Network Working Group
Request for Comments: 2167
Obsoletes: RFC 1714
Category: Informational

S. Williamson
M. Kosters
D. Blacka
J. Singh
K. Zeilstra
Network Solutions, Inc.

June 1997

Referral Whois (RWhois) Protocol V1.5

Status of this Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Abstract

This memo describes Version 1.5 of the client/server interaction of RWhois. RWhois provides a distributed system for the discovery, retrieval, and maintenance of directory information. This system is primarily hierarchical by design. It allows for the deterministic routing of a query based on hierarchical tags, referring the user closer to the maintainer of the information. While RWhois can be considered a generic directory services protocol, it distinguishes itself from other protocols by providing an integrated, hierarchical architecture and query routing mechanism.

1. Introduction

Early in the development of the ARPANET, the SRI-NIC established a centralized Whois database that provided host and network information about the systems connected to the network and the electronic mail (email) addresses of the users on those systems [RFC 954]. The ARPANET experiment evolved into a global network, the Internet, with countless people and hundreds of thousands of end systems. The sheer size and effort needed to maintain a centralized database necessitates an alternate, decentralized approach to storing and retrieving this information.

The original Whois function was to be a central directory of resources and people on ARPANET. However, it could not adequately meet the needs of the expanded Internet. RWhois extends and enhances the Whois concept in a hierarchical and scaleable fashion. In accordance with this, RWhois focuses primarily on the distribution of "network objects", or the data representing Internet resources or people, and uses the inherently hierarchical nature of these network objects (domain names, Internet Protocol (IP) networks, email addresses) to more accurately discover the requested information.

RWhois synthesizes concepts from other, established Internet protocols. The RWhois protocol and architecture derive a great deal of structure from the Domain Name System (DNS) [RFC 1034] and borrow directory service concepts from other directory service efforts, primarily [X.500]. The protocol is also influenced by earlier established Internet protocols, such as the Simple Mail Transport Protocol (SMTP) [RFC 821].

This RWhois specification defines both a directory access protocol and a directory architecture. The directory access protocol specifically describes the syntax of the client/server interaction. It describes how an RWhois client can search for data on an RWhois server, or how the client can modify data on the server. It also describes how the server is to interpret input from the client, and how the client should interpret the results returned by the server. The architecture portion of this document describes the conceptual framework behind the RWhois protocol. It details the concepts upon which the protocol is based and describes its structural elements. The protocol implements the architecture.

This document uses language like SHOULD and SHALL that have special meaning as specified in "Key words for use in RFCs to Indicate Requirement Levels". [RFC2119]

2. Architecture

2.1 Overview

As a directory service, RWhois is a distributed database, where data is split across multiple servers to keep database sizes manageable. The architecture portion of this document details the concepts upon which the protocol is based and describes its structural elements. Specifically, the architecture is concerned with how the data is split across the different servers. The basis of this splitting is the lexically hierarchical label (or tag), which is a text string whose position in a hierarchy can be determined from the structure of the string itself.

All data can follow some sort of hierarchy, even if the hierarchy seems somewhat arbitrary. For example, person names can be arranged into hierarchical groups via geography. If all the people in particular towns are grouped into town groups, then all of the town groups can be grouped into state (or province) groups, and then all of the state groups can be grouped into a country group. Then, a particular name would belong in a town group, a state group, and a country group. However, just given a name, it would be impossible to determine where in the hierarchy it belongs. Therefore, a person name is not lexically hierarchical.

However, there are certain types of data whose position in the hierarchy can be determined by deciphering the data itself, for example, phone numbers. A phone number is grouped according to country code, area code, local exchange, and local extension. By looking at a phone number, it is possible to determine to which of all these groups the number belongs: 1-303-555-2367 is in country code 1, area code 303, local exchange 555, and has a local extension of 2367. Therefore, a phone number is lexically hierarchical.

On the Internet, two such types of data are widely used: domain names and IP networks. Domain names are organized via a label-dot system, reading from a more specific label to a more general label left to right; for example, war.west.netsol.com is a part of west.netsol.com, which is a part of netsol.com, which is a part of com. IP networks are also lexically hierarchical labels using the Classless Inter-Domain Routing (CIDR) notation, but their hierarchy is not easily determined with simple text manipulation; for example, 198.41.0.0/22 is a part of 198.41.0.0/16, which is a part of 198.40.0.0/15. Instead, an IP network’s hierarchy is determined by converting the network to binary notation and applying successively shorter bit masks.

It is important to note that, while very little real data is lexically hierarchical in nature, people often create label systems (or namespaces) to help manage the data and provide an element of uniqueness, for example, Social Security Numbers, ISBNs, or the Dewey Decimal System. RWhois leverages lexically hierarchical labels, domain names and IP networks, for its data splitting using the concepts of authority areas and referrals. An authority area is associated with an RWhois server and a lexically hierarchical label, which is considered to be its name. An authority area is a piece of the distributed database that speaks with authority about its assigned part of the hierarchy. All data associated with a particular lexically hierarchical tag should be located within that authority area's database. Authority areas are further explained in Section 2.4.

RWhois directs clients toward the appropriate authority area by generating referrals. Referrals are pointers to other servers that are presumed to be closer to the desired data. The client uses this referral to contact the next server and ask the same question. The next server may respond with data, an error, or another referral (or referrals). By following this chain of referrals, the client will eventually reach the server with the appropriate authority area. In the RWhois architecture, referrals are generated by identifying a lexically hierarchical label and deciphering the label to determine the next server. Referrals are further explained in Section 2.5.

When a number of RWhois servers containing authority areas are brought on line and informed about each other, they form an RWhois tree. The tree has a root authority area, which is the group that contains all other groups. The root authority area must keep pointers to the servers and authority areas that form the first level of the hierarchy. The authority areas in the first level of the hierarchy are then responsible for keeping pointers to the authority areas below them and for keeping a pointer to the root.

2.2 Design Philosophy

The design goals for the RWhois protocol are as follows.

* It should be a directory access protocol. The server should be able to access and update the data residing on it.
* It should facilitate query routing. An unresolved query should be redirected to a server that is presumed to be closer to the desired data.
* It should enable data replication. The server should be able to duplicate its data on another server.
* The server should be lightweight and delegate more functions to the client.

The concepts used to achieve these design goals are explained in the remaining document.

2.3 Schema Model

As a directory service, RWhois uses various database schema to store and represent data. Schema, in this document, has two definitions. First, it refers to the entire structure of a database, all the tables and fields forming a complete database. When schema is used in this context, it is called the "database schema". Database schema consists of attributes, classes, and objects. Schema may also refer to a single piece of the database, a single table with fields. When schema is used in this context, it is just called "schema" or it is preceded by the name of the particular piece: contact schema or domain schema, for example. In this usage, schema is equivalent to "class", defined below.

There is no standard database schema in the RWhois architecture. Each authority area is presumed to be able to define its own local schema. However, an authority area that is part of a larger RWhois tree is expected to have some part of its schema pertain to the lexically hierarchical label upon which the RWhois tree is based. An authority area schema may not change throughout much of an RWhois tree.

2.3.1 Attributes

An attribute is a named field and is the smallest typed unit in the database schema. It is equivalent to a relational database's field. An attribute is not considered to be data by itself; it is simply used to give data a type. When a piece of data has been typed by an attribute, it is typically referred to as a value and is represented as an attribute-value pair. The RWhois syntax for the attribute-value pair is to separate them with a colon, for example:

```
First-Name:Bill
```

Attributes have a number of properties, some mandated by the RWhois protocol and some that are implementation dependent. These properties are usually a reflection of the database system used by the server. The following is a list of the protocol-mandated properties and their descriptions.

Attribute	This is the name of the attribute.
Description	This is a natural language description of the attribute.

Type — This is a parameter that broadly indicates the use of the attribute to the protocol. There are three standard types: TEXT, ID, and SEE-ALSO. The default is TEXT, which indicates that the value is a text string. ID indicates that the attribute contains the ID of another RWhois object. This type of attribute is used for database normalization. SEE-ALSO indicates that the attribute contains a pointer (a Uniform Resource Identifier (URI)) to some other kind of external data; for example, a World Wide Web page or FTP site.

Format — This is an interpretable string that describes the acceptance format of the value. The server (and optionally the client) should match the value to the format string to determine if the value is acceptable. The format of this property is a keyword indicating the syntax of the format string, followed by a colon, followed by the format string itself. Currently, the only keyword recognized is "re" for POSIX.2 extended regular expressions.

Indexed — This is a true or false flag indicating that this attribute should be indexed (and therefore able to be searched).

Required — This is a true or false flag indicating that this attribute must have a value in an instance of the class.

Multi-Line — This is a true or false flag indicating that this attribute may have multiple instances in a class, but all of the instances are to be considered as multiple lines of the same attribute instance. This allows normal line terminators to terminate values.

Repeatable — This is a true or false flag indicating that there may be multiple instances of this attribute in a class and each instance is to be interpreted as a separate instance (in contrast to Multi-Line). This flag is mutually exclusive with Multi-Line: if Multi-Line is true, then Repeatable must be false and vice versa.

Primary This is a true or false flag that indicates that this attribute is a primary key. If more than one attribute in a class is marked as primary, then these attributes together form a single primary key. The primary key is intended to be used to force uniqueness among class instances. Therefore, there can be only one instance of a primary key in a database. The Primary flag implies that the attribute is also required.

Hierarchical This is a true or false flag that indicates that this attribute is lexically hierarchical.

Private This is a true or false flag that indicates whether or not this attribute is private (that is, publicly not viewable). It defaults to false. If it is true, then only the clients that satisfy the authentication/encryption requirements of a guardian (described below) are able to view the attribute-value pair.

2.3.2 Class

A class is a collection of attributes; it is a structure, not data. The concept is equivalent to that of a relational database table. It is also equivalent to the second definition of schema, above.

A class also has some properties that are sometimes referred to as its "meta" information. These properties are listed below.

Version This is a time/date stamp that is used to quickly detect when a class definition has been changed.

Description This is a natural language description of the class.

2.3.3 Object

An object is an instance of a class. It is data with a type of <class>.

2.3.4 Base Class

While RWhois does not have or advocate using a specific, standardized schema, it does impose a few requirements. It requires that all defined classes inherit attributes from a particular base class (or base schema). The RWhois specification does not require the actual implementation of inheritance. Instead, all classes must include the attributes defined in the base class.

The base class has the following attributes.

Class-Name This attribute contains the name of the class to which the object belongs. It is the type of the object itself. It is of type TEXT and is required.

Auth-Area This attribute contains the name of the authority area to which the object belongs. It, along with Class-Name, definitively defines the type of the object. It is of type TEXT and is required.

ID This attribute is a universal identifier for the object. It is formed by choosing a string that is unique within an authority area and appending the authority area to it, separating the local string from the authority area name with a period. The only restrictions on the local string are that it must be unique within the authority area and not contain the period character. This attribute is hierarchical in nature. It is always generated by the server (for example, during a register operation). It is of type TEXT and is required.

Updated This attribute is a time/date stamp that indicates the time of last modification of the object. It is both informational and a form of record locking. It prevents two clients from modifying the same object at the same time. It is of type TEXT and is required.

Guardian This attribute is a link to a guardian object (described below). Its value is the ID of a guardian object. It is of type ID and is optional. It is repeatable, since an object may have multiple guardians.

Private This attribute is a true or false flag that indicates whether or not an object is private (that is, publicly not viewable). It defaults to false. If it is true, then only the clients that satisfy the authentication/encryption requirements of one of the object’s guardians are able to view the object. If the object is publicly viewable, then the Private attribute property of each of its attributes still applies.

