Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: December 2, 2018

A. Fregly
S. Sheth
S. Hollenbeck
Verisign Labs

May 31, 2018

Registration Data Access Protocol (RDAP) Search Using POSIX Regular Expressions
draft-fregly-regext-rdap-search-regex-04

Abstract

The Registration Data Access Protocol (RDAP) provides limited search functionality based on pattern matching. This document describes an RDAP query extension that provides additional search functionality using POSIX extended regular expressions.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 2, 2018.

Table of Contents

1. Introduction

The search patterns for Registration Data Access Protocol (RDAP) search as described in RFC 7482 [RFC7482] are limited. The protocol described in this specification extends RDAP search capabilities by adding path segments for RDAP search functions using a RESTful web service and POSIX [IEEE.1003.1_2013_EDITION] extended regular expressions. The service is implemented using the Hypertext Transfer Protocol (HTTP) [RFC7230] and the conventions described in RFC 7480 [RFC7480].

1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. RDAP Path Segment Specification

The path segments defined in this section are OPTIONAL extensions of path segments defined in RFC 7482 [RFC7482]. The resource type path segments for search are:

- 'domains': Used to identify a domain name information search using a pattern to match a fully-qualified domain name.
- 'nameservers': Used to identify a name server information search using a pattern to match a host name.
- 'entities': Used to identify an entity information search using a pattern to match a string identifier.

The search patterns in the path segments MUST be POSIX extended regular expressions. Non-URL-safe characters in Search patterns MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. Percent-encoding will eliminate errors that might occur due to web-server or app-server interpretation of certain unsafe characters and will eliminate errors due to inconsistent encoding and decoding semantics for certain characters. For instance, the space character may be encoded as "+" when submitted through a HTML form and encoded as "%20" when submitted through the address bar of a Web browser. Detailed results can be retrieved using the HTTP GET method and the path segments specified here.

This document defines an RDAP query parameter, "searchtype", that is used to identify search requests that require specialized processing beyond the limited functionality described in RFC 7482 [RFC7482]. Search processing using POSIX [IEEE.1003.1_2013_EDITION] extended regular expressions is indicated with a query string parameter value of "regex", e.g. "searchtype=regex". Other forms of search processing are possible and can be described in other specifications using other values for the "searchtype" query parameter. See Section 2.4 for additional information.

2.1. Domain Search

Syntax: domains?name=<domain search pattern>&searchtype=regex

Syntax: domains?nsLdhName=<domain search pattern>&searchtype=regex

Syntax: domains?nsIp=<domain search pattern>&searchtype=regex

Searches for domain information by name are specified using this form:

domains?name=XXXX&searchtype=regex

If the URL query string parameter "searchtype" has a value of "regex", then XXXX MUST be a POSIX extended regular expression. Non-URL-safe characters in XXXX MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against domains in a name space administered by the server operator. Domain names are as defined by RFC 5890 [RFC5890] in "letters, digits, hyphen" format. The following URL would be used to find information for domain names matching the "e[a-z]ample\.com" pattern:

```
https://example.com/rdap/domains?name=e%5Ba-
z%5Dample%5C.com&searchtype=regex
```

Internationalized Domain Names (IDNs) in U-label format [RFC5890] can also be matched by POSIX extended regular expression search patterns. Search patterns for these names are of the form /domains?name=XXXX&searchtype=regex, where XXXX is a POSIX extended regular expression. Non-URL-safe characters in XXXX MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against domain names in U-label format. See section 6.1 of RFC 7482 [RFC7482] for information describing U-label character encoding. See Section 5 for other considerations relative to regular expression matching of IDNs.

