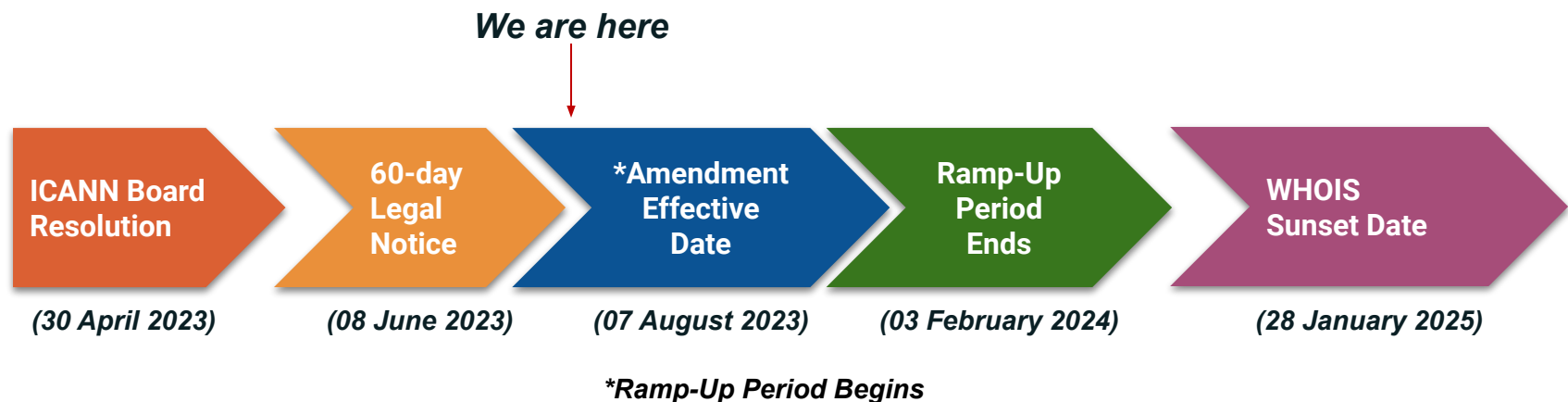
Past:

- Co-author of the foundational RDAP RFCs: 7480, 8521, 9082, & 9083.
- Co-author of two RDAP extensions.
- Original implementer of ARIN's Registry RDAP server.
- Original implementer of ARIN's Bootstrap RDAP server.
- Original implementer of NicInfo.

Though insights are from lessons learned, opinions offered are my own.

RDDS timeline in the gTLDs

- The Registration Data Directory Service (RDDS) is provided over WHOIS service available via port 43, web-based WHOIS, and the Registration Data Access Protocol (RDAP).
- The ICANN board has approved an RDAP Amendment to the contracts of the Registries and Registrars, and it's expected to be effective shortly.
- As part of the RDAP Amendment, RDAP will be the only required mechanism for RDDS in the future.



RDAP @ the command prompt

Complaint Form
RDNS Inaccuracy Complaint Form: <https://www.icann.org/wicf>
<https://www.icann.org/wicf> of type 'application/rdap+json' for <https://www.icann.org/wicf>

Domain icann.org

Identifiers	
LDH Name	icann.org
Unicode Name	icann.org
Handle	G28dbbcb4edc464b9401cbadea0a08b2-LROR

Events

Transfer	• Wed, 12-Aug-2009 17:40:26 +00:00
Expiration	• Tue, 7-Dec-2027 17:04:26 +00:00
Registration	• Mon, 14-Sep-1998 04:00:00 +00:00
Last Changed	• Tue, 14-Feb-2023 22:18:25 +00:00
Last Update Of RDAP Database	• Wed, 24-May-2023 11:29:11 +00:00

Links

Related	• https://rdap.godaddy.com/v1/domain/icann.org • application/rdap+json • https://rdap.godaddy.com/v1/domain/icann.org
Self	• https://rdap.publicinterestregistry.org/rdap/domain/icann.org • application/rdap+json • https://rdap.publicinterestregistry.org/rdap/domain/icann.org

Registrant

Identifiers	
Handle	

Events

Last Update Of RDAP Database	• Wed, 24-May-2023 11:29:11 +00:00
------------------------------	------------------------------------

Checks

SpecWarn	• RFC 9083 recommends self links for all object classes
-----------------	---

in this object has been removed.

