Abstract

This document describes how queries work in the latest version of the RIPE Database. (See page footer for version number.) This version uses the Routing Policy Specification Language (RPSL) [1] to represent many of the database objects. It uses the Routing Policy System Security (RPSS) [2] for authorisation. This means better security for Internet Routing Registries (IRR). It makes use of RPSL next generation specifications [14].

Though this document is self-contained, you may also wish to read the RPSL [1] and RPSS [2] specifications. For a tutorial on RPSL, you can read the RPSL applications document [3]. You may also want to read the "RIPE Database Update Reference Manual" [19]. It explains how updates work in the RIPE Database. There is also a single page "Whois Queries Reference Card" [18].

Intended Audience

This reference manual is for casual and advanced users who query the RIPE Database. If you are new to this database, you might find the "RIPE Database User Manual – Getting Started" [5] to be a more helpful place to start.

Conventions Used in This Document

We use <label> for a placeholder or to indicate syntax.
We use [option] to indicate an optional text or command argument.
We use a **bold** font to indicate an object type.
We use "attribute:" to indicate an attribute of an object.
"RIPE Database" usually means the user interface rather than the information in the database. Where there may be any doubt, this manual will make clear what is being discussed.

Table of Contents

Introduction

The RIPE Network Management Database (often called the “RIPE Database”) contains information about IP address space allocations and assignments, routing policies, reverse delegations and contacts in the RIPE NCC service region [16]. The data itself is a mix of registry information entered and maintained by the RIPE NCC and related data entered and maintained by RIPE NCC members and other users.

While the information in the RIPE Database is made freely available to the public, it is subject to the RIPE Database Terms and Conditions [22].

1.0 Database Objects and Attributes

The RIPE Database contains records of:

- Allocations and assignments of IP address space (the IP address registry)
- Reverse domain registrations
- Routing policy information (the Internet Routing Registry)
- Contact information - details of people who are registered as the contacts for the Internet number resources used in the operation of networks or routers and their organisations

The RIPE NCC maintains the Internet number resources in the database that are allocated or assigned by the RIPE NCC. These resource objects are tagged as “RIPE-REGISTRY-RESOURCE”. For other resource objects tagged as “RIPE-USER-RESOURCE”, the registered contacts (Registrants) and the holders of the referenced **mntner** objects maintain this data.

A database object is defined in RPSL as a list of attribute-value pairs in plain text form. Attributes can be mandatory, optional or generated. Mandatory attributes will always be present in an instance of an object. Optional attributes may be present if considered necessary or useful by the creator of the object or if required by the business rules in the software. Generated attributes can be included by the creator of the object, but their values will always be checked and included, when necessary, by the database software.

The attributes are indexed in a number of ways to allow the queries to search the database. An attribute can be a primary key, lookup key, inverse key, or a combination of these, or a part of one of these.

The characteristics of an attribute are determined by the type of object the attribute appears in. These are shown for each object in the object templates. They can be listed using the query:

whois –t <object-type>

There is also a detailed description available by querying:

whois –v <object-type>

2.0 Querying the RIPE Database

This section describes how you can find information by querying the RIPE Database. We describe the general format of a query and the query flags that you can use to change the default behaviour of a query.

We also describe how the database server automatically tracks query responses and limits how much contact information you can take from the RIPE Database. We do this to reduce the chance that someone will use the database to send spam e-mails to addresses that they find. There are four ways to query the database:

- Using a whois client running the whois protocol [12]

- Using the web interface from the RIPE NCC website [20]
- Using telnet to port 43 of the whois.ripe.net host
- Using the REST API running at rest.db.ripe.net

There is a set of general rules about server responses:

The same response will be returned from the server for each of these four methods.

Output lines starting with the % sign are either a server response code or server messages (a comment, information message or an error with description). A message contains a white space after the % sign, while a server response code line starts right after the % sign. See Appendix A2, “RIPE Database Query Server Response Codes and Messages” for more information.

Do not write scripts to parse the messages. The text is subject to change at any time.

A database object in the output is terminated by an empty line. This is a line containing only a newline character (\n).

Two empty lines, each containing only a newline character (\n), mean the end of a server response.

The general format of a query is:

[optional query flags] <query argument>

The format of a query flag is:

-short_query_flag [optional query flag argument]

or

--long_query_flag [optional query flag argument]

You may list each query flag separately, as in:

-B –i tech-c --no-grouping ABC999-RIPE

You can also group together short query flags. In this case, only the last flag in a list may have a flag argument, as in:

-Bi tech-c --no-grouping ABC999-RIPE

You can find a list of query flags and query flag arguments in Tables 2.1 to 2.6.

2.1 Queries Using Primary and Lookup Keys

Most queries use the primary and lookup keys of an object as an argument to the query. You can find a list of these queries in Table 2.1 at the end of this section.

2.2 Queries for IP Networks

The RIPE Database provides information about IP networks allocated or assigned within the RIPE NCC service region [16]. This information is mainly stored as **inetnum**, **inet6num**, **route** and **route6** objects. These objects store information about a single IP address or ranges of addresses.

The **route** and **route6** objects use “prefix notation” to specify the single address or range of addresses about which they contain information.

“Prefix notation” specifies ranges using two parts: the prefix and its length.