Searches for domain information by name server name are specified using this form:

```
domains?nsLdhName=YYYY&searchtype=regex
```

If the URL query string parameter "searchtype" has a value of "regex", then YYYY MUST be a POSIX extended regular expression. Non-URL-safe characters in YYYY MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against host names in a name space administered by the server operator. Host names are as defined by RFC 5890 [RFC5890] in "letters, digits, hyphen" format. The following URL would be used to search for domains delegated to name servers matching the "ns[1-9]\.e[a-z]ample\.com" pattern:

```
https://example.com/rdap/domains?nsLdhName=ns%5B1-9%5D%5C.e%5Ba-
z%5Dample%5C.com&searchtype=regex
```

Searches for domain information by name server IP address are specified using this form:

```
domains?nsIp=ZZZZ&searchtype=regex
```

If the URL query string parameter "searchtype" has a value of "regex", then ZZZZ MUST be a POSIX extended regular expression. Non-URL-safe characters in ZZZZ MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against IPv4 addresses [RFC1166] and IPv6 addresses [RFC5952] associated with specific name servers. The following URL would be used to search for domains that have been delegated to name servers that have IP addresses matching the "192\.0\.[1-9]\.0" pattern:

https://example.com/rdap/
domains?nsIp=192%5C.0%5C.%5B1-9%5D%5C.0&searchtype=regex

2.2. Name Server Search

Syntax: nameservers?name=<name server search pattern>&searchtype=regex

Syntax: nameservers?ip=<name server search pattern>&searchtype=regex

Searches for name server information by name server name are specified using this form:

nameservers?name=XXXX&searchtype=regex

If the URL query string parameter "searchtype" has a value of "regex", then XXXX MUST be a POSIX extended regular expression. Non-URL-safe characters in XXXX MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against name server names in a name space administered by the server operator. Name server names are as defined in RFC 5890 [RFC5890] in "letters, digits, hyphen" format. Matches will return information for the matching name servers. The following URL would be used to find information for name server names matching the "ns[1-9]\.e[a-z]ample\.com" pattern:

https://example.com/rdap/nameservers?name=ns%5B1-9%5D%5C.e%5Ba-z%5Dample%5C.com&searchtype=regex

Internationalized name server names in U-label format [RFC5890] can also be matched by POSIX extended regular expression search patterns. Search patterns for these names are of the form /nameservers?name=XXXX&searchtype=regex, where XXXX is a POSIX extended regular expression. Non-URL-safe characters in XXXX MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against name server names in U-label format. See section 6.1

of RFC 7482 [RFC7482] for information describing U-label character encoding. See Section 5 for other considerations relative to regular expression matching of U-labels.

Searches for name server information by name server IP address are specified using this form:

nameservers?ip=YYYY&searchtype=regex

If the URL query string parameter "searchtype" has a value of "regex", then YYYY MUST be a POSIX extended regular expression. Non-URL-safe characters in YYYY MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against IPv4 addresses [RFC1166] and IPv6 addresses [RFC5952] associated with specific name servers. The following URL would be used to search for name server names that resolve to addresses matching the "192\.0\.[1-9]\.0" pattern:

https://example.com/rdap/
nameservers?ip=192%5C.0%5C.%5B1-9%5D%5C.0&searchtype=regex

2.3. Entity Search

Syntax: entities?fn=<entity name search pattern>&searchtype=regex

Syntax: entities?handle=<entity handle search pattern>&searchtype=regex

Searches for entity information by name are specified using this form:

entities?fn=XXXX&searchtype=regex

If the URL query string parameter "searchtype" has a value of "regex", then XXXX MUST be a POSIX extended regular expression. Non-URL-safe characters in XXXX MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against the "FN" property of an entity (such as a contact, registrant, or registrar) name as specified in Section 5.1 of RFC 7483 [RFC7483]. The following URL would be used to find information for entity names matching the "Bobby[[:space:]]Joe[a-z]*" pattern:

https://example.com/rdap/
entities?fn=Bobby%5B%5B%3Aspace%3A%5D%5DJoe%5Ba-z%5D%2A&searchtype=regex

Searches for entity information by handle are specified using this form:

entities?handle=XXXX&searchtype=regex

If the URL query string parameter "searchtype" has a value of "regex", then XXXX MUST be a POSIX extended regular expression. Non-URL-safe characters in XXXX MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against an entity (such as a contact, registrant, or registrar) identifier whose syntax is specific to the registration provider. The following URL would be used to find information for entity handles matching the "CID-4[0-9]*" pattern:

https://example.com/rdap/entities?handle=CID-4%5B0-9%5D%2A&searchtype=regex

2.4. Future Path Segments

OPTIONAL extensions to new RDAP path segments defined in future RDAP specifications MAY be implemented to support POSIX extended regular expressions search capability. The syntax for such OPTIONAL extensions MUST be modeled on the syntax defined in Section 2.1, Section 2.2, and Section 2.3. The following syntax template MUST be followed:

