RDAP Conformance Tool

Functional Specifications

1. Overview

This document describes the requirements for the RDAP Conformance Tool.

The RDAP Conformance Tool is a stand-alone tool that verifies that an RDAP server implements the specifications developed by the IETF and the ICANN gTLD RDAP profile (https://www.icann.org/gtld-rdap-profile).

2. General requirements

The tool supports two schemes for URIs: "http" and "https". An URI is required, and the tool will try to connect to the specified server to obtain an RDAP response by using the HTTP GET method.

If the scheme of the URI is "https", the tool will initiate a TLS connection to the server:

- The tool shall not try to match CA certs (if available) to a well-known CA.
- If the CRL or OCSP is unavailable, this won't constitute an error, but if CRL or OCSP are accessible and indicate that the server certificate is revoked, the revocation constitutes an error.

3. Input data

The tool shall support the below optional (except where indicated) parameters:

- Definition file (--*config* <file>, required)
- Timeout for connecting to the server (--*timeout* <seconds>, default=20s)
- Maximum number of redirects to follow (--*maximum-redirects* <number>, default=3)
- Use locally-persisted datasets (--*use-local-datasets*)
- Use RDAP Profile February 2019 (--*use-rdap-profile-february-2019*)
- Validate the response as coming from a gTLD registry (--*gtld-registry*)
- Validate the response as coming from a gTLD registrar (--*gtld-registrar*)
- The TLD uses the thin model (--*thin*)

Note: if --*use-rdap-profile-february-2019* is set, one of --*gtld-registry* or --*gtld-registrar* shall be set. The --*thin* option can only be used if --*gtld-registry* is set.

After the parameters, the URI to be tested shall be specified.

3.1. Configuration definition

The definition file specifies which single tests are errors, warnings, or ignored (i.e. not tested for).
Note: a definition file is required for starting a test.
A configuration definition file specifies:

- *definitionIdentifier*: a required JSON string that identifies the configuration definition file. The string is copied verbatim to the *definitionIdentifier* element of the results file.
- *definitionError*: an optional JSON array of objects.
    - Each object contains the following elements:
        - *code*: a required JSON number that identifies a single test.
        - *notes*: a required JSON string that is copied verbatim if the test fails, generating an entry in the *results* section in the results file.
- *definitionWarning*: an optional JSON array of objects.
    - Each object contains the following elements:
        - *code*: a required JSON number that identifies a single test.
        - *notes*: an optional JSON string that is copied verbatim if the test fails, generating an entry in the *results* section in the results file.
- *definitionIgnore*: an optional JSON array of single test identifiers that are ignored (i.e. not tested for). The contents of this element are copied verbatim to the *ignore* section in the results file.
- *definitionNotes*: an optional JSON array of strings that are copied verbatim to the *notes* section in the results file.

An example of a configuration definition file is shown below:

```
{
	"definitionIdentifier": "gTLD Profile Version 1.0",
	"definitionError": [{
		"code": -1102,
		"notes": "If the gTLD is a legacy gTLD, this may not indicate an error, review by a
person is required."
	}],
	"definitionWarning": [{
		"code": -2186,
		"notes": "This only applies for a few gTLDs."
	}],
	"definitionIgnore": [-2323, -2345, -2346],
	"definitionNotes": ["This is a configuration definition for a legacy gTLD.", "Developed by
ICANN."]
}
```

4. Output data

After the tool executes, it produces one return code of the following:

0. A response was available to the tool, the Content-Type is application/rdap+JSON in the response, a HTTP Status of 200 or 404 was received, the RDAP response was successfully parsed and the results file was generated.
1. The configuration definition file is syntactically invalid.
2. The tool was not able to download a dataset.
3. The RDAP query is not supported by the tool.
4. The RDAP query is domain/<domain name> or nameserver/<nameserver>, but A-labels and U-labels are mixed.
5. A response was available to the tool, but the Content-Type is not application/rdap+JSON in the response.
6. A response was available to the tool, the Content-Type is application/rdap+JSON in the response, but the RDAP response is not a syntactically valid JSON object.
7. A response was available to the tool, the Content-Type is application/rdap+JSON in the response, but the HTTP Status is not 200 nor 404.
8. A response was available to the tool, the Content-Type is application/rdap+JSON in the response, the HTTP Status is 200 or 404, but the expected objectClassName in the topmost object was not found for a lookup query or the JSON array for a search query was not found a search query.
9. The RDAP query is invalid because the TLD uses the thin model.
10. Failed to connect to host.
11. The TLS handshake failed.
12. TLS server certificate - common name invalid.
13. TLS server certificate - revoked.
14. TLS server certificate - expired.
15. Other errors with the TLS server certificate.
16. Too many redirects.
17. HTTP errors.
18. HTTP/2 errors.
19. Failure sending network data.
20. Failure in receiving network data.

Additionally, a results file may be produced.

4.1. Results file

The results file is stored in a folder called results.

The filename of the results file follows the pattern: results-*YYYYMMDDhhmmss*.json:

- *YYYY*, year when the file was generated.
- *MM*, month when the file was generated.
- *DD*, day when the file was generated
- *hh*, hour when the file was generated.
- *mm*, minute when the file was generated.
- *ss*, second when the file was generated.

Note: the date and time in the filename is in UTC.

The results file contains (all JSON names are always outputted, however, values may be empty) the following elements:

- *definitionIdentifier*: a JSON string containing the value of the definition identifier element from the configuration definition file.
- *testedURI*: a JSON string containing the URI that was tested.
- *testedDate*: a JSON string containing the date and time (using Internet Date format, see RFC3339) in UTC when the test was executed.
- *receivedHttpStatusCode*: a JSON number containing the HTTP status code received from the server.
- *groupOK*: a JSON array of strings that contain the tests groups for which all the tests passed.
- *groupErrorWarning*: a JSON array of strings that contain the test groups for which at least one test resulted in error or warning.
- *results*: a JSON object with the following elements:
    - *error*: an optional JSON array of objects denoting tests that failed and are considered non-compliance issues.
        - Each object contains the following elements:
            - *code*: a JSON number containing the test identifier.
            - *value*: a JSON object where the issue was found encoded in Base64 (RFC4648).
            - *message*: a JSON string explaining the issue.
            - *notes*: a JSON string containing the notes defined in the configuration definition file.
    - *warning*: an optional JSON array of objects denoting tests that failed but are not considered non-compliance issues.
        - Each object contains the following elements:
            - *code*: a JSON number containing the test identifier.
            - *value*: a JSON string where the issue was found.
            - *message*: a JSON string explaining the issue.
            - *notes*: a JSON string containing the notes defined in the configuration definition file.
    - *ignore*: a JSON array of test identifiers that were ignored.
    - *notes*: a JSON array of strings containing the value of the *definitionNotes* element in the configuration definition file.

An example of the results file is shown below:


```
{
  "definitionIdentifier": "gTLD Profile Version 1.0",
  "testedURI": "https://localhost/domain/tested.example",
  "testedDate": "1990-12-31T23:59:59Z",
  "receivedHttpStatusCode": 200,
  "groupOK": ["domainNameValidation", "stdRdapConformanceValidation"],
  "groupErrorWarning": ["ipv4Validation", "ipv6Validation"],
  "results": {
    "error": [{
        "code": -1100,
        "value": "MjU2LjI1Ni4yNTYuMjU2LjI1Ng==",
        "message": "The IPv4 address is not syntactically valid in dot-decimal notation. See RFC791"
      },
      {
        "code": -1102,
        "value": "MTI3LjAuMC4x",
        "message": "The IPv4 address is included in the IANA IPv4 Special-Purpose Address Registry. Dataset: specialIPv4Addresses",
        "notes": "notes1"
      }
    ],
    "warning": [{
      "code": -2186,
      "value":
"ewoJImV2ZW50cyI6IFt7CgkJCSJldmVudEFjdGlvbiI6ICJyZWdpc3RyYXRpb24iCgkJfSwKCQl7CgkJCSJldmVudEFjdGlvbiI6ICJsYXN0IGNoYW5nZWQiLAoJCQkiZXZlbn
REYXRlIjogIjE5OTEtMTItMzFUMjM6NTk6NTlaIiwKCQkJImV2ZW50QWN0b3IiOiAiam9lQGV4YW1wbGUuY29tIgoJCX0KCV0KfQ==",
      "message": "The eventDate element does not exist.",
      "notes": "notes2"
    }],
    "ignore": [-2323, -2345, -2346],
    "notes": ["notes3", "notes4"]
  }
}
```

5. Workflow of the tool

1. Parse the configuration definition file, and if the file is not parsable, exit with a return code of 1.
2. If the parameter (*--use-local-datasets*) is set, use the datasets found in the filesystem, download the datasets not found in the filesystem, and persist them in the filesystem. If the parameter (*--use-local-datasets*) is not set, download all the datasets, and overwrite the datasets in the filesystem. If one or more datasets cannot be downloaded, exit with a return code of 2.
3. Verify the URI represent one of the following RDAP queries, and if not, exit with a return code of 3.
    a. *domain/<domain name>*
    Note: the domain name must pass Domain Name validation [domainNameValidation], however, A-labels and U-labels (see IDNA_RFCs) shall not be mixed. If A-labels and U-labels are mixed, the tool shall exit with the return code of 4.

    b. *nameserver/<nameserver name>*
    Note: the nameserver name must pass Domain Name validation [domainNameValidation], however A-labels and U-labels (see IDNA_RFCs) shall not be mixed. If A-labels and U-labels are mixed, the tool shall exit with the return code of 4.

    c. *entity/<handle>*
    d. *help*
    e. *nameservers?ip=<nameserver search pattern>*
4. Connect to the RDAP server, and if there is a connection error, exit with the corresponding return code.
5. If a response is available to the tool, and the header *Content-Type* is not *application/rdap+JSON*, exit with a return code of 5.
6. If a response is available to the tool, but it's not syntactically valid JSON object, exit with a return code of 6.
7. If a response is available to the tool, but the HTTP status code is not 200 nor 404, exit with a return code of 7.
8. If a response is available to the tool, but the expected *objectClassName* in the topmost object was not found for a lookup query (i.e. *domain/<domain name>*, *nameserver/<nameserver name>* and *entity/<handle>*) nor the expected JSON array (i.e. *nameservers?ip=<nameserver search pattern>*, just the JSON array should exist, not validation on the contents) for a search query, exit with an return code of 8.
9. If the HTTP status code is 404, perform Error Response Body [stdRdapErrorResponseBodyValidation].

10. If the option --*use-rdap-profile-february-2019* is not set and:
    - the RDAP query is *domain/<domain name>*, perform Domain Lookup Validation [stdRdapDomainLookupValidation].
    - the RDAP query is *nameserver/<nameserver name>*, perform Nameserver lookup validation [stdRdapNameserverLookupValidation].
    - the RDAP query is *entity/<handle>*, perform Entity lookup validation [stdRdapEntityLookupValidation] if the flag --thin is not set. If the flag --thin is set, exit with a return code of 9.
    - the RDAP query is *help*, perform Help validation [stdRdapHelpValidation].
    - the RDAP query is *nameservers?ip=<nameserver search pattern>*, perform Nameservers search validation [stdRdapNameserversSearchValidation].

    Note: the collection of tests included in the section Collection of RDAP Profile February-2019 tests shall **not** be used, if the option --*use-rdap-profile-february-2019* is not set.

11. If the option --*use-rdap-profile-february-2019* is set and:
    - the RDAP query is *domain/<domain name>*, perform Domain Lookup Validation [stdRdapDomainLookupValidation].
    - the RDAP query is *nameserver/<nameserver name>*, perform Nameserver lookup validation [stdRdapNameserverLookupValidation].
    - the RDAP query is *entity/<handle>*, perform Entity lookup validation [stdRdapEntityLookupValidation] if the flag --thin is not set. If the flag --thin is set, exit with a return code of 9.
    - the RDAP query is *help*, perform Help validation [stdRdapHelpValidation].
    - the RDAP query is *nameservers?ip=<nameserver search pattern>*, perform Nameservers search validation [stdRdapNameserversSearchValidation].