- For IPv4, the prefix is a 32-bit integer written in dotted quad notation with the value of the lowest IP address in the range. The prefix length is a whole number in the range 0-32 (for example 193.0.0.0/22 specifies the range of 1024 IPv4 addresses starting with, and including, 193.0.0.0).

- For IPv6 address ranges, the prefix length must be in the range 0-128 and is a 128-bit whole number, written in hexadecimal groups of two bytes separated by colons and with the possible use of shorthand notation for strings of consecutive 0s.

The **inetnum** objects represent an IPv4 address space in range notation where the range is explicitly specified as two 32-bit whole numbers written in dotted quad notation separated by a dash (for example 193.0.0.0 - 193.0.3.255, this is the same range as in the above example).

The **inet6num** objects represent IPv6 address space. Only the standard IPv6 prefix notation is allowed (as described above).

When you query the database for information about IP addresses, you can specify query arguments as search keys with the following notations:

- A prefix (this has the same meaning as above).
- An explicit range (also as above).
- A single IP number. This is interpreted as a range of just one address for IPv4 or a prefix length of one for IPv6.

For IPv4 address space, the query argument can be specified with either prefix or range notation. When prefix notation is used, the server software converts this into range notation before it executes the query. An information message is included in the server output showing the conversions performed.

For IPv6 address space, the query argument can only be specified in prefix notation. You can find a list of queries for IP networks in Table 2.2 at the end of this section. The rest of this section describes how you can retrieve different types of information relative to a particular range of IP addresses.

We use three terms in these types of queries. These are all defined relative to the specified (reference) range:

- An exact match refers to a range that is identical to the reference range.
- A more specific range is contained within the reference range and is smaller. It contains fewer IP addresses than the reference range. We also call this a sub range.
- A less specific range contains the whole of the reference range and is bigger. It has a greater number of IP addresses than the reference range. We also call this an encompassing range.

2.2.1 Default Queries for IP Networks

If you do not specify a query flag, and your query argument is a range of IP addresses in any one notation, the RIPE Database server will try to find an exact match for that range. If found, the exact match is returned. If an exact match cannot be found, the server looks for the smallest less specific range. This will be the smallest existing, encompassing range.

2.2.2 Exact Match Queries

If you want to change the default behaviour, so that the server returns only an exact match, you need to use the "–x" query flag. This flag stops the server from looking for any less specific ranges if no exact match exists, not even the smallest encompassing range will be returned.

2.2.3 More and Less Specific Queries

If the exact match is not the information you wanted, you can modify the information returned by the database server, by using one of these query flags which provide two generic types of queries known as more and less specific queries:

"-M", "-m", "-L" and "-l".

2.2.3.1 More Specific Queries

These refer to queries qualified by the use of the "-M" and "-m" query flags.

These will return information about ranges of IP addresses that are fully contained in the user-supplied reference range and contain fewer addresses. More specific queries do not return the exact match.

- "-M" - requests that the server should return all the sub-ranges completely contained within the reference range. When there are hierarchies of sub-ranges, all of these will be returned down to the smallest sub-ranges.
- "-m" - requests that the server should return all sub-ranges that are completely contained within the reference range. When there are hierarchies of sub-ranges, only the top level of the hierarchies will be returned. These are usually called one level more specific ranges.

2.2.3.2 Less Specific Queries

These refer to queries qualified by the use of the "-L" and "-l" query flags.

These will return information about ranges of IP addresses that fully contain the user-supplied reference range and may contain a greater number of addresses.

- "-L" - requests that the server returns the exact match, if any, and all encompassing ranges that are bigger than the reference range and that fully contain it.
- "-l" - requests that the server does NOT return the exact match. It returns only the smallest of the encompassing ranges that is bigger than the reference range and that fully contains it. This is usually referred to as the "one level less specific range".

2.2.4 Less Specific Queries Referencing irt Objects

In this section, "**inet(6)num**" is used to mean "**inetnum** or **inet6num**". This is to make the text clearer.

An **irt** object represents a Computer Security Incident Response Team (CSIRT). It includes contact and security information. It is optionally referenced from **inet(6)num** objects using the "mnt-irt:" attribute (even though the **irt** object is not a **mntner**). This shows which CSIRT is responsible for handling computer and network incidents for that address range.

A reference to an **irt** object does not apply only to the **inet(6)num** object that contains the reference. It also applies to all the **inet(6)num** objects that are "more specific" to the one containing the reference. Not every **inet(6)num** object needs to contain a reference to the **irt** object. The **irt** reference only needs to be placed in the least specific encompassing object to apply to a whole hierarchy of objects. This makes it easier to apply and maintain.

There may be more than one **inet(6)num** object in a hierarchy referencing an **irt** object. In this case, the one referenced from the smallest encompassing object is the one that applies to the range in question.

There is a "-c" query flag to make it easy to find the **inet(6)num** object containing the reference to an **irt** object for any specific range.

This flag makes the server search up the hierarchy from the range specified as the query argument. The search will stop when the first object is found containing a reference to an **irt** object. This can either be the specified range or an encompassing **inet(6)num** object. The query will return the **inet(6)num** object found for the specified range. The **irt** object will also be returned along with the usual contact data objects.

Sometimes, no **inet(6)num** object is found in the hierarchy containing a reference to an **irt** object. In this the query will return only the **inet(6)num** object found for the specified range.

2.3 Queries for Autonomous Systems

AS numbers can be either 32-bit or 16-bit. They are both described by a 32-bit number. From the point of view of the RIPE Database there is no difference. The difference is mainly historical.