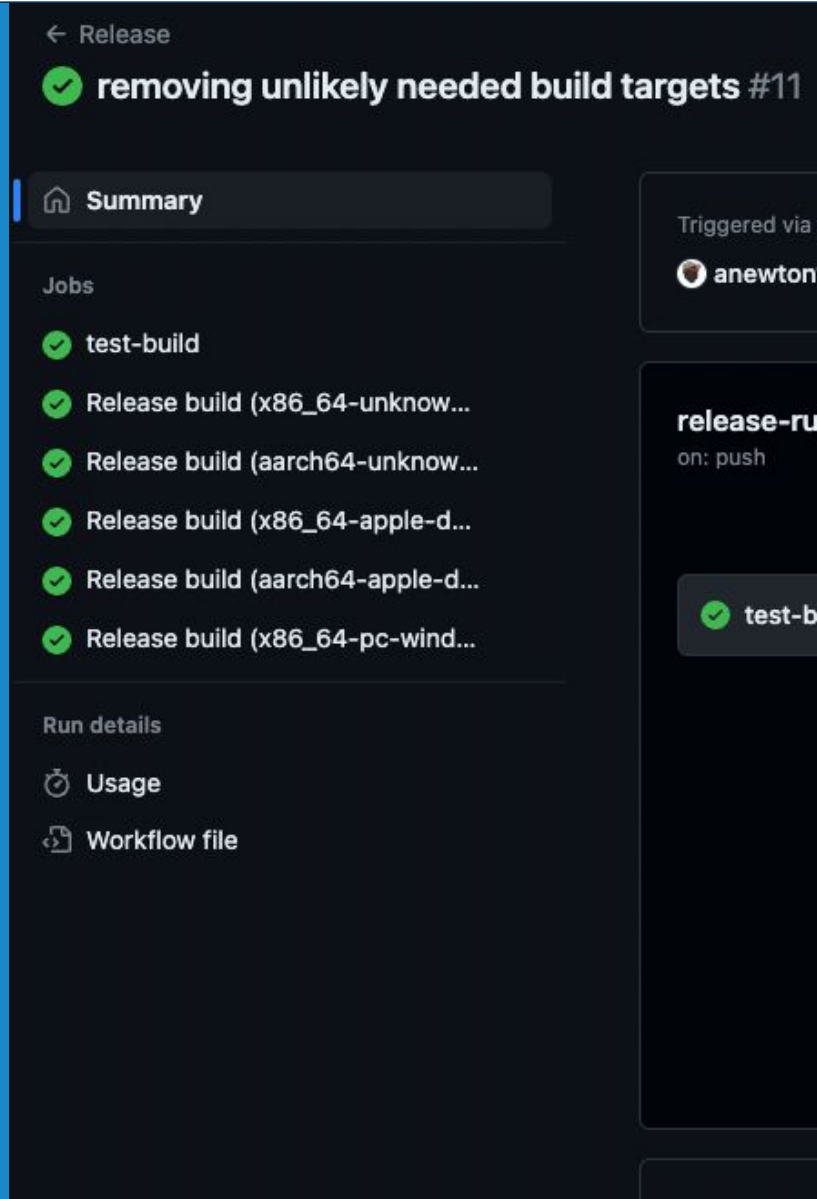
PRIVACY
RDNS service of the Registrar of Record identified in this output for information on how to contact the Registrar

ICANN has the web-based RDAP Lookup Tool (<https://lookup.icann.org>). Why create an RDAP command line interface (CLI)?

- The command prompt is where network operators and many other technicians live.
- CLI tools can be incorporated into scripts for automation purposes.
- As a library, it can be incorporated into more sophisticated programs (e.g. intrusion detection systems, spam filtering, etc...)

RDAP CLI Plan & Goals

- Code written in Rust.
 - Known for its correctness and memory safety.
 - Vibrant eco-system for developing CLI tools.
 - High adoption-plane for library incorporation.
 - Contact API to help deal with JCard.
- Release code as open source on GitHub.
- Produce executable binaries for popular architectures.
 - Best for getting into OS distributions.
- Use experience and learned-lessons as feedback to the standards process.



The screenshot shows a GitHub Actions workflow run for the repository 'removing unlikely needed build targets #11'. The workflow is triggered via a push event. The 'Jobs' section lists several jobs, all of which are completed successfully (indicated by green checkmarks): 'test-build', 'Release build (x86_64-unknow...', 'Release build (aarch64-unknow...', 'Release build (x86_64-apple-d...', 'Release build (aarch64-apple-d...', and 'Release build (x86_64-pc-wind...'. The 'Run details' section is partially visible, showing 'Usage' and 'Workflow file'.