Additionally, apply the collection tests described in the table below when the option *--use-rdap-profile-february-2019* is set:

	domain/<domain name>			nameserver/<nameserver name>		entity/<handle>		help		nameservers?ip=...	
	--gtld-registry										
	--thin set	--thin not set	--gtld-registrar	--gtld-registry	--gtld-registrar	--gtld-registry	--gtld-registrar	--gtld-registry	--gtld-registrar	--gtld-registry	--gtld-registrar
Technical Implementation Guide – General	☑										
[tigRegistrySection_1_11_1_Validation]	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	
[tigRegistrySection_3_2_Validation]	☑	☑									
[tigRegistrySection_6_1_Validation]	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	
[tigRegistrarSection_1_12_1_Validation]	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	
RDAP Response Profile - General	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	
[rdapResponseProfile_2_3_1_3_and_2_7_6_and_3_3_and_4_4_Validation]	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
RDAP Response Profile - Domain	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	
RDAP Response Profile - Nameserver		☑	☑	☑							
RDAP Response Profile – Entities within Domain		☑	☑								
RDAP Response Profile – Entities within Domain - Registry		☑									
RDAP Response Profile - Entities within Domain - Registrar			☑			☑					
RDAP Response Profile – Entity - Registrar											

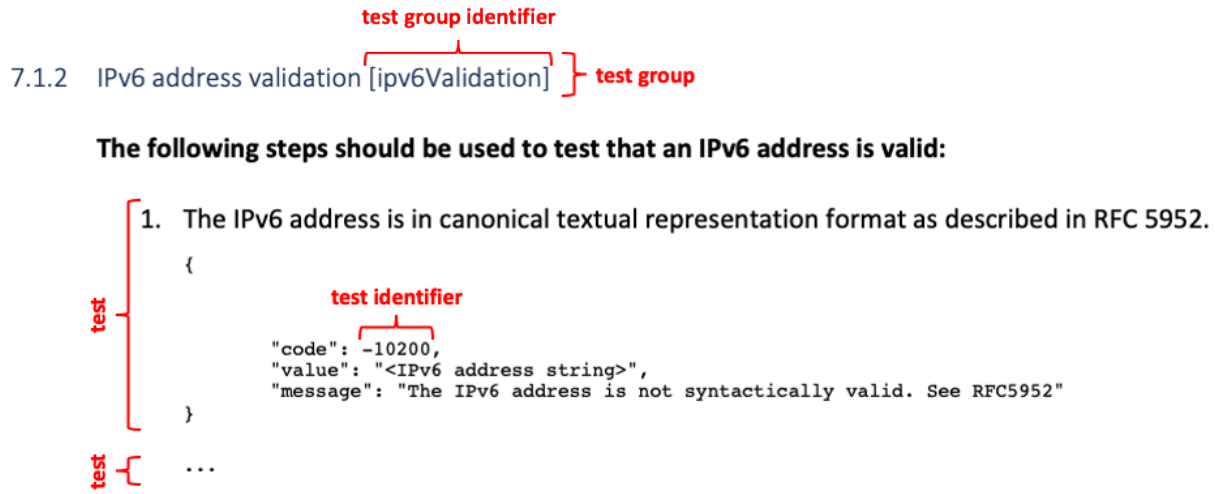
6. Understanding tests

6.1. Anatomy of a test

Tests are the atomic unit of processing of the tool. If the test results in error, the following JSON elements are added to the results file: *code*, *value*, and *message*. The error code is also used as a test identifier in the configuration definition and results file.
One or more tests are aggregated in a test group. The test group identifier is specified in this document between square brackets, and the test group identifier is used in the results file.
In this document, several test groups are aggregated in the following collections:

- Collection of Standard Tests
- Collection of RDAP Profile February-2019 tests

The collections are only used in this document for explanatory purposes, and they are not used in the configuration definition or results file.
Example of a test and test group:



6.2. Datasets

The downloaded datasets are stored in a folder called datasets.

The filename of the dataset is the identifier (e.g. ipv4AddressSpace) with the appropriate extension (i.e. xml or json).

The following datasets are used in the tests:

- **ipv4AddressSpace**, IANA IPv4 Address Space Registry, https://www.iana.org/assignments/ipv4-addressspace/ipv4-address-space.xml
- **specialIPv4Addresses**, IANA IPv4 Special-Purpose Address Registry, https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xml
- **ipv6AddressSpace**, Internet Protocol Version 6 Address Space, https://www.iana.org/assignments/ipv6-address-space/ipv6-address-space.xml
- **specialIPv6Addresses**, IANA IPv6 Special-Purpose Address Registry, https://www.iana.org/assignments/iana-ipv6-special-registry/iana-ipv6-special-registry.xml
- **RDAPExtensions**, RDAP Extensions, https://www.iana.org/assignments/rdap-extensions/rdap-extensions.xml
- **linkRelations**, Link Relations, https://www.iana.org/assignments/link-relations/link-relations.xml
- **mediaTypes**, Media Types, https://www.iana.org/assignments/media-types/media-types.xml
- **RDAPJSONValues**, RDAP JSON Values, https://www.iana.org/assignments/rdap-JSON-values/rdap-JSON-values.xml
- **dsRrTypes**, Delegation Signer (DS) Resource Record (RR) Type Digest Algorithms, https://www.iana.org/assignments/ds-rr-types/ds-rr-types.xml
- **dnsSecAlgNumbers**, Domain Name System Security (DNSSEC) Algorithm Numbers, https://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xml

- **bootstrapDomainNameSpace**, Bootstrap Service Registry for Domain Name Space, https://data.iana.org/rdap/dns.json

- **registrarId**, Registrar IDs, https://www.iana.org/assignments/registrar-ids/registrar-ids.xml

- **EPPROID**, Extensible Provisioning Protocol (EPP) Repository Identifiers, https://www.iana.org/assignments/epp-repository-ids/epp-repository-ids.xml

7. Collection of Standard Tests

7.1. General tests

7.1.1. IPv4 address validation [ipv4Validation]

The following steps should be used to test that an IPv4 address is valid:

1. The IPv4 address is syntactically valid in dot-decimal notation.

```
{
        "code": -10100,
        "value": "<IPv4 address string>",
        "message": "The IPv4 address is not syntactically valid in dot-decimal
notation."
}
```

2. The IPv4 address MUST be part of a prefix categorized as "ALLOCATED" or "LEGACY" in the "Status" field in the **ipv4AddressSpace**.

```
{
        "code": -10101,
        "value": "<IPv4 address string>",
        "message": "The IPv4 address is not included in a prefix categorized as
ALLOCATED or LEGACY in the IANA IPv4 Address Space Registry. Dataset:
ipv4AddressSpace"
}
```

3. The IPv4 address MUST NOT be part of the **specialIPv4Addresses**.

```
{
        "code": -10102,
        "value": "<IPv4 address string>",
        "message": "The IPv4 address is included in the IANA IPv4 Special-Purpose
Address Registry. Dataset: specialIPv4Addresses"
}
```

The following normative references are required to understand this test group:

- RFC791

7.1.2. IPv6 address validation [ipv6Validation]

The following steps should be used to test that an IPv6 address is valid:

1. The IPv6 address is in canonical textual representation format.

```
{
        "code": -10200,
        "value": "<IPv6 address string>",
        "message": "The IPv6 address is not syntactically valid."
}
```

2. The IPv6 address MUST be part of an allocation of type "Global Unicast" in the **ipv6AddressSpace**.

```
{
        "code": -10201,
        "value": "<IPv6 address string>",
        "message": "The IPv6 address is not included in a prefix categorized as
Global Unicast in the Internet Protocol Version 6 Address Space. Dataset:
ipv6AddressSpace"
}
```

3. The IPv6 address MUST NOT be part of the **specialIPv6Addresses**.

```
{
        "code": -10202,
        "value": "<IPv6 address string>",
        "message": "The IPv6 address is included in the IANA IPv6 Special-Purpose
Address Registry. Dataset: specialIPv6Addresses"
}
```

The following normative references are required to understand this test group:

- RFC5952

7.1.3. Domain Name validation [domainNameValidation]

The following steps should be used to test that a domain name is valid:

1. The length of each label is between 1 and 63.

```
{
        "code": -10300,
        "value": "<domain name>",
        "message": "A DNS label with length not between 1 and 63 was found."
}
```

2. A maximum total length of 253 characters not including the last ".".

```
{
        "code": -10301,
        "value": "<domain name>",
        "message": "A domain name of more than 253 characters was found. "
}
```

3. At least two labels shall exist in the domain name. See, RDAP_Technical_Implementation_Guide_2_1 section 1.10.

```
{
        "code": -10302,
        "value": "<domain name>",
        "message": "A domain name with less than two labels was found. See
RDAP_Technical_Implementation_Guide_2_1 section 1.10"
}
```

4. Each label of the domain name is a valid "A-label", "U-label", or "NR-LDH label".

```
{
        "code": -10303,
        "value": "<domain name>",
        "message": "A DNS label not being a valid "A-label", "U-label", or "NR-LDH
label" was found."
}
```

   Note: the latest version of the IANA IDNA Rules and Derived Property Values shall be used. See, https://www.iana.org/assignments/idna-tables-11.0.0/idna-tables-11.0.0.xml

The following normative references are required to understand this test group:

- DNS_RFCs
- IDNA_RFCs

7.1.4. Web URI validation [webUriValidation]

The following steps should be used to test that a *Web URI* is valid:

1. The URI shall be syntactically valid according to RFC3986.

```
{
	"code": -10400,
	"value": "<URI>",
	"message": "The URI is not syntactically valid according to RFC3986."
}
```

2. The scheme of the URI shall be "http" or "https".

```
{
	"code": -10401,
	"value": "<URI>",
	"message": "The scheme of the URI is not 'http' nor 'https'".
}
```

3. The host of the URI shall pass the test Domain Name validation [domainNameValidation], IPv4 address validation [ipv4Validation] or IPv6 address validation [ipv6Validation].

```
{
	"code": -10402,
	"value": "<URI>",
	"message": "The host does not pass Domain Name validation
[domainNameValidation], IPv4 address validation [ipv4Validation] nor IPv6 address
validation [ipv6Validation]".
}
```

The following normative references are required to understand this test group:

- RFC3986

7.1.5. Domain label case folding validation [domainCaseFoldingValidation]

The following steps should be used to test that an RDAP server is processing label case conversion correctly for domain name lookups:

1. A subsequent RDAP lookup may be performed in the case of a domain name lookup to validate correct support for case insensitive label matching:
   1.1. For any "NR-LDH label" or "A-label" present, the RDAP response must match the response of a subsequent request converting any "NR-LDH label" or "A-label" alternating uppercase and lowercase characters (e.g., if the domain is "test.example" the RDAP response must match also for converted domain name "tEsT.ExAmPlE").
   1.2. For any "U-Label" present, in case that any of the code points support case folding, the u-label should be case-folded for the subsequent request. (e.g., if the domain is "CAFÉ.EXAMPLE" the RDAP response must match also for converted domain name "café.ExAmPlE").
      - In case that the domain name in the query contains all u-labels and none of the labels can be case-folded (i.e., the script or code points do not support case folding) a subsequent query is not required.
      - In case that the domain name in the query contains all u-labels and the resulting domain name to query after case-folding is the same as the original, a subsequent query is not required.

```
{
       "code": -10403,
       "value": "<converted domain name>",
       "message": "RDAP responses do not match when handling domain label case
folding."
}
```

The following normative references are required to understand this test group:

- DNS_RFCs
- IDNA_RFCs

7.2. Standard RDAP Common Data Structures Validations

7.2.1. RDAP Conformance validation [stdRdapConformanceValidation]

The following steps should be used to test that an *RDAP Conformance* data structure is valid:

1. The *RDAP Conformance* data structure must be a syntactically valid JSON array.

```
{
        "code": -10500,
        "value": "<rdapConformance structure>",
        "message": "The RDAP Conformance structure is not syntactically valid."
}
```

2. Every value of the JSON array shall be a JSON string data type.

```
{
        "code": -10501,
        "value": "<JSON value>",
        "message": "The JSON value is not a string."
}
```

3. Each of the JSON string values in the JSON array, with the exception of "rdap_level_0", shall be included as an Extension Identifier in **RDAPExtensions**.

```
{
        "code": -10502,
        "value": "<JSON string>",
        "message": "The JSON string is not included as an Extension Identifier in
RDAPExtensions."
}
```

4. The JSON string value "rdap_level_0" is not included in the *RDAP Conformance* data structure.

```
{
        "code": -10503,
        "value": "<rdapConformance>",
        "message": "The RDAP Conformance data structure does not include
rdap_level_0."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.2.2. Links validation [stdRdapLinksValidation]

The following steps should be used to test that a links data structure is valid:

1. The *links* data structure must be a syntactically valid JSON array.

```
{
        "code": -10600,
        "value": "<links structure>",
        "message": "The links structure is not syntactically valid."
}
```

2. For every object (i.e. link) of the JSON array, verify that the *link* structure complies with:

   2.1. The name of every name/value pair shall be *value*, *rel*, *href*, *hreflang*, *title*, *media* or *type*.

```
{
        "code": -10601,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: value, rel, href,
hreflang, title, media or type."
}
```

   2.2. The JSON name/value pairs of *rel*, *href*, *hreflang*, *title*, *media* and *type* shall appear only once.

```
{
        "code": -10602,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a link structure was found
more than once."
}
```

   2.3. If the JSON name *media* exists, the allowed values are: screen, tty, tv, projection, handheld, print, braille, embossed, speech, and all.

```
{
        "code": -10603,
        "value": "<name/value pair>",
        "message": "The value for the JSON name media is not of: screen, tty, tv,
projection, handheld, print, braille, embossed, speech, or all."
}
```

   2.4. If the JSON name *rel* exists, the value shall be included as a "Relation Name" in **linkRelations**.

```
{
        "code": -10604,
        "value": "<name/value pair>",
        "message": "The JSON value is not included as a Relation Name in
linkRelations."
}
```

   2.5. If the JSON name *type* exists, the value shall be included as a "Type Name/Subtype Name" as registered in **mediaTypes**.

```
{
        "code": -10605,
        "value": "<name/value pair>",
        "message": "The JSON value is not included as a Name in mediaTypes."
}
```

2.6. If the JSON name *title* exists, the value shall be a JSON string data type.