Syntax: {path_segment}?{property}=XXXX&searchtype=regex

If the URL query string parameter "searchtype" has a value of "regex", then XXXX MUST be a POSIX extended regular expression. Non-URL-safe characters in XXXX MUST be percent-encoded. Percent-encoding MUST be as described in section 2.1 of RFC 3986 [RFC3986]. The supplied regular expression will be matched against the property specified by {property} for the path segment specified by {path_segment}. For example, if a new RDAP path segment "foo" is defined and has a property "bar", the following URL would be used to find information for the "foo" resource type with a "bar" property matching the "widget:.*mech.*" pattern:

https://example.com/rdap/foo?bar=widget%3A.%2Amech.%2A&searchtype=regex

3. Search Pattern Syntax

POSIX extended regular expression search pattern syntax is defined in Section 9 of IEEE Std 1003.1, 2013 Edition [IEEE.1003.1_2013_EDITION]. An RDAP service implementation MAY

implement a subset of the extended regular expression syntax and capabilities defined by the specification. An RDAP service implementation MUST specify the regular expression syntax and capabilities it supports in response to a query to the /help path segment as specified in section 3.1.6 of RFC 7482 [RFC7482].

Characters within a regular expression search pattern may be URI reserved characters. To avoid ambiguity in parsing a URL containing a regular expression search pattern, non-URL-safe character in the regular expression search pattern MUST be percent-encoded as described in RFC 3986 [RFC3986].

4. Query Processing

RDAP clients using regular expression search patterns MUST percent-encode non-URL-safe characters in the regular expression search pattern as described in RFC 3986 [RFC3986]. The regular expression SHOULD be consistent with the regular expression syntax and capabilities supported by the RDAP service implementation that is being queried in order to provide predictable results. The use of a regular expression that is not consistent with the capabilities of the RDAP service implementation MUST result in the return of an HTTP 400 response code as described in section 5.4 of RFC 7480 [RFC7480].

An RDAP service implementation will receive regular expressions search patterns that contain percent-encoded characters. Prior to processing a regular expression, the RDAP service MUST decode the received percent-encoded characters in regular expressions as described in RFC 3986 [RFC3986]. After decoding the received regular expression, the regular expression MUST be matched as described in Section 2.1, Section 2.2 and Section 2.3. Matching records related to the search are then returned in the client.

Server operating systems are typically configured to use a collection of regional and language rules that describe default processing conventions, such as sort order, date format, etc., as part of a "locale" setting. The regular expression library used for an RDAP service implementation will typically acquire all the information it requires for the current locale from the underlying operating system. The locale used by a regular expression library may impact the results of regular expression searches based on locale-specific processing.For example, a POSIX locale can have collating sequences to describe how certain characters or groups of characters can be ordered. In the Czech language, for example, "ch" can be treated as if it were one character. You can use the collating sequence element [.ch.] inside a bracketed expression to match "ch" when the Czech locale (cs-CZ) is active, but a similar collating sequence would not match the string if the system locale was, for example, en_US. Users

submitting regular expression searches that do not take into account locale-specific processing may receive misleading or inaccurate results. As such, it is RECOMMENDED to identify the underlying locale in the "help" path segment as specified in section 3.1.6 of RFC 7482 [RFC7482]. This will help RDAP clients construct regular expressions that can be processed in a predictable way.

The POSIX regular expression specification [IEEE.1003.1_2013_EDITION] allows implementations to provide case insensitive searching. RDAP service implementations SHOULD implement case insensitive searching as described in the specification. This will allow for consistency in search results regardless of the case of the RDAP data being searched. For example, some RDAP service implementations may represent domain names in upper case during searching while other RDAP service implementations may represent domain names in lower case or mixed case during searching. Case insensitive searching will alleviate the need for search clients to know how each RDAP service implementation represents the case of searchable data. RDAP service implementations that do not perform case insensitive searching may produce unexpected search results for entities that are not aware of how the service represents the case of searchable data.

