Network Working Group B. FiorelliInternet-Draft RIPEIntended status: Informational June 9, 2011Expires: December 11, 2011

The RIPE Database REST API draft-fiorelli-weirds-rws-00

Abstract This document describes the RIPE Database REST API. The purpose of this document is to facilitate discussion and serve as input into a standards process in this area, currently being discussed on the WHOIS-based Extensible Internet Registration Data Service (WEIRDS) mailing list (https://www.ietf.org/mailman/listinfo/weirds).

Table of Contents

1. Introduction

This document describes a pilot service for querying and displaying data in the RIPE database. The service is implemented using the HTTP protocol [RFC2616] and conforms to the architectural constraints of REST [REST]. In this document, we explain our reasons for new interfaces (and limits of existing ones) as well as high level functionalities of the API. We did not include technical details of our implementation, which can be found as a separate document and is available at: http://labs.ripe.net/Members/bfiorell/api-documentation.

1.1. Why REST?

The RIPE Database has a large number of power users: organisations and individuals that make or maintain software which depends on the RIPE Database. The way these specialized clients interact with the RIPE Database is dictated by historical and technical aspects of RPSL and the Whois protocol. RPSL ([RFC2622]) is the routing policy specification language, but for historical and convenience reasons it actually specifies much more than routing policies, there are many RFCs related to the RPSL (see [RFC2622], [RFC2650], [RFC4012]) and in practice the language describes more than 20 different types of RPSL objects, it is more a formalisation of the way policy records have been stored and exchanged in the existing Whois systems since to the late 80's instead of a high level domain language specification. The result is that the policy specification language and its extensions are tightly coupled with the way policies are stored in the existing Whois systems and vice versa, so any new system that is implementing the language ends up reproducing a Whois system. The qualiti

es of RPSL and Whois are well known. In this section we highlight their limits by describing the way a client would interact with the two Whois interfaces of query and update. First scenario, a client needs to extract sensible informati on from one or more Internet Registries like the RIPE Database, the steps are : prepare one or more queries and invoke a command line Whois client pr ogram or open a TCP socket connection to one or more Whois servers, par se and post-process the RPSL responses, facing many intricacies like continu ation lines, end of line comments, comma separatedFiorelli Expires December 11, 2011 [Page 3]
 val ues, attribute values that can reference to different types of RPSL objects depending on various conditions, etc., if object references need to be resol ved, extract referenced primary keys, execute other queries, parse again RPS L to identify the right referenced objects; this step can be recursive, merge results from multiple queries, maybe queries from multiple internet r egistries that may adopt different RPSL flavours, extract subsets of objects , by applying various criteria, extract subsets of attributes from set of ob jects, by applying various criteria. The other standard scenarios is that of update: in case of update or delete, fetch the object, this operation may require several of the steps described in the previous workflow, in ca se of a new object, build RPSL text, maybe starting converting business data into RPSL text, decorate the object with authorisation credentials, adding password hashes to the RPSL text or signing the RPSL text with a PGP ke y, send the text data via mail to the Mail Update interface or via HTTP post to the Syncupdates interface, in case of error, parse a human readable text response if specific error conditions must be handled in a special way . As shown above, such processes are modeled on a human centered workflow, th ey are not the optimal to build new services, extend existing ones or build too ls on top of them. Concretely, this means clients have to handle too much compl exity, many clients have problems at handling all the complexity of RPSL, man y are too expensive to be extended and difficult to maintain. RPSL and Whois ar e still invaluable but it’s evident that there is a need for simpler interfaces and machine ready data formats in order to simplify the development of service s and tools and increase the value of Registries by exposing new domain specifi c interfaces. The first step can be the identification of a layer of Service Provider Interfaces and a data schema simple enough to be agnostic ofFiorelli Expires December 11, 2011 [Page 4]
 the un derlying Registry implementation. The SPI would sit on top of a Registry Databa se allowing the composition of domain specific Services but also making real-ti me interoperation between Registries and services on multiple authoritative D atabases possible to achieve. With these forces and this vision the RIPE Databa se Group has been prototyping the RIPE Whois RESTful Query and CRUD API and the related XML schemas. The choice of REST has been quite obvious, HTTP is nowa days the most accessible protocol and it represents a good architectural bluepr int for stateless services, with its different HTTP methods, the resource loc ator protocol, the HTTPS, etc. For the representation schema we have designed a very relaxed attribute oriented XML Schema. We only apply a structural valida tion via XSD, after some testing we decided to remove any form of attribute o r type validation in order to reuse the same schema on different RPSL flavors. The services support representational styles JSON, HTML and plain text, all d erived via XSL transformation, with the latter two to showcase the transformati on powers of XML and also to demo resource navigation via lookup references on any HTML browser. Content negotiation can be done using HTTP headers or appendi ng a file extension to the request URL. The query services can be used on any RPSL based Whois server or mirror. For example in one request it is possible to execute the same Lookup or Search request on all the Regional Internet Regi stries and return all the responses as a unique set of resources. Similarly c an be done for the CRUD interfaces, building adapter modules for the different update mechanisms provided by different Registries given that they adopt the sa me set of resource types. Actually we implemented an adapter for the Whois Sync updates interface in the form of an HTTP proxy.2. Some features of the Query S