2.4 Inverse Queries

Inverse queries are performed using an object's inverse key as an argument to a query. The query flag "-i" (or --inverse) must also be specified with appropriate query flag arguments. Inverse keys are defined in the templates of the RIPE Database objects. These can be listed using the query:

whois –t <object type>

Table 2.3, at the end of this section, gives a complete listing of the inverse query flag arguments.

By performing this type of query, the user requests all objects to be returned by the database that reference the specified query argument in the attribute(s) specified in the query flag arguments.

As an example:

whois -i admin-c <nic-handle>

will return all objects where the "admin-c:" attribute contains the <nic-handle> specified as the query argument.

You can specify several query flag arguments to request searches against several attributes in objects. If you want to do this, the query flag arguments should be entered as a comma-separated list with no white spaces. All the attributes searched must contain the same type of value. In other words, all the values must be maintainers or nic-handles or one of the other values listed in Table 2.3.

As an example:

whois --inverse mb,mnt-lower <mntner name>

will return all objects where the "mnt-by:" or the "mnt-lower:" attributes contain the <mntner name> specified as the query argument.

2.5 Abuse Contacts

There are many attributes in objects within the RIPE Database containing e-mail addresses. These addresses cover a number of functions. A growing concern to engineers and administrators that maintain networks is receiving spam and abuse complaints that are sent to every e-mail address displayed. This will get the message to the right person, but it also causes more spam and abuse to people who are not responsible for solving these problems.

To solve this issue, an "abuse-c:" attribute is available in the **organisation** object. This optional attribute references a role object which is required to contain an optional "abuse-mailbox:" attribute. Any Internet resource object (**inetnum**, **inet6num**, **aut-num**) that references this **organisation** object is then "covered" by this abuse email address.

By default, any query for an Internet resource object will return the related abuse email address. This appears in the query output as a comment line (starting with the % ' characters).

There is also a "-b" query flag to find the "abuse-mailbox:" attributes for any specific range. It returns the resource primary key with any abuse contact email address found. Also the prefix of any corresponding **route** or **route6** objects, followed by the "abuse-mailbox:" attributes.

If no abuse contact is found in any encompassing objects then no object summaries will be returned.

A complaint will not be handled any quicker by copying your message to any other e-mail address found in the database.

The "-b" query flag cannot be used with many other query flags.

2.6 Grouping the RIPE Database Output

There are two ways to order the results of a query.

One way is for the first part of the results to list the primary objects like **inetnum** and **mntner**. Then the second part of the results lists all the secondary objects associated with the primary objects, like **organisation** and **person**. If any of these secondary objects are referenced by more than one of the returned objects, it will only be listed once in the returned results.

The other way is to group the returned objects to show the association between the primary and secondary objects. In this way, each of the primary objects is followed immediately by all of its secondary objects. The secondary objects may appear more than once in this type of output. The default output is grouped. If you include the "-G" or "--no-grouping" query flag then the output will not be grouped.

2.7 Filtering the RIPE Database Output

A filtering process restricts some data from default query results. This applies to e-mail contact data. When a user is searching for abuse contact data, they sometimes take all e-mail addresses found in all objects returned from a query. This may include the correct address. However, it often also includes many other addresses for people who are not responsible for handling such complaints.

To help overcome this issue, some attributes containing e-mail addresses are filtered out of the default output. One exception to this is where a **role** object includes an "abuse-mailbox:" attribute. The abuse email address is never filtered. If you include the "-B" or "--no-filtering" query flag then the output will not be filtered

When any attribute has been filtered out of an object in the query results returned to the user, a "Note:" is added to the output to warn the user. In addition, the "source:" attribute of each filtered object will have a comment added to the end of the line saying "# Filtered". If this filtered output is cut and pasted into an update message, including this end of line comment on the "source:" attribute, the update will fail. This is because some mandatory attributes will be missing and the "source:" will not be recognised. Filtered output can therefore not be accidentally used for updates.

The MD5 hash in the "auth:" attribute of **mntner** objects is always filtered when these objects are queried. This filtering cannot be turned off with any query flags. The only way to see an unfiltered **mntner** object is to query it for update in the Webupdates form and enter a valid password for the object.

2.8 Query Support for Tools

There are several query types in the RIPE Whois Server that support various client tools. Other whois clients can also use these.

2.8.1 IRRToolset Support

The IRRToolset [6] is a third party suite of routing policy analysis tools. Some of the tools in this set access Routing Registry servers through an authorisation whois interface.

The RIPE Database Server includes support for these query types. This section describes the additions to the RIPE Database user interface that allow it to support the IRRToolset. The required queries are:

- Return the prefixes of all **route** and **route6** objects with a specified origin
- Return only the primary keys of the **route** and **route6** objects, not full objects
- Return the prefixes of all **route** and **route6** objects referenced in a given route-set
- Return all the members (**aut-num** or **as-set** object) of a specified **as-set**
- Return only the "members:" attributes, not the full object
- Optionally, include support for expansion of the previous query, if the returned value contains references to as-sets, so that the result contains only a list of **aut-num** objects

The RIPE Database server does not support this and it is up to the client to perform the expansion. The IRRToolset currently does the expansion.