An RDAP service implementation MUST specify its support or lack of support for case insensitive searching in response to a query to the /help path segment as specified in section 3.1.6 of RFC 7482 [RFC7482].

Servers indicate the success or failure of query processing of a regular expression search pattern by returning an appropriate HTTP response code to the client. Response codes not specifically identified in this document are described in RFC 7480 [RFC7480].

5. Internationalization Considerations

An RDAP service implementation that supports regular expression search patterns MUST support pattern construction and pattern matching using UTF-8 encoded character strings. Other character encoding considerations are described in section 6.1 of RFC 7482 [RFC7482].

6. Implementation Considerations

The set of related records that may be returned in response to a search with a regular expression search pattern are subject to the constraints specified in section 4.2 of RFC 7482 [RFC7482].

An RDAP service implementation MAY choose to limit the scope of searches to RDAP data that is managed by the RDAP service

implementation. For example, an RDAP response to a query that could be matched against multiple TLDs or data in related RDAP repositories (such as those distributed between domain registry and domain registrar) need only return matches for the data managed by the RDAP service implementation.

Server operating systems are typically configured to use a collection of regional and language rules that describe default processing conventions, such as sort order, date format, etc., as part of a 'locale' setting. The regular expression library used for an RDAP service implementation will typically acquire all the information it requires for the current locale from the underlying operating system. The locale used by a regular expression library may impact the results of regular expression searches based on locale-specific processing. For example, a POSIX locale can have collating sequences to describe how certain characters or groups of characters can be ordered. In the Czech language, for example, 'ch' can be treated as if it were one character. You can use the collating sequence element [.ch.] inside a bracketed expression to match 'ch' when the Czech locale (cs-CZ) is active, but a similar collating sequence would not match the string if the system locale was, for example, en_US. Users submitting regular expression searches that do not take into account locale-specific processing may receive misleading or inaccurate results. As such, it is RECOMMENDED to identify the underlying locale in the 'help' path segment as defined in section 3.1.6 of RFC 7482 [RFC7482]. This will help RDAP clients construct regular expressions that can be processed in a predictable way.

Implementors should take care to ensure that decoding of percent-encoded characters in a received regular expression is only performed once. Standard APIs for processing HTTP requests will likely perform decoding of percent-encoded characters prior to providing a received regular expression to the RDAP service implementation code. In such case, the RDAP service implementation code should not attempt to perform decoding for percent-encoded characters.

Regular expression matching results for some search patterns may vary based on the regular expression search engine used, the version of the engine used, and configuration of the search engine. For example, POSIX [IEEE.1003.1_2013_EDITION] defines different semantics based on whether a search is using Basic Regular Expressions (BRE) or Extended Regular Expressions (ERE). Search mechanisms that perform search processing compliant with Perl Compatible Regular Expressions (PCRE) as defined by pcre.org [PCRE] and in Perl 5 [PERLRE] may also produce matches that differ from matches produced by POSIX compatible regular expression matching. Differences in regular expression matching between POSIX BRE, POSIX ERE and PCRE are illustrated in the examples below, where the "sed" command without the "-E" option is

used for POSIX BRE matching, the "sed" command with the "-E" option is used for POSIX ERE matching, and the "perl" command is used for PCRE matching.

```
$ echo 'abcdef' | sed 's/ab(cd)?(cdef)?/[xxxx]/'
abcdef
$ echo 'abcdef' | sed -E 's/ab(cd)?(cdef)?/[xxxx]/'
[xxxx]
$ echo 'abcdef' | perl -p -e 's/ab(cd)?(cdef)?/[xxxx]/'
[xxxx]ef

$ echo 'aaa' | sed 's/a\{3,\}/[xxxx]/'
[xxxx]
$ echo 'aaa' | sed 's/a{3,}/[xxxx]/'
aaa
$ echo 'aaa' | sed -E 's/a\{3,\}/[xxxx]/'
aaa
$ echo 'aaa' | sed -E 's/a{3,}/[xxxx]/'
[xxxx]
$ echo 'aaa' | perl -p -e 's/a\{3,\}/[xxxx]/'
aaa
$ echo 'aaa' | perl -p -e 's/a{3,}/[xxxx]/'
[xxxx]
```