 Sin
gle resource Lookup Service, given a primary key a type and a registry it al
ways return one and only one object. It can also be used to identify resour
ces by URL bookmark. Resolution of referenced resources. Given a resource,
all the attribute values that represent references to other resource co
ntain an xlink anchor that can be followed to navigate and browse networks o
f resources. HATEOAS, the client can navigate through any network of resourc
es via xlinks without requiring any stateful information to be stored o
n the servers, this comes as a benefit of the two previous features. No
rmalisation of comma separated values, when they represent references to mul
tiple resources. Normalisation of continuation lines, end of line comments a
nd other RPSL intricacies. This is still a work in progress and still
not complete because is difficult to test all the side effects of this opera
tion. The only real solution would be to deprecate continuation lines in RP
SL and split them into multiple attribute value pairs. All the Query Se
rvices are available also on HTTPS.3. Some sample invocation of the query servic
es A Lookup Service URL for an organization object: http://apps.db.ripe.net/w
hois/lookup/ripe/organisation/ORG-BME1-RIPE A Lookup Service URL for a route ob
ject: http://apps.db.ripe.net/whois/lookup/ripe/route/193.6.23.0/24/AS2547 A
Search Service URL on multiple registries: http://apps.db.ripe.net/ whois/ se
arch.xml?flags=r&source=ripe&source=apnic&source=afrinic& query-string=AS25474.
Some details on the CRUD Services The CRUD Services are an attempt to split t
he single overloaded generic update interface provided by Mail Updates and Sync
updates into separate low level interfaces, each defining a simpler contract,
designed for programmatic use rather than for human interaction.Fiorelli
 For ex
ample the existing update interfaces always require clients to provide the enti
re RPSL resource in exactly the same text-format as it is stored in the Whois d
atabase, even if the requested operation is just a deletion. A client will hav
e to fetch the RPSL object from the RIPE Database doing an appropriate query, t
han extract ony the text blob that represent the resource to be deleted and fin
ally it will have to execute a request to the update service providing the en
tire text and adding delete attributes and authorisation credentials. The new
CRUD Services on the other hand provide a Delete interface that only requires
a primary key, a type, a registry identifier and one or more passwords. It is
the equivalent of a lookup but is executed with the HTTP Delete method. It is
exposed only on HTTPS. The client will just have to send a request like: HTTP D
ELETE: https: //lab.db.ripe.net/whois/delete/test/person/pp16-test?password=123
: The server will just respond with an HTTP Status code indicating success o
r failure, in case of failure different status codes will indicate different er
ror conditions. On top of Create, Update and Delete methods we also prototyped
some attribute modification services that in one only request can implement c
omplex update workflow. For example with one only HTTP request will be possible
to execute commands like: replace all the attributes of a given type with
a new set of attribute, remove all the attributes that have value matc
hing the given regular expression, add a new set of attributes after th
e Nth attribute of type X.5. Internationalization Considerations TBA6. IANA C
onsiderations TBAFiorelli Expires December 11, 2011
[Page 7]
 Secur
ity Considerations TBA.8. References8.1. Normative References [REST] Fi
elding, R. and R. Taylor, "Principled Design of the Modern Web Archi
tecture", ACM Transactions on Internet Technology Vol. 2, No. 2, May
2002. [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H.,
Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer P
rotocol -- HTTP/1.1", RFC 2616, June 1999. [RFC3986] Berners-Lee, T., Fielding
, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic S
yntax", STD 66, RFC 3986, January 2005. [RFC3987] Duerst, M. and
M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 39
87, January 2005.8.2. Informative References [RFC2622] Alaettinoglu, C., Vill

amizar, C., Gerich, E., Kessens, D., Meyer, D., Bates, T., Karrenberg, D., and M. Terpstra, "Routing Policy Specification Language (RPSL)", RFC 2622, June 1999. [RFC2650] Meyer, D., Schmitz, J., Orange, C., Prior, M., and C. Alaettinoglu, "Using RPSL in Practice", RFC 2650, August 1999. [RFC3707] Newton, A., "Cross Registry Internet Service Protocol (CRISP) Requirements", RFC 3707, February 2004. [RFC3912] Daigle, L., "WHOIS Protocol Specification", RFC 3912, September 2004. [RFC4012] Blunk, L., Damas, J., Parent, F., and A. Robachevsky, "Routing Policy Specification Language next generation (RPSLng)", RFC 4012, March 2005.Author’s Address Benedetto Fiorelli RIPE NCC P.O. Box 10096 Amsterdam 1001EB The Netherlands Phone: +31.20.535.4444 Email: bfiorell@ripe.net

Network Working Group
Internet-Draft
Intended status: Informational
Expires: February 10, 2012

M. Kucherawy
Cloudmark

August 9, 2011

Requirements For Internet Registry Services
draft-kucherawy-weirds-requirements-01

Abstract

This document enumerates a base set of requirements that should be included in any system that provides registration information for Internet entities, be they network assignments or domain name assignments. Some of these, in turn, will define requirements for registrars; this, however, is an issue outside of the scope of this document.

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 http://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 February 10, 2012.

Copyright Notice

Table of Contents

1. Introduction

The ubiquitous [WHOIS] service can be used today to query for domain name registration or network or subnetwork assignment information by the general public. It is however a very simple protocol, whose output is free-form and thus not amenable to machine parsing.

The CRISP working group created a workable and extensible standard for replacing WHOIS, called [IRIS]. Unfortunately, IRIS has seen little to no deployment for various reasons, mostly its complexity compared to WHOIS and some political and technical inertia.

Thus, this effort confronts anew the need for a better service than WHOIS provides, and also presents several working APIs for standardization to improve service to all constituents.

2. Document Series

This memo represents the introduction to a series of others that define the overall problem and some available solutions. The series is as follows:

1. RFCxxxx: Requirements for Internet Registry Services (this memo)

2. RFCxxxx+1: The ICANN Internet Registry Service API

3. RFCxxxx+2: The RIPE Internet Registry Service API

4. RFCxxxx+3: The ARIN Internet Registry Service API

5. RFCxxxx+4: RESTful WHOIS

The intent is to publish one of the API specifications as a standards track document, and the remainder (including this memo) as informational documents.

3. Terminology and Definitions

This section defines terms used in the rest of the document.

3.1. Keywords

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 [KEYWORDS]. In particular, since this is not a standards track document, these key

words are meant to describe requirements for those proposals for a WHOIS replacement that seek standards track status.

3.2. Incorporated Requirements

Many of the requirements distilled from the input provided by various communities in [CRISP] will apply to this effort as well. It is certainly the case that the research presented there should be considered prerequisite reading for this new work.

4. Requirements

This section enumerates the basic requirements of any WHOIS replacement system.

4.1. Clients

The client-side requirements are as follows:

1. To support internationalized values, a client SHOULD be able to handle replies that contain data that are not exclusively 7-bit clean.

2. A client SHOULD support caching of replies. It MAY apply its own default and MAY use a time-to-live provided as part of the reply.

3. A client SHOULD be able to handle a reply that is effectively a referral or redirect to another server.

4. A client MUST be able to decode a reply using a well-established generic encoding mechanism such as XML or JSON.

4.2. Servers

The server-side requirements are as follows:

1. A server MUST be able to return internationalized data.

2. A server MUST be able to return a result that indicates to a client that the answer sought is not available here, but can be found elsewhere (i.e., a referral mechanism). The DNS has well-established facilities for doing so and this effort might do well to consider those methods.

3. A server SHOULD be able to provide class-of-service facilities for different types of users. At least the following cases need to be considered:

anonymous: Users with no prior arrangement for access to the data; typically all available data will be provided in response to a query, but the query rate may be severely limited. No authentication is typically required.

security: Users that have an interest in a specific subset of a registration's data for the purpose of analysis and correlation while evaluating the trustworthiness of the source. Examples include email client evaluation, email content evaluation, web site security, etc. The subset will typically include creation/registration dates, assigned nameserver names and IP addresses, registrar ID and registrant ID. Users in this class would be required to authenticate in some way, but such clients would not typically be subjected to rate limiting given the prior arrangement.

law enforcement: Users with a bona fide interest in as much registration data, including change history, as is available. Typically, queries would be rare but have extremely high priority. These clients would definitely require authentication and probably also require encryption.

4. A server MUST reply in a univerally standard format; free-form replies MUST NOT be used, although the standard format may have provisions for some fields that are free-form within it. In particular:

   * All date and/or time fields MUST be formatted as per [DATETIME].

5. NOTE: The standard format is expected to be a significant portion of the work on the way to describing a new overall WHOIS specification. In any case, machine-parsability of replies is crucial to the success of this work.