- Return **route** and **route6** objects that exactly match a specified prefix
- Return **route** and **route6** objects that exactly match a specified prefix (as above), but return only the "route:" or "route6:" attributes

Table 2.4, at the end of this section, gives details of query support for tools.

2.8.2 Persistent Connections and Keeping State

If you are carrying out batched queries, your database client can request a persistent connection. The server will not close this connection after sending a reply to your client. This avoids having to set up a new TCP connection for every query.

The client can request this by sending the "-k" or "--persistent-connection" query flag to the server. This flag may be sent without a query argument to start the connection. It may also be included as a query flag with the first query.

During a persistent connection, the server operates a "stop-and-wait" protocol. This means that the next query cannot be sent until the reply has been received to the previous query. If you want to be able to send queries in batch mode, you must use the RIPE whois client.

To exit a persistent connection, send the "-k" or "--persistent-connection" flag with no query argument or an empty query (\n) to the server. The connection will also time out after a period of inactivity.

2.9 Getting All the Members of Set Objects

In RPSL [3], an object can be a member of a set object in two ways.

- You can list objects in a "members:" attribute in the set object. This is the kind of member relationship present in "Representation of IP Routing Policies in a Routing Registry" [4].
- You can use the "member-of:" attribute. You can use this in **route**, **route6**, **aut-num** and **inet-rtr** object types. The value of the "member-of:" attribute identifies a set object that this object wants to be a member of.

However, just specifying "member-of:" is not enough. The referenced set object must also have a "mbrs-by-ref:" attribute. This lists the maintainer of the object that wants to be a member of the set. This means that the set owner must validate the membership claim of an object with a "member-of:" attribute. It does this by matching the "mnt-by" line of the object with one of the maintainers in the "mbrs-by-ref:" attributes of the **set** object.

2.10 More and Less Specific Lookups For Reverse Domains

The RIPE Database supports IP network queries including the "-x", "-M", "-m", "-L" and "-l" functionality for reverse delegation domains. To request that reverse delegation domains be queried for with the more (or less) specific query flags, you must also include the "-d" query flag.

Note that there is no hierarchy allowed with reverse **domain** objects. These queries work on the address space hierarchies and return the corresponding **domain** object, if found, for any address space object.

2.11 Access Control for Queries

The control mechanism is based on the amount of contact information (contained in **person** and **role** objects) that is returned because of any queries made. Limits are based on the IP address of a whois client sending queries to the database server. Sometimes an IP address may be acting as a proxy and submitting queries on behalf of other IP addresses (for example, a webserver providing an interface to the RIPE Database). The database server provides a facility for such proxy clients that allows accounting to be based on the IP address of the clients using the proxy to query the RIPE Database and not on the IP address of the proxy server. This is done using the "-V" flag as follows:

-V <version>,<ipv4-address>

where

- <version> is a client tag that usually represents the software version that the proxy uses
- <ipv4-address> is the IPv4 address of the client that queries the database using the proxy

Not all users can use this "-V" flag. You must contact RIPE Database Administration and tell us why you need this facility. If we approve your request, we will add the IP address of the proxy server to an access control list. You can then use the "-V" flag, but *only* from your stated IP address.

Attempting to use the "-V" flag without approval may result in permanent denial of access to the RIPE Database. This denial of access will apply to the IP address that submits the query.

We restrict access to stop people using the RIPE Database to collect excessive amounts of contact data. If the amount of contact data returned by all your queries in a day (defined by Amsterdam time) exceeds defined limits, a temporary block on access is applied to that IP address. This block will be automatically released at midnight (Amsterdam time) to allow querying to continue. There is also a limit on the number of times an IP address can be blocked and recover. When this limit is reached, that IP address is permanently blocked from accessing the RIPE Database. This permanent block will not be automatically removed. The limits are defined in the RIPE Database Acceptable Use Policy [23].

There are many reasons why you could find yourself in this position. One is that you are mining the RIPE Database for contact data to use for non-agreed purposes. In this case, the denial of access is justified and your IP address will remain on the blocked list. However, there may be other reasons. Queries for object types other than **person** and **role** objects return contact information by default. Using the "-r" or "--no-referenced" flag to prevent contact data being included in your query results can turn off this default. Alternatively, you may have an error in a script that runs automatically, retrieving contact data that you did not know about. If you believe there was a genuine error or mistake that led to the permanent block, you need to contact RIPE Database Administration. Explain the error and tell us what steps you have taken to stop it happening again. RIPE Database Administration will decide whether to remove the block. It will remain on record that this IP address has been permanently blocked and unblocked. If another permanent block occurs, we will be less likely to consider removing it a second time.

Each time a **person** or **role** object is retrieved, a counter increases. When it reaches the limit defined in the AUP [23], the query execution is aborted and the connection is terminated, displaying an error message to the client (see "Access errors" in Appendix A1, "RIPE Database Query Server Response Codes and Messages"). Also a count of denials increases. Retrieving any other object type does not affect these counters.

Any role object used for abuse contacts with an "abuse-mailbox:" attribute is exception to this rule. No accounting is done on these objects.

There is also a limit on the number of simultaneous connections from a host. When this limit is reached, further connections from the same host are refused.

If we block your access, you will not be able to query for any object types. We will not just block your access to contact date alone.