```
{
        "code": -10606,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

2.7. If the JSON name *hreflang* exists, the value shall be a JSON string data type or a valid JSON array where every value is a JSON string data type.

```
{
        "code": -10607,
        "value": "<name/value pair>",
        "message": "The value for the JSON name hreflang is not a JSON string data
type or a valid JSON array where every value is a JSON string data type."
}
```

2.8. If the JSON name *hreflang* exists, every one of the *JSON string* data values shall conform to the Language-Tag syntax.

```
{
        "code": -10608,
        "value": "<name/value pair>",
        "message": "The value of the JSON string data in the hreflang does not
conform to Language-Tag syntax."
}
```

2.9. If the JSON name *value* exists, the value shall pass the test Web URI validation [webUriValidation] defined in this document.

```
{
        "code": -10609,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Web URI
validation [webUriValidation]."
}
```

2.10. The JSON name *href* shall exist.

```
{
        "code": -10610,
        "value": "<links structure>",
        "message": "The href element does not exist."
}
```

2.11. For the JSON name *href*, the value shall pass the test Web URI validation [webUriValidation] defined in this document.

```
{
        "code": -10611,
        "value": "<name/value pair>",
        "message": "The value for the JSON name href does not pass Web URI
validation [webUriValidation]."
}
```

The following normative references are required to understand this test group:

- RFC5646
- RFC7159
- RFC8288
- RDAP_RFCs

- Media_Queries

7.2.3. Notices and Remarks Validation [stdRdapNoticesRemarksValidation]

The following steps should be used to test that a notices or remarks data structure is valid:

1. The *notices* or *remarks* data structure must be a syntactically valid JSON array.

```
{
        "code": -10700,
        "value": "<notices or remarks structure>",
        "message": "The notices or remarks structure is not syntactically valid."
}
```

2. For every object of the JSON array, verify that the structure complies with:

   2.1. The name of every name/value pair shall be *title*, *type*, *description* or *links*.

```
{
        "code": -10701,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: title, type,
description or links."
}
```

   2.2. The JSON name/values of *title*, *type*, *description* and *links* shall exist only once.

```
{
        "code": -10702,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a link structure was found
more than once."
}
```

   2.3. If the JSON name *title* exists, the value shall be a JSON string data type.

```
{
        "code": -10703,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

   2.4. If the JSON name *links* exists, the value shall pass the test Links validation [stdRdapLinksValidation] defined in this document.

```
{
        "code": -10704,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Links
validation [stdRdapLinksValidation]."
}
```

   2.5. If the JSON name *type* exists, the value shall be a JSON string data type.

```
{
        "code": -10705,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

   2.6. If the JSON name *type* exists, the value shall be included in the **RDAPJSONValues** with Type="notice and remark type".

```
{
        "code": -10706,
        "value": "<JSON string>",
```

```
        "message": "The JSON string is not included as a Value with Type="notice
and remark type" in the RDAPJSONValues dataset.
}
```

2.7. The JSON name *description* shall exist.

```
{
        "code": -10707,
        "value": "<notices or remarks structure>",
        "message": "The description element does not exist."
}
```

2.8. The *description* data structure must be a syntactically valid JSON array.

```
{
        "code": -10708,
        "value": "<description structure>",
        "message": "The description structure is not syntactically valid."
}
```

2.9. Every value of the JSON array of the *description* data structure shall be a JSON string data type.

```
{
        "code": -10709,
        "value": "<JSON value>",
        "message": "The JSON value is not a string."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.2.4. Language Identifier Validation [stdRdapLanguageIdentifierValidation]

The following steps should be used to test that a lang data structure is valid:

1. For the JSON name *lang*, the value shall conform to the Language-Tag syntax.

```
{
        "code": -10800,
        "value": "<name/value pair>",
        "message": "The value of the JSON string data in lang does not conform to
Language-Tag syntax."
}
```

The following normative references are required to understand this test group:

- RFC5646
- RFC7159
- RDAP_RFCs

7.2.5. Events Validation [stdRdapEventsValidation]

The following steps should be used to test that a events data structure is valid:

1. The *events* data structure must be a syntactically valid JSON array.

```
{
        "code": -10900,
        "value": "<events structure>",
        "message": "The events structure is not syntactically valid."
}
```

2. For every object of the JSON array, verify that the structure complies with:

   2.1. The name of every name/value pair shall be any of: *eventAction*, *eventActor*, *eventDate* or *links*.

```
{
        "code": -10901,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: eventAction,
eventActor, eventDate or links."
}
```

   2.2. The JSON name/value pairs of *eventAction*, *eventActor*, *eventDate* and *links* shall exist only once.

```
{
        "code": -10902,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a link structure was found
more than once. "
}
```

   2.3. The JSON name *eventAction* shall exist.

```
{
        "code": -10903,
        "value": "<links structure>",
        "message": "The eventAction element does not exist."
}
```

   2.4. For the JSON name *eventAction*, the value shall be a JSON string data type.

```
{
        "code": -10904,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

   2.5. For the JSON name *eventAction*, the value shall be included in the **RDAPJSONValues** with Type="event action".

```
{
        "code": -10905,
        "value": "<JSON string>",
        "message": "The JSON string is not included as a Value with Type="event
action" in the RDAPJSONValues data set.
}
```

   2.6. The JSON name *eventDate* shall exist.

```
{
        "code": -10906,
        "value": "<links structure>",
        "message": "The eventDate element does not exist."
```

```
}
```

2.7. For the JSON name *eventDate*, the value shall be a JSON string data type.

```
{
        "code": -10907,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

2.8. For the JSON name *eventDate*, the value shall be syntactically valid time and date according to RFC3339.

```
{
        "code": -10908,
        "value": "<name/value pair>",
        "message": "The JSON value shall be a syntactically valid time and date
according to RFC3339."
}
```

2.9. If the JSON name *eventActor* exists, the value shall be a JSON string data type.

```
{
        "code": -10909,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

2.10. If the JSON name *links* exists, the JSON name *eventActor* shall also exist.

```
{
        "code": -10910,
        "value": "<events structure>",
        "message": "A links structure was found but an eventActor was not."
}
```

2.11. If the JSON name *links* exists, the value shall pass the test Links validation [stdRdapLinksValidation] defined in this document.

```
{
        "code": -10911,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Links
validation [stdRdapLinksValidation]."
}
```

3. An *eventAction* value shall only appears once within the events array.

```
{
        "code": -10912,
        "value": "<events structure>",
        "message": "An eventAction value exists more than once within the events
array."
}
```

The following normative references are required to understand this test group:

- RFC3339
- RFC7159
- RDAP_RFCs

7.2.6. Status validation [stdRdapStatusValidation]

The following steps should be used to test that a status data structure is valid:

1. The *status* data structure must be a syntactically valid JSON array.

```
{
        "code": -11000,
        "value": "<status structure>",
        "message": "The status structure is not syntactically valid."
}
```

2. Every value of the JSON array shall be a JSON string data type.

```
{
        "code": -11001,
        "value": "<JSON value>",
        "message": "The JSON value is not a string."
}
```

3. Each of the JSON string values in the JSON array shall be included in the **RDAPJSONValues** with Type="status".

```
{
        "code": -11002,
        "value": "<JSON string>",
        "message": "The JSON string is not included as a Value with
Type="status"."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.2.7. Port 43 WHOIS Server [stdRdapPort43WhoisServerValidation]

The following steps should be used to test that a port43 data structure is valid:

1. For the JSON name *port43*, the value shall pass the test [IPv4Validation], [IPv6Validation] or [DomainNameValidation] defined in this document.

```
{
        "code": -11100,
        "value": "<name/value pair>",
        "message": "The value for the JSON name port43 does not pass
[IPv4Validation], [IPv6Validation] or [DomainNameValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.2.8. Public IDs validation [stdRdapPublicIdsValidation]

The following steps should be used to test that a publicIds data structure is valid:

1. The *publicIds* data structure must be a syntactically valid JSON array.

```
{
        "code": -11200,
        "value": "<publicIds structure>",
        "message": "The publicIds structure is not syntactically valid."
}
```

2. For every object of the JSON array, verify that the structure complies with:

   2.1. The name of every name/value pairs shall be *type* or *identifier*.

```
{
        "code": -11201,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: type or
        identifier."
}
```

   2.2. The JSON name/values of *type* or *identifier* shall appear only once.

```
{
        "code": -11202,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a domain structure was
found more than once."
}
```

   2.3. The JSON name/values of *type* and *identifier* shall appear.

```
{
        "code": -11203,
        "value": "<publicIds structure>",
        "message": "The following name/values shall exist: type or identifier."
}
```

   2.4. For the JSON name *type*, the value shall be a JSON string data type.

```
{
        "code": -11204,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

   2.5. For the JSON name *identifier*, the value shall be a JSON string data type.

```
{
        "code": -11205,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.2.9. asEventActor Validation [stdRdapAsEventActorValidation]

The following steps should be used to test that an asEventActor data structure is valid:

1. The *asEventActor* data structure must be a syntactically valid JSON array.

```
{
	"code": -11300,
	"value": "<asEventActor structure>",
	"message": "The asEventActor structure is not syntactically valid."
}
```

2. The *asEventActor* data structure must be embedded within an entity object, and the entity object must be embedded within another object.

```
{
	"code": -11301,
	"value": "<asEventActor structure>",
	"message": "The asEventActor structure is not embedded within an entity
object and the entity object is not embedded within another object."
}
```

3. For every object of the JSON array, verify that the structure complies with:

   3.1. The name of every name/value pair shall be any of: *eventAction* or *eventDate*.

```
{
	"code": -11302,
	"value": "<name/value pair>",
	"message": "The name in the name/value pair is not of: eventAction and
eventDate."
}
```

   3.2. The JSON name/values of *eventAction* or *eventDate* shall appear only once.

```
{
	"code": -11303,
	"value": "<name/value pair>",
	"message": "The name in the name/value pair of a link structure was found
more than once."
}
```

   3.3. The JSON name *eventAction* shall exist.

```
{
	"code": -11304,
	"value": "<links structure>",
	"message": "The eventAction element does not exist."
}
```

   3.4. For the JSON name *eventAction*, the value shall be a JSON string data type.

```
{
	"code": -11305,
	"value": "<name/value pair>",
	"message": "The JSON value is not a string."
}
```

3.5. For the JSON name *eventAction*, the value shall be included in the **RDAPJSONValues** with Type="event action".

```
{
        "code": -11306,
        "value": "<JSON string>",
        "message": "The JSON string is not included as a Value with Type="event
action" in the RDAPJSONValues dataset."
}
```

3.6. The JSON name *eventDate* shall exist.

```
{
        "code": -11307,
        "value": "<links structure>",
        "message": "The eventDate element does not exist."
}
```

3.7. For the JSON name *eventDate*, the value shall be a JSON string data type.

```
{
        "code": -11308,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

3.8. For the JSON name *eventDate*, the value shall be syntactically valid time and date according to RFC3339.

```
{
        "code": -11309,
        "value": "<name/value pair>",
        "message": "The JSON value shall be a syntactically valid time and date
according to RFC3339."
}
```

4. An *eventAction* shall only appear once within the events array.

```
{
        "code": -11310,
        "value": "<events structure>",
        "message": "An eventAction exists more than once within the events array."
}
```

The following normative references are required to understand this test group:

- RFC3339
- RFC7159
- RDAP_RFCs

7.2.10. IP Addresses Validation [stdRdapIpAddressesValidation]

The following steps should be used to test that an ipAddresses data structure is valid:

1. The *ipAddresses* data structure must be a syntactically valid JSON object.

```
{
        "code": -11400,
        "value": "<ipAddresses structure>",
        "message": "The ipAddresses structure is not syntactically valid."
}
```

2. The name of every name/value pair shall be any of: *v4* or *v6*.

```
{
        "code": -11401,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: v4 or v6."
}
```

3. The JSON name/values of *v4* and *v6* shall appear only once.

```
{
        "code": -11402,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of an ipAddresses structure
was found more than once."
}
```

4. One *v4* name/values or one *v6* name/value shall appear.

```
{
        "code": -11403,
        "value": "<name/value pair>",
        "message": "v4 nor v6 name/value pair exists."
}
```

5. If the JSON name *v4* exists, the value shall pass the following:

   5.1. The *v4* data structure must be a syntactically valid JSON array.

```
{
        "code": -11404,
        "value": "<v4 structure>",
        "message": "The v4 structure is not syntactically valid."
}
```

   5.2. For every object of the JSON array, verify that the structure complies with:

      5.2.1. Every value of the JSON array shall be a JSON string data type.

```
{
        "code": -11405,
        "value": "<JSON value>",
        "message": "The JSON value is not a string."
}
```

5.2.2. The IPv4 address is syntactically valid in dot-decimal notation.

```
{
        "code": -11406,
        "value": "<IPv4 address string>",
        "message": "The IPv4 address is not syntactically valid in dot-decimal
        notation."
}
```

6. If the JSON name *v6* exists, the value shall pass the following:

6.1. The *v6* data structure must be a syntactically valid JSON array.

```
{
       "code": -11407,
       "value": "<v6 structure>",
       "message": "The v6 structure is not syntactically valid."
}
```

6.2. For every object of the JSON array, verify that the structure complies with:

6.2.1. Every value of the JSON array shall be a JSON string data type.

```
{
        "code": -11408,
        "value": "<JSON value>",
        "message": "The JSON value is not a string."
}
```

6.2.2. The IPv6 address is syntactically valid.

```
{
        "code": -11409,
        "value": "<IPv6 address string>",
        "message": "The IPv6 address is not syntactically valid."
}
```

The following normative references are required to understand this test group:

- RFC791
- RFC5952
- RFC7159
- RDAP_RFCs

7.2.11. Variants validation [stdRdapVariantsValidation]

The following steps should be used to test that a variants data structure is valid:

1. The *variants* data structure must be a syntactically valid JSON array.

```
{
        "code": -11500,
        "value": "<variants structure>",
        "message": "The variants structure is not syntactically valid."
}
```

2. For every object of the JSON array, verify that the structure complies with:

   2.1. The name of every name/value pair shall be *relation*, *idnTable* or *variantNames*.

```
{
        "code": -11501,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: relation, idnTable
or variantNames."
}
```

   2.2. The JSON name/value pairs of *relation*, *idnTable* and *variantNames* shall appear only once.