TTL — This attribute is the "time-to-live" of a given object. It is included only if an object has a different time-to-live than the default given in the Start of Authority information. Its value is specified in seconds. It is of type TEXT and is optional.

The RWhois specification defines two standard classes that should be included in all implementations: the referral and guardian classes.

2.3.5 Referral Class

The referral class is defined to hold referral information (typically for link referrals). It consists of attributes defined as part of the base class, the protocol-specific attributes described below, and any installation-specific attributes.

Referred-Auth-Area — This attribute contains the name of the authority area to which the referral points. It is used as a search key during the query routing. It is of type TEXT and is required. It is repeatable, since referrals can point to servers hosting more than one authority area.

Referral — This attribute contains the referral itself. It is an RWhois URL. It is of type TEXT and is required. It is repeatable, since more than one server can host a Referred-Auth-Area.

2.3.6 Guardian Class

The guardian class is defined to hold security information. The fundamental concept behind the guardian class is that an object (or another structure) is "guarded" by containing a pointer to a guardian object [Guardian]. To modify, delete, or possibly view the guarded object, the authentication (or encryption, or both) scheme must be satisfied. Guardians are intended to not have rank: if an object is guarded by more than one guardian object, satisfying any one of those guardians is sufficient. A guardian object that does not have any Guardian attribute linking it to other guardians guards itself. That is, the authentication scheme in the guardian object itself must be satisfied to modify, delete, or possibly view it.

Guardian objects are typically linked to actual database objects with the Guardian attribute found in the base class. However, a guardian may also be linked to an entire authority area, in which case the guardian becomes implicitly linked to all of the objects contained within the authority area.

The guardian class consists of the base class, the protocol-specific attributes described below, and any installation-specific attributes.

Guard-Scheme This attribute contains a keyword indicating the authentication methodology. Its value must be understood by both the client and server, and its value dictates the contents of the Guard-Info attribute. It is of type TEXT and is required.

Guard-Info This attribute contains that data that is used by the Guard-Scheme to verify the authentication. Its actual format is dictated by the Guard-Scheme, for example, it could contain a password or Pretty Good Privacy (PGP) public key id [RFC 1991]. For security reasons, it should not be displayed, and its Private attribute property should be set to true. It is of type TEXT and is required.

2.4 Authority Areas

The concept of authority areas is pivotal to the RWhois architecture. When an RWhois tree is created for a particular lexically hierarchical namespace, the different pieces of the hierarchy are mapped to authority areas. The most important concept behind an authority area is the ability for a portion of the RWhois tree to definitively control that portion of the hierarchy. This means that an authority area is able to state whether or not a hierarchical tag is in the whole RWhois tree. It does this either by returning the object containing this tag, returning a referral to a sub-authority area, or returning a response indicating that no objects were found.

This structure enables efficient routing of queries based on the hierarchical label to the piece of the hierarchy responsible for it. For example, in the domain name namespace as served by RWhois, the root of the tree would be an authority area named ".", which would delegate a "us" sub-authority area, which would delegate "va", "co", "md", and "ca" authority areas, and so forth. When the server with the "va.us" authority area is asked about "loudoun.va.us", it will be able to authoritatively state that either no "loudoun.va.us" exists or it will provide an object for or a referral to "loudoun.va.us". Further, if the server is asked about "howard.md.us", it cannot answer authoritatively, so it must provide a referral to its hierarchical parent ("us" or the root).

This use of authority area strongly indicates where data should be stored within an RWhois tree. Because RWhois uses a specific query routing model, data needs to be placed under the proper authority area. It is certainly possible to place a piece of data under the

wrong authority area, for example, putting an object for "howard.md.us" under the "va.us" authority area. In such cases, the data is considered to be misplaced and unable to be found within the RWhois tree. However, while data should be placed under the lowest (most specific) authority area, it is also possible that it could be placed in a higher (least specific) authority area, for example, putting an object for "loudoun.va.us" under the "us" authority. This may be acceptable since, in most cases, the data would be able to be found.

In addition to controlling a part of an RWhois hierarchy, an authority area is considered to be autonomous. Each authority area is treated as a separate database by the protocol. However, it is recommended that an authority area share some core schema with the rest of the RWhois tree for interoperability reasons. Each authority area, however, is not bound by the database schema of its hierarchical parent or by any of its sub-authority areas.

2.5 Query Routing

RWhois is not only a directory access protocol but it can also route queries. Routing a query involves redirecting the query to another server that is presumed to be closer to the desired data. To route a query, the server first determines the location of the next server. It then either forwards the query to that server and returns the result to the client or returns the location of that server to the client. The location of the server must contain its host name (or IP address), port number, and authority area.

The location of the server to which a query is routed is called a referral. There are two types of referrals: punt and link referrals. A punt referral is a pointer to a server that is further up an RWhois tree, and a link referral is a pointer to a server that is further down the tree. For example, in Figure 1, when the server for the "va.us" authority area routes a query up to the server for the "us" authority area, it generates a punt referral. Alternatively, when it routes a query down to the server for the "loudon.va.us" authority area, it generates a link referral.

Query routing depends on whether or not the search value in a query is lexically hierarchical. If the search value is hierarchical, the server can generate punt or link referrals using the association of authority areas with lexically hierarchical labels. Otherwise, the server may send the query to a special index server that gathers the indexing information for both hierarchical and non-hierarchical data from the directory servers and returns referrals to these servers [CIP]. If the server receives one or more referrals from the index server, it should return them to the client.

It is important to note that the server may route a query whether it could resolve the query or not. Even if a query has been resolved locally, the server may also return referrals to the client by sending the query to the index server. For example, if the server for the "com" authority area receives the "domain Org-Name=IBM" query, it may return all the domain objects for IBM within the "com" authority area. In addition, it may also return referrals to the server for the "nl" authority area if that server contains domain objects for IBM in the Netherlands and has fed the corresponding indexing information to the index server. This way the client can get back information for both "ibm.com" and "ibm.nl" domains.

2.5.1 Query Routing Rules

An RWhois server routes a query based on certain rules. The objective is to determine the location of a server to which to route the query. A query may contain one or more query terms. The query routing rules are applied on each query term until a referral is found. The rules are listed below.

* Is the search value in the query term hierarchical? If not, go to the next query term.
* Parse the hierarchical portion of the search value. Is it is within one of the authority areas? If not, go to the next query term.
* Does the found authority area have any referral objects (instances of the referral class)? If not, return the "230 No objects found" error to the client.
* Is the hierarchical portion of the search value within the Referred-Auth-Area attribute of one of the referral objects? If it is, return the value of the Referral attribute of the found referral object as a link referral to the client.
* Are the search values of some of the query terms hierarchical but not within any of the authority areas? If they are, return a punt referral to the client.
* Are the search values of all the query terms non-hierarchical? If they are, send the query to a special index server that gathers the indexing information for both hierarchical and non-hierarchical data from the directory servers and returns referrals to these servers. If the server receives one or more referrals from the index server, return them to the client.

Note that there can be more than one referral returned to the client. These referrals may point to servers serving different authority areas. The client may follow them in any order.

The pseudo code for the above rules is:

```
for each query term in the query
 if the search value in the query term is hierarchical
  if the search value is within one of the authority areas
    if the search value is within one of the referred authority areas
     the server sends link referral(s)
    else
     the server sends a "230 No objects found" error
    endif
  endif
 endif
endfor

if the search values of some of the query terms are hierarchical but
  not within any of the authority areas
 the server sends Punt referral(s)
endif

if the search values of all the query terms are non-hierarchical
 the server sends Referral(s) from an index server
endif
```

2.6 Data Replication

An RWhois server can replicate (duplicate) data from another RWhois server on a per-authority area basis. Data replication makes the RWhois service more reliable. Further, it increases throughput by distributing queries to more than one server.

There can be two types of servers serving an authority area: a master server and a slave server. A master server is where data is registered for an authority area. It answers authoritatively to queries in that authority area. There must be one and only one master server for an authority area. A master server is also called a primary server.

A slave server is where data is replicated from the master server for an authority area. It also answers authoritatively to queries in that authority area. There may be one or more slave servers for an authority area. A slave server is also called a secondary server. Note that a slave server must not register data for an authority area.

It is recommended that the master and slave servers for an authority area be geographically separate. Therefore, network unreachability at one site will not completely shut down the RWhois service for that authority area.

2.6.1 Data to Replicate

In RWhois, data is replicated on a per-authority area basis. The smallest type of data a slave server can replicate is an attribute of a class. Therefore, a slave server can replicate data for all the classes, some classes, or some attributes of some classes.

The amount of data a slave server can replicate each time is either all of the data or the data that has changed since the last replication. The process of replicating all of the data is called complete replication. The process of replicating the data that has changed since the last replication is called incremental replication.

2.6.2 Start Of Authority Variables

Each authority area has some administrative variables, defined at the master server, to control data replication. These variables are called the Start Of Authority (SOA) variables. They are listed below.

Serial-Number	This is the serial number of the data in an authority area. The master server should update this variable whenever the data in the authority area is changed. Its value is a time/date stamp.
Refresh-Interval	This is the time interval before a slave server checks for complete replication. Its value is specified in seconds.
Increment-Interval	This is the time interval before a slave server checks for incremental replication. Its value is specified in seconds.
Retry-Interval	This is the time interval before a slave server tries again to connect to a master server that appears to be out-of-service. Its value is specified in seconds.
Time-To-Live	This is the default time to live for the data in an authority area at a slave server. The slave server should not answer authoritatively to queries for such stale data. Its value is specified in seconds.
Admin-Contact	This is the email address of an individual or a role account responsible for the data integrity in an authority area at the master server.

Tech-Contact	This is the email address of an individual or a role account responsible for the operation of the master server for an authority area.
Hostmaster	This is the email address of an individual or a role account to whom email messages to update the data in an authority area at the master server are sent.
Primary-Server	This is the location of the master server for an authority area. Its value must contain both the host name (or IP address) and port number of the master server.

3. Protocol

3.1 Overview

The above sections describe the directory service architecture based on the RWhois protocol. The remaining sections describe the syntax of the protocol; the sequence and syntax of the information exchanged between a server and a client. There are five types of information that may be exchanged during a client/server session: directive, response, query, result, and info.

3.1.1 Directive

A directive is a command that a client sends to a server to set a control parameter for the session, get the meta-information (class definitions and SOA information) about an authority area, or get the data in an authority area. The first character of a directive must be a "-". The server must support the "-rwhois" directive; all other directives are optional. The server must indicate in the banner which directives are implemented (see Section 3.1.9).

3.1.2 Response

A response is the information that a server returns to a client for a directive. It is comprised of one or more lines, and the last line always indicates the success or failure of the directive. The first character of each response line must be a "%". If a server runs a directive successfully, the last response line must be "%ok". Otherwise, it must be "%error <error-code> <error-text>". A line with the string "%ok" or "%error" in the first position must occur only once in a server response and must always be the last line. The server may send the "%info" response for special messages.

A client must understand the "%ok", "%error", and "%info" responses. The client must also understand directive specific responses, if it uses the related directives to communicate with the server. For example, if the client sends the "-schema" directive to the server, the client must understand the "%schema" response.

3.1.3 Query

A query is a command that a client sends to a server to access the data in an authority area. The first character of a query must not be a "-", since the server checks the first character of each command from a client to determine whether it is a directive or a query.

3.1.4 Result

A result is the information that a server returns to a client for a query. It can be either the accessed data or referrals to other servers. It is comprised of one or more lines, and the last line always indicates the success or failure of the query. If a server returns either data or referrals for a query, the last result line must be "%ok". Otherwise, it must be "%error <error-code> <error-text>".

3.1.5 Info

An info message contains miscellaneous information that a server sends to a client. The server may use it to send special messages, for example a "message of the day" (MOTD), to the client. The first info line must be "%info on", and the last info line must be "%info off".