6. A server MUST provide a minimum set of data about a given query. It is expected that this minimum set will be different for a network allocation registry than a domain name registry, however the following MUST be provided:

   * The creation date of the record

   * The date on which the record most recently changed owners/ registrants

   * For domain name registration records, the identifier of the registrar that created the record

   * For domain name registration records, the identifier of the registrant that created the record

   * For network registration records, the size of the assigned subnet in terms of a number of bits

7. A server MAY provide different output based on the nature of the client, where such can be definitively determined.

5. IANA Considerations

This memo presents no actions for IANA, though later memos in this series are likely to do so.

6. Security Considerations

This memo introduces an overall protocol model, but no implementation details. Specific security considerations of the various approaches presented in this document series will be described in those other documents.

7. Informative References

[CRISP] Newton, A., "Cross Registry Internet Service Protocol (CRISP) Requirements", RFC 3707, February 2004.

[DATETIME] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

[IRIS] Newton, A. and M. Sanz, "IRIS: The Internet Registry Information Service (IRIS) Core Protocol", RFC 3981, January 2005.

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[WHOIS] Daigle, L., "WHOIS Protocol Specification", RFC 3912, September 2004.

Appendix A. Public Discussion

Public discussion of this suite of memos takes place on the

weirds@ietf.org mailing list. See
https://www.ietf.org/mailman/listinfo/weirds.

Author's Address

Murray S. Kucherawy
Cloudmark
128 King St., 2nd Floor
San Francisco, CA 94107
USA

Phone: +1 415 946 3800
Email: msk@cloudmark.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 24, 2012

A. Newton
ARIN
K. Ranjbar
RIPE NCC
A. Servin
LACNIC

September 21, 2011

A Uniform RESTful URL Query Pattern for RIRs
draft-newton-et-al-weirds-rir-query-00

Abstract

This document describes uniform patterns for which to construct HTTP URLs that may be used to retreive information from Regional Internet Registries (RIRs) using "RESTful" web access patterns.

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 http://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 March 24, 2012.

Copyright Notice

Table of Contents

1. Introduction

The Regional Internet Registries (RIRs) have begun experimenting with RESTful web services for access to Whois data. This document presents uniform patterns which may be used to contruct URLs for accessing data from these RESTful web services.

The patterns described in this document purposefully do not encompass all of the methods employed in the Whois and RESTful web services of all of the RIRs. The intent of the patterns described here are to enable lookups of networks by IP address, autonomous system numbers by number, and reverse DNS meta-data reverse DNS domain labels. It is envisioned that each RIR will continue to maintain NICNAME/WHOIS and/or RESTful web services specific to their needs and those of their constituencies, and the information retreived through the patterns described here may reference such services.

Whois services, in general, are read-only services. Therefore URL patterns presented here are only applicable to the HTTP GET and HEAD methods.

This document does not describe the results or entities returned from issuing the described URLs with an HTTP GET. It is envisioned that other documents will describe these entities in various serialization formats, such as XML and JSON.

Additionally, resource management, provisioning and update functions are out of scope for this document. RIRs have various and divergent methods covering these functions, and it is unlikely a uniform approach for these functions will ever be possible.

And while HTTP contains mechanisms for servers to authenticate clients and clients to authenticate servers, from which authorization schemes may be built, both authentication of clients and servers and authorization for access to data are out-of-scope of this document. In general, these matters require "policy" and are not the domain of technical standards body.

2. Path Specification

The uniform patterns start with a base URL specified by each RIR or any other service provider offering this service. The base URL will be appended with resource type specific path segments. The base URL may contain its own path segments (e.g. http://example.com/... or http://example.com/restful-whois/...).

The resource type path segments are:

'ip' IP networks and associated data referenced using either an IPv4 or IPv6 address (i.e. not CIDR notation).

'autnum' Autonomous system registrations and associated data referenced using an AS Plain autonomous system number.

'rdns' Reverse DNS information and associated data referenced using a fully-qualified domain name.

2.1. IP Networks

Queries for information about IP networks are of the form /ip/XXX.XXX.XXX.XXX/... where the path segment following 'ip' is either an IPv4 or IPv6 address. While an IP address may fall into multiple IP networks in a hierarchy of networks, this query targets the "most-specific" or lowest IP network in a hierarchy.

Path segments following the IP address target specific information associated with the targetted IP network in the following way:

'registration' The query is for the network registration data.

'operator' The query is for data about the network operator of the IP network. The network operator is not always considered to be the end user or end site customer of the IP network, a distinction made in some cases. For example, a residential Internet installation may be assigned IP addresses, but the provider from whom they receive Internet access is considered the network operator. Another rule of thumb is that the network operator is the entity contacted to coordinate network issues and has published contact information for this purpose, and operator information can be further decomposed into operator contact information, which is returned with the 'operator' query when not specifically targetted (see below).

When no path segment follows the IP address, the semantics of the query are that both registration and operator information are to be returned.

The following example URL is a query for the IP network registrion information.

```
http://example.com/somepath/ip/192.0.2.0/registration
```

The following example URL is a query for the network operator information of the most specific network containing 192.0.2.0

```
http://example.com/somepath/ip/192.0.2.0/operator
```

And this is an example URL for both the registration and operator information of the most specific network containing 192.0.2.0

```
http://example.com/somepath/ip/192.0.2.0
```

The contact information of an operator maybe specifically targetted by following it with a 'contacts' path segment. And the type of contact information may be further targetted by following that path segment with a type. The types are:

- tech
- admin
- abuse
- noc

For example:

```
/ip/192.0.2.0/operator/contacts
```

returns all the contact information for the network operator of the most specific network containing IP address 192.0.2.0.

And this path targets only the abuse contacts of that network operator.

```
/ip/192.0.2.0/operator/contacts/abuse
```

2.2. Autonomous Systems

Queries for information regarding autonomous system number registrations are of the form /autnum/XXX/... where XXX is an autonomous system number. In some registries, registration of autonomous system numbers is done on an individual number basis, while other registries may register blocks of autonomous system numbers. The semantics of this query is such that if a number falls within a range of registered blocks, the target of the query is the block registration, and that individual number registrations are considered a block of numbers with a size of 1.

For example, to find information on autonomous system number 65551, the following path would be used:

```
/autnum/65551
```

The autnum path segment may be followed by a 'registration' or 'operator' path segment or no additional path segment, all of which follow the semantics above (Section 2.1).

2.3. Reverse DNS

Queries for reverse DNS information are of the form /rdns/XXXXXXXXX/... where XXXX is a fully-qualified domain name in either the in-addr.arpa or ip6.arpa zones.

For example, to find information on the zone serving the network 192.0.2/24, the following path would be used:

```
/rdns/2.0.192.in-addr.arpa
```

The rdns path segment may be follwed by a 'registration' or 'operator' path segment or no additional path segment, all of which follow the semantics in Section 2.1.

