RIPE Database Query Reference Manual

Abstract

This document describes how queries work in version 3.2 of the RIPE Database. 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]. This means that you can register multicast and IPv6 routing policies. 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].

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 interface software 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
1.0 Database Objects and Attributes
1.1 Object Representation
1.2 Object Types
1.2.1 as-block
1.2.2 as-set
1.2.3 aut-num
1.2.4 domain
1.2.5 filter-set
1.2.6 inet6num
1.2.7 inetnum
1.2.8 inet-rtr
1.2.9 irt
1.2.10 key-cert
1.2.11 mntner
1.2.12 organisation
1.2.13 peering-set
1.2.14 person
1.2.15 poem
1.2.16 poetic-form
1.2.17 role
1.2.18 route
1.2.19 route6
1.2.20 route-set
1.2.21 rtr-set
2.0 Querying the RIPE Database
2.1 Queries Using Primary and Lookup Keys
2.2 Queries for IP Networks
2.2.1 Default Queries for IP Networks
2.2.2 Exact Match Queries
2.2.3 More and Less Specific Queries
2.2.3.1 More Specific Queries
2.2.3.2 Less Specific Queries
2.2.4 Less Specific Queries Referencing irt Objects
2.3 Queries for Autonomous Systems
2.4 Inverse Queries
2.5 Abuse Contacts
2.6 Grouping the RIPE Database Output
2.7 Filtering the RIPE Database Output
2.8 Query Support for Tools
2.8.1 IRRToolset Support
2.8.2 Persistent Connections and Keeping State
2.9 Getting All the Members of Set Objects
2.10 More and Less Specific Lookups For Reverse Domains

2.11 Referral Mechanism for Domains
2.11.1 The “refer:” Attribute
2.11.2 Domain Name Stripping
2.11.3 The “-R” Query Flag
2.11.4 The Referral Process
2.12 Access Control for Queries
2.13 Other Server Features
2.13.1 Mirroring Other Databases
2.13.2 The “–q” Query Flag
2.13.3. The “-t” Query Flag
2.13.4. The “-v” Query Flag
2.13.5. The “-F” Query Flag
2.13.6. The “-K” Query Flag
2.13.7. The “-T” Query Flag
2.13.8. The “-a” Query Flag
Tables of Query Types Supported by the RIPE Database
Table 2.1 Queries Using Primary and Lookup Keys
Table 2.2 Queries For IP Networks
Table 2.3 Query Flag Arguments to the "-i" Query Flag and the Corresponding Inverse Keys.
Table 2.4 Query Support For Tools
Table 2.5 Miscellaneous Queries
Table 2.6 Informational Queries
Appendices
A1. Object Attributes
A2. RIPE Database Query Server Response Codes and Messages
A2.1 Query Errors
A2.2 Access Errors
A2.3 Connection Errors
A2.4 NRTM Errors
A2.5 Warnings
A2.6 Referral Text
A3. Copyright Information
A3.1 RIPE Database Copyright
A3.2 RIPE NCC Copyright
Acknowledgements
References

Introduction

The RIPE Network Management Database (often called the "RIPE Database") contains information about IP address space allocations and assignments, routing policies, reverse delegations, contacts in the RIPE NCC service region [16] and ENUM delegations worldwide.

The information in the RIPE Database is available to the public for agreed Internet operation purposes, but it is copyright. See Appendix A3, "Copyright Information". This document describes how queries work in version 3.2 of the RIPE Database. This version uses the Routing Policy Specification Language (RPSL) [1] to represent all database objects. It uses the Routing Policy System Security (RPSS) [2] for authorisation. This means better security for Internet Routing Registries (IRR). The RIPE Database includes an IRR. It makes use of RPSL next generation specifications [14]. This means that you can register multicast and IPv6 routing policies.

This document is self-contained, but does not contain examples of usage and illustrations of how the RIPE Database works. If this is what you want, you should read the RPSL [1] and RPSS [2] specifications. If you are looking for a tutorial on RPSL, you should read the RPSL applications document [3]. The "RIPE Database User Manual – Getting Started" [5] contains some examples. 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].

1.0 Database Objects and Attributes

The RIPE Database contains records of:

- allocations and assignments of IP address space (the IP address registry);
- domain names (mainly for reverse domains);
- routing policy information (the routing registry);
- contact information (details of people who are responsible for the operation of networks or routers. As the RIPE NCC does not maintain the contents of the database, you can find contact details here of the people who do).