Use of POSIX extended regular expressions is motivated by broad support in the form of API availability [GNU] and database support, with the following major databases supporting POSIX extended regular expressions:

Oracle [ORACLE]
MySQL [MYSQL]
Postgres [POSTGRES]

7. IANA Considerations

FOR DISCUSSION: The URL query parameter "searchtype" with a value of "regex" is specified here-in as syntax for specifying that the RDAP query search pattern is a POSIX extended regular expression. The same approach could be used for specifying future OPTIONAL RDAP search mechanisms. An IANA-maintained registry of RDAP search mechanisms is recommended for recording a list of allowable values for the "searchtype" query parameter.

8. Implementation Status

Note to RFC Editor: Please remove this entire section before publication along with the reference to RFC7942 [RFC7942].

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in RFC 7942. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to RFC 7942, "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

8.1. Verisign Labs

Responsible Organization: Verisign Labs
Location: https://rdap.verisignlabs.com/
Description: This implementation includes support for POSIX extended regular expression domain registry RDAP queries using live data from the .cc and .tv country code top-level domains. This implementation also supports federated authentication using OpenID Connect providers as described in [RDAPOPENID]. Three access levels are provided based on the authenticated identity of the client:

1. Unauthenticated: Limited information is returned in response to queries from unauthenticated clients.
2. Basic: Clients who authenticate using a publicly available identity provider like Google Gmail or Microsoft Hotmail will receive all of the information available to an unauthenticated client plus additional registration metadata, but no personally identifiable information associated with entities.
3. Advanced: Clients who authenticate using a more restrictive identity provider will receive all of the information available to a Basic client plus whatever information the server operator deems appropriate for a fully authorized

client. Currently supported identity providers include those developed by Verisign Labs (https://testprovider.rdap.verisignlabs.com/) and CZ.NIC (https://www.mojeid.cz/).

Level of Maturity: This is a "proof of concept" research implementation.
Coverage: This implementation includes all of the features described in this specification.
Contact Information: Swapneel Sheth, ssheth@verisign.com

8.2. APNIC RDAP Service

Responsible Organization: Asia-Pacific Network Information Centre (APNIC)
Location: https://testrdap.apnic.net/
Description: This implementation includes support for POSIX extended regular expression RDAP queries for the domain and entity object classes. The data source is a subset of a snapshot of the production registry data.
Level of Maturity: This is a "proof of concept" research implementation.
Coverage: Aside from character class expressions, collating symbols, and equivalence class expressions, all of the features described in this specification are implemented.
Contact Information: Tom Harrison, tomh@apnic.net

9. Security Considerations

Security services for the operations specified in this document are described in RFC 7481 [RFC7481].

Search functionality typically requires more server resources (such as memory, CPU cycles, and network bandwidth) when compared to basic lookup functionality. This increases the risk of server resource exhaustion and subsequent denial of service due to abuse. This risk can be mitigated by developing and implementing controls to restrict search functionality to identified and authorized clients. If those clients behave badly, their search privileges can be suspended or revoked. Rate limiting as described in Section 5.5 of RFC 7480 [RFC7480] can also be used to control the rate of received search requests. Server operators can also reduce their risk by restricting the amount of information returned in response to a search request.

Search functionality also increases the privacy risk of disclosing object relationships that might not otherwise be obvious. For example, a search that returns IDN variants [RFC6927] that do not explicitly match a client-provided search pattern can disclose information about registered domain names that might not be otherwise

available. Implementers need to consider the policy and privacy implications of returning information that was not explicitly requested.

Note that there might not be a single, static information return policy that applies to all clients equally. Client identity and associated authorizations can be a relevant factor in determining how broad the response set will be for any particular query.

10. Acknowledgements

The author would like to acknowledge the following individuals for their contributions to the development of this document: TBD.