```
{
        "code": -11502,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a link structure was found
more than once."
}
```

   2.3. The RDAP *relation* data structure must be a syntactically valid *JSON array*.

```
{
        "code": -11503,
        "value": "<relation structure>",
        "message": "The RDAP Conformance structure is not syntactically valid."
}
```

   2.4. For every object of the JSON array, verify that the structure complies with:

      2.4.1. Every value of the JSON array shall be a JSON string data type.

```
{
        "code": -11504,
        "value": "<JSON value>",
        "message": "The JSON value is not a string."
}
```

      2.4.2. Each of the JSON string values in the JSON array shall be included in the **RDAPJSONValues** with Type="domain variant relation".

```
{
        "code": -11505,
        "value": "<JSON string>",
        "message": "The JSON string is not included as a Value with
        Type="domain variant relation "."
}
```

   2.5. If the JSON name *idnTable* exists, the value shall be a JSON string data type.

```
{
```

```
        "code": -11506,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

2.6. The *variantNames* data structure must be a syntactically valid JSON array.

```
{
        "code": -11507,
        "value": "<variantNames structure>",
        "message": "The variantNames structure is not syntactically valid."
}
```

2.7. For every object of the JSON array, verify that the structure complies with:

2.7.1. The name of every name/value pair shall be any of: *ldhName* or *unicodeName*

```
{
        "code": -11508,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: ldhName or
        unicodeName."
}
```

2.7.2. The JSON name/value pairs of *ldhName* or *unicodeName* shall exist only once.

```
{
        "code": -11509,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a link structure was
        found more than once."
}
```

2.7.3. If the JSON name title *ldhName* exists, the value shall pass the test LDH name [stdRdapLdhNameValidation] defined in this document.

```
{
        "code": -11510,
        "value": "<name/value pair>",
        "message": " The value for the JSON name value does not pass LDH name
        [stdRdapLdhNameValidation]."
}
```

2.7.4. If the JSON name *unicodeName* exists, the value shall pass the test Unicode name [stdRdapUnicodeNameValidation] defined in this document.

```
{
        "code": -11511,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Unicode
        name [stdRdapUnicodeNameValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs
- IDNA_RFCs

7.2.12. Unicode name [stdRdapUnicodeNameValidation]

The following steps should be used to test that a unicodeName is valid:

1. The length of each label is between 1 and 63.

```
{
        "code": -11600,
        "value": "<domain name>",
        "message": "A DNS label with length not between 1 and 63 was found."
}
```

2. A maximum total length of 253 characters not including the last ".".

```
{
        "code": -11601,
        "value": "<domain name>",
        "message": "A domain name of more than 253 characters was found."
}
```

3. At least two labels shall exist in the domain name. See, RDAP_Technical_Implementation_Guide_2_1 section 1.10.

```
{
        "code": -11602,
        "value": "<domain name>",
        "message": "A domain name with less than two labels was found. See
RDAP_Technical_Implementation_Guide_2_1 section 1.10"

}
```

4. Each label of the domain name is a valid "U-label or "NR-LDH label".

```
{
        "code": -11603,
        "value": "<domain name>",
        "message": "A label not being a valid "U-label" or "NR-LDH label" was
found."
}
```

   Note: the latest version of the IANA IDNA Rules and Derived Property Values shall be used. See, https://www.iana.org/assignments/idna-tables-11.0.0/idna-tables-11.0.0.xml

   Note: some legacy gTLDs may fail this test, because they have a few domain name registrations that comply with IDNA2003 but not IDNA2018. Such names are not recommended to be used when testing an RDAP response with this tool.

The following normative references are required to understand this test group:

- DNS_RFCs
- IDNA_RFCs

7.2.13. LDH name [stdRdapLdhNameValidation]

The following steps should be used to test that a ldhName is valid:

1. The length of each label is between 1 and 63.

```
{
        "code": -11700,
        "value": "<domain name>",
        "message": "A DNS label with length not between 1 and 63 was found."
}
```

2. A maximum total length of 253 characters not including the last ".".

```
{
        "code": -11701,
        "value": "<domain name>",
        "message": "A domain name of more than 253 characters was found."
}
```

3. At least two labels shall exist in the domain name. See, RDAP_Technical_Implementation_Guide_2_1 section 1.10.

```
{
        "code": -11702,
        "value": "<domain name>",
        "message": "A domain name with less than two labels was found. See
RDAP_Technical_Implementation_Guide_2_1 section 1.10"
}
```

4. Each label of the domain name is a valid "A-label or "NR-LDH label".

```
{
        "code": -11703,
        "value": "<domain name>",
        "message": "A label not being a valid "A-label" or "NR-LDH label" was
found."
}
```

Note: the latest version of the IANA IDNA Rules and Derived Property Values shall be used. See, https://www.iana.org/assignments/idna-tables-11.0.0/idna-tables-11.0.0.xml

Note: some legacy gTLDs may fail this test, because they have a few domain name registrations that comply with IDNA2003 but not IDNA2018. Such names are not recommended to be used when testing an RDAP response with this tool.

The following normative references are required to understand this test group:

- DNS_RFCs
- IDNA_RFCs

7.2.14. Roles validation [stdRdapRolesValidation]

The following steps should be used to test that a roles data structure is valid:

1. The *roles* data structure must be a syntactically valid JSON array.
```
{
	"code": -11800,
	"value": "<roles structure>",
	"message": "The roles structure is not syntactically valid."
}
```

2. Every value of the JSON array shall be a JSON string data type.
```
{
	"code": -11801,
	"value": "<JSON value>",
	"message": "The JSON value is not a string."
}
```

3. Each of the JSON string values in the JSON array shall be included in the **RDAPJSONValues** with Type="role".
```
{
	"code": -11802,
	"value": "<JSON string>",
	"message": "The JSON string is not included as a Value with Type="role"."
}
```

4. The role value shall only appear once in the JSON array.
```
{
	"code": -11803,
	"value": "<roles structure>",
	"message": "A role value appeared more than once."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.2.15. Entities validation [stdRdapEntitiesValidation]

The following steps should be used to test that an entities data structure is valid:

1. The *entities* data structure must be a syntactically valid JSON array.

```
{
        "code": -11900,
        "value": "<entities structure>",
        "message": "The entities structure is not syntactically valid."
}
```

2. Every value of the JSON array shall pass the test Entity lookup validation [stdRdapEntityLookupValidation] defined in this document.

```
{
        "code": -11901,
        "value": "<JSON value>",
        "message": "The JSON value does not pass Entity lookup validation
[stdRdapEntityLookupValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.2.16. Secure DNS validation [stdRdapSecureDnsValidation]

The following steps should be used to test that a secureDNS data structure is valid:

1. The *secureDNS* data structure must be a syntactically valid JSON object.

```
{
        "code": -12000,
        "value": "<secureDNS structure>",
        "message": "The domain structure is not syntactically valid."
}
```

2. The name of every name/value pairs shall be *zoneSigned*, *delegationSigned*, *maxSigLife*, *dsData* or *keyData*.

```
{
        "code": -12001,
        "value": "<name/value pair>",
"message": "The name in the name/value pair is not of: zoneSigned,
delegationSigned, maxSigLife, dsData or keyData."
}
```

3. The JSON name/values of *zoneSigned*, *delegationSigned*, *maxSigLife*, *dsData* and *keyData* shall appear only once.

```
{
        "code": -12002,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a domain structure was
found more than once."
}
```

4. If the JSON name *zoneSigned* appears, the value shall be a JSON boolean data type.

```
{
        "code": -12003,
        "value": "<name/value pair>",
        "message": "The JSON value is not a boolean."
}
```

5. If the JSON name *delegationSigned* appears, the value shall be a JSON boolean data type.

```
{
        "code": -12005,
        "value": "<name/value pair>",
        "message": "The JSON value is not a boolean."
}
```

6. If the JSON name *maxSigLife* exists, the value shall be a JSON number data type between 1 and 2147483647.

```
{
        "code": -12006,
        "value": "<name/value pair>",
        "message": "The JSON value is not a number between 1 and 2147483647."
}
```

-12007

7. If the JSON name *dsData* appears, the value shall pass the following:

7.1. The *dsData* data structure must be a syntactically valid array of JSON objects.

```
{
        "code": -12008,
        "value": "<dsData structure>",
        "message": "The dsData structure is not syntactically valid."
}
```

7.2. The name of every name/value pair shall be any of: *keyTag*, *algorithm*, *digest*, *digestType*, *events* or *links*.

```
{
        "code": -12009,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: keyTag, algorithm,
digest, digestType, events or links."
}
```

7.3. The JSON name/values of *keyTag*, *algorithm*, *digest*, *digestType*, *events* or *links* shall appear only once.

```
{
        "code": -12010,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a dsData structure was
found more than once."
}
```

7.4. The JSON name/values of *keyTag*, *algorithm*, *digest* and *digestType* shall appear.

```
{
        "code": -12011,
        "value": "<dsData structure>",
        "message": "The following name/values shall exist: keyTag, algorithm,
digest and digestType."
}
```

7.5. For the JSON name *keyTag*, the value shall be a JSON number data type between 1 and 65535.

```
{
        "code": -12012,
        "value": "<name/value pair>",
        "message": "The JSON value is not a number between 1 and 65535."
}
```

7.6. For the JSON name *algorithm*, the value shall be a JSON number listed with Zone Signing=Y in **dnsSecAlgNumbers**. The values 253 and 254 are not valid for this test.

```
{
        "code": -12013,
        "value": "<name/value pair>",
        "message": "The JSON value is not listed with Zone Signing=Y in
dnsSecAlgNumbers, or it's 253 or 254."
}
```

7.7. For the JSON name *digest*, the value shall be a JSON string of case-insensitive hexadecimal digits. Whitespace is allowed within the hexadecimal text.

```
{
        "code": -12014,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string of case-insensitive hexadecimal
digits. Whitespace is allowed within the hexadecimal test."
}
```

7.8. For the JSON name *digestType*, the value shall be a JSON number assigned in **dsRrTypes**.

```
{
        "code": -12015,
        "value": "<name/value pair>",
        "message": "The JSON value is not assigned in dsRrTypes."
}
```

7.9. If the JSON name *events* exists, the value shall pass the test Events Validation [stdRdapEventsValidation] defined in this document.

```
{
        "code": -12016,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Events
Validation [stdRdapEventsValidation]."
}
```

7.10. If the JSON name *links* exists, the value shall pass the test Links validation [stdRdapLinksValidation] defined in this document.

```
{
        "code": -12017,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Links
validation [stdRdapLinksValidation]."
}
```

8. If the JSON name *keyData* exists, the value shall pass the following:

8.1. The *keyData* data structure must be a syntactically valid array of JSON objects.

```
{
        "code": -12018,
        "value": "<keyData structure>",
        "message": "The keyData structure is not syntactically valid."
}
```

8.2. The name of every name/value pair shall be *flags*, *protocol*, *publicKey*, *algorithm*, *events* or *links*.

```
{
        "code": -12019,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair is not of: flags, protocol,
publicKey, algorithm, events or links."
}
```

8.3. The JSON name/values of *flags*, *protocol*, *publicKey*, *algorithm*, *events* or *links* shall appear only once.

```
{
        "code": -12020,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a keyData structure was
found more than once."
}
```

8.4. The JSON name/values of *flags*, *protocol*, *publicKey* and *algorithm* shall appear.

```
{
        "code": -12021,
        "value": "<dsData structure>",
        "message": "The following name/values shall exist: flags, protocol,
publicKey and algorithm."
}
```

8.5. For the JSON name *flags*, the value shall be a JSON number data type with values: 256 or 257.

```
{
        "code": -12022,
        "value": "<name/value pair>",
        "message": "The JSON value is not 256 or 257."
}
```

8.6. For the JSON name *protocol*, the value shall be a JSON number data type with value: 3.

```
{
        "code": -12023,
        "value": "<name/value pair>",
        "message": "The JSON value is not 3."
}
```

8.7. For the JSON name *publicKey*, the value shall be a JSON string, and the key is represented as a Base64. Whitespace is allowed within the text.

```
{
        "code": -12024,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string of case-insensitive hexadecimal
digits. Whitespace is allowed within the hexadecimal text."
}
```

8.8. For the JSON name *algorithm*, the value shall be a JSON number listed with Zone Signing=Y in **dnsSecAlgNumbers**. The values 253 and 254 are not valid for this test.

```
{
        "code": -12025,
        "value": "<name/value pair>",
        "message": "The JSON value is not listed with Zone Signing=Y in
dnsSecAlgNumbers, or it's 253 or 254."
}
```

8.9. If the JSON name *events* exists, the value shall pass the test Events Validation [stdRdapEventsValidation] defined in this document.

```
{
        "code": -12026,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Events
Validation [stdRdapEventsValidation]."
}
```

8.10. If the JSON name *links* exists, the value shall pass the test Links validation [stdRdapLinksValidation] defined in this document.