3. Response Formats

URLs may contain the 'format' query parameter. The value of the query parameter contains the MIME type of the desired format of the response. This parameter is used instead of the HTTP Accept header to defeat web caches and better support HTTP clients that do not support alteration of the Accept header.

The following is an example of a URL for AS 65551 requesting the response be in text/plain format:

```
http://example.com/autnum/65551?format=text%2Fplain
```

This document does not specify the contents of the responses. It is envisioned that multiple format types may be supported. Other documents will specify the contents of responses.

Authors' Addresses

Andrew Lee Newton
American Registry for Internet Numbers
3635 Concorde Parkway
Chantilly, VA 20151
US

Email: andy@arin.net
URI: http://www.arin.net

Kaveh Ranjbar
RIPE Network Coordination Centre
Singel 258
Amsterdam 1016AB
NL

Email: kranjbar@ripe.net
URI: http://www.ripe.net

Arturo L. Servin
Latin American and Caribbean Internet Address Registry
Rambla Republica de Mexico 6125
Montevideo 11300
UY

Email: aservin@lacnic.net
URI: http://www.lacnic.net

Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 5, 2011

A. Newton
American Registry for Internet Numbers

June 3, 2011

ARIN's RESTful Web Service for Whois Data
draft-newton-weirds-arin-whoisrws-00

Abstract

This document describes the RESTful Web Service for Whois data as implemented and fielded by the American Registry for Internet Numbers (ARIN). ARIN calls this service Whois-RWS.

The purpose of this document is to facilitate discussion and serve as input into a standards process in this area, currently being discussed on the WHOIS-based Extensible Internet Registration Data Service (WEIRDS) mailing list (https://www.ietf.org/mailman/listinfo/weirds).

Please excuse this very rough initial draft. It is roughly based on information currently published on ARIN's website. However, future revisions of this document are planned, including discussions on lessons learned by ARIN from the deployment of Whois-RWS and thoughts on a future, unified standard.

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 http://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 5, 2011.

Table of Contents

1. Requirements notation

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[RFC2119].

2. Background

ARIN offers public access to ARIN registration data via a number of services. Traditionally, these services are known in the industry as "Whois" in reference to the public data service of the ARPANET, precursor of today's modern Internet. Whois services are offered by all the Regional Internet Registries (RIRs), most Internet Routing Registries (IRRs) and most domain name registries and registrars.

Traditionally, Whois services have been offered using the NICNAME/WHOIS protocol as described by RFC 954 and RFC 3912. This protocol is a simple, text based TCP protocol registered with in the Internet Assigned Numbers Authority (IANA) for well-known port 43. RFC 3912, the most recent specification for the protocol, does not define either data types or data formats. As a consequence, Whois data varies from service provider to service provider and is far from ideal for programmatic consumption.

ARIN provides to the general public services for read-only access to ARIN's registration data. These services include a user-friendly web site (http://whois.arin.net), a RESTful Web Service, and a NICNAME/WHOIS port 43 service. For the purposes of programmatic consumption, ARIN recommends the use of the RESTful Web Service and strongly discourages the use of the NICNAME/WHOIS port 43 service.

This document describes the RESTful Web Service for ARIN's Whois data, which is known as Whois-RWS.

3. Data Model

ARIN's Whois-RWS data model is composed of six first order objects: networks, autonomous system numbers, delegations, organizations, points of contact, and customers. Each of these types, except delegations, is directly addressable in a Whois service via a handle (i.e. identifier). Within the Whois RESTful Web Service, these handles are part of URLs that may be used to identify objects in ARIN's Whois registration database by external, non-ARIN systems.

3.1. Networks and Autonomous System Numbers

Networks and Autonomous System Numbers (ASNs) are collectively referred to as "resources" in ARIN parlance (this should not be confused with the term "resources" in the context of RESTful Web Services and Resource Oriented Architectures). They are the pieces of information assigned or allocated to organizations for the coordinated administration and operation of the Internet.

Networks signify the allocation or assignment of IP address space and the contiguous IP CIDR blocks that make up that IP address space. Handles for IPv4 networks start with "NET-", and handles for IPv6 networks start with "NET6-".

Autonomous System Numbers (ASNs) are used for the proper routing of Internet packets. ARIN assigns ASNs in ranges, therefore a single ASN is part of an ASN range allocation. The handles for these registrations start with "AS" and are usually appended with the first number of the ASN range.

3.2. Delegations

Delegations contain the information necessary for Reverse DNS, including the associated nameservers, and DNS DS record information. Unlike the other first order objects, delegations do not have a Handle. Rather, they are directly addressable in a Whois service via a delegation name (i.e. 0.192.in-addr.arpa.).

3.3. Organizations and Customers

Organizations are the registrants of resources and have a direct relationship with ARIN. Organizations may be associated with many resources. Customers do not have a direct relationship with ARIN and, at present, may only be associated with one network registration.

Customer handles start with "C", while organization handles have no prefix.

3.4. Points of Contact

A Point of Contact (POC) is the registration of names, mailboxes, and/or phone numbers which fulfill technical or administrative functions on behalf of either an organization or a resource. POC handles are usually appended with "-ARIN" though there are a few exceptions.

3.5. General Query Behavior

For the first order objects addressable via handle, ARIN's Whois services also allow for searching against names and other appropriate fields contained within those objects. Unless otherwise specified, these searches are case insensitive. Because ARIN's networks are composed of multiple CIDR based network blocks, searches by IP address or CIDR notation may sometimes yield what may, at first blush, appear incorrect. For instance, a search for a particular CIDR block may yield a network that covers the given CIDR block and additional CIDR blocks composing the network, while a search by an IP address will always yield the network or networks in which the IP address falls.

4. Use of REST over HTTP

4.1. Data Formats and Versions

RESTful web services are not strictly tied to XML. ARIN's Whois RESTful Web Service offers data in XML, JSON, plain text, and HTML (actually, XHTML). However, ARIN's first order data format is XML as there are many format and validation tools readily available for it, and the other formats are offered on a best-effort basis.

If no desired format is specified, XML is the default format.

Specifying a desired data format may be accomplished in one of two ways: either using the HTTP Accept header or using "file extensions". A "file extension" appends a DOS like file extension to the path.

Though there only exists one version of this RESTful interface, it is possible that the "data model" of the structured data such as XML and JSON that is output by this service will need revision. Should it be possible for ARIN to provide a backward compatible version, the HTTP Accept header is the mechanism for specifying the desired version as well as the data format.

The MIME type used in the Accept header will follow the format of "application/arin.whoisrws-{version}+{format}". To use the latest version of this service you would simply use the default MIME types of "application/xml" or "application/json".

The following table lists the data types and their associated MIME types and file extensions.

Data Type	Current Version MIME Type	Version 1 MIME Type	File Extension
XML	application/xml	application/arin.whoisrws-v1+xml	xml
JSON	application/json	application/arin.whoisrws-v1+json	json
plain text	text/plain		txt

HTML	text/html		html

Table 1

4.2. Schemas

Relax NG stuff goes here.