2.12 Other Server Features

2.12.1 RIPE NCC Global Resource Service

The RIPE NCC mirrors several other databases, in near real time, that are considered to be of use to anyone who queries the RIPE Database. These include the other Regional Internet Registries

[24]. This allows queries to the RIPE Database to retrieve information from one or more of the mirrored databases. A local copy is maintained of each mirrored database. Daily updates are obtained from the mirrored databases to keep the local copies up to date.

The mirrored databases are referenced by the source of the data. A list of currently available sources and the data they can provide can be obtained by using the “-q” query flag. This is described in Section 2.12.2.

All mirrored databases are made available by the RIPE NCC in RPSL syntax with the same RPSL extensions as the RIPE Database. If the mirrored data stream has some other syntax, it is converted into RIPE Database syntax before submission to the local database copy. This means that the data returned to a user who queries a RIPE NCC mirror may not be in the same format as data returned by querying the original database directly.

Users can query the RIPE NCC mirrored databases as if they were querying the RIPE Database by adding the “-s” or "-a” query flags. Bulk access to any of the mirrored databases is not possible. The Access Control mechanism described in Section 2.12 applies to all data returned, regardless of the source.

2.12.2 The “–q” Query Flag

The RIPE Database server supports the retrieval of certain information about itself and the data sets served, using a “-q” query flag.

The “-q” query flag requests the server to reply with information about the system setup. It does not return any information extracted from any of the databases that it serves. This query flag takes a single argument which has three possible values:

- **Version** (usage: -q version). This will display version information for the server software
- **Types** (usage: -q types). This will list all the object types recognised by the RIPE Database
- **Sources** (usage: -q sources). This will list all available sources. That is, the local RIPE Database and all the mirrored databases

2.12.3. The “-t” (“--template”) Query Flag

This query flag returns to the user a brief description of the specified object type.

2.12.4. The “-v” (“--verbose”) Query Flag

This query flag returns to the user a verbose description of the specified object type.

2.12.5. The “-F” (“--brief”) Query Flag

This query flag changes the format of the returned objects. The attribute names are represented in a short hand notation. For example, "person:" becomes "*pn:". Using the –F query flag includes the non-recursive action of the –r query flag.

2.12.6. The “-K” (“--primary-keys”) Query Flag

This query flag returns only the primary keys of each object.

There are some exceptions to this:

- With **set** objects, the "members:" attributes will also be returned.
- No information is returned for **person**, **role** or **organisation** objects. If a query would normally only return these types of objects, no data is returned. In this case you do not get the "ERROR:101: no entries found". The entries were found but filtered because of using the "-K" flag.

2.12.7. The “-T” (“--select-types”) Query Flag

This query flag restricts the type of the objects returned. The query flag argument is a comma-separated list of object types.

2.12.8. The “-a” (“--all-sources”) Query Flag

This query flag requests that the server searches all the sources available to it. These are the sources listed by using the “–q sources” query.

2.13 Tagging

The RIPE Database allows system tags to be added to any (group of) objects. This is meta data that gives some additional information or meaning to certain objects. For example, all resource objects maintained by the RIPE NCC are tagged with “RIPE-REGISTRY-RESOURCE”. Other resources created in the database by users that are more specific to objects tagged with “RIPE-REGISTRY-RESOURCE” are tagged with “RIPE-USER-RESOURCE”.

Currently these tags are only generated by the database software and are re-generated nighty.

Tables of Query Types Supported by the RIPE Database

Table 2.1 Queries Using Primary and Lookup Keys

There are side effects to these types of queries. Other objects may be returned besides the ones that you are expecting. For example, if you enter a netname you may only expect to get back the **inetnum** and **inet6num** objects with this netname. You will also get any **person**, **role** or **mntner** objects back whose name matches the netname specified. The query is done as a text search on the primary and lookup keys. So any object with a matching string will be returned. The results can be limited by using the “–T” option to specify the object types you are interested in.

Lookup Key(s)	**Objects Returned**
<ip-lookup> (IPv4 address prefix, range or single address)	**inetnum**, **route** objects with exact match on the specified key. If the exact match does not exist, the objects with the smallest less specific match are returned. When you specify a single address, an **inet-rtr** object whose “ifaddr:” attribute matches the query argument is also returned.
<ip-lookup> (IPv6 address prefix or single address)	**inet6num, route6** objects with exact match on a specified key. If the exact match does not exist, the objects with the smallest less specific match are returned. If you specify a single address, an **inet-rtr** object whose “interface:” attribute matches the query argument is also returned.
<netname>	**inetnum** and **inet6num** objects whose “netname:” attribute matches the query argument.
<as-number>	**aut-num** object whose “aut-num:” attribute matches the query argument and an **as-block** object with the range containing the **aut-num** object, if it exists.
<as-number> - <as-number> (range of <as-number> separated by “-“)	**as-block** object whose primary key defines a range that exactly matches or fully contains the range specified in the query argument.
<reverse-domain>	**domain** and **inet-rtr** objects whose primary keys match the query argument.
<router interface	inet-rtr with an ifaddr containing the IPv4 or IPv6 address

address>	specified in the query.
<irt-name>	**irt** object whose primary key matches the query argument.
<person name> or <role name>	**person** and **role** objects whose “person:” or “role:” attributes contain the name specified in the query argument.
<email>	person, role and organisation objects with a matching email address.
<set-name>	A **set** whose primary key matches the query argument.
<nic-handle>	**person** or **role** object whose “nic-hdl:” attribute matches the query argument.
<mntner-name>	**mntner** object whose primary key matches the query argument.
<org name>	**organisation** object with an org-name containing the name specified in the query argument.
<org-id>	**organisation** object whose primary key matches the query argument.
<key-cert-id>	**key-cert** object whose primary key matches the query argument.
<poem>	**poem** object whose primary key matches the query argument.
<poetic-form>	**poetic-form** object whose primary key matches the query argument.