Planned Features

- Automatic query type detection.
- Built-in Bootstrapping.
 - including mnemonic base URLs using RDAP object tags.
- Caching.
- Multiple output types.
 - human readable tables with ANSI colors.
 - Legacy Whois.
 - JSON
- Built-in paging.
- Support for many RDAP extensions.

```
Compiling futures v0.3.27
Compiling predicates v3.0.3
Compiling predicates-tree v1.0.9
Compiling wait-timeout v0.2.0
Compiling icann-rdap-client v0.0.5 (/Users/andy.newton/proj
Compiling rstest v0.17.0
Compiling icann-rdap-srv v0.0.5 (/Users/andy.newton/project
Compiling icann-rdap-cli v0.0.5 (/Users/andy.newton/project
Compiling assert_cmd v2.0.11
Finished test [unoptimized + debuginfo] target(s) in 8.94s
Running unittests src/main.rs (target/debug/deps/rdap-c40
```

```
running 1 test
```

```
test tests::cli_debug_assert_test ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0
```

```
Running tests/integration/main.rs (target/debug/deps/inte
```

```
running 7 tests
```

```
2023-05-31T11:44:16.100763Z INFO icann_rdap_srv::server: rdap
2023-05-31T11:44:16.100762Z INFO icann_rdap_srv::server: rdap
2023-05-31T11:44:16.100761Z INFO icann_rdap_srv::server: rdap
2023-05-31T11:44:16.100760Z INFO icann_rdap_srv::server: rdap
2023-05-31T11:44:16.100901Z WARN icann_rdap_srv::server: Serv
2023-05-31T11:44:16.100937Z WARN icann_rdap_srv::server: Serv
2023-05-31T11:44:16.100937Z WARN icann_rdap_srv::server: Serv
2023-05-31T11:44:16.100963Z WARN icann_rdap_srv::server: Serv
2023-05-31T11:44:16.106669Z INFO icann_rdap_srv::server: rdap
2023-05-31T11:44:16.106714Z WARN icann_rdap_srv::server: Serv
2023-05-31T11:44:16.107233Z INFO icann_rdap_srv::server: rdap
2023-05-31T11:44:16.107258Z WARN icann_rdap_srv::server: Serv
2023-05-31T11:44:16.116491Z INFO icann_rdap_srv::server: rdap
2023-05-31T11:44:16.116518Z WARN icann_rdap_srv::server: Serv
test check::GIVEN_domain_with_check_WHEN_query_THEN_failure ..
test queries::GIVEN_authnum_WHEN_query_THEN_success ... ok
test queries::GIVEN_network_ip_WHEN_query_THEN_success ... ok
test queries::GIVEN_network_cidr_WHEN_query_THEN_success ... o
```

Future Possibilities

```
4 | nameserver::Nameserver,
5 | network::Network,
6 | types::{Common, Events, Links, ObjectCommon, PublicIds},
7 | };
8 |
9 | /// Represents an RDAP variant name.
10 | #[derive(Serialize, Deserialize, Builder, Clone, Debug, PartialEq, Eq)]
11 | pub struct VariantName {
12 |     #[serde(rename = "ldhName")]
13 |     #[serde(skip_serializing_if = "Option::is_none")]
14 |     pub ldh_name: Option<String>,
15 |
16 |     #[serde(rename = "unicodeName")]
17 |     #[serde(skip_serializing_if = "Option::is_none")]
18 |     pub unicode_name: Option<String>,
19 | }
20 |
21 | /// Represents an RDAP IDN variant.
22 | #[derive(Serialize, Deserialize, Builder, Clone, Debug, PartialEq, Eq)]
23 | pub struct Variant {
24 |     pub relation: Option<Vec<String>>,
25 |
26 |     #[serde(rename = "idnTable")]
27 |     #[serde(skip_serializing_if = "Option::is_none")]
28 |     pub idn_table: Option<String>,
29 |
30 |     #[serde(rename = "variant_names")]
31 |     #[serde(skip_serializing_if = "Option::is_none")]
32 |     pub variant_names: Option<Vec<VariantName>>,
33 | }
34 |
35 | #[derive(Serialize, Deserialize, Builder, Clone, Debug, PartialEq, Eq)]
36 | pub struct DsDatum {
37 |     #[serde(rename = "keyTag")]
38 |     #[serde(skip_serializing_if = "Option::is_none")]
39 |     pub key_tag: Option<u32>,
40 |
41 |     #[serde(skip_serializing_if = "Option::is_none")]
42 |     pub algorithm: Option<u8>,
43 |
44 |     #[serde(skip_serializing_if = "Option::is_none")]
45 |     pub digest: Option<String>,
46 |
47 |     #[serde(rename = "digest_type")]
48 |     #[serde(skip_serializing_if = "Option::is_none")]
49 |     pub digest_type: Option<u8>,
50 |
51 |     #[serde(skip_serializing_if = "Option::is_none")]
```