3.1.6 Client/Server Session

A typical RWhois client/server session has the following sequence of messages.

* The client connects to the server.
* The server returns a banner identifying its protocol versions and capabilities.
* The client sends one or more directives to the server.
* The server returns the response to each directive.
* The client finally sends a query to the server.
* The server returns the query results.
* The server closes the connection, unless the client has directed it not to close the connection.

3.1.7 Examples

This section gives some common examples of the client/server interaction. The notation in the examples uses a prefix to indicate from where the information comes. A "C" indicates that the client sends the data to the server. An "S" indicates that the server sends the data to the client. The line is a comment when "#" is used. The space after the prefix is not part of the data.

The following example illustrates a successful query.

```
# The client connects to the server.
# The server returns a banner identifying its protocol versions and
# capabilities.
S %rwhois V-1.5:00ffff:00 master.rwhois.net (Network Solutions V-1.5)
# The client sends a directive to limit the number of search hits
# to 20.
C -limit 20
# The server returns a successful response.
S %ok
# The client sends a query to search for rwhois.net domain.
C domain rwhois.net
# The server returns the data for rwhois.net domain.
S domain:ID:dom-1.rwhois.net
S domain:Auth-Area:rwhois.net
S domain:Class-Name:domain
S domain:Updated:19970107201111000
S domain:Domain:rwhois.net
S domain:Server;I:hst-1.rwhois.net
S domain:Server;I:hst-2.rwhois.net
S
S %ok
# The server closes the connection.
```

The following example illustrates the link and punt referrals.

```
# The client connects to the server.
# The server returns a banner identifying its protocol versions and
# capabilities.
S %rwhois V-1.5:00ffff:00 master.rwhois.net (Network Solutions V-1.5)
# The client sends a directive to hold the connection until it sends
# a directive to close the connection.
C -holdconnect on
# The server returns a successful response.
S %ok
# The client sends a query to search for a.b.rwhois.net domain.
C domain a.b.rwhois.net
# The server returns a link referral to a server serving the
```

```
   # b.rwhois.net authority area.
   S %referral rwhois://master.b.rwhois.net:4321/auth-area=b.rwhois.net
   S %ok
   # The client sends a query to search for internic.net domain.
   C domain internic.net
   # The server returns a punt referral to a server serving the root
   # authority area.
   S %referral rwhois://rs.internic.net:4321/auth-area=.
   S %ok
   # The client sends a directive to close the connection.
   C -quit
   S %ok
   # The server closes the connection.
```

The following example illustrates a query error.

```
   # The client connects to the server.
   # The server returns a banner identifying its protocol versions and
   # capabilities.
   S %rwhois V-1.5:00ffff:00 master.rwhois.net (Network Solutions V-1.5)
   # The client sends a query to search for c.rwhois.net domain.
   C domain c.rwhois.net
   # The server returns an error, since neither data nor referrals for
   # c.rwhois.net domain are found within the rwhois.net authority area.
   S %error 230 No objects found
   # The server closes the connection.
```

3.1.8 Notation

The following sections use the Augmented Backus-Naur Form (ABNF) notation to describe the syntax of the protocol. For further information, see Section 2 of [RFC822]. The notation in the examples uses a prefix to indicate from where the information comes. A "C" indicates that the client sends the data to the server. An "S" indicates that the server sends the data to the client. The line is a comment when "#" is used. The space after the prefix is not part of the data.

3.1.9 General ABNF definitions

Lexical Tokens

```
   alpha = "a".."z" / "A".."Z"
   digit = "0".."9"
   hex-digit = digit / "a.."f" / "A".. "F"
   id-char = alpha / digit / "_" / "-"
   any-char = <ASCII 1..255,
               except LF (linefeed) and CR (carriage return)>
```

```
dns-char = alpha / digit / "-"
email-char = <see [RFC 822]>
space = " "
tab = <ASCII TAB (tab)>
lf = <ASCII LF (linefeed)>
cr = <ASCII CR (carriage return)>
crlf = cr lf
```

Grammar

```
year = 4digit
month = 2digit
day = 2digit
hour = 2digit
minute = 2digit
second = 2digit
milli-second = 3digit
host-name = dns-char *(dns-char / ".")
ip-address = 1*3digit "." 1*3digit "." 1*3digit "." 1*3digit
email = 1*email-char "@" host-name
authority-area = (dns-char / ".") *(dns-char / "." / "/")
object-id = 1*id-char "." authority-area
host-port = (host-name / ip-address) ":" 1*5digit
class-name = 1*id-char
attribute-name = 1*id-char
attribute-value = 1*any-char
time-stamp = year month day hour minute second milli-second
on-off = "on" / "off"
```

Note that the time-stamp must be in the Greenwich Mean Time (GMT) time zone. Also note that since in the above any-char is 1..255 ASCII that the RWhois protocol is an 8 bit protocol.

Response

The general response for every directive and query is either "%ok" or "%error". In addition, a "%info" response may be sent.

```
response = ok-response crlf / error-response crlf / info-response
ok-response = "%ok"
error-response = "%error" space error-code space error-text
error-code = 3digit
error-text = 1*any-char
info-response = "%info" space "on" crlf *(*any-char crlf) "%info"
        space "off" crlf
```

Banner

The server must send a banner to the client when the connection is opened. The banner contains the version(s) of the protocol the server supports and a capability ID of encoded bit flags that indicates which directives are implemented. If the server supports more than one version of the protocol, the lowest-numbered version must be specified first. The bits in extra-id are reserved for future use. The end of the banner should contain a free-form string indicating the name of the server implementation. A server must support at least one version of the protocol, and may accept more versions for compatibility reasons.

```
rwhois-banner = "%rwhois" space version-list space host-name
      [space implementation] crlf
version-list = version *("," version)
version = version-number [":" capability-id]
        / "V-1.5" ":" capability-id
version-number = "V-" 1*digit "." 1*digit
capability-id = response-id ":" extra-id
response-id = 6hex-digit
extra-id = 2hex-digit
implementation = 1*any-char
```

Protocol

The entire RWhois protocol can be defined as a series of directives, responses, queries, and results.

```
rwhois-protocol = client-sends / server-returns
client-sends = *(directives / rwhois-query)
server-returns = *(responses / rwhois-query-result)
```

3.2 Required Directives

The server must implement the following directives.

3.2.1 rwhois

Description

The "-rwhois" directive may be issued by the client at the start of every session . It tells the server which version of the protocol the client can handle. The server must respond with a banner containing the protocol version and directives it implements. This banner is the same banner that is sent by the server when the connection is opened, except that the server must indicate only one version number. The banner issued when opening a connection may contain more than one version number. The directive flags are encoded into three octets, which are described in Appendix D.

ABNF

```
rwhois-dir = "-rwhois" space version-number [space implementation]
             crlf
rwhois-response = "%rwhois" space version space host-name
        [space implementation] crlf
```

Errors

300 Not compatible with version
338 Invalid directive syntax

Examples

```
# When a connection is opened, the server issues the banner.
S %rwhois V-1.0,V-1.5:00ffff:00 rs.internic.net (NSI Server 1.5.4)
# The client sends the rwhois directive.
C -rwhois V-1.5 NSI Client 1.2.3
S %rwhois V-1.5:00ffff:00 rs.internic.net (NSI Server 1.5.4)
S %ok
```

3.3 Optional Directives

The server should implement the following directives.

3.3.1 class

Description

The "-class" directive can be used by the client to get the meta-information for one or more classes in an authority area. The response must contain the description and version number of each specified class and may be expanded in the future with additional attributes. When no class name is given, the server must return the meta-information for all the classes in the authority area. Every class record must end with an empty "%class" line.

ABNF

```
class-dir = "-class" space authority-area *(space class-name) crlf
class-response = *class-record response
class-record = *class-line "%class" crlf
class-line = "%class" space class-name ":" "description" ":"
             1*any-char crlf
   / "%class" space class-name ":" "version" ":" time-stamp crlf
   / "%class" space class-name ":" meta-field ":" meta-value crlf
meta-field = 1*id-char
meta-value = 1*any-char
```

The following fields are required.

meta-field	meta-value	Description
description	1*any-char	Class description.
version	time-stamp	Time/date stamp indicating version of class, must be updated after class definition is changed.

Errors

```
338 Invalid directive syntax
340 Invalid authority area
341 Invalid class
400 Directive not available
401 Not authorized for directive
```

Examples

```
C -class rwhois.net domain host
S %class domain:description:Domain information
S %class domain:version:19970103101232000
S %class
```

```
S %class host:description:Host information
S %class host:version:19970214213241000
S %class
S %ok
```

3.3.2 directive

Description

The "-directive" directive can be used by the client to get information about the directives that the server supports. The response must contain the name and description of each specified directive and may be expanded in the future with additional attributes. When no directive name is given, the server must return information about all the directives. Every directive record must end with an empty "%directive" line.

ABNF

```
directive-dir = "-directive" *(space directive-name) crlf
directive-name = 1*id-char
directive-response = *directive-record response
directive-record = "%directive" space "directive" ":" directive-name
                   crlf *directive-line "%directive" crlf
directive-line = "%directive" space "description" ":" 1*any-char crlf
        / "%directive" space attribute-name ":" attribute-value crlf
```

Errors

```
338 Invalid directive syntax
400 Directive not available
401 Not authorized for directive
```

Examples

Without parameters:

```
C -directive
S %directive directive:rwhois
S %directive description:RWhois directive
S %directive
S %directive directive:quit
S %directive description:Quit connection
S %directive
S %ok
```

With parameters:

```
C -directive quit
S %directive directive:quit
S %directive description:Quit connection
S %directive
S %ok
```

3.3.3 display

Description

By default, the server uses the dump format for the output of a query result. The output format can be changed with the "-display" directive. When no parameter is given, the server must list all the display formats it supports. Every display record must end with an empty "%display" line.

Currently, only the dump format is standard and must be supported by the server. Other output formats may be added in the future. See Section 3.4 for the definition of the dump format.

ABNF

```
display-dir = "-display" crlf
    / "-display" space display-name crlf
display-name = 1*id-char
display-response = *(display-record) response
display-record = "%display" space "name" ":" display-name crlf
*display-line "%display" crlf
display-line = "%display" space attribute-name ":"
               attribute-value crlf
```

Errors

```
338 Invalid directive syntax
400 Directive not available
401 Not authorized for directive
436 Invalid display format
```

Examples

```
# Get the available display formats.
C -display
S %display name:dump
S %display
S %ok
```

```
# Change the active display format.
C -display dump
S %ok
```

3.3.4 forward

Description

The "-forward" directive instructs the server to follow all the referrals and return the results to the client. This directive can be used to run an RWhois server as a proxy server. The default value must be "off". When the value is set to "on", the server must not return referrals.

ABNF

```
forward-dir = "-forward" space on-off crlf
forward-response = response
```

Errors

```
338 Invalid directive syntax
400 Directive not available
401 Not authorized for directive
```

Examples

```
C -forward on
S %ok

C -forward off
S %ok
```

3.3.5 holdconnect

Description

Normally, the server closes the connection after each query. This behavior is controlled by the holdconnect state, which can be changed with the "-holdconnect" directive. When the holdconnect state is set to "off", the server must close the connection after a query; when it is set to "on", the server must not close the connection after a query. By default, the holdconnect state must be set to "off" for each connection.

ABNF

```
holdconnect-dir = "-holdconnect" space on-off crlf
holdconnect-response = response
```

Errors

```
338 Invalid directive syntax
400 Directive not available
401 Not authorized for directive
```

Examples

```
C -holdconnect on
S %ok

C -holdconnect off
S %ok
```

3.3.6 limit

Description

When returning a query result, the server should limit the number of objects returned to the client. The "-limit" directive changes this limit. The default and maximum limit is server-dependent. The client can get the current limit by using the "-status" directive (see Section 3.3.13).