Table 2.2 Queries For IP Networks

Flag	**Lookup Key(s)**	**Objects Returned** or **Effect**
-x --exact	<ip-lookup>	Only **inetnum**, **inet6num**, **route**, **route6** or **domain** objects that exactly match the prefix. If no exact match is found, no objects are returned.
-M --all-more	<ip-lookup>	All level more specific **inetnum**, **inet6num**, **route**, **route6** or **domain** objects, excluding exact matches.
-m --one-more	<ip-lookup>	First level more specific **inetnum**, **inet6num**, **route** or **route6** objects, excluding exact matches.
-L --all-less	<ip-lookup>	All level less specific **inetnum**, **inet6num**, **route**, **route6** or **domain** objects, including exact matches.
-l --one-less	<ip-lookup>	First level less specific **inetnum**, **inet6num**, **route**, **route6** or **domain** objects, excluding exact matches.
-d --reverse-domain	<ip-lookup>	When used with “-x”, “-M”, “-m”, “-L” and “-l” flags, both **address** and **route** object types and **domain** object types are returned.

-c --irt	<ip-lookup>	The smallest, less specific **inetnum** or **inet6num** object found encompassing the range specified in the query argument. Also any **irt** objects found referenced from the returned **inetnum** or **inet6num,** or referenced from the first less specific **inetnum** or **inet6num** to the returned object that has such a reference.
-b --abuse-contact		Provides a brief output of address ranges or prefixes with associated abuse contact information.

Table 2.3 Query Flag Arguments to the "-i" Query Flag and the Corresponding Inverse Keys.

These can be combined in a list. There must be no white space in the list of values. For example: "-i mb,ml,mu,md"

Flag Argument (Alternative Form)	**Lookup Key(s)**	**Objects Returned**
am (abuse-mailbox)	<e-mail>	Objects whose "abuse-mailbox:" attribute matches the query argument.
ac (admin-c)	<nic-handle>	Objects whose "admin-c:" attributes match the query argument.
ah (author)	<nic-hdl>	Objects with a matching author
at (auth)	<PGP-id>	**mntner** objects whose "auth:" attribute matches the query argument. Please note that encrypted passwords and SSO usernames cannot be inverse-searched, but only PGPKEY.
fp (fingerprint)	<fingerprint>	**key-cert** objects whose "fingerpr:" attribute matches the query argument.
pn (person)	<nic-handle>	Objects whose "admin-c:", "tech-c:", "zone-c:", "ping-hdl:" or "author:" attribute matches the query argument.
iy (irt-nfy)	<e-mail>	**irt** objects whose "irt-nfy:" attribute matches the query argument.
la (local-as)	<as-number>	**inet-rtr** objects whose "local-as:" attribute matches the query argument.
mi (mnt-irt)	<irt-name>	**inetnum** and **inet6num** objects whose "mnt-irt:" attribute matches the query argument.
mr (mbrs-by-ref)	<mntner-name>	Set objects (**as-set**, **route-set** and **rtr-set**) whose "mbrs-by-ref:" attribute matches the query argument.
mo (member-of)	<set-name>	Objects whose "member-of:" attribute matches the query argument and their membership claim is validated by the "mbrs-by-ref:" attribute of the set. Absence of the "mbrs-by-

		ref:" attribute means that the membership is only defined by the "members:" attribute of the set.
mb (mnt-by)	<mntner-name>	Objects whose "mnt-by:" attribute matches the query argument.
md (mnt-domains)	<mntner-name>	Objects whose "mnt-domains:" attribute matches the query argument.
ml (mnt-lower)	<mntner-name>	Objects whose "mnt-lower:" attribute matches the query argument.
mn (mnt-nfy)	<e-mail>	**mntner** objects whose "mnt-nfy:" attribute matches the query argument.
mu (mnt-routes)	<mntner-name>	**aut-num**, **inetnum**, **route** and **route6** objects whose "mnt-routes:" attribute matches the query argument.
mz (mnt-ref)	<mntner-name>	Returns all objects whose "mnt-ref:" attribute matches the query argument.
ny (notify)	<e-mail>	Objects whose "notify:" attribute matches the query argument.
ns (nserver)	<domain-name>	**domain** objects whose "nserver:" attribute matches the query argument.
or (origin)	<as-number>	**route** and **route6** objects whose "origin:" attribute matches the query argument.
og (org)	<org-id>	Objects whose "org:" attribute matches the query argument.
rn ref-nfy	<e-mail>	Organisation objects whose "ref-nfy:" attribute matches the query argument.
tc (tech-c)	<nic-handle>	Objects whose "tech-c:" attribute matches the query argument.
dt (upd-to)	<e-mail>	**mntner** objects whose "upd-to:" attribute matches the query argument.
zc (zone-c)	<nic-handle>	Objects whose "zone-c:" attribute matches the query argument.

Table 2.4 Query Support For Tools

Flag	**Lookup Key(s)**	**Effect**
-F		Produces output using shorthand notation for attribute names. Produces slower responses.