Explanation of JSON and Badgerfish goes here.

4.3. Resource Oriented URLs

URLs point to resources. A typical URL might be http://whois.arin.net/rest/poc/ALN-ARIN, which is a URL pointing to the author of this document. Conceptually, this can be broken into a base URL and the resource identifier:

base URL: http://whois.arin.net/rest

resource: /poc/ALN-ARIN

The base URL is just where ARIN is hosting the service. The real interesting parts are the bits that identify the resources:

/poc this indicates the resource is a Point of Contact.

/ALN-ARIN this is the unique identifier for the Point of Contact, or the POC Handle.

The Whois-RWS data model has six types of addressable resources. These are reflected in Whois-RWS in the following way:

/poc point of contact

/org organization

/net a network

/asn autonomous system number(s)

/customer a customer of an organization

/rdns delegations in the reverse DNS

4.3.1. Resources Related to Resources

In the ARIN Whois data model, resources have relationships to other resources. Getting references to these resources can be accomplished by addressing the resource in question and applying a resource type qualifier.

Here is the list of relationships (where "XXXX" signifies a handle):

/poc/XXXX

- /orgs (i.e. /poc/XXXX/orgs) lists the organizations associated with a given POC
- /asns (i.e. /poc/XXXX/asns) lists the autonomous system numbers associated with a given POC
- /nets (i.e. /poc/XXXX/nets) lists the networks associated with a given POC

/org/XXXX

- /pocs (i.e. /org/XXXX/pocs) lists the points of contact associated with a given organization
- /asns (i.e. /org/XXXX/asns) lists the autonomous system numbers associated with a given organization
- /nets (i.e. /org/XXXX/nets) lists the networks associated with a given organization

/asn/XXXX

- /pocs (i.e. /asn/XXXX/pocs) lists the points of contact associated with a given autonomous system number

/net/XXXX

- /pocs (i.e. /net/XXXX/pocs) lists the points of contact associated with a given network
- /parent (i.e. /net/XXXX/parent) lists the parnet network of a given network
- /children (i.e. /net/XXXX/children) lists the child networks associated with a given network

/rdns (i.e. /net/XXXX/rdns) lists the reverse DNS delegations associated with a given network

/rdns/XXXX

/nets (i.e. /rdns/XXXX/nets) lists the networks associated with a given reverse DNS delegation

4.3.2. Unrelated Lists of Resources

Unrelated lists of resources may be addressed using URL matrix parameters. As an example, to find organizations with the name "ARIN", the /orgs list is appended with the matrix parameter name to form /orgs;name=ARIN (this is without the base URL).

Here is the list of resources and their supported matrix parameters:

/orgs

handle the handle of the organization

name the name of the organization

dba the "doing-business-as" name of the organization

/customers

handle the handle of the customer

name the name of the customer

/pocs

handle the handle of the POC

domain the domain name of an email address of the POC

first the first name of the POC

middle the middle name of the POC

last the last name of the POC

company the company name registered by the POC

city the city registered by the POC

/asns

handle the handle of the autonomous system number

name the name of the autonomous system number

/nets

handle the handle of the network

name the name of the network

4.3.3. IP Addresses

In the ARIN data model, an IP "network" is a set of associated IP address blocks and the information related to these IP address blocks and the set as a whole. A network can be composed of one IP address block or of multiple IP address blocks. Networks are also hierarchical, meaning that a network can have a parent and can have children. An IP address or range of IP addresses can therefore fall within the IP address block of multiple networks.

To get the networks that a particular IP address may fall within, one can use the /ip/XX.XX.XX.XX resource, where XX.XX.XX.XX signifies the IP address.

Networks may be looked up as well by supplying the full CIDR notation of a range. To use the full CIDR notation, the resource looks like /cidr/XX.XX.XX.XX/YY where XX.XX.XX.XX is the IP address prefix and YY is the CIDR length.

Resources relative to the networks may be fetched using the /less and /more path prefixes. /less will find the networks that are less specific in scope (or wider), and /more will find the networks that are more specific in scope (or narrower).

4.3.4. Aggregate Resources

Though Whois-RWS breaks down the ARIN data model into distinct resources, end users have been accustomed to seeing many related resources with one query over the NICNAME/WHOIS port 43 service. While this can be accomplished by any Whois-RWS clients with multiple queries, an aggregate resources can reduce the number of queries and make this use case easier on client implementers.

For resources under /org, /net, /asn, and /ip, the URL may be

appended with "/pft" to produce an aggregate resource which includes the resource being queried and related resources. If you can guess what /pft means, I'll buy you a beer.

4.3.5. Query Parameters

4.3.5.1. Showing More of Detail

By default, lists of resources only show references to the resources. However, lists can be expanded to show full detail by tacking on a showDetails=true URL query parameter (e.g.http://whois.arin.net/rest/org/ARIN/pocs?showDetails=true). This query parameter also forces a resource to list its related resources in line.

4.3.5.2. Showing Only Points of Contact

Some organizations have many, many associated resources. Having all those resources returned when gathering information about an organization maybe very undesirable. To allow getting all the POCs of an organization without taking the penalty of retrieving all the resources of the organization, the showPocs=true parameter may be used. This parameter is only recognized when an organization's information is referenced by handle.

4.3.5.3. Showing ARIN Resources

In the past, ARIN's NICNAME/WHOIS service yielded a "No Match" result for searches of IP address space unallocated by ARIN but within ARIN's IP address pool. This was somewhat confusing and could possibly mislead a person into thinking the address space is simply unallocated to any available address pool (in other words, still held by the IANA). This has been changed in ARIN's NICNAME/WHOIS service so it will return the appropriate ARIN network allocation for IP addresses unallocated by ARIN. By default, the RESTful service will also return the appropriate ARIN network allocation for IP addresses unallocated by ARIN to ARIN constituents.

If the behavior is desired, the showARIN=false query parameter may be used on IP and CIDR queries.

5. HTTP Caching

One of the advantages of a RESTful Web Service is that HTTP infrastructure may be employed to make it more reliable and/or robust. HTTP caching is one such tool to help with these goals. ARIN employs caching at multiple levels and customized to address the query pattern seen against ARIN's services.

If HTTP caching is be employed on the client side, there are a number of issues to consider to craft a custom caching solution.

First caching should be restricted to URLs that have the highest likelihood of remaining in the cache using the caches retention strategy. In almost any scenario, caching RESTful resources under /rest/ip will lead to a large dataset and a low cache hit ratio. However, it is likely acceptable to cache resources under /rest/org, /rest/poc, and /rest/net. It is recommended that cache statistics be available for proper cache tuning.