```
{
        "code": -12027,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Links
validation [stdRdapLinksValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs
- DNSSEC_RFCs

7.2.17. Error Response Body [stdRdapErrorResponseBodyValidation]

The following steps should be used to test that an error data structure is valid:

1. The *error* data structure must be a syntactically valid JSON object.
```
{
        "code": -12100,
        "value": "<error structure>",
        "message": "The error structure is not syntactically valid."
}
```

2. At least the following name/value pairs shall appear: *errorCode*, *title* and *description*.
```
{
        "code": -12101,
        "value": "<name/value pair>",
        "message": "At least the following name/value pairs shall exist:
errorCode, title and description."
}
```

3. The JSON name/values of *errorCode*, *title*, and *description* shall appear only once.
```
{
        "code": -12102,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of an error structure was
found more than once."
}
```

4. For the JSON name *errorCode*, the value shall be a JSON number data type.
```
{
        "code": -12103,
        "value": "<name/value pair>",
        "message": "The JSON value is not a number."
}
```

5. For the JSON name *title*, the value shall be a JSON string data type.
```
{
        "code": -12104,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

6. The *description* data structure must be a syntactically valid JSON array.
```
{
        "code": -12105,
        "value": "<description structure>",
        "message": "The description structure is not syntactically valid."
}
```

7. Every value of the JSON array of the *description* data structure shall be a JSON string data type.
```
{
        "code": -12106,
        "value": "<JSON value>",
        "message": "The JSON value is not a string."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.3. Standard RDAP Object Classes Validations

7.3.1. Domain Lookup Validation [stdRdapDomainLookupValidation]

The following steps should be used to test that a domain data structure is valid:

1. The *domain* data structure must be a syntactically valid JSON object.

```
{
        "code": -12200,
        "value": "<domain structure>",
        "message": "The domain structure is not syntactically valid."
}
```

2. The name of every name/value pairs shall be any of: *objectClassName*, *handle*, *ldhName*, *unicodeName*, *variants*, *nameservers*, *secureDNS*, *entities*, *status*, *publicIds*, *remarks*, *links*, *port43*, *events*, *notices* or *rdapConformance*.

```
{
        "code": -12201,
        "value": "<name/value pair>",
"message": "The name in the name/value pair is not of: objectClassName, handle,
ldhName, unicodeName, variants, nameservers, secureDNS, entities, status,
publicIds, remarks, links, port43, events, notices or rdapConformance."
}
```

3. The JSON name/values of *objectClassName*, *handle*, *ldhName*, *unicodeName*, *variants*, *nameservers*, *secureDNS*, *entities*, *status*, *publicIds*, *remarks*, *links*, *port43*, *events*, *notices* or *rdapConformance* shall appear only once.

```
{
        "code": -12202,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a domain structure was
found more than once."
}
```

4. For the JSON name *objectClassName*, the value shall be "domain".

```
{
        "code": -12203,
        "value": "<name/value pair>",
        "message": "The JSON value is not "domain"."
}
```

5. If the JSON name *handle* exists, the value shall be a JSON string data type.

```
{
        "code": -12204,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

6. If the JSON name *ldhName*, the value shall pass the test LDH name [stdRdapLdhNameValidation] defined in this document.

```
{
        "code": -12205,
        "value": "<name/value pair>",
        "message": " The value for the JSON name value does not pass LDH name
[stdRdapLdhNameValidation]."
}
```

7. If the JSON name *unicodeName* exists, the value shall pass the test Unicode name [stdRdapUnicodeNameValidation] defined in this document.

```
{
        "code": -12206,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Unicode name
[stdRdapUnicodeNameValidation]."
}
```

8. If the JSON name *variants* exists, the value shall pass the test Variants validation [stdRdapVariantsValidation] defined in this document.

```
{
        "code": -12207,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Variants
validation [stdRdapVariantsValidation]."
}
```

9. If the JSON name *nameservers* exists, the value shall pass the test Nameserver lookup validation [stdRdapNameserverLookupValidation] defined in this document.

```
{
        "code": -12208,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Nameserver
lookup validation [stdRdapNameserverLookupValidation]."
}
```

10. If the JSON name *secureDNS* exists, the value shall pass the test Secure DNS validation [stdRdapSecureDnsValidation] defined in this document.

```
{
        "code": -12209,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Secure DNS
validation [stdRdapSecureDnsValidation]."
}
```

11. If the JSON name *entities* exists, the value shall pass the test Entities validation [stdRdapEntitiesValidation] defined in this document.

```
{
        "code": -12210,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Entities
validation [stdRdapEntitiesValidation]."
}
```

12. If the JSON name *status* exists, the value shall pass the test Status validation [stdRdapStatusValidation] defined in this document.

```
{
        "code": -12211,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Status
validation [stdRdapStatusValidation]."
}
```

13. If the JSON name *publicIds* exists, the value shall pass the test Public IDs validation [stdRdapPublicIdsValidation] defined in this document.

```
{
        "code": -12212,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Public IDs
validation [stdRdapPublicIdsValidation]."
}
```

14. If the JSON name *remarks* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.

```
{
        "code": -12213,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

15. If the JSON name *links* exists, the value shall pass the test Links validation [stdRdapLinksValidation] defined in this document.

```
{
        "code": -12214,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Links
validation [stdRdapLinksValidation]."
}
```

16. If the JSON name *port43* exists, the value shall pass the test Port 43 WHOIS Server [stdRdapPort43WhoisServerValidation] defined in this document.

```
{
        "code": -12215,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Port 43 WHOIS
Server [stdRdapPort43WhoisServerValidation]."
}
```

17. If the JSON name *events* exists, the value shall pass the test Events Validation [stdRdapEventsValidation] defined in this document.

```
{
        "code": -12216,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Events
Validation [stdRdapEventsValidation]."
}
```

18. If the JSON name *notices* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.

```
{
        "code": -12217,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

19. If the JSON name *notices* exists and the domain object is not the topmost JSON object.

```
{
        "code": -12218,
        "value": "<name/value pair>",
        "message": "The value for the JSON name notices exists but domain object
is not the topmost JSON object."
}
```

20. If the JSON name *rdapConformance* exists, the value shall pass the test RDAP Conformance validation [stdRdapConformanceValidation] defined in this document.

```
{
        "code": -12219,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass RDAP
Conformance validation [stdRdapConformanceValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.3.2. Entity lookup validation [stdRdapEntityLookupValidation]

The following steps should be used to test that an entity data structure is valid:

1. The *entity* data structure must be a syntactically valid JSON object.

```
{
        "code": -12300,
        "value": "<entity structure>",
        "message": "The entity structure is not syntactically valid."
}
```

2. The name of every name/value pairs shall be any of: *objectClassName*, *handle*, *vcardArray*, *roles*, *publicIds*, *entities*, *remarks*, *links*, *events*, *asEventActor*, *status*, *port43*, *notices* or *rdapConformance*.

```
{
        "code": -12301,
        "value": "<name/value pair>",
"message": "The name in the name/value pair is not of: objectClassName, handle,
vcardArray, roles, publicIds, entities, remarks, links, events, asEventActor,
status, port43, notices or rdapConformance."
}
```

3. The JSON name/values of *objectClassName*, *handle*, *vcardArray*, *roles*, *publicIds*, *entities*, *remarks*, *links*, *events*, *asEventActor*, *status*, *port43*, *notices* or *rdapConformance* shall exist only once.

```
{
        "code": -12302,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a domain structure was
found more than once."
}
```

4. For the JSON name *objectClassName*, the value shall be "entity".

```
{
        "code": -12303,
        "value": "<name/value pair>",
        "message": "The JSON value is not "entity"."
}
```

5. If the JSON name *handle* exists, the value shall be a JSON string data type.

```
{
        "code": -12304,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

6. If the JSON name title *vcardArray* exists, the value shall be syntactically valid.

```
{
        "code": -12305,
        "value": "<name/value pair>",
        "message": " The value for the JSON name value is not a syntactically
valid vcardArray."
}
```

Note: it's expected that a library for testing jCard objects will be used.

7. If the JSON name *roles* exists, the value shall pass the test Roles validation [stdRdapRolesValidation] defined in this document.

```
{
        "code": -12306,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Roles
validation [stdRdapRolesValidation]."
}
```

8. If the JSON name *publicIds* exists, the value shall pass the test Public IDs validation [stdRdapPublicIdsValidation] defined in this document.

```
{
        "code": -12307,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Public IDs
validation [stdRdapPublicIdsValidation]."
}
```

9. If the JSON name *entities* exists, the value shall pass the test Entities validation [stdRdapEntitiesValidation] defined in this document.

```
{
        "code": -12308,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Entities
validation [stdRdapEntitiesValidation]."
}
```

10. If the JSON name *remarks* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.

```
{
        "code": -12309,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

11. If the JSON name *links* exists, the value shall pass the test Links validation [stdRdapLinksValidation] defined in this document.

```
{
        "code": -12310,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Links
validation [stdRdapLinksValidation]."
}
```

12. If the JSON name *events* exists, the value shall pass the test Events Validation [stdRdapEventsValidation] defined in this document.

```
{
        "code": -12311,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Events
Validation [stdRdapEventsValidation]."
}
```

13. If the JSON name *asEventActor* exists, the value shall pass the test asEventActor Validation [stdRdapAsEventActorValidation] defined in this document.

```
{
        "code": -12312,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass asEventActor
Validation [stdRdapAsEventActorValidation]."
}
```

14. If the JSON name *status* exists, the value shall pass the test Status validation [stdRdapStatusValidation] defined in this document.

```
{
        "code": -12313,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Status
validation [stdRdapStatusValidation]."
}
```

15. If the JSON name *port43* exists, the value shall pass the test Port 43 WHOIS Server [stdRdapPort43WhoisServerValidation] defined in this document.

```
{
        "code": -12314,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Port 43 WHOIS
Server [stdRdapPort43WhoisServerValidation]."
}
```

16. If the JSON name *notices* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.

```
{
        "code": -12315,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

17. If the JSON name *notices* exists and the entity object is not the topmost JSON object.

```
{
        "code": -12316,
        "value": "<name/value pair>",
        "message": "The value for the JSON name notices exists but entity object
is not the topmost JSON object."
}
```

18. If the JSON name *rdapConformance* exists, the value shall pass the test RDAP Conformance validation [stdRdapConformanceValidation] defined in this document.

```
{
        "code": -12317,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass RDAP
Conformance validation [stdRdapConformanceValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.3.3. Nameserver lookup validation [stdRdapNameserverLookupValidation]

The following steps should be used to test that a nameserver data structure is valid:

1. The *nameserver* data structure must be a syntactically valid JSON object.

```
{
        "code": -12400,
        "value": "<nameserver structure>",
        "message": "The nameserver structure is not syntactically valid."
}
```

2. The name of every name/value pairs shall be any of: *objectClassName*, *handle*, *ldhName*, *unicodeName*, *ipAddresses*, *entities*, *status*, *remarks*, *links*, *port43*, *events*, *notices* or *rdapConformance*.

```
{
        "code": -12401,
        "value": "<name/value pair>",
"message": "The name in the name/value pair is not of: objectClassName, handle,
ldhName, unicodeName, ipAddresses, entities, status, remarks, links, port43,
events, notices or rdapConformance."
}
```

3. The JSON name/values of *objectClassName*, *handle*, *ldhName*, *unicodeName*, *ipAddresses*, *entities*, *status*, *remarks*, *links*, *port43*, *events*, *notices* or *rdapConformance* shall exist only once.

```
{
        "code": -12402,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a link structure was found
more than once."
}
```

4. For the JSON name *objectClassName*, the value shall be "nameserver".

```
{
        "code": -12403,
        "value": "<name/value pair>",
        "message": "The JSON value is not "nameserver"."
}
```

5. If the JSON name *handle* exists, the value shall be a JSON string data type.

```
{
        "code": -12404,
        "value": "<name/value pair>",
        "message": "The JSON value is not a string."
}
```

6. If the JSON name *ldhName* exists, the value shall pass the test LDH name [stdRdapLdhNameValidation] defined in this document.

```
{
        "code": -12405,
        "value": "<name/value pair>",
        "message": " The value for the JSON name value does not pass LDH name
[stdRdapLdhNameValidation]."
}
```

7. If the JSON name *unicodeName* exists, the value shall pass the test Unicode name [stdRdapUnicodeNameValidation] defined in this document.

```
{
        "code": -12406,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Unicode name
[stdRdapUnicodeNameValidation]."
}
```

8. If the JSON name *ipAddresses* exists, the value shall pass the test IP Addresses Validation [stdRdapIpAddressesValidation] defined in this document.

```
{
        "code": -12407,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass IP Addresses
Validation [stdRdapIpAddressesValidation]."
}
```

9. If the JSON name *entities* exists, the value shall pass the test Entities validation [stdRdapEntitiesValidation] defined in this document.

```
{
        "code": -12408,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Entities
validation [stdRdapEntitiesValidation]."
}
```

10. If the JSON name *status* exists, the value shall pass the test Status validation [stdRdapStatusValidation] defined in this document.

```
{
        "code": -12409,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Status
validation [stdRdapStatusValidation]."
}
```

11. If the JSON name *remarks* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.

```
{
        "code": -12410,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

12. If the JSON name *links* exists, the value shall pass the test Links validation [stdRdapLinksValidation] defined in this document.

```
{
        "code": -12411,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Links
validation [stdRdapLinksValidation]."
}
```

13. If the JSON name *port43* exists, the value shall pass the test Port 43 WHOIS Server [stdRdapPort43WhoisServerValidation] defined in this document.

```
{
        "code": -12412,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Port 43 WHOIS
Server [stdRdapPort43WhoisServerValidation]."
}
```

14. If the JSON name *events* exists, the value shall pass the test Events Validation [stdRdapEventsValidation] defined in this document.

```
{
        "code": -12413,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Events
Validation [stdRdapEventsValidation]."
}
```

15. If the JSON name *notices* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.