--brief		
-K --primary-keys		Requests that only the primary keys of an object be returned. The exceptions are set objects, where the "members:" attributes will also be returned. This flag does not apply to **person** and **role** objects.
-k --persistent-connection	(optional normal query)	Requests a persistent connection. After returning the result, the connection will not be closed by the server and a client may issue multiple queries on the same connection. The server implements a "stop-and-wait" protocol, whereby no next query can be sent before receiving a reply for the previous one. Use RIPE Whois client to be able to send queries in batch mode. Except the first "-k query", "-k" without an argument closes the persistent connection.

Table 2.5 Miscellaneous Queries

Flag	**Flag Argument**	**Effect**
-r --no-referenced		Switches off lookups for referenced contact information after retrieving the objects that match the lookup key.
-B --no-filtering		Switches off default filtering of objects.
-G --no-grouping		Switches off grouping of associated objects and lists objects according to type.
-T --select-types	Comma separated list of object types. No white space is allowed in the list.	Restricts the types of primary objects to lookup in the query.
-a --all-sources		Specifies that the server should perform lookups in all available sources. See also "-q sources" query.
-s --sources	Comma separated list of sources. No white space is allowed in the list.	Specifies which sources are to be looked up when performing a query and in which order.
--list-versions		Shows a list of timestamps for historical versions for the object. Can not be used on person or role objects. History only goes back as far as the most recent delete.
--diff-versions	<version-number: version-number>	Indicates the difference between two given historical versions of the object.
--show-version	<version-number>	Returns the historical version of the object with this index number. Must use --list-versions to get the index numbers. Output has personal data removed.
--types		Lists available RPSL object types.

--version		Displays the current software version.
--list-sources		Returns the current set of sources along with the information required for mirroring.
--resource		Search all sources for the specified resource and returns the authoritative one. Placeholders are omitted.
--show-personal		Include referenced person and role objects in results.
--no-personal		Filter referenced person and role objects from results. A client can be blocked for excessive querying of these objects.
--show-tag-info		Include tagging information for each object in the results.
--no-tag-info		Do not include any tagging information in the results.
--filter-tag-include	<tag>	Return only objects with given tag in the results.
--filter-tag-exclude	<tag>	Filter out objects with given tag from the results.

Table 2.6 Informational Queries

The following notations are used in this table:
<object-type> means full or abbreviated name;
<client-tag> is a string without a white space that usually bears the name of the client's software.

Flag	Flag Argument	Effect
-q	sources	Returns the current set of sources along with the information required for mirroring.
-q	version	Displays the current version of the server.
-t --template	<object-type>	Requests a template for the specified object type.
-v --verbose	<object-type>	Requests a verbose template for the specified object type, including some help text about each attribute.
-V --client	<client-tag>	Sends information about the client to the server.

Appendices

A1. RIPE Database Query Server Response Codes and Messages

If the server encounters a problem, an error message is returned as a query result. The format of an error message is as follows:

%ERROR:#:<message>,

where # is the error or response code and <message> is a short description of the problem. There are no white spaces in this line, except in the <message> string. This may be followed by a more descriptive message, each line of which starts with % followed by a white space and some text.

Example:

```
% This is the RIPE Database query server #1.
% The objects are in RPSL format.
% The RIPE Database is subject to Terms and Conditions.
% See http://www.ripe.net/db/support/db-terms-conditions.pdf
%ERROR:101: no entries found
%
% No entries found in the selected source(s).
```

A1.1 Query Errors

%ERROR:101: no entries found
No entries were found in the selected source(s).

%ERROR:102: unknown source
Unknown source was supplied as argument to the "-s" query flag. Use "-q sources" for a list of available sources.

%ERROR:103: unknown object type
Unknown object type is specified as an argument to the "-T" query flag.

%ERROR:104: unknown attribute
Unknown argument is specified to the inverse query flag ("-I"). See Section 2.0, "Querying the RIPE Database" for more information.

%ERROR:105: attribute is not searchable
The argument specified for the inverse query flag is not a searchable attribute. See Section 2.0, "Querying the RIPE Database" for more information.

%ERROR:106: no query argument specified
No query argument has been specified in the query.

%ERROR:107: input line too long
Input exceeds the maximum line length.

%ERROR:108: bad character in input
An invalid character was passed in the query. The only allowed characters are letters, numbers and -_:+=.,@/?'

%ERROR:109: invalid combination of flags passed
The specified query flags cannot be included in the same query.

%ERROR:110: multiple use of flag

The same flag cannot be used multiple times.

%ERROR:111: invalid option supplied

Query flag that does not exist was given. Use the help query to see the valid options.

%ERROR:112: unsupported query

"-mM" query options are not allowed on very large ranges/prefixes.

%ERROR:114: unsupported query

Search key doesn't match any known query types.

%ERROR:115: invalid search key

Search key entered is not valid for the specified object type(s).

%ERROR:116: unsupported query

Versions are not supported for PERSON/ROLE objects.

%ERROR:117: version cannot exceed X for this object

Versions are numbers greater or equal to 1 but cannot exceed the object's current version number.

A1.2 Access Errors

%ERROR:201: access denied
Access from the host has been temporarily or permanently denied because of excessive querying. You should contact a customer service representative at ripe-dbm@ripe.net to discuss this problem.