Second, many HTTP caches base the cache objects on URLs. However, the Accept header dictates the content of the object and may not be part of the cache key of cached objects. Therefore it is possible to get a cached object of one type when another type was requested. For instance, /rest/poc/KOSTE-ARIN might return XML when JSON was requested. To overcome this, it is advisable to use file extensions to specify data format as file extensions are part of the URL.

6. Security Considerations

Of course there are some... but I’m not gonna tell you what they are.

Seriously.... to be discussed in future revisions.

7. Lessons Learned

to be discussed in a future revision, the lessons we have learned from this

8. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

Author’s Address

Andrew Lee Newton
American Registry for Internet Numbers

Network Working Group S. Sheng
Internet-Draft F. Arias
Intended status: Informational ICANN
Expires: December 6, 2011 June 4, 2011

A RESTful Web Service for Domain Name Registration Data (RWS-DNRD)
draft-sheng-weirds-icann-rws-dnrd-00

Abstract

This document describes a pilot RESTful Web Service for querying Domain Name Registration Data (aka WHOIS data).

The purpose of this document is to facilitate discussion and serve as input into a standards process in this area, currently being discussed on the WHOIS-based Extensible Internet Registration Data Service (WEIRDS) mailing list (https://www.ietf.org/mailman/listinfo/weirds).

Table of Contents

1. Introduction

This document describes a pilot implemention by ICANN staff for querying domain name registration data through a RESTful Web-based Interface. The service is implemented using the HTTP protocol [RFC2616] and conforms to the architectural constraints of REST [REST].

1.1. Domain Name Registration Data

Domain Name Registration Data are data that a registrant provides when s/he acquires or is assigned a domain name. By contract with the Internet Corporation for Assigned Names and Numbers (hereinafter ICANN), domain name system registries - domain registries and registrars as defined by [RFC3707] - are to provide public access to some of these data both via the WHOIS protocol ([RFC3912]) as well as via a web interface.

1.2. REST and RESTful Web Service

REST stands for Representational State Transfer. It is a set of architectural constraints that is developed as an abstract model of the Web architecture. These constraints include: client-server model, stateless, cacheable, layered system, code on demand (optional), and uniform interface. REST was used to guide the redesign of the Hypertext Transfer Protocol (HTTP) and Uniform Resource Identifiers. It is widely regarded as the architecture of the Web today. Principles of REST have been used to design other protocols such as the ATOM publishing protocol.

A RESTful web service is a web service implemented using HTTP and the principles of REST. It is a collection of resources, with three defined aspects: 1) The "verbs" of the service are strictly those defined by the HTTP methods GET, PUT, POST, and DELETE, 2) The "verbs" are used to act upon resources, and 3) resources are addressable using URLs.

1.3. Why RESTful?

Compared to the WHOIS protocol as defined in RFC 3912, we felt the RESTful approach offers the following advantages:

Standardized output and error format: The base response output format is XML, which when paired with a well-defined schema would allow for automated processing.

Support for internationalisation: RWS has complete support for internationalised registration data, as well as IDNs with

U-labels, by using the XML data format, which uses Unicode.

Authentication and access control: HTTP, the transport for RWS, already supports authentication, and by means of using these capabilities, RWS makes technically possible to implement granular permissions over registration data if required.

Addressable Whois Service: RWS requires the use of a URI/URL standard structure for each object/resource. This has the additional benefit of providing a widely recognized manner to refer unambiguously to objects in Whois.

Increased Usability: Some of the inherent capabilities of the HTTP protocol (such as redirects) can be used to provide additional functionality such as automatic referrals to more specific WHOIS data sources without requiring specialized parsing by the client.

Authenticity of Origin: RWS provided over HTTPS offers confidence in the origin of the information.

Leverage existing infrastructure and expertise: RWS is HTTP-based and can be supported using popular web server infrastructures. Web administration is a skill-set and resource likely already commonplace inside registries and registrars. Similarly, RWS can benefit from existing technology to implement load-balance servers, cache answers to minimize network traffic, etc.

2. Terminology

For convenience, this implementation can be referred to as the "RESTful Web Service for Domain Name Registration Data" or "RWS - DNRD". The following terminology is used by this specification:

Domain Name Registration Data (aka WHOIS data) - refers to the data that registrants provide when registering a domain name. Part of these data are sometimes contractually required to be made publicly available, these includes the name of the registrant, name servers, the expiration date of the registration, etc. (see section 3.3 section of Registrar Accreditation Agreement).

URI - A Uniform Resource Identifier as defined in [RFC3986].

IRI - An internationalized Resource Identifier as defined in [RFC3987].

Resource - A network-accessible data object or service identified by an URI, as defined in [RFC2616]. In this context, resources

refers to the registration data objects.

Representation - An entity included with a request or response as defined in [RFC2616].

3. Implmentation Description

ICANN's RWS-DNRD specifies the URL structure, the methods to be used, the responses and its format, and the result codes.

As its name implies RWS-DNRD is Web-based, i.e., uses HTTP [RFC2616] as its transport. Given its RESTful nature it only uses the standard HTTP methods. And given it is read-only, it only usea the GET and HEAD methods.

3.1. The Request

The server accepts standard HTTP "GET" requests for the resources it serves. The client sends its request with the following URI structure.

3.1.1. URL structure

Meta Data - URL: / Display information about resources, URI scheme and allowed operations

Domain Name Request - /domain/<name>/ example:
http://whois.test/domain/example.test/

Contact Request - /contact/<id>/ example:
http://whois.test/conact/CID-4005/

Host Request - /host/<name>/ example:
http://whois.test/host/ns1.example.test/

3.2. The Response

The RWS-RDNRD server provide responses in XML format as specified below and can offer responses in other formats; currently only HTML and plain text. The client signals the preferred format using the standard HTTP "Accept:" header. The client can also signal the preferred format by adding a DOS-file-style extension to the resource. For example, "/xreg/rsrc1.xml". If the client specifies no preferred format, the server responds in XML. If the client signals one format with the HTTP "Accept:" header and another with the extension style, the server ignores the preference signaled with the former.

The pilot implementation currently support the following values for the Accept header: application/xml for XML, text/html for HTML, and text/plain for plain text.

The root element for a RWS-DNRD response is <rws>. This element contains one or more <result> elements that are explained in the following section.