```
{
        "code": -12414,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

16. If the JSON name *notices* exists and the nameserver object is not the topmost JSON object.

```
{
        "code": -12415,
        "value": "<name/value pair>",
        "message": "The value for the JSON name notices exists but nameserver
object is not the topmost JSON object."
}
```

17. If the JSON name *rdapConformance* exists, the value shall pass the test RDAP Conformance validation [stdRdapConformanceValidation] defined in this document.

```
{
        "code": -12416,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass RDAP
Conformance validation [stdRdapConformanceValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.3.4. Help validation [stdRdapHelpValidation]

The following steps should be used to test that a help data structure is valid:

1. The *help* data structure must be a syntactically valid JSON object.
```
{
	"code": -12500,
	"value": "<help structure>",
	"message": "The help structure is not syntactically valid."
}
```

2. The name of every name/value pairs shall be *notices* or *rdapConformance*.
```
{
	"code": -12501,
	"value": "<name/value pair>",
"message": "The name in the name/value pair is not of: notices or
rdapConformance."
}
```

3. The JSON name/values of *notices* or *rdapConformance* shall exist only once.
```
{
	"code": -12502,
	"value": "<name/value pair>",
	"message": "The name in the name/value pair of a link structure was found
more than once."
}
```

4. If the JSON name *notices* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.
```
{
	"code": -12503,
	"value": "<name/value pair>",
	"message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

5. If the JSON name *rdapConformance* exists, the value shall pass the test RDAP Conformance validation [stdRdapConformanceValidation] defined in this document.
```
{
	"code": -12504,
	"value": "<name/value pair>",
	"message": "The value for the JSON name value does not pass RDAP
Conformance validation [stdRdapConformanceValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

7.3.5. Nameservers search validation [stdRdapNameserversSearchValidation]

The following steps should be used to test that a nameserverSearchResults data structure is valid:

1. The *nameserverSearchResults* data structure must be a syntactically valid JSON object.

```
{
        "code": -12600,
        "value": "<nameserver structure>",
        "message": "The nameserver structure is not syntactically valid."
}
```

2. The name of every name/value pairs shall be any of: *nameserverSearchResults*, *remarks*, *events*, *notices* or *rdapConformance*.

```
{
        "code": -12601,
        "value": "<name/value pair>",
"message": "The name in the name/value pair is not of: nameserverSearchResults,
remarks, events, notices or rdapConformance."
}
```

3. The JSON name/values of *nameserverSearchResults*, *remarks*, *events*, *notices* or *rdapConformance* shall exist only once.

```
{
        "code": -12602,
        "value": "<name/value pair>",
        "message": "The name in the name/value pair of a link structure was found
more than once."
}
```

3. The *nameserverSearchResults* data structure must be a syntactically valid JSON array.

```
{
        "code": -12603,
        "value": "<nameserverSearchResults structure>",
        "message": "The nameserverSearchResults structure is not syntactically
valid."
}
```

4. For every object (i.e. nameserver) of the JSON array, verify that the *nameserverSearchResults* structure complies with:

    4.1. The object (i.e. nameserver) shall pass the Nameserver lookup validation [stdRdapNameserverLookupValidation] test.

```
{
       "code": -12604,
       "value": "<nameserver object>",
       "message": "The nameserver object does not pass Nameserver lookup
       validation [stdRdapNameserverLookupValidation]."
}
```

5. If the JSON name *remarks* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.

```
{
        "code": -12605,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

6. If the JSON name *events* exists, the value shall pass the test Events Validation [stdRdapEventsValidation] defined in this document.

```
{
        "code": -12606,
```

```
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Events
Validation [stdRdapEventsValidation]."
}
```

7. If the JSON name *notices* exists, the value shall pass the test Notices and Remarks Validation [stdRdapNoticesRemarksValidation] defined in this document.

```
{
        "code": -12607,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass Notices and
Remarks Validation [stdRdapNoticesRemarksValidation]."
}
```

8. If the JSON name *notices* exists and the object is not the topmost JSON object.

```
{
        "code": -12608,
        "value": "<name/value pair>",
        "message": "The value for the JSON name notices exists but object is not
the topmost JSON object."
}
```

9. If the JSON name *rdapConformance* exists, the value shall pass the test RDAP Conformance validation [stdRdapConformanceValidation] defined in this document.

```
{
        "code": -12609,
        "value": "<name/value pair>",
        "message": "The value for the JSON name value does not pass RDAP
Conformance validation [stdRdapConformanceValidation]."
}
```

The following normative references are required to understand this test group:

- RFC7159
- RDAP_RFCs

8. Collection of RDAP Profile February-2019 tests

8.1. Technical Implementation Guide – General

8.1.1. [tigSection_1_2_Validation]

The following steps should be used to test the RDAP protocol section 1.2 of the RDAP_Technical_Implementation_Guide_2_1:

1. If the scheme of the URI to be tested is "http":

```
{
        "code": -20100,
        "value": "<URI>",
        "message": "The URL is HTTP, per section 1.2 of the
RDAP_Technical_Implementation_Guide_2_1 shall be HTTPS only."
}
```

2. If the scheme of the URI to be tested is "https", perform the same RDAP query using "http". If the HTTP URI provides a response (other than redirect)::

```
{
        "code": -20101,
        "value": "<RDAP response provided over HTTP> + "\n/\n" + <RDAP response
provided over HTTPS>",
        "message": "The RDAP response was provided over HTTP, per section 1.2 of
the RDAP_Technical_Implementation_Guide_2_1 shall be HTTPS only."
}
```

Note: If redirects are present, the test [tigSection_1_2_Validation] shall be performed on the URL on the last HTTP redirect.

8.1.2. [tigSection_1_3_Validation]

The following steps should be used to test the RDAP protocol section 1.3 of the RDAP_Technical_Implementation_Guide_2_1:

1. If the scheme of the URI to be tested is "https", verify that SSLv2 and SSLv3 are not offered by the RDAP server.

```
{
        "code": -20200,
        "value": "<URI>",
        "message": "The RDAP server is offering SSLv2 and/or SSLv3."
}
```

Note: the test [tigSection_1_3_Validation] shall be performed on the URL on every HTTP redirect.

8.1.3. [tigSection_1_6_Validation]

The following steps should be used to test the RDAP protocol section 1.6 of the RDAP_Technical_Implementation_Guide_2_1:

1. The tool shall use the HTTP HEAD method on the URI to be tested. If the HTTP Status code is different from the status code obtained when doing the GET method:

```
{
        "code": -20300,
        "value": "<HTTP Status code when using the GET method> + "\n/\n" "<HTTP
Status code when using the HEAD method>",
"message": "The HTTP Status code obtained when using the HEAD method is different
from the GET method. See section 1.6 of the
RDAP_Technical_Implementation_Guide_2_1."
}
```

8.1.4. [tigSection_1_8_Validation]

The following steps should be used to test the RDAP protocol section 1.8 of the RDAP_Technical_Implementation_Guide_2_1:

1. Obtain the Resource Record for the A QTYPE for the host in the URI. Validate that the status of the DNS response is not NOERROR. Validate that all IPv4 addresses in the RDATA pass IPv4 address validation [ipv4Validation]:

```
{
        "code": -20400,
        "value": "<IPv4 addresses>",
        "message": "The RDAP service is not provided over IPv4. See section 1.8 of
the RDAP_Technical_Implementation_Guide_2_1."
}
```

2. Obtain the Resource Record for the AAAA QTYPE for the host in the URI. Validate that the status of the DNS response is not NOERROR. Validate that all IPv6 addresses in the RDATA pass IPv6 address validation [ipv6Validation]:

```
{
        "code": -20401,
        "value": "<IPv6 addresses>",
        "message": "The RDAP service is not provided over IPv6. See section 1.8 of
the RDAP_Technical_Implementation_Guide_2_1."
}
```

Note: the test [tigSection_1_8_Validation] shall be performed on the URL on every HTTP redirect.

8.1.5. [tigSection_1_13_Validation]

The following steps should be used to test the RDAP protocol section 1.13 of the RDAP_Technical_Implementation_Guide_2_1:

1. Validate that the HTTP header "Access-Control-Allow-Origin: *" is included in the RDAP response.

```
{
	"code": -20500,
	"value": "<HTTP headers>",
	"message": "The HTTP header "Access-Control-Allow-Origin: *" is not
included in the HTTP headers. See section 1.13 of the
RDAP_Technical_Implementation_Guide_2_1."
}
```

Note: the test [tigSection_1_13_Validation] shall be performed on the URL on every HTTP redirect.

8.1.6. [tigSection_1_14_Validation]

The following steps should be used to test the RDAP protocol section 1.14 of the RDAP_Technical_Implementation_Guide_2_1:

1. Validate that the JSON string value "icann_rdap_technical_implementation_guide_0" is included in the RDAP Conformance data structure.

```
{
	"code": -20600,
	"value": "<rdapConformance data structure>",
	"message": "The RDAP Conformance data structure does not include
icann_rdap_technical_implementation_guide_0. See section 1.14 of the
RDAP_Technical_Implementation_Guide_2_1."
}
```

8.1.7. [tigSection_3_3_and_3_4_Validation]

The following steps should be used to test the RDAP protocol section 3.3 and 3.4 of the RDAP_Technical_Implementation_Guide_2_1:

1. Validate that at least one links data structure exists within the notices object in the topmost object.

```
{
	"code": -20700,
	"value": "<notices data structure>",
	"message": "A links object was not found in the notices object in the
topmost object. See section 3.3 and 3.4 of the
RDAP_Technical_Implementation_Guide_2_1."
}
```

8.1.8. [tigSection_4_1_Validation]

The following steps should be used to test the RDAP protocol section 4.1 of the TIG:

1. Validate that all the *entities* in the RDAP Response contain structured address. If a street address has more than one line, it MUST be structured as an array of strings.

```
{
        "code": -20800,
        "value": "<entity data structure>",
        "message": "An entity with a non-structured address was found. See section
4.1 of the TIG."
}
```

8.1.9. [tigSection_7_1_and_7_2_Validation]

The following steps should be used to test the RDAP protocol section 7.1 and 7.2 of the TIG:

1. Validate that at all the *tel* properties in the *entities* in the RDAP Response contain voice or fax as type parameter.

```
{
        "code": -20900,
        "value": "<entity data structure>",
        "message": "An entity with a tel property without a voice or fax type was
found. See section 7.1 and 7.2 of the TIG."
}
```

8.2. Technical Implementation Guide - Registry

8.2.1. [tigSection_1_11_1_Validation]

The following steps should be used to test the RDAP protocol section 1.11.1 and 1.2 of the RDAP_Technical_Implementation_Guide_2_1:

1. Verify that the TLD of the domain name is listed in the bootstrapDomainNameSpace.

```
{
	"code": -23100,
	"value": "<TLD> + "\n/\n" <bootstrapDomainNameSpace>",
	"message": "The TLD is not included in the bootstrapDomainNameSpace. See section 1.11.1 of the RDAP_Technical_Implementation_Guide_2_1."
}
```

2. Validate that at least one base URL exists in the bootstrapDomainNameSpace for the TLD.

```
{
	"code": -23101,
	"value": "<TLD element in bootstrapDomainNameSpace>",
	"message": "The TLD entry in bootstrapDomainNameSpace does not contain a base URL. See section 1.11.1 of the RDAP_Technical_Implementation_Guide_2_1."
}
```

3. For the entry of the TLD in bootstrapDomainNameSpace verify that every one of the base URLs contain a schema of "https".

```
{
	"code": -23102,
	"value": "<TLD entry in bootstrapDomainNameSpace>",
	"message": "One or more of the base URLs for the TLD contain a schema different from https. See section 1.2 of the RDAP_Technical_Implementation_Guide_2_1."
}
```

8.2.2. [tigSection_3_2_Validation]

The following steps should be used to test the RDAP protocol section 3.2 of the RDAP_Technical_Implementation_Guide_2_1:

1. Validate that a links data structure in the topmost object exists, and the links object shall contain the elements *rel*:related and *href*.

```
{
	"code": -23200,
	"value": "<links data structure>",
	"message": "A links data structure in the topmost object exists, and the links object shall contain the elements rel:related and href, but they were not found. See section 3.2 of the RDAP_Technical_Implementation_Guide_2_1."
}
```

8.2.3. [tigSection_6_1_Validation]

The following steps should be used to test the RDAP protocol section 6.1 of the RDAP_Technical_Implementation_Guide_2_1:

1. For the *entity* with the registrar role within the domain object, validate that a *publicIds* member is included.

```
{
        "code": -23300,
        "value": "<entity data structure>",
        "message": "A publicIds member is not included in the entity with the
registrar role."
}
```

2. For the *entity* with the registrar role within the domain object, if a *publicIds* member is included, validate that the identifier member is a positive integer.

```
{
        "code": -23301,
        "value": "<publicIds data structure>",
        "message": "The identifier of the publicIds member of the entity with the
registrar role is not a positive integer."
}
```

8.3. Technical Implementation Guide – Registrar

8.3.1. [tigSection_1_12_1_Validation]

The following steps should be used to test the RDAP protocol section 1.12.1 of the RDAP_Technical_Implementation_Guide_2_1:

1. Get the identifier in the *publicIds* element in the *entity* with the registrar role.

```
{
        "code": -26100,
        "value": "<publicIds data structure>",
        "message": "An identifier in the publicIds within the entity data
structure with the registrar role was not found. See section 1.12.1 of the
RDAP_Technical_Implementation_Guide_2_1."
}
```

2. For the *identifier* found in the previous step, validate that an entry exists in the **registrarId**.

```
{
        "code": -26101,
        "value": "<identifier> "\n/\n" <registrarId>",
        "message": "The registrar identifier is not included in the registrarId.
See section 1.12.1 of the RDAP_Technical_Implementation_Guide_2_1."
}
```

3. For the *identifier* found in the previous step, verify that every of the base URLs contain a schema of "https".

```
{
        "code": -26102,
        "value": "<Registrar entry in registrarId>",
        "message": "One or more of the base URLs for the registrar contain a
schema different from https. See section 1.2 of the
RDAP_Technical_Implementation_Guide_2_1."
}
```

8.4. RDAP Response Profile - General

8.4.1. [rdapResponseProfile_1_2_2_Validation]

The following steps should be used to test the RDAP protocol section 1.2.2 of the RDAP_Response_Profile_2_1:

1. Validate that the RDAP response does not contain browser executable code (e.g., JavaScript).

```
{
        "code": -40100,
        "value": "<rdap response>",
        "message": "The RDAP response contains browser executable code (e.g.,
JavaScript). See section 1.2.2 of the RDAP_Response_Profile_2_1.""
}
```

Note: a library for HTML sanitizing (https://en.wikipedia.org/wiki/HTML_sanitization) may be used for this test.