ABNF

```
limit-dir = "-limit" space 1*digit crlf
limit-response = response
```

Errors

```
331 Invalid limit
338 Invalid directive syntax
400 Directive not available
401 Not authorized for directive
```

Examples

```
C -limit 100
S %ok
```

3.3.7 notify

Description

The "-notify" directive performs several functions.

* If the server returns a referral that results in an error, the client can report the bad referral to the server using the "badref" option.
* When the client follows referrals and goes through the same referral twice, that referral is a recursive referral and causes a referral loop. The client can report the recursive referral to the server using the "recurref" option.
* When the data in an authority area changes, a master server can use the "update" option to notify its slave servers to update the data.
* The "inssec" option allows an RWhois server to register itself as a slave server for an authority area with a master server. The master server may reject the request on the basis of its registration policy.
* The "delsec" option allows a slave server to cancel its registration with the master server.

ABNF

```
notify-dir = "-notify" space "badref" space referral-query crlf
        / "-notify" space "recurref" space referral-query crlf
   / "-notify" space "update" space host-port ":" authority-area crlf
        / "-notify" space "inssec" space host-port ":"
          authority-area crlf
        / "-notify" space "delsec" space host-port ":"
          authority-area crlf
referral-query = referral-url space [class-name space] query
notify-response = response
```

See Section 3.4 for the definitions of referral-url and query.

Errors

338 Invalid directive syntax
340 Invalid authority area
342 Invalid host/port
400 Directive not available
401 Not authorized for directive

Examples

```
# The client reports a bad referral to rwhois.foobar.com to the
# server.
C -notify badref rwhois://rwhois.foobar.com:4321/auth-area=foobar.com
  domain foobar.com
S %ok

# The client reports a recursive referral to rwhois.foobar.com to the
# server.
C -notify recurref rwhois://rwhois.foobar.com:4321/auth-area=
  foobar.com contact Last-Name="Beeblebrox"
S %ok

# The master server for the foobar.com authority area notifies its
# slave servers to update the data.
C -notify update master.foobar.com:4321:foobar.com
S %ok

# The server rwhois2.foobar.com registers as a slave server for the
# foobar.com authority area.
C -notify inssec rwhois2.foobar.com:4321:foobar.com
S %ok

# The server rwhois2.foobar.com cancels its registration as a slave
# server for the foobar.com authority area.
C -notify delsec rwhois2.foobar.com:4321:foobar.com
S %ok
```

3.3.8 quit

Description

The "-quit" directive can be used by the client to close the connection. Before the server closes the connection, it must respond with "%ok".

ABNF

```
quit-dir = "-quit" crlf
quit-response = response
```

Errors

No errors.

Examples

```
C -quit
S %ok
```

3.3.9 register

Description

The "-register" directive can be used by the client to add, modify, or delete objects in the server's database. The client must wait to send the registration data until the "%ok" response is received from the server. This directive has the following options.

* The "add" option indicates that the object being sent should be added to the server's database.
* The "mod" option indicates that the object being sent is a modification of an object that already resides on the server's database. During a modify operation, the "_NEW_" tag is used to delineate the end of the original (unmodified) object and the beginning of the replacement object. That is, the identifying characteristics of the original object are sent first, then the "_NEW_" separator is sent, and then the entire replacement object is sent.

The "del" option indicates that the object being sent should be deleted from the server's database.

After a register operation (add, modify, or delete an object) in an authority area, the server should update the "Serial-Number" variable in the SOA information for the authority area. This is useful for data replication because a slave server checks the "Serial-Number" variable to detect a data change at the master server (see Section 3.6.2).

ABNF

```
register-dir = register-on space "add" space maintainer-id crlf
               register-add register-off
        / register-on space "mod" space maintainer-id crlf
          register-mod register-off
        / register-on space "del" space maintainer-id crlf
          register-del register-off
register-on = "-register" space "on"
register-off = "-register" space "off" crlf
register-add = 1*(register-line crlf)
register-mod = 1*(register-line crlf) "_NEW_" crlf
               1*(register-line crlf)
register-del = 1*(register-line crlf)
```

```
maintainer-id = email
register-line = attribute-name ":" attribute-value
register-on-response = response
register-off-response = "%register" space "ID" ":" object-id crlf
                        response
        / "%register" space "Updated" ":" time-stamp crlf response
        / response
```

* The server must return the register-on-response for the "-register on" directive and the register-off-response for the "-register off" directive.
* The maintainer-id identifies, for maintenance purposes, the sender of registration information. The server should not use it to authenticate the sender.
* For the "add" option, the client must send all the required attributes for the object, including the Class-Name and Auth-Area attributes. However, the client must not send the ID and Updated attributes. These attributes are assigned by the server and returned in the response.
* For the "mod" option, the client must send the identifying information for the object to be modified, followed by the "_NEW_" separator and the entire replacement object. The identifying information must contain the ID and Updated attributes; it may contain other attributes, but the server may not check them. The ID, Auth-Area, and Class-Name attributes must match in both the original object data and the replacement object. The original object data is sent before the replacement object to enable the server to lock the record in the database.
* For the "del" option, the client must send the identifying information for the object to be deleted. The identifying information must contain the ID and Updated attributes; it may contain other attributes, but the server may not check them.

Errors

```
120 Registration deferred
320 Invalid attribute
321 Invalid attribute syntax
322 Required attribute missing
323 Object reference not found
324 Primary key not unique
325 Failed to update outdated object
336 Object not found
338 Invalid directive syntax
340 Invalid authority area
341 Invalid class
400 Directive not available
401 Not authorized for directive
```

Examples

```
# Add an object.
C -register on add joe@netsol.com
S %ok
C Class-Name:contact
C Auth-Area:a.com
C First-Name:Scott
C Last-Name:Williamson
C Name:Williamson, Scott
C Email:scottw@a.com
C -register off
S %register ID:23456789.a.com
S %register Updated:19961205224403000
S %ok

# Modify an object.
C -register on mod joe@netsol.com
S %ok
C ID:23456789.a.com
C Updated:19961205124403000
C _NEW_
C Class-Name:contact
C Auth-Area:a.com
C ID:23456789.a.com
C First-Name:Scott
C Last-Name:Williamson
C Name:Williamson, Scott
C Email:sw@a.com
C -register off
S %ok

# Delete an object.
C -register on del joe@netsol.com
S %ok
C ID:23456789.a.com
C Updated:19961205224403000
C -register off
S %ok
```

3.3.10 schema

Description

The "-schema" directive can be used by the client to get the attribute definitions of one or more classes in an authority area. If the client specifies class names, the server must return the attribute definitions of the specified classes. Otherwise, the server

must return the attribute definitions of all the classes in the authority area. Every schema record must end with an empty "%schema" line.

ABNF

```
schema-dir = "-schema" space authority-area *(space class-name) crlf
schema-response = *schema-record response
schema-record = *schema-line "%schema" crlf
schema-line = "%schema" space class-name ":" attribute-name ":"
              attribute-value crlf
```

Errors

338 Invalid directive syntax
340 Invalid authority area
341 Invalid class
400 Directive not available
401 Not authorized for directive

Examples

```
C -schema map
S %schema map:attribute:Class-Name
S %schema map:description:Type of the object
S %schema map:type:TEXT
S %schema map:format:re:[a-zA-Z0-9-]+
S %schema map:indexed:OFF
S %schema map:required:ON
S %schema map:multi-line:OFF
S %schema map:repeatable:OFF
S %schema map:primary:OFF
S %schema map:hierarchical:OFF
S %schema map:private:OFF
S %schema
S %schema map:attribute:ID
S %schema map:description:Globally unique object identifier
S %schema map:type:TEXT
S %schema map:format:re:[0-9]+.[a-zA-Z0-9.-]+
```

```
S %schema map:indexed:ON
S %schema map:required:ON
S %schema map:multi-line:OFF
S %schema map:repeatable:OFF
S %schema map:primary:ON
S %schema map:hierarchical:OFF
S %schema map:private:OFF
S %schema
# This is an abbreviated example, more attributes usually follow.
S %ok
```

3.3.11 security

Description

The "-security" directive enables either a client request or a server response to be authenticated and/or encrypted. Currently, RWhois uses two standard security methods: password and PGP. Password provides authentication only, and PGP provides both authentication and encryption. This directive can be used to securely access or update any information (meta or data) in an authority area that is protected by one or more guardian objects.

ABNF

```
security-dir = "-security" space "on" space direction space
        security-method [space security-data] crlf
        security-payload ["-security" space "off" crlf]
direction = "request" / "response"
security-method = "password" / "pgp" / 1*id-char
security-data = password-data / pgp-data / 1*any-char
password-data = 1*any-char
pgp-data = "signed" / "encrypt" [space key-id] / "signed-encrypt"
           [space key-id]
security-payload = *(*any-char crlf)
security-response = response
```

* The "password" security-method is available in the "request" direction only. For password, the security-data is a cleartext password.
* The "pgp" security-method is available in both the "request" and "response" directions. For PGP, the security-data indicates how to treat the security-payload: signed, encrypted, or signed and encrypted. To encrypt the security-payload in the "response" direction, the security-data must include the public key ID with which to encrypt it.

Errors

338 Invalid directive syntax
352 Invalid security method
353 Authentication failed
354 Encryption failed
400 Directive not available
401 Not authorized for directive

Examples

```
# Authenticate a request using password.
C -security on request password hello!1
S %ok

# Authenticate a PGP signed request.
C -security on request pgp signed
S %ok
C -register on mod joe@netsol.com
S %ok
C -----BEGIN PGP MESSAGE-----
C Version: 2.6.2
C
C owHrZJjKzMpgdP9D9crUhdpBYnwHGRnPbmVhmHlV7Hef9je/n7vyzhmE6589/+Dg
C jPpVm59tNz92vPSmrFB/4ankBRz+xgY+7z9OUYjefGahbWSNwzzxbw6TpWZGerU+
C uOUg/Cygs33JBdHqjwEc+wyfZPp+N5p2bu+ywoaOu8eLPyn+m2Mt/T9p1UaG68vP
C Zd2d9EPw+Ywpio7dco6yh3b/v7zmQxJHcWpyaVFmSSUDEHi6WBkZm5iamVtY6iXq
C JefnKnCFFqQklqSmWBlaWpoZGhmYGhqZmBgYGxgYKHA55yQWF+v6JeamWiXn55Uk
C JpcocDmWlmToOhalJlpB9cf7uYbHE6kWi/VumUXFJRB9wcn5JUBdPokwgfDMnJzM
C xNzi/DwFLjQBHQWoatfcxMwcq+JyB6h5AA==
C =a0sQ
C -----END PGP MESSAGE-----
C -register off
S %ok

# Encrypt a response using PGP. 52160EC1 is the public key ID with
# which the response is encrypted.
C -security on response pgp encrypt 52160EC1
S %ok
C -xfer com class=domain attribute=Domain-Name
  attribute=Organization-Name
S -----BEGIN PGP MESSAGE-----
S Version: 2.6.2
S
S hIwDqWWhK1IWDsEBBACOXssTzD2CbB7Vjj2cNURScpJc2as2TbUDOQiwkT+8qFgG
S ZyRfktpwNNTawRIcGOk1Kcs84z8a3vvTA/oje9vZexHtzfJwBHFdiIZxPuCEpvgv
S 2ppK7WqlmHGcQKVBJJHYw7Fq83CUkeGJB9P1M3CQiXeW8h8MwAuhxSgbgt23PKYA
S AABuhknJrXeh9Owm81+MvyzgLOyM7sjDYmttU9sj/yuOYmAhS9V+34MT/Mwn4wO8
```

```
S 2BCsJqBHXbwOuYKs02p0se4jyKFtZR8MDPWNm9QyAP+oNMTjsufy6ZRa9PegUC6t
S HDhXymkiP03mKMMVK1//7X0=
S =vZ2x
S -----END PGP MESSAGE-----
S %ok
```

3.3.12 soa

Description

The "-soa" directive can be used by the client to retrieve the SOA information for one or more authority areas. When no authority area name is given, the server must return the SOA information for all the authority areas. Every SOA record must end with an empty "%soa" line.