11. References

11.1. Normative References

[IEEE.1003.1_2013_EDITION]
IEEE, "Standard for Information TechnologyPortable Operating System Interface (POSIX(R)) Base Specifications, Issue 7", IEEE 1003.1, 2013 Edition, DOI 10.1109/ieeestd.2013.6506091, April 2013, <http://ieeexplore.ieee.org/servlet/opac?punumber=6506089>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <https://www.rfc-editor.org/info/rfc5890>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/info/rfc5952>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7480] Newton, A., Ellacott, B., and N. Kong, "HTTP Usage in the Registration Data Access Protocol (RDAP)", RFC 7480, DOI 10.17487/RFC7480, March 2015, <https://www.rfc-editor.org/info/rfc7480>.

[RFC7481] Hollenbeck, S. and N. Kong, "Security Services for the Registration Data Access Protocol (RDAP)", RFC 7481, DOI 10.17487/RFC7481, March 2015, <https://www.rfc-editor.org/info/rfc7481>.

[RFC7482] Newton, A. and S. Hollenbeck, "Registration Data Access Protocol (RDAP) Query Format", RFC 7482, DOI 10.17487/RFC7482, March 2015, <https://www.rfc-editor.org/info/rfc7482>.

[RFC7483] Newton, A. and S. Hollenbeck, "JSON Responses for the Registration Data Access Protocol (RDAP)", RFC 7483, DOI 10.17487/RFC7483, March 2015, <https://www.rfc-editor.org/info/rfc7483>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

11.2. Informative References

[GNU] gnu.org, "GNU Regular Expression Matching", <https://www.gnu.org/software/libc/manual/html_node/Regular-Expressions.html>.

[MYSQL] mysql.com, "MySQL Regular Expressions", <http://dev.mysql.com/doc/refman/5.7/en/regexp.html>.

[ORACLE] Oracle Corporation, "Oracle SQL and POSIX Regular Expression Standard", <https://docs.oracle.com/database/121/ADFNS/adfns_regexp.htm#ADFNS231>.

[PCRE] pcre.org, "Perl Compatible Regular Expressions", <http://www.pcre.org/>.

[PERLRE] perl.org, "Perl regular expressions", <http://perldoc.perl.org/perlre.html>.

[POSTGRES] postgresql.org, "PostgreSQL POSIX Regular Expressions", <https://www.postgresql.org/docs/9.3/static/functions-matching.html>.

[RDAPOPENID] ietf.org, "Federated Authentication for the Registration Data Access Protocol (RDAP) using OpenID Connect", <https://tools.ietf.org/html/draft-hollenbeck-regext-rdap-openid-01.txt>.

[RFC1166] Kirkpatrick, S., Stahl, M., and M. Recker, "Internet numbers", RFC 1166, DOI 10.17487/RFC1166, July 1990, <https://www.rfc-editor.org/info/rfc1166>.

[RFC6927] Levine, J. and P. Hoffman, "Variants in Second-Level Names Registered in Top-Level Domains", RFC 6927, DOI 10.17487/RFC6927, May 2013, <https://www.rfc-editor.org/info/rfc6927>.

Appendix A. Change Log

00: Initial version.
01: Renewed and moved invalid Normative References to Informative References
02: Specified use of percent encoding for reserved URL reserved characters in regular expressions and removed specification for base64url encoding for regular expressions
03: Added information related to implications of system locale on processing regular expression search. Also, updated the implementation status section with APNIC's information.
04: Keepalive Refresh and minor spelling fixes

Authors' Addresses

Andrew Fregly
Verisign Labs
12061 Bluemont Way
Reston, VA 20190
USA

Email: afregly@verisign.com
URI: http://www.verisignlabs.com/

Swapneel Sheth
Verisign Labs
12061 Bluemont Way
Reston, VA 20190
USA

Email: ssheth@verisign.com
URI: http://www.verisignlabs.com/

Scott Hollenbeck
Verisign Labs
12061 Bluemont Way
Reston, VA 20190
USA

Email: shollenbeck@verisign.com
URI: http://www.verisignlabs.com/