8.4.2. [rdapResponseProfile_1_3_Validation]

The following steps should be used to test the RDAP protocol section 1.3 of the RDAP_Response_Profile_2_1:

1. Validate that the JSON string value "icann_rdap_response_profile_0" is included in the *RDAP Conformance* data structure.

```
{
        "code": -40200,
        "value": "<rdapConformance data structure>",
        "message": "The RDAP Conformance data structure does not include
icann_rdap_response_profile_0. See section 1.3 of the RDAP_Response_Profile_2_1."
}
```

8.4.3. [rdapResponseProfile_1_4_Validation]

The following steps should be used to test the RDAP protocol section 1.4 of the RDAP_Response_Profile_2_1:

1. Validate that the country name parameter is empty in the *adr* of all the jCard objects in the RDAP response.

```
{
        "code": -40400,
        "value": "<vcard object>",
        "message": "A vcard object with a country name parameter with data was
found. "
}
```

8.5. RDAP Response Profile - Miscellaneous

8.5.1. [rdapResponseProfile_2_3_1_3_and_2_7_6_and_3_3_and_4_4_Validation]

The following steps should be used to test the RDAP protocol section 2.3.1.3, 2.7.6, 3.3 and 4.4 of the RDAP_Response_Profile_2_1:

1. Validate that an *eventAction* type "last update of RDAP database" exists in events structure included in the topmost object.

```
{
        "code": -43100,
        "value": "<events data structure>",
        "message": "An eventAction type last update of RDAP database does not
exists in the topmost events data structure. See section 2.3.1.3, 2.7.6, 3.3 and
4.4 of the RDAP_Response_Profile_2_1."
}
```

8.6. RDAP Response Profile - Domain

8.6.1. [rdapResponseProfile_2_1_Validation]

The following steps should be used to test the RDAP protocol section 2.1 of the RDAP_Response_Profile_2_1:

1. If domain/<domain name> in the RDAP Query URI contains only A-label or NR-LDH labels, the topmost domain object shall contain a *ldhName*.

```
{
        "code": -46100,
        "value": "<domain object>",
        "message": "The RDAP Query URI contains only A-label or NR-LDH labels, the
topmost domain object does not contain a ldhName member. See section 2.1 of the
RDAP_Response_Profile_2_1."
}
```

2. If domain/<domain name> in the RDAP Query URI contains one or more U-label, the topmost domain object shall contain an *unicodeName*.

```
{
        "code": -46101,
        "value": "<domain object>",
        "message": " The RDAP Query URI contains one or more U-label, the topmost
domain object does not contain a unicodeName member. See section 2.1 of the
RDAP_Response_Profile_2_1."
}
```

8.6.2. [rdapResponseProfile_2_2_Validation]

The following steps should be used to test the RDAP protocol section 2.2 of the RDAP_Response_Profile_2_1:

1. The handle in the topmost *domain* object shall comply with the following format specified in RFC5730: "(\w|_){1,80}-\w{1,8}".

```
{
        "code": -46200,
        "value": "<domain object>",
        "message": "The handle in the domain object does not comply with the
format (\w|_){1,80}-\w{1,8} specified in RFC5730"."
}
```

2. If the handle in the topmost *domain* object comply with the format: "(\w|_){1,80}-\w{1,8}", validate that the string followed by a hyphen ("-", ASCII value 0x002D) is registered in **EPPROID**.

```
{
        "code": -46201,
        "value": "<domain object>",
        "message": "The globally unique identifier in the domain object handle is
not registered in EPPROID."
}
```

8.6.3. [rdapResponseProfile_2_3_1_1_Validation]

The following steps should be used to test the RDAP protocol section 2.3.1.1 of the RDAP_Response_Profile_2_1:

1. Validate that an *eventAction* of type "registration" exists in the topmost events structure.

```
{
        "code": -46300,
        "value": "<events data structure>",
        "message": "An eventAction of type registration does not exists in the
topmost events data structure. See section 2.3.1.1 of the
RDAP_Response_Profile_2_1."
}
```

8.6.4. [rdapResponseProfile_2_3_1_2_Validation]

The following steps should be used to test the RDAP protocol section 2.3.1.2 of the RDAP_Response_Profile_2_1:

1. Validate that an *eventAction* type "expiration" exists in the topmost events structure.

```
{
        "code": -46400,
        "value": "<events data structure>",
        "message": "An eventAction of type expiration does not exists in the
topmost events data structure. See section 2.3.1.2 of the
RDAP_Response_Profile_2_1."
}
```

8.6.5. [rdapResponseProfile_notices_included_Validation]

The following steps should be used to test that a notices member appear in the RDAP response:

1. Validate that a *notices* member appears in the RDAP response.

```
{
        "code": -46500,
        "value": "<RDAP response>",
        "message": "A notices members does not appear in the RDAP response."
}
```

8.6.6. [rdapResponseProfile_2_6_3_Validation]

The following steps should be used to test the RDAP protocol section 2.6.3 of the RDAP_Response_Profile_2_1:

1. Validate that the *notices* member contains an element in the JSON array with a title “Status Codes”, a description containing the string “For more information on domain status codes, please visit https://icann.org/epp” and a links member with an *href* containing "https://icann.org/epp".

```
{
        "code": -46600,
```

```
        "value": "<notices structure>",
        "message": "The notice for https://icann.org/epp was not found."
}
```

8.6.7. [rdapResponseProfile_2_11_Validation]

The following steps should be used to test the RDAP protocol section 2.11 of the RDAP_Response_Profile_2_1:

1. Validate that the *notices* member contains an element in the JSON array with a title "RDDS Inaccuracy Complaint Form", a description containing the string "URL of the ICANN RDDS Inaccuracy Complaint Form: https://icann.org/wicf" and a links member with an *href* containing "https://icann.org/wicf".

```
{
        "code": -46700,
        "value": "<notices structure>",
        "message": "The notice for https://icann.org/wicf was not found."
}
```

8.6.8. [rdapResponseProfile_2_10_Validation]

The following steps should be used to test the RDAP protocol section 2.10 of the RDAP_Response_Profile_2_1:

1. Validate that a *secureDNS* member is included in the domain object.

```
{
        "code": -46800,
        "value": "<domain object>",
        "message": "A secureDNS member does not appear in the domain object."
}
```

2. The JSON name *delegationSigned* shall appear.

```
{
        "code": -46801,
        "value": "<secureDNS structure>",
        "message": "The delegationSigned element does not exist."
}
```

3. **If *delegationSigned* has a value of true,** one dsData name/values or one *keyData* name/value shall appear.

```
{
        "code": -46802,
        "value": "<secureDNS structure>",
        "message": "delegationSigned value is true, but no dsData nor keyData
name/value pair exists."
}
```

8.6.9. [rdapResponseProfile_rfc5731_Validation]

The following steps should be used to test that the status values comply with RFC5731:

1. Validate that the values of the *status* member in the topmost object comply with the following:
    a. "active" status MUST NOT be combined with any other status.
    b. "pending delete" status MUST NOT be combined with either "client delete prohibited" or "server delete prohibited" status.
    c. "pending renew" status MUST NOT be combined with either "client renew prohibited" or "server renew prohibited" status.
    d. "pending transfer" status MUST NOT be combined with either "client transfer prohibited" or "server transfer prohibited" status.
    e. "pending update" status MUST NOT be combined with either "client update prohibited" or "server update prohibited" status.
    f. The pending create, pending delete, pending renew, pending transfer, and pending update status values MUST NOT be combined with each other.

```
{
        "code": -46900,
        "value": "<status data structure>",
        "message": "The values of the status data structure does not comply with
RFC5731."
}
```

8.6.10. [rdapResponseProfile_rfc3915_Validation]

The following steps should be used to test that the status values comply with RFC3915:

1. Validate that the values of the *status* member in the topmost object comply with the following:
    a. "redemption period" status MUST only be combined with "pending delete".
    b. "pending restore" status MUST only be combined with "pending delete".

```
{
        "code": -47000,
        "value": "<status data structure>",
        "message": "The values of the status data structure does not comply with
RFC3915."
}
```

8.6.11. [rdapResponseProfile_2_6_1_Validation]

The following steps should be used to test the RDAP protocol section 2.6.1 of the RDAP_Response_Profile_2_1:

1. Validate that a *status* member in the topmost object contain at least one value.

```
{
        "code": -47100,
        "value": "<status data structure>",
        "message": "The status member does not contain at least one value."
}
```

8.6.12. [rdapResponseProfile_2_9_1_and_2_9_2_Validation]

The following steps should be used to test the RDAP protocol section 2.9.1 and 2.9.2 of the RDAP_Response_Profile_2_1:

1. If the *nameservers* member is included within the *domain* object, validate that all nameserver objects contain the *ldhName* element.

```
{
        "code": -47200,
        "value": "<nameservers data structure>",
        "message": "A nameserver object without ldhName was found."
}
```

2. If the *nameservers* member is included within the *domain* object, validate that all handles in the nameserver objects comply with the following format specified in RFC5730: "(\w|_){1,80}-\w{1,8}".

```
{
        "code": -47201,
        "value": "<nameserver object>",
        "message": "The handle in the nameserver object does not comply with the
format (\w|_){1,80}-\w{1,8} specified in RFC5730"."
}
```

3. If the *nameservers* member is included within the *domain* object, validate that the string followed by a hyphen ("-", ASCII value 0x002D) is registered in **EPPROID** for all the handles that comply with the format "(\w|_){1,80}-\w{1,8}".

```
{
        "code": -47202,
        "value": "<nameserver object>",
        "message": "The globally unique identifier in the nameserver object handle
is not registered in EPPROID."
}
```

4. If the *nameservers* member is included within the *domain* object and at least one nameserver object contains a *handle* or a *status* element, validate that all nameserver objects include a *handle* and a *status* element.

```
{
        "code": -47203,
        "value": "<nameserver object>",
        "message": "The handle or status in the nameserver object is not
included."
}
```

5. If the *nameservers* member is included within the domain object, validate that all *status* elements included in the nameserver objects comply with the following:
    a. "active" status MAY only be combined with "associated" status.
    b. "associated" status MAY be combined with any status.
    c. "pending delete" status MUST NOT be combined with either "client delete prohibited" or "server delete prohibited" status.
    d. "pending update" status MUST NOT be combined with either "client update prohibited" or "server update prohibited" status.
    e. The pending create, pending delete, pending renew, pending transfer, and pending update status values MUST NOT be combined with each other.

```
{
        "code": -47204,
        "value": "<status data structure>",
        "message": "The values of the status data structure does not comply with
RFC5732."
}
```

8.6.13. [rdapResponseProfile_2_4_1_Validation]

The following steps should be used to test the RDAP protocol section 2.4.1 of the RDAP_Response_Profile_2_1:

1. An *entity* with the registrar role within the topmost domain object shall exist.

```
{
        "code": -47300,
        "value": "<domain object data structure>",
        "message": "An entity with the registrar role was not found in the domain
topmost object."
}
```

2. Only one *entity* with the registrar role within the topmost domain object shall exist.

```
{
        "code": -47301,
        "value": "<domain object data structure>",
        "message": "More than one entities with the registrar role were found in
the domain topmost object."
}
```

3. For the *entity* with the registrar role within the topmost domain object, validate that a *fn* member is included in all of the vcard objects.

```
{
	"code": -47302,
	"value": "<entity data structure>",
	"message": "An fn member was not found in one or more vcard objects of the
entity with the registrar role."
}
```

8.6.14. [rdapResponseProfile_2_4_2_and_2_4_3_Validation]

The following steps should be used to test the RDAP protocol section 2.4.2 and 2.4.3 of the RDAP_Response_Profile_2_1:

1. For the *entity* with the registrar role within the topmost object, validate that a *publicIds* member is included.

```
{
	"code": -47400,
	"value": "<entity data structure>",
	"message": "A publicIds member is not included in the entity with the
registrar role."
}
```

2. For the *entity* with the registrar role within the domain object, if a *publicIds* member is included, validate that the identifier member is a positive integer.

```
{
	"code": -47401,
	"value": "<publicIds data structure>",
	"message": "The identifier of the publicIds member of the entity with the
registrar role is not a positive integer."
}
```

3. For the *entity* with the registrar role within the domain object, validate that the *handle* member is a positive integer.

```
{
	"code": -47402,
	"value": "<publicIds data structure>",
	"message": "The handle of the entity with the registrar role is not a
positive integer."
}
```

4. For the *entity* with the registrar role within the domain object, if a *publicIds* member is included, validate that the *identifier* member equals the *handle* member.

```
{
	"code": -47403,
	"value": "<entity data structure>",
	"message": "The identifier of the publicIds member of the entity with the
registrar role is not equal to the handle member."
}
```