ABNF

```
soa-dir = "-soa" *(space authority-area) crlf
soa-response = *soa-record response
soa-record = *soa-line "%soa" crlf
soa-line = "%soa" space "authority" ":" authority-area crlf
   / "%soa" space "ttl" ":" 1*digit crlf
   / "%soa" space "serial" ":" time-stamp crlf
   / "%soa" space "refresh" ":" 1*digit crlf
   / "%soa" space "increment" ":" 1*digit crlf
   / "%soa" space "retry" ":" 1*digit crlf
   / "%soa" space "tech-contact" ":" email crlf
   / "%soa" space "admin-contact" ":" email crlf
   / "%soa" space "hostmaster" ":" email crlf
   / "%soa" space "primary" ":" host-port crlf
   / "%soa" space attribute-name ":" attribute-value crlf
```

The server must return the following SOA information for an authority area.

attribute-name	attribute-value	Comments
authority	authority-area	This is the name of the authority area.
ttl	1*digit	This is the default time to live for the data in the authority area.
serial	time-stamp	This is the serial number of the data in the authority area; it changes when the data changes.

refresh	1*digit	This is the time interval before a slave server checks for complete replication.
increment	1*digit	This is the time interval before a slave server checks for incremental replication.
retry	1*digit	This is the time interval before a slave server tries again to connect to a master server that appears to be out-of-service.
tech-contact	email	This is the contact for the operation of the master server.
admin-contact	email	This is the contact for the data integrity at the master server.
hostmaster	email	This is the contact for sending update requests at the master server.
primary	host-port	This is the host name (or IP address) and port number of the master server.

Errors

338 Invalid directive syntax
340 Invalid authority area
400 Directive not available
401 Not authorized for directive

Examples

```
C -soa org
S %soa authority:org
S %soa ttl:86400
S %soa serial:19961119111535000
S %soa refresh:3600
S %soa increment:1800
S %soa retry:180
S %soa tech-contact:tech@internic.net
S %soa admin-contact:admin@internic.net
S %soa hostmaster:hostmaster@internic.net
S %soa primary:rs.internic.net:4321
S %soa
S %ok
```

3.3.13 status

Description

The "-status" directive can be used by the client to get various status flags from the server. The response must include the number of objects in all the authority areas, the current display format, the server contact information, and the status flags for the state-oriented directives: "-limit", "-holdconnect", and "-forward".

ABNF

```
status-dir = "-status" crlf
status-response = *status-line response
status-line = "%status" space "limit" ":" 1*digit crlf
   / "%status" space "holdconnect" ":" on-off crlf
   / "%status" space "forward" ":" on-off crlf
   / "%status" space "objects" ":" 1*digit crlf
   / "%status" space "display" ":" 1*any-char crlf
   / "%status" space "contact" ":" email crlf
   / "%status" space attribute-name ":" attribute-value crlf
```

Errors

```
338 Invalid directive syntax
400 Directive not available
401 Not authorized for directive
```

Examples

```
C -status
S %status limit:20
S %status holdconnect:OFF
S %status forward:OFF
S %status objects:12345
S %status display:dump
S %status contact:joe@rwhois.net
S %ok
```

3.3.14 xfer

Description

The "-xfer" directive can be used by the client (generally, a slave server) to transfer the data in an authority area. The client can control the amount of data transferred using one of the following options.

* serial-number: The client can transfer all the objects that have been added, modified or deleted since a certain time, specifying the serial-number that indicates that time. This option is used for incremental replication.
* class: The client can limit the data transfer to one or more classes, using the "class=<class-name>" option. The server must return data for only the specified classes. If no class name is specified, the server must return data for all the classes.
* attribute: The client can limit the data transfer to one or more attributes of a class, using the "attribute=<attribute-name>" option in combination with the "class=<class-name>" option. The server must return data for only the specified attributes of the class. The client can specify multiple "class=" and "attribute=" pairs.

ABNF

```
xfer-dir = "-xfer" space authority-area *attribute-def
        [space serial-number] crlf
attribute-def = [space "class=" class-name] *(space "attribute="
                attribute-name)
serial-number = time-stamp
xfer-response = *xfer-record response
xfer-record = *xfer-line "%xfer" crlf
xfer-line = "%xfer" space class-name ":" attribute-name ":"
            attribute-value crlf
```

Errors

```
332 Nothing to transfer
333 Not master for authority area
338 Invalid directive syntax
340 Invalid authority area
341 Invalid class
342 Invalid attribute
400 Directive not available
401 Not authorized for directive
```

Examples

```
C -xfer com class=domain attribute=Domain-Name
  attribute=Organization-Name
S %xfer domain:Domain-Name:acme.com
S %xfer domain:Organization-Name:Acme Inc.
S %xfer
S %xfer domain:Domain-Name:vogon.com
S %xfer domain:Organization-Name:Vogon Heavy Industries
S %xfer
S %ok
```

3.3.15 X

Description

The "-X" directive is used to specify an additional, non-standard directive. It can be implemented by executing an external program, by internal functions, or by other means. It may interact with the client or simply produce output like one of the standard directives.

ABNF

```
x-dir = "-X-" x-directive [space x-arguments] crlf *x-line
x-directive = 1*id-char
x-arguments = *any-char
x-response = *(*any-char crlf) response
x-line = *any-char crlf
```

Errors

```
338 Invalid directive syntax
400 Directive not available
401 Not authorized for directive
```

Examples

The following example uses an implementation that executes an external program, the UNIX "date" command. The server runs the "date" command and returns its output to the client.

```
C -X-date
S Mon Jan 6 13:21:20 EST 1997
S %ok
```

3.4 Query

Description

The query allows the client to retrieve objects from the server's database. The server must support the following types of queries.

* Unrestricted query: It is a single word or a quoted string. The server must return all the matching objects where one or more attributes match the query, regardless of the class.
* Class-restricted query: It is a class name specified in front of the unrestricted query. The server must return all the matching objects where one or more attributes of the specified class match the query.
* Attribute-restricted query: It is of the "<attribute-name>=<search-string>" form. The server must return all the matching objects where the specified attribute matches the query.

The server may implement the following types of queries.

* Boolean operator query: It consists of simpler queries combined using the "and" and "or" operators.
* Wild card query: It consists of an asterisk ("*") in the front and/or at the end of the search string. The server may support partial matching using the asterisk.

In response to the query, the server will return the objects that match the query. If the server does not support complex queries, with, for example, wild cards or boolean operators, the server may return the "351 Query too complex" error. When the number of objects found exceeds the limit (set by the "-limit" directive), the server should return the objects, followed by the "330 Exceeded maximum objects limit" error.

The default object output format is the dump format that uses the "<class-name>:<attribute-name>;<type character>:<attribute-value>" form. The type character is optional and identifies the type of the attribute value. The type character is a shorthand for the Type field of the attribute definition (see Section 2.3.1). The type characters are defined as follows.

Type character	Attribute Type
T	TEXT
I	ID
S	SEE-ALSO

When no type character is given, the client should assume the "T" type character. The server must provide the type character when the attribute type is ID or SEE-ALSO. The purpose of the type character is to aid the client in displaying the data. For example, when an attribute value is an ID, the client may indicate to the end-user that it is possible to retrieve the object indicated by the ID.

The server may return one or more referrals in the "%referral rwhois://<host-name>:<port-number>/auth-area=<authority area>" form. The client can distinguish multiple referrals by comparing their authority areas; if all the referrals refer to the same authority area, the client should follow only one of them. Otherwise, the client should follow all of them. To follow a referral, the client must connect to the specified host name and port number, and issue the same query.

ABNF

```
rwhois-query = [class-name space] query crlf
query = query-string / attribute-query / query bin-boolean query
query-char = <any-char, except """, space, tab>
quoted-query-char = query-char / space / tab / "
query-string = ["*"] 1*query-char ["*"] / """ ["*"]
        1*quoted-query-char ["*"] """
attribute-query = attribute-name "=" query-string
bin-boolean = "and" / "or"

rwhois-query-result = *(query-record / referral-record) response
query-record = 1*query-line crlf
query-line = class-name ":" attribute-name [";" type-char] ":"
        attribute-value crlf
type-char = "T" / "I" / "S"
referral-record = 1*(referral-line crlf)
referral-line = "%referral" space referral-url
referral-url = "rwhois" ":" "//" host-port "/" "auth-area="
               authority-area
```

Errors

130 Object not authoritative
230 No objects found
330 Exceeded maximum objects limit
340 Invalid authority area
341 Invalid class
342 Invalid attribute
350 Invalid query syntax
351 Query too complex

Examples

This example illustrates a query, where no objects are found.

```
C vogon
S %error 230 No objects found
```

This example illustrates a query, where two different objects are returned.

```
C ibm
S domain:ID:IBMLIFEPRO-DOM.com
S domain:Auth-Area:com
S domain:Domain-Name:IBMLIFEPRO.COM
S domain:Org-Name:IBM
S domain:Server;I:NS12345-HST.NET
S domain:Server;I:NS12345-HST.NET
S domain:Admin-Contact;I:TW1234.COM
S domain:Tech-Contact;I:BN123.NET
S domain:Updated:19961120123455000
S domain:Updated-By:autoreg@internic.net
S domain:Class-Name:domain
S
S network:ID:NET-IBMNET-3.0.0.0/0
S network:Auth-Area:0.0.0.0/0
S network:Network-Name:IBMNET-3
S network:IP-Network:123.45.67.0/24
S network:Org-Name:IBM
S network:Street-Address:1234 Maneck Avenue
S network:City:Black Plains
S network:State:NY
S network:Postal-Code:12345
S network:Country-Code:US
S network:Tech-Contact;I:MG305.COM
S network:Updated:19931120123455000
S network:Updated-By:joeblo@nic.ddn.mil
S network:Class-Name:network
```

```
S
S %ok
```

This example illustrates a query with a class restrictor, where the number of objects found exceeds the limit set by the "-limit" directive.

```
C -limit 1
S %ok
C domain ibm
S domain:ID:IBMLIFEPRO-DOM.com
S domain:Auth-Area:com
S domain:Domain-Name:IBMLIFEPRO.COM
S domain:Org-Name:IBM
S domain:Server;I:NS12345-HST.NET
S domain:Server;I:NS12345-HST.NET
S domain:Admin-Contact;I:TW1234.COM
S domain:Tech-Contact;I:BN123.NET
S domain:Updated:19961120123455000
S domain:Updated-By:erice@internic.net
S domain:Class-Name:domain
S
S %error 330 Exceeded maximum objects limit
```

This is an example of attribute matching.

```
C domain Domain-Name=konabo.com
S domain:ID:12345678.com
S domain:Auth-Area:com
S domain:Domain-Name:konabo.com
S domain:Org-Name:ACME
S domain:Server;I:12345670.com
S domain:Server;I:12345671.com
S domain:Admin-Contact;I:12345660.com
S domain:Tech-Contact;I:12345665.com
S domain:Updated:19961120123455000
S domain:Updated-By:joeblo@internic.net
S domain:Class-Name:domain
S
S %ok
```