Hopefully with the support of the community, there are some interesting things that can be done in the future:

- Create native library bindings for other languages, such as Python.
- Be a platform for experimenting...
 - Device flow for OpenID Connect.
 - Proving out other RDAP extensions.
- Enable RDAP-based research tooling.

Observations on the RDAP protocol, so far...

Experience gained from writing a client provides different insight than that of writing a server.

Server Source

- Users often want to know who provided the data.
- But there is no data in an RDAP response identifying the server providing the response.

Raw Registry RDAP Response



Raw Registrar RDAP Response



(from the ICANN registration data Lookup Tool - <https://lookup.icann.org>)

This could be done with a simple RDAP extension.

- Display name of server operator.
- Type of service (domain registrar, INR, etc...).
- Base RDAP URL.
- Cacheable mnemonic.

Object Tags Make Great Mnemonics

- RFC 8521 defines RDAP Object Tags
 - Bootstrapping for contacts using handle suffix.
 - The -ARIN in foo-ARIN points to ARIN's base URL.
- Can also be used by clients to issue queries directly to an RDAP server.
 - instead of -B `https://rdap.lunarnic.com/v1`
 - use -b `lunar`
- *Hint! Hint! - registration is first come, first serve.*

Client Extension Signaling

- RDAP has an extension mechanism.
- Servers can signal to clients which extensions are supported.
- But there is no way for clients to signal desired extensions to servers.

An RDAP With Extensions Media Type

Abstract

This document defines a media type for RDAP that can be used to describe RDAP content with RDAP extensions. Additionally, this document describes the usage of this media type with RDAP.

(shameless plug)

- RDAP-X media type
- <https://www.ietf.org/archive/id/draft-newton-regext-rdap-x-media-type-00.html>

Attn IETF: Dog Food Tastes Awful!



Eating your own dog food or "dogfooding" is the practice of using one's own products or services.

In the 1970s television advertisements for Alpo dog food, Lorne Greene pointed out that he fed Alpo to his own dogs.

Another possible origin is from the president of Kal Kan Pet Food, who was said to eat a can of his dog food at shareholders' meetings.

-Wikipedia

Why Does RDAP use JCard?

The first RDAP I-Ds did not use JCard. Contacts looked more like EPP contacts.

The IETF working group was asked to “eat its own dogfood” and use JCard.

What is wrong with JCard?

- Easy to mess up serialization (examples abound).
- Hard to parse (see the code base).
- *“Many of the types of information that can be represented with jCard have little or no use in RDAP, such as birthdays, anniversaries, and gender.” - RFC 9083*



Will JSContact Make it Better? - IMHO: No



The REGEXT WG is currently working on a new contact format using JSContact.

This is more “eating your own dogfood” and does not fix the underlying problem.

- Too complex for RDAP needs.
- Easy to get wrong.

RDAP could benefit from a SimpleContact extension:

- Use EPP contact as a base (or EPP contact 2.0?).
- Add the few extra things used by ccTLDs and RIRs.

This is just my opinion and not a consensus opinion.

- I intend to present on this at the next IETF.
- Please participate in REGEXT wg.

Common Problems w/ RDAP Servers

Commonly found problems while testing the RDAP CLI.

Some Common Issues w/ RDAP Servers

- Connectivity:
 - IPv6 reachability issues
 - Obsolete/Vulnerable TLS version support
 - Restrictive rate-limiting
- HTTP headers in the RDAP response
 - Not honoring RDAP media type in Accept
 - Not using RDAP media type in Content-Type

Some More Common Issues w/ RDAP Servers

- JSON errors
 - Non-RDAP error responses
 - Incorrect capitalization of JSON names
- jCard errors
 - No “fn” property
 - Misaligned postal address arrays
- Link objects
 - Missing ‘rel’ and ‘value’ attributes.
 - Objects without ‘self’ links.

Summary / Q&A

To Summarize

A new RDAP CLI is on the horizon.

This will broaden the RDAP ecosystem.

Lessons-learned will be fed back into the standardization process.

Q&A



HTTP 406: Dog Food Not Accepted