Example of root element object:

```
<?xml version="1.0" encoding="UTF-8"?>
<rws:rws xmlns:rws="urn:ietf:params:xml:ns:rws-1.0"
    ...
    <rws:result>
    ...
    </rws:result>

    <rws:result>
    ...
    </rws:result>
    ...
</rws:rws>
```

3.2.1. Response for Domain Names

Example Query: http://whois.test/domain/example.test/

Response:

```
<?xml version="1.0" encoding="UTF-8"?>
<rws xmlns="urn:ietf:params:xml:ns:rws-1.0"
     xmlns:rws="urn:ietf:params:xml:ns:rws-1.0">
  <result>
    <rws:domain xmlns="urn:ietf:params:xml:ns:rwsDomain-1.0"
         xmlns:rwsDomain="urn:ietf:params:xml:ns:rwsDomain-1.0">
      <name>example.test</name>
      <roid>9690-TEST</roid>
      <status s="clientHold"/>
      <status s="clientRenewProhibited"/>
      <status s="clientUpdateProhibited"/>

      <registrant href="/contact/4447">4447</registrant>
      <contact type="admin" href="/contact/4447">4447</contact>
      <contact type="tech" href="/contact/4447">4447</contact>
      <ns>
        <hostObj href="/host/ns1.example.test">
            ns1.example.test
        </hostObj>
        <hostObj href="/host/ns2.example.test">
            ns2.example.test
        </hostObj>
        <hostObj href="/host/ns3.example.test">
            ns3.example.test
        </hostObj>
      </ns>
      <clID>793</clID>
      <clName>XYZ Corporation</clName>
      <crID>1289</crID>
      <crDate>1992-07-26T09:10:56Z</crDate>

      <exDate>2019-01-21T10:11:18Z</exDate>
    </rws:domain>
  </result>
</rws>
```

3.2.2. Response for Contacts

Example Query: http://whois.test/conact/4447/

Response:

```
<?xml version="1.0" encoding="UTF-8"?>
<rws xmlns="urn:ietf:params:xml:ns:rws-1.0"
     xmlns:rws="urn:ietf:params:xml:ns:rws-1.0">
  <result>
    <rws:contact xmlns="urn:ietf:params:xml:ns:rwsContact-1.0"
         xmlns:rwsContact="urn:ietf:params:xml:ns:rwsContact-1.0">
      <id>4447</id>
      <roid>4447-TEST</roid>
      <status s="clientDeleteProhibited"/>
      <status s="clientTransferProhibited"/>
      <status s="ok"/>

      <rwsContact:postalInfo
         xmlns="urn:ietf:params:xml:ns:contact-1.0" type="int">
        <name>Nova D Janes</name>
        <addr>
          <street>1755 XYZ Avenue</street>
          <city>ABC</city>
          <sp>PA</sp>
          <pc>15206</pc>
          <cc>US</cc>
        </addr>
      </rwsContact:postalInfo>
      <voice>+1.1234567890</voice>
      <fax>+1.1234567890</fax>
      <email>Nova.D.Janes@xyz.com</email>
      <clID>1289</clID>
      <crID>1289</crID>
      <crDate>1995-11-29T03:17:09Z</crDate>
      <trDate>1997-02-10T21:56:43Z</trDate>
    </rws:contact>
  </result>
</rws>
```

3.2.2.1. Response for Host Names

Example Query: http://whois.test/host/ns1.example.test/

Response:

```
<?xml version="1.0" encoding="UTF-8"?>
<rws xmlns="urn:ietf:params:xml:ns:rws-1.0"
    xmlns:rws="urn:ietf:params:xml:ns:rws-1.0">
  <result>
    <rws:host xmlns="urn:ietf:params:xml:ns:rwsHost-1.0">
      <name>ns1.example.test</name>
      <roid>8998-TEST</roid>
      <status s="ok"/>
      <status s="clientDeleteProhibited"/>
      <status s="serverDeleteProhibited"/>

      <addr ip="v4">188.17.219.3</addr>
      <clID>1289</clID>
      <crID>1289</crID>
      <crDate>1995-01-12T01:26:27Z</crDate>
      <upID>1289</upID>
      <upDate>2005-03-01T08:43:11Z</upDate>

    </rws:host>
  </result>
</rws>
```

4. Error Codes

In compliance with the REST paradigm any error information is returned in the form of a standard HTTP response with an HTTP status code describing the error and a text/plain body message describing the exception causing the error response. In this version we are using only standard HTTP codes (http://www.iana.org/assignments/http-status-codes).

We plan to define specialized error codes in the future.

5. Formal XML Syntax

The formal syntax presented here is a complete schema representation suitable for automated validation of an XML instance. We base our implementation on the Extension Provisioning Protocol (EPP). We reference and include the following EPP schemas:

[RFC5730] - Extensible Provisioning Protocol (EPP)

[RFC5731] - Extensible Provisioning Protocol (EPP) Domain Name Mapping

[RFC5732] - Extensible Provisioning Protocol (EPP) Host Mapping

[RFC5733] - Extensible Provisioning Protocol (EPP) Contact Mapping

5.1. Contact XML Schema

```
<schema targetNamespace="urn:ietf:params:xml:ns:rwsContact-1.0"
     elementFormDefault="qualified">

<!--
Import common element types.
-->
<import namespace="urn:ietf:params:xml:ns:eppcom-1.0"/>
<import namespace="urn:ietf:params:xml:ns:rws-1.0" />
<import namespace="urn:ietf:params:xml:ns:contact-1.0" />

<annotation>
    <documentation>
        RESTful Whois schema for contact.
    </documentation>
</annotation>

<!--
Child elements of Contact object
-->
<complexType name="rwsContactType">
 <sequence>
   <element name="id" type="eppcom:clIDType"/>
   <element name="roid" type="eppcom:roidType"/>
   <element name="status" type="contact:statusType" maxOccurs="7"/>
   <element name="postalInfo" type="contact:postalInfoType"
       maxOccurs="2"/>
   <element name="voice" type="contact:e164Type" minOccurs="0"/>
   <element name="fax" type="contact:e164Type" minOccurs="0"/>
   <element name="email" type="eppcom:minTokenType"/>
   <element name="clID" type="eppcom:clIDType"/>
   <element name="crID" type="eppcom:clIDType"/>
   <element name="crDate" type="dateTime"/>
   <element name="upID" type="eppcom:clIDType" minOccurs="0"/>
   <element name="upDate" type="dateTime" minOccurs="0"/>
   <element name="trDate" type="dateTime" minOccurs="0"/>
   <element name="extension" type="rws:extAnyType" minOccurs="0"/>
 </sequence>
</complexType>

<!--
End of schema.
-->
</schema>
```