This example illustrates a link referral.

```
C domain a.b.rwhois.net
# The server returns a link referral to a server serving the
# b.rwhois.net authority area.
S %referral rwhois://master.b.rwhois.net:4321/auth-area=b.rwhois.net
S %ok
```

This example illustrates a punt referral.

```
C domain internic.net
# The server returns a punt referral to a server serving the root
# authority area.
S %referral rwhois://rs.internic.net:4321/auth-area=.
S %ok
```

This example illustrates multiple referrals that refer to the same authority area. The client should follow only one of them.

```
C domain a.b.rwhois.net
# The server returns link referrals to two RWhois servers serving the
# b.rwhois.net authority area.
S %referral rwhois://master.b.rwhois.net:4321/auth-area=b.rwhois.net
S %referral rwhois://slave.b.rwhois.net:4321/auth-area=b.rwhois.net
S %ok
```

This example illustrates multiple referrals that refer to different authority areas. The client should follow all of them.

```
C contact Last-Name="Beeblebrox"
# The server returns a link referral to a server serving the
# b.rwhois.net authority area.
S %referral rwhois://master.b.rwhois.net:4321/auth-area=b.rwhois.net
# The server also returns a punt referral to a server serving the
# net authority area since the query matched an entry in the
# non-hierarchical index received from it.
S %referral rwhois://rs.internic.net:4321/auth-area=net
S %ok
```

This is an example of a boolean operator and wildcard matching.

```
C ibm and jubliana*
S host:ID:JUBLIANA-HST.root
S host:Auth-Area:.
S host:Host-Name:JUBLIANA.TRL.IBM.CO.JP
S host:IP-Address:123.156.220.68
S host:Org-Name:IBM
S host:Street-Address:1234 Maneck Avenue
S host:City:Black Plains
S host:State:NY
S host:Postal-Code:12345
S host:Country-Code:US
S host:Updated:19961120123455000
S host:Updated-By:joeblo@nic.ddn.mil
S host:Class-Name:host
S
S %ok
```

3.5 Connection Model

An RWhois client can connect to an RWhois server using one of the following transport protocols.

3.5.1 Transmission Control Protocol (TCP)

TCP provides a reliable stream transport service between a client and a server. In RWhois, TCP is the default transport protocol because, during a particular session, a client can send more than one query and a server can reliably return a large amount of data for each of those queries. By default, a TCP RWhois server should run on the standard, Internet Assigned Number Authority (IANA)-assigned port 4321. However, if port 4321 is not available, it may run on an available port in the non-reserved range (1024 - 65535).

3.5.2 User Datagram Protocol (UDP)

UDP provides an unreliable connectionless transport service between a client and a server. In RWhois, UDP may be used as the transport protocol if a client wants to quickly send only one query, without incurring the overhead of establishing a TCP connection with a server. By default, a UDP RWhois server should run on the standard, IANA-assigned port 4321. However, if port 4321 is not available, it may run on an available port in the non-reserved range (1024 - 65535). A separate document will describe the use of UDP as the transport protocol in RWhois.

3.6 Data Replication

This section discusses when and how a slave server should replicate data. Further, it describes the server registration and location mechanisms.

3.6.1 When to Replicate Data

The time when a slave server may replicate data for an authority area is determined by the SOA variables for that authority area. The possible times are the following.

* When the "Refresh-Interval" expires, a slave server may completely replicate data.
* When the "Increment-Interval" expires, a slave server may incrementally replicate data.
* A slave server fails to connect to its master server to replicate data. When the "Retry-Interval" expires, it tries again to replicate data.
* When the data in an authority area is changed and its "Serial-Number" updated, a master server may notify its slave servers to immediately update the data. To notify about the data change, the master server should send the "-notify update <host-name>:<port-number>:<authority-area>" directive to its slave servers.

3.6.2 How to Replicate Data

To replicate data, a slave server sends a series of directives to its master server and checks each response before sending the next directive. The following sections describe the protocols for complete and incremental replication.

Complete Replication

The protocol between a master server and a slave server to completely replicate data for an authority area is as follows.

1. The slave server should connect to the master server. If there is a connection error, the slave server should log an error and exit.

2. The slave server should send the "-soa <authority-area>" directive to the master server and parse the SOA variables from the response. Let the "Serial-Number" variable in this response be called the "old-serial-number".

3. The slave server should send the "-class <authority-area>" directive to the master server and parse the versions of all the classes from the response.
4. The slave server should send the "-schema <authority-area>" directive to the master server and parse the definitions of all the classes from the response.
5. The slave server should send the "-xfer <authority-area>" directive to the master server and parse the data objects from the response. The master server should return all the data objects, excluding the deleted ones, in the authority area. The slave server should index these data objects.
6. When the "Refresh-Interval" expires, the slave server should to the master server. If there is a connection error, the slave server should try again after the "Retry-Interval".
7. The slave server should send the "-soa <authority-area>" directive to the master server and parse the SOA variables from the response. Let the "Serial-Number" variable in this response be called the "new-serial-number". If the "new-serial-number" is not greater than the "old-serial-number", go back to step 6. Otherwise, it indicates a data change at the master server.
8. The slave server should send the "-class <authority-area>" directive to the master server and parse the versions of all the classes from the response. If the version of any of the classes has changed, the slave server should send the "-schema <authority-area>" directive to the master server and parse the definitions of all the classes from the response.
9. The slave server should send the "-xfer <authority-area>" directive the master server and parse the data objects from the response. The master server should return all the data objects, excluding the deleted ones, in the authority area. The slave server should index these data objects and seamlessly replace the old index with the new one. Further, it should assign the "new-serial-number" to the "old-serial-number".
10. Go back to step 6.

Note that the "-class", "-schema", and "-xfer" directives change when a slave server replicates data for only a subset of the schema for an authority area.

In the following example, a slave server completely replicates data for all the classes in an authority area. The notation in the example uses a prefix to indicate from where the information is coming. An "M" indicates that the master server sends the data to the slave server. An "S" indicates that the slave server sends the data to the master server. The line is a comment when "#" is used. The space after the prefix is not part of the data. The example authority area is "rwhois.net".

```
# The slave server connects to the master server.
M %rwhois V-1.5:00ffff:00 master.rwhois.net
S -soa rwhois.net
M ...
M %soa serial:19970103102258000
M %soa refresh:3600
M ...
S -class rwhois.net
# The master server returns the versions of all the classes in the
# rwhois.net authority area.
S -schema rwhois.net
# The master server returns the definitions of all the classes in the
# rwhois.net authority area.
S -xfer rwhois.net
# The master server returns all the data objects, excluding the
# deleted ones, in the rwhois.net authority area. The slave server
# indexes these data objects.
# The refresh interval of 3600 seconds expires.
S -soa rwhois.net
M ...
M %soa serial:19970103103258000
M %soa refresh:3600
M ...
# The new serial number 19970103103258000 is greater than the old
# serial number 19970103102258000. It indicates a data change at the
# master server.
S -class rwhois.net
# The master server returns the versions of all the classes in the
# rwhois.net authority area. If the version of any of the classes has
# changed, the slave server logs an error and closes the connection.
S -xfer rwhois.net
# The master server returns all the data objects, excluding the
# deleted ones, in the rwhois.net authority area. The slave server
# indexes these data objects and seamlessly replaces the old index.
# The refresh interval of 3600 seconds expires.
S ...
```

Incremental Replication

The protocol between a master server and a slave server to incrementally replicate data for an authority area is as follows.

1. The slave server should connect to the master server. If there is a connection error, the slave server should log an error and exit.

2. The slave server should send the "-soa <authority-area>" directive to the master server and parse the SOA variables from the response. Let the "Serial-Number" variable in this response be called the "old-serial-number".
3. The slave server should send the "-class <authority-area>" directive to the master server and parse the versions of all the classes from the response.
4. The slave server should send the "-schema <authority-area>" directive to the master server and parse the definitions of all the classes from the response.
5. The slave server should send the "-xfer <authority-area>" directive to the master server and parse the data objects from the response. The master server should return all the data objects, excluding the deleted ones, in the authority area. The slave server should index these data objects.
6. When the "Increment-Interval" expires, the slave server should connect to the master server. If there is a connection error, the slave server should try again after the "Retry-Interval".
7. The slave server should send the "-soa <authority-area>" directive to the master server and parse the SOA variables from the response. Let the "Serial-Number" variable in this response be called the "new-serial-number". If the "new-serial-number" is not greater than the "old-serial-number", go back to step 6. Otherwise, it indicates a data change at the master server.
8. The slave server should send the "-class <authority-area>" directive to the master server and parse the versions of all the classes from the response. If the version of any of the classes has changed, the slave server should send the "-schema <authority-area>" directive to the master server and parse the definitions of all the classes from the response. The slave server should then send the "-xfer <authority-area>" directive to the master server and parse the data objects from the response. The master server should return all the data objects, excluding the deleted ones, in the authority area. The slave server should index these data objects and seamlessly replace the old index with the new one. Further, it should assign the "new-serial-number" to the "old-serial-number". If the version of any of the classes has changed, go back to step 6.
9. The slave server should send the "-xfer <authority-area> <old-serial-number>" directive to the master server and parse the data objects from the response. The master server should return all the data objects in the authority area that have been inserted, updated, or deleted since the "old-serial-number". The slave server should index all the data again after purging stale data objects and seamlessly replace the old index with the new one. Further, it should assign the "new-serial-number" to the "old-serial-number".
10. Go back to step 6.

Note that the "-class", "-schema", and "-xfer" directives change when a slave server replicates data for only a subset of the schema for an authority area.

In the following example, a slave server incrementally replicates data for all the classes in an authority area. The notation in the example uses a prefix to indicate from where the information is coming. An "M" indicates that the master server sends the data to the slave server. An "S" indicates the slave server sends the data to the master server. The line is a comment when "#" is used. The space after the prefix is not part of the data. The example authority area is "rwhois.net".

```
# The slave server connects to the master server.
M %rwhois V-1.5:00ffff:00 master.rwhois.net
S -soa rwhois.net
M ...
M %soa serial:19970103102258000
M %soa increment:1800
M ...
S -class rwhois.net
# The master server returns the versions of all the classes in the
# rwhois.net authority area.
S -schema rwhois.net
# The master server returns the definitions of all the classes in the
# rwhois.net authority area.
S -xfer rwhois.net
# The master server returns all the data objects, excluding the
# deleted ones, in the rwhois.net authority area. The slave server
# indexes these data objects.
# The increment interval of 1800 seconds expires.
S -soa rwhois.net
M ...
M %soa serial:19970103103258000
M %soa increment:1800
M ...
# The new serial number 19970103103258000 is greater than the old
# serial number 19970103102258000. It indicates a data change at
# the master server.
S -class rwhois.net
# The master server returns the versions of all the classes in the
# rwhois.net authority area. If the version of any of the classes has
# changed, the slave server logs an error and closes the connection.
S -xfer rwhois.net 19970103102258000
```

```
# The master server returns all the data objects in the rwhois.net
# authority area that have been inserted, updated, or deleted since
# 19970103102258000. The slave server indexes all the data again
# after purging stale data objects and seamlessly replaces the old
# index. The increment interval of 1800 seconds expires.
S ...
```

3.6.3 Server Registration

This section discusses how an RWhois server can register itself or cancel its registration as a slave server for an authority area with a master server.

The initial list of slave servers for an authority area should be manually configured at the master server. To register itself as a slave server, the server should send the "-notify inssec <host-name>:<port-number>:<authority-area>" directive to the master server. The master server may reject the request on the basis of its registration policy. To cancel its registration as a slave server, the server should send the "-notify delsec <host-name>:<port-number>:<authority-area>" directive to the master server. Note that the "host-name" and "port-number" in the above directives correspond to the requesting server.