%ERROR:202: access control limit reached
Limit of returned objects has been reached. The connection is terminated. Continued attempts to excessively query the database will result in permanent denial of service. See Section 2.11, "Access Control for Queries" for more information.

%ERROR:203: address passing not allowed
The host is not registered as a proxy and is not allowed to pass addresses on the query line ("-V" flag). See Section 2.11, "Access Control for Queries" for more information.

A1.3 Connection Errors

%ERROR:301: connection has been closed
The connection is administratively or abnormally closed.

%ERROR:302: referral timeout
The connection was closed due to referral timeout.

%ERROR:303: no referral host
Referral host cannot be found.

%ERROR:304: referral host not responding
The connection to the referral host cannot be established.

%ERROR:305: connection has been closed
The connection to the server has been closed after a period of inactivity.

%ERROR:306: connections exceeded
Number of connections from a single IP address has exceeded the maximum number allowed.

A1.4 NRTM Errors

%ERROR:401: invalid range: Not within <first>-<last>
This happens when the requested range or part of it is outside the serial numbers available at the server. <first> is the lowest serial number available. <Last> is the most recent serial number available.

A1.5 Warnings

%WARNING:901: duplicate IP flags passed
More than one IP flag (-x, -M, -m, -L, -l, -c, or -b) was passed to the server. Only the last one in the list of query flags will be used for this query.

%WARNING:902: useless IP flag passed
An IP flag (-x, -M, -m, -L, -l, -c, or -b) was passed to the server when query did not include an IP key as the argument.

A2. Copyright Information

A2.1 RIPE Database Copyright

The information in the RIPE Database is available to the public subject to the RIPE Database Terms and Conditions [22].

A2.2 RIPE NCC Copyright

Acknowledgements

The authors wish to acknowledge the work done by the original developers of version 3.0 of the RIPE Database software and infrastructure at the RIPE NCC:

Andrei Robachevsky, Daniele Arena, Marek Bukowy, Engin Gunduz, Roman Karpiuk, Shane Kerr, Ambrose M.R. Magee, Chris Ottrey and Filippo Portera.

Those who have continued its development include Denis Walker, Katie Petruska, Agoston Horvath, Can Bican, Tiago Anteo,

Jos I Boumans, Luis Motta Campos, Erik Broes and Menno Blom.

Edward Shryane, Angela Sjoholm, Theodoros Polychniatis

References

[1] C. Alaettinoglu, C. Villamizar, E. Gerich, D. Kessens, D. Meyer, T. Bates, D. Karrenberg and M. Terpstra, “Routing Policy Specification Language (RPSL)”, RFC 2622, June 1999

[2] C. Villamizar, C. Alaettinoglu, D. Meyer and S. Murphy, “Routing Policy System Security”, RFC 2725, December 1999

[3] D. Meyer, J. Schmitz, C. Orange, M. Prior, and C. Alaettinoglu, “Using RPSL in Practice”, RFC 2650, August 1999

[4] T. Bates, E. Gerich, L. Joncheray, J.M. Jouanigot, D. Karrenberg, M. Terpstra and J. Yu, “Representation of IP Routing Policies in a Routing Registry”, ripe-181, October 1994. See http://www.ripe.net/docs/ripe-181.html

[5] RIPE Database - Getting Started

[6] IRRToolset. See http://www.isc.org/sw/IRRToolSet/

[7] P. Mockapetris, “Domain names - Concepts and Facilities”, RFC 1034, November 1987

[8] P. Resnick, ed., “Internet Message Format”, RFC 2822, April 2001

[9] J. Zsako, “PGP Authentication for RIPE Database Updates”, RFC 2726, December 1999

[10] N. Nimpuno, A.Robachevsky, “New Value of the “status:” Attribute for Inetnum Objects (LIR-PARTITIONED)”, ripe-239, June 2002

[11] A. Cormack, D. Stikvoort, W. Woeber, and A. Robachevsky, “IRT Object in the RIPE Database”, ripe-254, July 2002

[12] K. Harrenstien, M.K. Stahl, E.J. Feinler. “NICNAME/WHOIS”, RFC 954, October 1985

[13] J.S.L. Damas and L. Vegoda, “New Values of the “status:” Attribute for inet6num Objects”, ripe-243, August 2002

[14] L. Blunk, J. Damas, F. Parent and A. Robachevsky, Routing Policy Specification Language next generation (RPSLng), RFC 4012

[15] C. Bican, RIPE 43 presentation on Webupdates, December 2002, http://www.ripe.net/ripe/meetings/ripe-43/presentations/ripe43-database-syncupdates/index.html

[16] RIPE NCC service region http://www.ripe.net/membership/maps/index.html

[17] The IANA ccTLD Database contains a full list of the ccTLD administrators, http://www.iana.org/cctld/cctld-whois.htm

[18] RIPE Database Queries Reference Card

[19] RIPE Database Update Reference Manual

[20] RIPE Database web query, http://www.ripe.net/whois

[21] Regional Internet Registries (RIR), http://www.ripe.net/info/resource-admin/index.html

[22] Ripe Database Terms and Conditions, http://www.ripe.net/db/support/db-terms-conditions.pdf

[23] RIPE Database Acceptable Use Policy, http://www.ripe.net/data-tools/support/documentation/aup

[24] Regional Internet Registries - RIPE NCC, ARIN, APNIC, AFRINIC, LACNIC