5.2. Domain Name XML Schema

```
<schema targetNamespace="urn:ietf:params:xml:ns:rwsDomain-1.0"
     elementFormDefault="qualified">

<!--
Import common element types
-->
<import namespace="urn:ietf:params:xml:ns:eppcom-1.0" />
<import namespace="urn:ietf:params:xml:ns:domain-1.0" />
<import namespace="urn:ietf:params:xml:ns:contact-1.0" />
<import namespace="urn:ietf:params:xml:ns:rws-1.0" />

<annotation>
  <documentation>
     RESTful Whois schema for domains
  </documentation>
</annotation>

<!--
Child elements of a Domain object
-->
<complexType name="rwsDomainType">
  <sequence>
        <element name="name" type="eppcom:labelType"/>
        <element name="uName" type="eppcom:labelType" minOccurs="0"/>
        <element name="roid" type="eppcom:roidType"/>
        <element name="status" type="domain:statusType"
       maxOccurs="unbounded"/>
    <element name="registrant" type="rwsDomain:registrantType"
       minOccurs="0"/>
        <element name="contact" type="rwsDomain:contactType"
           minOccurs="0" maxOccurs="unbounded"/>
        <element name="ns" type="rwsDomain:nsType" minOccurs="0"/>
        <element name="host" type="rwsDomain:hostObjType"
           minOccurs="0" maxOccurs="unbounded"/>
        <element name="clID" type="eppcom:clIDType"/>
        <element name="clName" type="contact:postalLineType"/>
        <element name="crID" type="eppcom:clIDType" minOccurs="0"/>
        <element name="crDate" type="dateTime" minOccurs="0"/>
        <element name="upID" type="eppcom:clIDType" minOccurs="0"/>
        <element name="upDate" type="dateTime" minOccurs="0"/>
        <element name="exDate" type="dateTime" minOccurs="0"/>
        <element name="trDate" type="dateTime" minOccurs="0"/>
        <element name="extension" type="rws:extAnyType"
           minOccurs="0"/>
```

```
    </sequence>
  </complexType>

  <complexType name="registrantType">
      <simpleContent>
          <extension base="eppcom:clIDType">
              <attribute name="href" type="anyURI"/>
          </extension>
      </simpleContent>
  </complexType>

  <complexType name="contactType">
      <simpleContent>
          <extension base="eppcom:clIDType">
              <attribute name="type" type="domain:contactAttrType"/>
              <attribute name="href" type="anyURI"/>
          </extension>
      </simpleContent>
  </complexType>

  <complexType name="nsType">
      <choice>
          <element name="hostObj" type="rwsDomain:hostObjType"
              maxOccurs="unbounded"/>
          <element name="hostAttr" type="domain:hostAttrType"
              maxOccurs="unbounded"/>
      </choice>
  </complexType>

  <complexType name="hostObjType">
      <simpleContent>
          <extension base="eppcom:labelType">
              <attribute name="href" type="anyURI"/>
          </extension>
      </simpleContent>
  </complexType>

  <!--
    End of schema.
  -->
  </schema>
```

5.3. Host XML Schema

```
<schema targetNamespace="urn:ietf:params:xml:ns:rwsHost-1.0"
    elementFormDefault="qualified">

<!--
Import common element types
-->
<import namespace="urn:ietf:params:xml:ns:eppcom-1.0" />
<import namespace="urn:ietf:params:xml:ns:rws-1.0" />
<import namespace="urn:ietf:params:xml:ns:host-1.0" />

<annotation>
    <documentation>
       RESTful Whois schema for hosts
    </documentation>
</annotation>

<!--
Child elements of Host object
-->
<complexType name="rwsHostType">
  <sequence>
     <element name="name" type="eppcom:labelType"/>
     <element name="roid" type="eppcom:roidType"/>
     <element name="status" type="host:statusType" maxOccurs="7"/>
     <element name="addr" type="host:addrType" minOccurs="0"
        maxOccurs="unbounded"/>
     <element name="clID" type="eppcom:clIDType"/>
     <element name="crID" type="eppcom:clIDType"/>
     <element name="crDate" type="dateTime"/>
     <element name="upID" type="eppcom:clIDType" minOccurs="0"/>
     <element name="upDate" type="dateTime" minOccurs="0"/>
     <element name="trDate" type="dateTime" minOccurs="0"/>
     <element name="extension" type="rws:extAnyType" minOccurs="0"/>
  </sequence>
</complexType>

<!--
End of schema.
-->
</schema>
```

5.4. RWS XML Schema

```
   <schema targetNamespace="urn:ietf:params:xml:ns:rws-1.0"
       elementFormDefault="qualified">

   <!--
       Import common element types
   -->
   <import namespace="urn:ietf:params:xml:ns:eppcom-1.0" />
   <import namespace="urn:ietf:params:xml:ns:rwsContact-1.0" />
   <import namespace="urn:ietf:params:xml:ns:rwsHost-1.0" />
   <import namespace="urn:ietf:params:xml:ns:rwsDomain-1.0" />

   <annotation>
       <documentation>
               RESTful Whois schema
       </documentation>
   </annotation>

   <!--
       Root element
   -->
   <element name="rws" type="rws:rwsType"/>
   <attribute name="uri" type="anyURI"/>

   <!--
       RWS types
   -->
   <complexType name="rwsType">
     <sequence>
       <element name="result" type="rws:resultType" minOccurs="0"
           maxOccurs="unbounded"/>
     </sequence>
   </complexType>

   <complexType name="resultType">
     <choice>
       <element name="domain" type="rwsDomain:rwsDomainType"/>
       <element name="contact" type="rwsContact:rwsContactType"/>
       <element name="host" type="rwsHost:rwsHostType"/>
       <element name="extension" type="rws:extAnyType"/>
     </choice>
   </complexType>

   <!--
       Extension framework type
   -->
   <complexType name="extAnyType">
     <sequence>
       <any namespace="##other" maxOccurs="unbounded"/>
```

```
    </sequence>
  </complexType>

  <!--
      End of schema.
  -->
  </schema>
```

6. Internationalization Considerations

Information published in RWS is represented in XML, which provides native support for encoding information using the Unicode character set and its more compact representations including UTF-8. Conformant XML processors recognize both UTF-8 and UTF-16. Though XML includes provisions to identify and use other character encodings through use of an "encoding" attribute in an <?xml?> declaration, use of UTF-8 is preferred.

7. IANA Considerations

TBD

8. Security Considerations

8.1. URIs and IRIs

RWS-DNRD implementations use URIs and IRIs. See Section 7 of [RFC3986] and Section 8 of [RFC3987] for security considerations related to their handling and use.

9. Acknowledgments

The authors would like to acknowledge the following individuals for their input: Andrew Sullivan, and Dave Piscitello.

10. References

10.1. Normative References

[REST] Fielding, R. and R. Taylor, "Principled Design of the Modern Web Architecture", ACM Transactions on Internet Technology Vol. 2, No. 2, May 2002.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, January 2005.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, August 2009.

[RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, RFC 5731, August 2009.

[RFC5732] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Host Mapping", STD 69, RFC 5732, August 2009.

[RFC5733] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Contact Mapping", STD 69, RFC 5733, August 2009.

10.2. Informative References

[RFC3707] Newton, A., "Cross Registry Internet Service Protocol (CRISP) Requirements", RFC 3707, February 2004.

[RFC3912] Daigle, L., "WHOIS Protocol Specification", RFC 3912, September 2004.

Authors' Addresses

Steve Sheng
Internet Corporation for Assigned Names and Numbers
4676 Admiralty Way, Suite 330
Marina del Rey 90292
United States of America

Phone: +1.310.823.9358
Email: steve.sheng@icann.org

Francisco Arias
Internet Corporation for Assigned Names and Numbers
4676 Admiralty Way, Suite 330
Marina del Rey 90292
United States of America

Phone: +1.310.823.9358
Email: francisco.arias@icann.org