3.6.4 Server Location

To resolve a query in a particular authority area, an RWhois client may need to first locate the master and slave servers for that authority area. The different server location mechanisms are as follows.

Referrals

An RWhois client should know about at least one RWhois server. It should send the "referral <authority-area>" query to that server. The query may be routed up or down the RWhois tree before getting resolved. If the query does get resolved, the result should be a referral object for that authority area. The client should parse the "Referral" attributes from the result to obtain a list of servers serving that authority area.

The client should then send the "-soa <authority-area>" directive to one of the above servers and parse the "Primary-Server" variable from the response. The value of this variable is the master server. Then, the remaining servers in the list are the slave servers.

SRV RRs

The Server Resource Record (SRV RR), defined for DNS, can be used to locate the master and slave servers for an authority area. An SRV RR specifies the location of a network service in an organization's DNS. It is defined in [RFC 2052] as follows.

Service.Proto.Name TTL Class SRV Priority Weight Port Target

Since an authority area identifier is generally a domain name or an IP address, the RWhois SRV RRs can be added to the DNS file for that domain or IP address. For example, the RWhois SRV RRs for the "rwhois.net" authority area could be:

```
rwhois.tcp.rwhois.net. 86400 IN SRV 10 0 4321 master.rwhois.net.
                                SRV 20 0 4322 slave.rwhois.net.
```

where the "master.rwhois.net" server has a higher priority than the "slave.rwhois.net" server. The client must try to connect to the server with a higher (lower-numbered) priority.

4. Security Considerations

RWhois provides security using the guardian class (see Section 2.3.6). Any information (meta or data) in an authority area can be guarded by containing pointers to one or more guardian objects; that is, it can be securely updated and accessed. Currently, there are two standard security methods: password and PGP (see Section 3.3.11). Password provides authentication only, and PGP provides both authentication and encryption. PGP is the recommended security method in RWhois.

The following sections discuss how to securely update and access the data in an authority area.

4.1 Data Update

This involves the ability to securely add, modify, or delete some information (meta or data) in an authority area. An authority area, on the whole, can be guarded by linking guardians to its SOA and schema information. Only these guardians should be allowed to add objects to the authority area and modify its SOA and schema information. In addition, they can also modify or delete existing objects in the authority area. However, the function of modifying or deleting existing objects can be delegated to other guardians by linking them to objects on a per-object basis.

4.2 Access Control

There are two access control issues; the first is the ability to securely transfer data between the slave and master servers. To transfer data for an authority area, a slave server can authenticate itself by satisfying one of the guardians linked to the SOA information of the authority area at the master server. In addition, the master server may encrypt the transferred data.

The second issue is the ability to make public only a subset of the data in an authority area. If all the objects of a particular class need to be private, the Private attribute of the class should be set to true. If only some attributes of all the objects of a particular class need to be private, the Private attribute property of each of those attributes should be set to true. The guardians of such objects must be able to view them completely.

5. Acknowledgments

The authors would like to acknowledge the following individuals.

Stan Borinski
C. Ming Lu
Leslie Meador
Michael Mealling
Greg Pierce
Amar Rao

6. References

[CIP] Allen, J., "The Common Indexing Protocol (CIP)", Bunyip Information Systems, November 1996, Work in Progress.

[Guardian] Singh, J., M. Kosters, "The InterNIC Guardian Object", ftp://rs.internic.net/policy/internic/internic-gen-1.txt, Network Solutions, February 1996.

[RFC 821] Postel, J., "Simple Mail Transfer Protocol", STD 10, RFC 821, ISI, August 1982.

[RFC 822] Crocker, D, "Standards for the Format of ARPA Internet Text Messages", STD 11, RFC 822, University of Delaware, August 1982.

[RFC 954] Harrenstien, K., Stahl, M., Feinler, E., "NICNAME/WHOIS", RFC 954, SRI, October 1985.

[RFC 1034] Mockapetris, P. V., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC 1714] Williamson, S., Kosters, M., "Referral Whois Protocol", RFC 1714, Network Solutions, November 1994.

[RFC 1738] T. Berners-Lee, L. Masinter, M. McCahill, "Uniform Resource Locators (URL)", RFC 1738, CERN, Xerox Corporation, University of Minnesota, December 1994.

[RFC 1991] Atkins, D., W. Stallings, P. Zimmermann, "PGP Message Exchange Formats", RFC 1991, MIT, Comp-Comm Consulting, Boulder Software Engineering, August 1996.

[RFC 2052] Gulbrandsen, A., P. Vixie, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2052, Troll Technologies, Vixie Enterprises, October 1996.

[X.500] "The Directory: Overview of Concepts, Models and Service", CCITT Recommendation X.500, 1988.

Authors' Addresses

Scott Williamson (scottw@rwhois.net)
Mark Kosters (markk@internic.net)
David Blacka (davidb@rwhois.net)
Jasdip Singh (jasdips@rwhois.net)
Koert Zeilstra (kzeil@rwhois.net)

Postal Address:
505 Huntmar Park Drive
Herndon, VA 22070-5100
Telephone: 703-742-0400

Appendix A: Glossary Of Terms

ABNF: Augmented Backus-Naur Form. Refined version of BNF, defined in [RFC 822]. See BNF.

Attribute: A named field and the smallest typed unit in a database schema. See Database Schema.

Authority Area: An autonomous part of an RWhois tree. It is associated and named after a particular piece of a hierarchy and is able to state authoritatively whether or not an instance of hierarchical data is present within the RWhois tree. See RWhois Tree.

Banner: A line sent by a server indicating which protocol versions it supports and which directives are implemented. This line is issued by the server after a connection is opened and as a response to the "-rwhois" directive. See Directive and Response.

Base Class: A class from which all defined classes in a database schema inherit attributes. See Attribute, Class, and Database Schema.

BNF: Backus-Naur Form. Language to precisely define the syntax of protocols and computer languages.

Class: A collection of attributes. See Attribute.

Complete Replication: The process of replicating all of the data for an authority area. See Replication.

Database Schema: A collection of all the classes forming an RWhois database. See Class.

Directive: A command that a client sends to a server to set a control parameter for the session, get the meta-information (class definitions and SOA information) about an authority area, or get the data in an authority area. See Class and SOA.

Guardian Class: A standard class that contains security information. An object is guarded by containing a pointer to a guardian object. See Class and Object.

Incremental Replication: The process of replicating the data that has changed since the last replication for an authority area. See Replication.

Info: The miscellaneous information that a server sends to a client.

Lexically Hierarchical Label: A text string whose position in a hierarchy is encoded in the string itself.

Link Referral: A pointer to another server that is further down an RWhois tree. It is used to route a query down the tree. See Referral and RWhois Tree.

Master Server: A server where the data is registered for an authority area. It answers authoritatively to queries in the authority area. It is also called a primary server. See Authority Area.

Namespace: A particular naming system defined by a set of rules describing the format of a name. Alternately, all of the names satisfying the rules.

Object: An instance of a class. It is data with a type of <class>. See Class.

PGP: Pretty Good Privacy. An authentication and encryption scheme.

Primary Server: See Master Server.

Punt Referral: A pointer to another server that is further up an RWhois tree. It is used to route a query up the tree. See Referral and RWhois Tree.

Query: A command that a client sends to a server to access the data in an authority area.

Query Routing: Redirecting a query to another server for resolution. See Query.

Referral: A pointer to another server that is presumed to be closer to the desired data. It is used to route a query. See Query Routing.

Referral Class: A standard class that contains referral information for an authority area. See Class and Referral.

Replication: A server duplicating data from another server on a per-authority area basis. See Authority Area.

Response: The information that a server returns to a client for a directive. See Directive.

Result: The information that a server returns to a client for a query. It can be either the accessed data or referrals to other servers. See Query and Referral.

RWhois Tree: A data information tree of RWhois servers where the data is arranged hierarchically in the authority areas. See Authority Area.

Schema: See Class.

Secondary Server: See Slave Server.

Slave Server: A server where the data is replicated from the master server for an authority area. It also answers authoritatively to queries in the authority area. It is also called a secondary server. See Master Server.

SOA: Start Of Authority. Administrative variables, defined at the master server, to control replication for an authority area. See Master Server and Replication.

Appendix B: RWhois ABNF

This specification uses the Augmented Backus-Naur Form (ABNF) notation, as defined in Section 2 of [RFC 822].

General Definitions

Lexical Tokens

```
alpha = "a".."z" / "A".."Z"
digit = "0".."9"
hex-digit = digit / "a".."f" / "A".. "F"
id-char = alpha / digit / "_" / "-"
any-char = <ASCII 1..255,
           except LF (linefeed) and CR (carriage return)>
dns-char = alpha / digit / "-"
email-char = <see [RFC 822]>
space = " "
tab = <ASCII TAB (tab)>
lf = <ASCII LF (linefeed)>
cr = <ASCII CR (carriage return)>
crlf = cr lf
```