5. For the *entity* with the registrar role within the domain object, validate that the value of the *handle* member exists in the **registrarId**.

```
{
	"code": -47404,
	"value": "<handle> + "\n/\n" + <registrarId>",
	"message": "The handle references an IANA Registrar ID that does not exist
in the registrarId."
```

```
}
```

8.6.15. [rdapResponseProfile_2_4_5_Validation]

1. For the *entity* with the registrar role within the *domain* object, validate that an *entity* with the abuse role is included, and the entity with the abuse role includes a *tel* and *email* members in all the vcard objects.

```
{
        "code": -47500,
        "value": "<entity data structure>",
        "message": "Tel and email members were not found for the entity within the
entity with the abuse role in the topmost domain object."
}
```

8.7. RDAP Response Profile - Nameserver

8.7.1. [rdapResponseProfile_4_1_Validation]

The following steps should be used to test the RDAP protocol section 4.1 of the RDAP_Response_Profile_2_1:

1. If nameserver/<nameserver name> in the RDAP Query URI contains only A-label or NR-LDH labels, the topmost domain object shall contain a *ldhName*.

```
{
        "code": -49100,
        "value": "<nameserver object>",
        "message": "The RDAP Query URI contains only A-label or NR-LDH labels, the
topmost nameserver object does not contain a ldhName member. See section 2.1 of
the RDAP_Response_Profile_2_1."
}
```

2. If nameserver/<nameserver name> in the RDAP Query URI contains one or more U-label, the topmost domain object shall contain an *unicodeName*.

```
{
        "code": -49101,
        "value": "<nameserver object>",
        "message": " The RDAP Query URI contains one or more U-label, the topmost
nameserver object does not contain a unicodeName member. See section 2.1 of the
RDAP_Response_Profile_2_1."
}
```

3. The handle in the topmost *nameserver* object shall comply with the following format specified in RFC5730: "(\w|_){1,80}-\w{1,8}".

```
{
        "code": -49102,
        "value": "<nameserver object>",
        "message": "The handle in the nameserver object does not comply with the
format (\w|_){1,80}-\w{1,8} specified in RFC5730"."
}
```

4. If the handle in the topmost *nameserver* object comply with the format: "(\w|_){1,80}-\w{1,8}", validate that the string followed by a hyphen ("-", ASCII value 0x002D) is registered in **EPPROID**.

```
{
        "code": -49103,
        "value": "<nameserver object>",
        "message": "The globally unique identifier in the nameserver object handle
is not registered in EPPROID."
}
```

8.7.2. [rdapResponseProfile_4_3_Validation]

The following steps should be used to test the RDAP protocol section 4.3 of the RDAP_Response_Profile_2_1.

The following steps shall only be executed if an entity with the registrar role exists within the topmost object, and the handle is different from "not applicable":

1. For the *entity* with the registrar role within the topmost object, validate that a *publicIds* member is included.

```
{
        "code": -49200,
        "value": "<entity data structure>",
        "message": "A publicIds member is not included in the entity with the
registrar role."
}
```

2. For the *entity* with the registrar role within the domain object, if a *publicIds* member is included, validate that the identifier member is a positive integer.

```
{
        "code": -49201,
        "value": "<publicIds data structure>",
        "message": "The identifier of the publicIds member of the entity with the
registrar role is not a positive integer."
}
```

3. For the *entity* with the registrar role within the domain object, validate that the *handle* member is a positive integer.

```
{
        "code": -49202,
        "value": "<publicIds data structure>",
        "message": "The handle of the entity with the registrar role is not a
positive integer."
}
```

4. For the *entity* with the registrar role within the domain object, if a *publicIds* member is included, validate that the *identifier* member equals the *handle* member.

```
{
        "code": -49203,
        "value": "<entity data structure>",
        "message": "The identifier of the publicIds member of the entity with the
registrar role is not equal to the handle member."
}
```

5. For the *entity* with the registrar role within the domain object, validate that the value of the *handle* member exists in the **registrarId**.

```
{
        "code": -49204,
        "value": "<handle> + "\n/\n" + <registrarId>",
        "message": "The handle references an IANA Registrar ID that does not exist
in the registrarId."
}
```

The following steps shall only be executed if an entity with the registrar role exists within the topmost object, and the handle is "not applicable":

6. For the *entity* with the registrar role within the topmost object, validate that a *publicIds* member is not included.

```
{
        "code": -49205,
        "value": "<entity data structure>",
        "message": "A publicIds member is included in the entity with the
registrar role."
}
```

8.7.3. [nameserver_status]

1. If a *status* element is included in the nameserver object, validate that it complies with the following:
    f. "active" status MAY only be combined with "associated" status.
    g. "associated" status MAY be combined with any status.
    h. "pending delete" status MUST NOT be combined with either "client delete prohibited" or "server delete prohibited" status.
    i. "pending update" status MUST NOT be combined with either "client update prohibited" or "server update prohibited" status.
    j. The pending create, pending delete, pending renew, pending transfer, and pending update status values MUST NOT be combined with each other.

```
{
        "code": -49300,
        "value": "<status data structure>",
        "message": "The values of the status data structure does not comply with
RFC5732."
}
```

8.8. RDAP Response Profile – Entities within Domain

8.8.1. [rdapResponseProfile_2_7_1_X_and_2_7_2_X_and_2_7_3_X_and_2_7_4_X_Validation]

The following steps should be used to test the RDAP protocol section 2.7.1.X and 2.7.2.X and 2.7.3.X and 2.7.4.X of the RDAP_Response_Profile_2_1:

1. For entities with the registrant, administrative, technical and billing role within the domain object, if a *remarks* member with the title "REDACTED FOR PRIVACY" is included, validate that the type member is "object redacted due to authorization".

```
{
        "code": -52100,
        "value": "<entity data structure>",
        "message": "An entity with the registrant, administrative, technical or
billing role with a remarks members with the title "REDACTED FOR PRIVACY" was
found, but the description and type does not contain the value in 2.7.4.3 of the
RDAP_Response_Profile_2_1."
}
```

2. For entities with the registrant, administrative, technical and billing role within the domain object, if a *remarks* member with the title "REDACTED FOR PRIVACY" is not included, validate that valid *handle*, *fn*, *adr*, *tel* members are included. For the *adr* member, validate that the following RDDS fields are included: Street and City.

```
{
        "code": -52101,
        "value": "<entity data structure>",
        "message": "An entity with the registrant, administrative, technical or
billing role with a remarks members with the title "REDACTED FOR PRIVACY" was
found, but the description and type does not contain the value in 2.7.4.3 of the
RDAP_Response_Profile_2_1."
}
```

3. For entities with the registrant, administrative, technical and billing role within the domain object, if a *remarks* member with the title "REDACTED FOR PRIVACY" is not included, validate that the handle comply with the following format specified in RFC5730: "(\w|_){1,80}-\w{1,8}".

```
{
        "code": -52102,
        "value": "<nameserver object>",
        "message": "The handle in the entity object does not comply with the
format (\w|_){1,80}-\w{1,8} specified in RFC5730"."
}
```

4. For entities with the registrant, administrative, technical and billing role within the domain object, if a *remarks* member with the title "REDACTED FOR PRIVACY" is not included and the handle conforms to the format: "(\w|_){1,80}-\w{1,8}", validate that the string followed by a hyphen ("-", ASCII value 0x002D) is registered in **EPPROID**.

```
{
        "code": -52103,
        "value": "<nameserver object>",
        "message": "The globally unique identifier in the entity object handle is
not registered in EPPROID."
}
```

5. Only one entity shall be assigned the following roles: registrant, administrative, technical and billing.

```
{
        "code": -52104,
        "value": "<entities data structure>",
        "message": "More than one entity with the following roles were found:
registrant, administrative, technical and billing."
}
```

6. For entities with the registrant role within the domain object, validate that the CC parameter is included in the entity as defined in RFC8605.

```
{
        "code": -52105,
        "value": "<entity data structure>",
        "message": "An entity with the registrant role without the CC parameter
was found. See section 2.7.3.1 of the RDAP_Response_Profile_2_1."
}
```

8.9. RDAP Response Profile – Entities within Domain - Registry

8.9.1. [rdapResponseProfile_2_7_5_3_Validation]

The following steps should be used to test the RDAP protocol section 2.7.5.3 of the RDAP_Response_Profile_2_1:

1. For the *entities* with the registrant, administrative, technical and billing role within the domain object, if the *email* property is omitted, validate that a remarks element containing a title member with a value "EMAIL REDACTED FOR PRIVACY" and a type member with a value "object redacted due to authorization" is included in the entity object.

```
{
        "code": -55000,
        "value": "<entity data structure>",
        "message": "An entity with the administrative, technical, or billing role
without a valid "EMAIL REDACTED FOR PRIVACY" remark was found. See section 2.7.5.3
of the RDAP_Response_Profile_2_1."
}
```

Note: this test also includes 2.7.5.1.

8.10. RDAP Response Profile - Entities within Domain - Registrar

8.10.1. [rdapResponseProfile_2_7_5_2_Validation]

The following steps should be used to test the RDAP protocol section 2.7.5.2 of the RDAP_Response_Profile_2_1:

1. For the *entities* with the registrant, administrative, technical and billing role within the domain object, if the *email* property is omitted, validate that a CONTACT-URI member is included.

```
{
        "code": -58000,
        "value": "<entity data structure>",
        "message": "An entity with the administrative, technical, or billing role
without a CONTACT-URI member was found. See section 2.7.5.2 of the
RDAP_Response_Profile_2_1."
}
```

2. For the *entities* with the registrant, administrative, technical and billing role within the domain object, if a CONTACT-URI member is included, validate that the content is an email address or an http/https link.

```
{
        "code": -58001,
        "value": "<entity data structure>",
        "message": "The content of the CONTACT-URI member of an entity with the
administrative, technical, or billing role does not contain an email or http/https
link. See section 2.7.5.2 of the RDAP_Response_Profile_2_1."
}
```

Note: this test also includes 2.7.5.1.

8.11. RDAP Response Profile – Entity - Registrar

8.11.1. [rdapResponseProfile_3_1_Validation]

The following steps should be used to test the RDAP protocol section 3.1 of the RDAP_Response_Profile_2_1:

1. An *entity* with the registrar role as the topmost object shall exist.

```
{
        "code": -60100,
        "value": "<topmost object>",
        "message": "An entity with the registrar role was not found as the topmost
object. See section 3.1 of the RDAP_Response_Profile_2_1"
}
```

2. Validate that valid *handle*, *fn*, *adr*, *tel*, *email* members are included in the topmost object. For the *adr* member, validate that the following RDDS fields are included: Street, City and Country.

```
{
        "code": -60101,
        "value": "<topmost object>",
        "message": "The required members for a registrar entity were not found.
See section 3.1 of the RDAP_Response_Profile_2_1."
}
```

8.11.2. [rdapResponseProfile_3_2_Validation]

The following steps should be used to test the RDAP protocol section 3.2 of the RDAP_Response_Profile_2_1:

1. If entities with the administrative and technical roles within the topmost object exist, validate that valid *fn*, *tel*, *email* members are included.

```
{
        "code": -60200,
        "value": "<entity data structure>",
        "message": "The required members for entities with the administrative and
technical roles were not found. See section 3.2 of the RDAP_Response_Profile_2_1."
}
```

9. Normative references

- RFC791, https://tools.ietf.org/html/rfc791
- RFC3339, https://tools.ietf.org/html/rfc3339
- RFC3986, https://tools.ietf.org/html/rfc3986
- RFC5646, https://tools.ietf.org/html/rfc5646
- RFC5730, https://tools.ietf.org/html/rfc5730
- RFC5732, https://tools.ietf.org/html/rfc5732
- RFC5952, https://tools.ietf.org/html/rfc5952
- RFC7159, https://tools.ietf.org/html/rfc7159
- RFC8288, https://tools.ietf.org/html/rfc8288

DNS_RFCs:

- RFC1034, https://tools.ietf.org/html/rfc1034
- RFC1035, https://tools.ietf.org/html/rfc1035

DNSSEC_RFCs:

- RFC4033, https://tools.ietf.org/html/rfc4033
- RFC4034, https://tools.ietf.org/html/rfc4034
- RFC4035, https://tools.ietf.org/html/rfc4035

IDNA_RFCs:

- RFC5890, https://tools.ietf.org/html/rfc5890
- RFC5891, https://tools.ietf.org/html/rfc5891
- RFC5892, https://tools.ietf.org/html/rfc5892
- RFC5893, https://tools.ietf.org/html/rfc5893
- RFC5894, https://tools.ietf.org/html/rfc5894
- RFC6055, https://tools.ietf.org/html/rfc6055
- RFC6365, https://tools.ietf.org/html/rfc6365

RDAP_RFCs:

- RFC7480, https://tools.ietf.org/html/rfc7480
- RFC7481, https://tools.ietf.org/html/rfc7481
- RFC7482, https://tools.ietf.org/html/rfc7482
- RFC7483, https://tools.ietf.org/html/rfc7483
- RFC7484, https://tools.ietf.org/html/rfc7484
- RFC8605, https://tools.ietf.org/html/rfc8605

- RDAP_Technical_Implementation_Guide_2_1, https://www.icann.org/en/system/files/files/rdap-technical-implementation-guide-15feb19-en.pdf

- RDAP_Response_Profile_2_1, https://www.icann.org/en/system/files/files/rdap-response-profile-15feb19-en.pdf

- Media_Queries, https://www.w3.org/TR/css3-mediaqueries/