Grammar

```
year = 4digit
month = 2digit
day = 2digit
hour = 2digit
minute = 2digit
second = 2digit
```

```
milli-second = 3digit
host-name = dns-char *(dns-char / ".")
email = 1*email-char "@" host-name
authority-area = (dns-char / ".") *(dns-char / "." / "/")
object-id = 1*id-char "." authority-area
host-port = (host-name / ip-address) ":" 1*5digit
ip-address = 1*3digit "." 1*3digit "." 1*3digit "." 1*3digit
class-name = 1*id-char
attribute-name = 1*id-char
attribute-value = 1*any-char
time-stamp = year month day hour minute second milli-second
on-off = "on" / "off"

Note that the time-stamp must be in the Greenwich Mean Time (GMT)
time zone.

response = ok-response crlf / error-response crlf / info-response
ok-response = "%ok"
error-response = "%error" space error-code space error-text
error-code = 3digit
error-text = 1*any-char
info-response = "%info" space "on" crlf *(*any-char crlf) "%info"
        space "off" crlf

rwhois-banner = "%rwhois" space version-list space host-name
        [space implementation] crlf
version-list = version *("," version)
version = version-number [":" capability-id]
        / "V-1.5" ":" capability-id
version-number = "V-" 1*digit "." 1*digit
capability-id = response-id ":" extra-id
response-id = 6hex-digit
extra-id = 2hex-digit
implementation = 1*any-char

rwhois-protocol = client-sends / server-returns
client-sends = *(directives / rwhois-query)
server-returns = *(responses / rwhois-query-result)

directives = rwhois-dir / class-dir / directive-dir / display-dir /
        holdconnect-dir / limit-dir / notify-dir / quit-dir /
        register-dir / schema-dir / security-dir / soa-dir /
        status-dir / xfer-dir / x-dir
```

```
responses = rwhois-response / class-response/ directive-response/
        display-response/ holdconnect-response/ limit-response/
        notify-response/ quit-response/ register-response/
        schema-response / security-response/ soa-response/
        status-response/ xfer-response/ x-response

Required Directives

rwhois

rwhois-dir = "-rwhois" space version-number [space implementation]
             crlf
rwhois-response = "%rwhois" space version space host-name
        [space implementation] crlf

Optional Directives

class

class-dir = "-class" space authority-area *(space class-name) crlf
class-response = *class-record response
class-record = *class-line "%class" crlf
class-line = "%class" space class-name ":" "description" ":"
             1*any-char crlf
   / "%class" space class-name ":" "version" ":" time-stamp crlf
   / "%class" space class-name ":" meta-field ":" meta-value crlf
meta-field = 1*id-char
meta-value = 1*any-char

directive

directive-dir = "-directive" *(space directive-name)crlf
directive-name = 1*id-char
directive-response = *directive-record response
directive-record = "%directive" space "directive" ":"
        directive-name crlf *directive-line "%directive" crlf
directive-line = "%directive" space "description" ":" 1*any-char crlf
        / "%directive" space attribute-name ":" attribute-value crlf
```

```
   display

   display-dir = "-display" crlf
      / "-display" space display-name crlf
   display-name = 1*id-char
   display-response = *display-record response
   display-record = "%display" space "name" ":" display-name crlf
   *display-line "%display" crlf
   display-line = "%display" space attribute-name ":" attribute-value
                  crlf

   holdconnect

   holdconnect-dir = "-holdconnect" space on-off crlf
   holdconnect-response = response

   limit

   limit-dir = "-limit" space 1*digit crlf
   limit-response = response

   notify

   notify-dir = "-notify" space "badref" space referral-query crlf
           / "-notify" space "recurref" space referral-query crlf
           / "-notify" space "update" space host-port ":" authority-area
             crlf
           / "-notify" space "inssec" space host-port ":" authority-area
             crlf
           / "-notify" space "delsec" space host-port ":" authority-area
             crlf
   referral-query = referral-url space [class-name space] query
   notify-response = response

   See the query section for the definitions of referral-url and query.

   quit

   quit-dir = "-quit" crlf
   quit-response = response
```

```
   register

   register-dir = register-on space "add" space maintainer-id crlf
           register-add register-off
           / register-on space "mod" space maintainer-id crlf
                   register-mod register-off
           / register-on space "del" space maintainer-id crlf
                   register-del register-off
   register-on = "-register" space "on"
   register-off = "-register" space "off" crlf
   register-add = 1*(register-line crlf)
   register-mod = 1*(register-line crlf) "_NEW_" crlf
                  1*(register-line crlf)
   register-del = 1*(register-line crlf)
   maintainer-id = email
   register-line = attribute-name ":" attribute-value
   register-on-response = response
   register-off-response = "%register" space "ID" ":" object-id crlf
                           response
           / "%register" space "Updated" ":" time-stamp crlf response
           / response

   schema

   schema-dir = "-schema" space authority-area *(space class-name) crlf
   schema-response = *schema-record response
   schema-record = *schema-line "%schema" crlf
   schema-line = "%schema" space class-name ":" attribute-name ":"
           attribute-value crlf

   security

   security-dir = "-security" space "on" space direction space
           security-method [space security-data] crlf security-payload
           ["-security" space "off" crlf]
   direction = "request" / "response"
   security-method = "password" / "pgp" / 1*id-char
   security-data = password-data / pgp-data / 1*any-char
   password-data = 1*any-char
   pgp-data = "signed" / "encrypt" [space key-id] / "signed-encrypt"
              [space key-id]
   security-payload = *(*any-char crlf)
   security-response = response
```

```
   soa

   soa-dir = "-soa" *(space authority-area) crlf
   soa-response = *soa-record response
   soa-record = *soa-line "%soa" crlf
   soa-line = "%soa" space "authority" ":" authority-area crlf
      / "%soa" space "ttl" ":" 1*digit crlf
      / "%soa" space "serial" ":" time-stamp crlf
      / "%soa" space "refresh" ":" 1*digit crlf
      / "%soa" space "increment" ":" 1*digit crlf
      / "%soa" space "retry" ":" 1*digit crlf
      / "%soa" space "tech-contact" ":" email crlf
      / "%soa" space "admin-contact" ":" email crlf
      / "%soa" space "hostmaster" ":" email crlf
      / "%soa" space "primary" ":" host-port crlf
      / "%soa" space attribute-name ":" attribute-value crlf

   status

   status-dir = "-status" crlf
   status-response = *status-line response
   status-line = "%status" space "limit" ":" 1*digit crlf
      / "%status" space "holdconnect" ":" on-off crlf
      / "%status" space "forward" ":" on-off crlf
      / "%status" space "authority" ":" 1*digit crlf
      / "%status" space "display" ":" 1*any-char crlf
      / "%status" space "contact" ":" email crlf
      / "%status" space attribute-name ":" attribute-value crlf

   xfer

   xfer-dir = "-xfer" space authority-area *attribute-def
           [space serial-number] crlf
   attribute-def = [space "class=" class-name]
                   *(space "attribute=" attribute-name)
   serial-number = time-stamp
   xfer-response = *xfer-record response
   xfer-record = *xfer-line "%xfer" crlf
   xfer-line = "%xfer" space class-name ":" attribute-name ":"
               attribute-value crlf

   X

   x-dir = "-X-" x-directive [space *[x-arguments]] crlf
   x-directive = 1*id-char
   x-arguments = *any-char
   x-response = *(*any-char crlf) response
```

Query

```
rwhois-query = [class-name space] query crlf
query = query-string / attribute-query / query bin-boolean query
query-char = <any-char, except """, space, tab>
quoted-query-char = query-char / space / tab / "
query-string = 1*query-char ["*"] / """ 1*quoted-query-char ["*"] """
attribute-query = attribute-name "=" query-string
bin-boolean = "and" / "or"

rwhois-query-result = *(query-record / referral-record) response
query-record = 1*query-line crlf
query-line = class-name ":" attribute-name [";" type-char] ":"
        attribute-value crlf
type-char = "T" / "I" / "S"
referral-record = 1*(referral-line crlf)
referral-line = "%referral" space referral-url
referral-url = "rwhois" ":" "//" host-port "/" "auth-area="
               authority-area
```

Appendix C: Error Codes

When a server fails to run a command (directive or query), it returns an error response. The ABNF for an error response is as follows.

```
error-response = "%error" space error-code space error-text
error-code = 3digit
error-text = 1*any-char
```

An error text may be modified, but its meaning must remain the same. The server may append additional information to it, for example "%error 333 Not master for authority area: foobar.com".

The following table describes the possible digits in the first, second, and third positions of an error code.

XXX	Description
1XX	Information only, no action required
2XX	Information, action required
3XX	Specific command error, retry that command or try another one
4XX	Serious for current command, may correct with another command
5XX	Fatal, must disconnect
X0X	System wide, no specific command
X1X	System wide, no specific command
X2X	Registration error
X3X	Specific command
X4X	Specific command
X5X	Specific command
X6X	Extended message (version specific)
XXX	Sequential order

The following table gives an ordered list of RWhois error codes. These codes may be extended with implementation- specific codes. An implementation- specific code must have a "6" in the second position.

Code	Text
120	Registration deferred
130	Object not authoritative
230	No objects found
300	Not compatible with version
320	Invalid attribute
321	Invalid attribute syntax
322	Required attribute missing
323	Object reference not found
324	Primary key not unique
325	Failed to update outdated object
330	Exceeded maximum objects limit
331	Invalid limit
332	Nothing to transfer
333	Not master for authority area
336	Object not found
338	Invalid directive syntax
340	Invalid authority area
341	Invalid class
342	Invalid host/port
350	Invalid query syntax
351	Query too complex
352	Invalid security method
353	Authentication failed
354	Encryption failed
400	Directive not available
401	Not authorized for directive

402	Unidentified error
420	Registration not authorized
436	Invalid display format
500	Memory allocation problem
501	Service not available
502	Unrecoverable error
503	Idle time exceeded

The following error codes, defined in [RFC 1714], have been made obsolete: 100, 200, 231, 334, 335, 337, 421, 431, 432, 433, 434, 460, 461, and 530.

Appendix D: Capability ID

The capability ID encodes which directives are implemented in the server. To create a capability ID, perform a logical OR on all the hexadecimal numbers corresponding to the implemented directives. The resulting number is used in the banner, which is sent by the server after opening a connection and as a response to the "-rwhois" directive. The eight most significant bits of the capability ID are reserved for future use:

class	000001h
directive	000002h
display	000004h
forward	000008h
holdconnect	000010h
limit	000020h
notify	000040h
quit	000080h
register	000100h
schema	000200h
security	000400h
soa	000800h
status	001000h
xfer	002000h
X	004000h

Appendix E: Schema Definitions

Attribute Definition Model

Name	Type	Description
Attribute	N	This is the name of the attribute.
Description	S	This is a free-form description of the attribute.
Type	T	This is a parameter that broadly indicates the use of the attribute to the protocol. There are three standard types: TEXT, ID, and SEE-ALSO. The default is TEXT, which indicates that the value is a text string. ID indicates that the attribute contains the ID of another RWhois object. This type of attribute is used for database normalization. SEE-ALSO indicates that the attribute contains a pointer (a Uniform Resource Identifier (URI)) to some other kind of external data; for example, a World Wide Web page or FTP site.
Format	S	This is an interpretable string that describes the acceptance format of the value. The server (and optionally the client) should match the value to the format string to determine if the value is acceptable. The format of this property is a keyword indicating the syntax of the format string, followed by a colon, followed by the format string itself. Currently, the only keyword recognized is "re" for POSIX.2 extended regular expressions.
Indexed	B	This is a true or false flag that indicates that this attribute should be indexed (and therefore able to be searched).
Required	B	This is a true or false flag that indicates that this attribute must have a value.
Multi-Line	B	This is a true or false flag that indicates that this attribute may have multiple instances in an object; all the instances are to be considered as multiple lines of the same attribute instance.

Repeatable	B	This is a true or false flag that indicates that there may be multiple instances of this attribute in a class and each instance is to be interpreted as a separate instance (in contrast to Multi-Line). This flag is mutually exclusive with Multi-Line: if Multi-Line is true, then Repeatable must be false and vice versa.
Primary	B	This is a true or false flag that indicates that this attribute is a primary key. If more than one attribute in a class is marked as primary, then these attributes together form a single primary key. The primary key is intended to be used to force uniqueness among class instances. Therefore, there can be only one instance of a primary key in a database. The Primary flag implies that the attribute is also required.
Hierarchical	B	This is a true or false flag that indicates that this attribute is lexically hierarchical.
Private	B	This is a true or false flag that indicates whether or not this attribute is private (that is, publicly not viewable). It defaults to false. If it is true, then only the clients that satisfy the authentication/encryption requirements of a guardian are able to view the attribute-value pair.

Type is defined as follows:

Type	ABNF Definition
B	"ON" / "OFF"
N	1*id-char
S	1*any-char
T	"ID" / "SEE-ALSO" / "TEXT"

Base Class

Name	Type	Required	Repeatable	Description
Class-Name	TEXT	Y	N	This attribute is the name of the class to which the object belongs.
Auth-Area	TEXT	Y	N	This attribute is the name of the authority area to which the object belongs.
ID	TEXT	Y	N	This attribute is the universal identifier of the object.
Updated	TEXT	Y	N	This attribute is a time/date stamp that indicates the time of last modification of the object.
Guardian	ID	N	Y	This attribute is a link to a guardian object. Its value is the ID of a guardian object.
Private	TEXT	N	N	This attribute is a true or false flag that indicates whether or not an object is private (that is, publicly not viewable). It defaults to false. If it is true, then only the clients that satisfy the authentication/encryption requirements of one of the object's guardians are able to view the object. If the object is publicly viewable, then the Private attribute property of each of its attributes still applies.

TTL	TEXT	N	N	This attribute is the "time-to-live" of a given object. It is included only if an object has a different time-to-live than the default given in the Start of Authority information. Its value is specified in seconds.

Appendix F: Changes RWhois V1.0 - V1.5

General

* Multiple authority areas per server.
* Data replication.
* Revised schema model.
* Revised query routing rules.
* Revised error codes.
* Removed unnecessary spaces in responses and results.

Directives

* Class: New. Returns meta-information for a class.
* Display: Can return supported display formats.
* Load: Obsolete.
* Notify: Syntax change.
* Private: Obsolete.
* Register: Syntax change.
* Schema: Syntax change.
* Security: Obsoletes Private.
* Xfer: Syntax change.

Query

* Display option removed.
* Output format: Only the dump format is standard; optional type character added.
* Attribute-restricted query.
* Revised referral syntax.