The RIPE NCC defines a database object 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. 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.

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>
```

1.1 Object Representation

The records in the RIPE Database are known as objects. RPSL [1] defines the syntax of these database objects (how they are written). An object belongs to one of the object types or classes. We use the two terms - ‘type’ and ‘class’ - interchangeably throughout this document. .

Object Types Supported by the RIPE Database:

Object type (Class name)	**Short name**	**Description**
as-block	ak	Delegation of a range of Autonomous System (AS) Numbers to a given Regional Internet Registry (RIR).
as-set	as	Set of aut-num objects.
aut-num	an	AS in the database. It describes the external routing policy of the AS.
domain	dn	Forward or reverse domain registrations.
filter-set	fs	Set of routes matched by its filter.
inet6num	i6	Allocations and assignments of IPv6 address space.
inetnum	in	Allocations and assignments of IPv4 address space.
inet-rtr	ir	Router in the database.
irt	it	Contact and authentication information about a Computer Security Incidence Response Team (CSIRT).
key-cert	kc	Public key certificate that is stored on the server and may be used with a **mntner** object for authentication when performing updates.
mntner	mt	Authentication information needed to authorise creation, deletion or modification of the objects protected by the **mntner**.
organisation	oa	Organisation that holds the resources.
peering-set	ps	Set of peerings.
person	pn	Technical or administrative contacts.
poem	po	Humorous poem.
poetic-form	pf	Type of humour for a **poem** object.

role	ro	Technical or administrative contacts - describes a role performed by one or more people.
route	rt	IPv4 route advertised on the Internet.
route6	r6	IPv6 route advertised on the Internet.
route-set	rs	Set of routes.
rtr-set	is	Set of routers.

All objects contain at least one "changed:" attribute. The information in this attribute may show who created or modified the object and when. It is not reliable as a full audit trail. The only database software conditions are that there be at least one of these attributes, and if there is more than one, the dates must be in ascending order. It is for the maintainer of the object to decide how to use this attribute. In some cases, it may be a full audit trail. In other objects, it may only have one attribute that is modified each time the object is modified. It may also have just the one attribute that represents the original creation or any one of many modifications made to the object over a period of time. It is used as a reference for the benefit of the maintainer of the object. It is not intended to give any reliable information to a user who queries for an object.

Some objects contain a "country:" attribute. It is difficult to strictly define what this value means. It may be the country where the IP address is used, or the country from where it was assigned. It could be the country where the head office of the organisation is based. As with the "changed:" attribute, it is not intended to give any reliable information to a user who queries for an object.

There are a number of attributes used for notifications. For example, "notify:", "mnt-nfy:", "upd-to:". These are used for administration of the data in the database. They show who is to be notified of changes to objects, or incorrectly authorised attempts to make changes.

There are a number of attributes used for the authorisation of changes to the data in the database. For example, "mnt-by:", "mnt-lower:". These contain references to objects containing the authentication tokens needed to make changes.

1.2 Object Types

This section describes the object types (classes) that the RIPE Database supports.

1.2.1 as-block

An as-block object delegates a range of AS Numbers to a given RIR [21]. These objects are created by the Database Administration. They are used to authorise creation of aut-num objects within the range specified by the "as-block:" attribute.

1.2.2 as-set

An as-set object defines a set of **aut-num** objects. The "as-set:" attribute defines the name of the set. It is an RPSL name that starts with "as-".

The "members:" attribute lists the members of the set. It can be either a list of AS Numbers, or other as-set names.

The name of an **as-set** object can be hierarchical. A hierarchical as-set name is a sequence of as-set names and AS Numbers separated by colons. The first component must be an actual as-set name (that means that it should start with "as-").

1.2.3 aut-num

The aut-num object specifies routing policies. It refers to a group of IP networks that have a single and clearly defined external routing policy, operated by one or more network operators – an Autonomous System (AS).

- The value of the "aut-num:" attribute is the AS Number of the AS that this object describes.
- The "as-name:" attribute is a symbolic name of the AS.
- The import, export and default routing policies of the AS are specified using the "import:", "export:" and "default:" attributes.
- Corresponding attributes with an "mp-" prefix (for example: "mp-import:") are used to specify IPv6 and multicast routing policies.
- Only a single value for the "org:" attribute is allowed in the **aut-num** object. This makes sure that only one organisation is responsible for this resource.

1.2.4 domain

The domain object mainly represents reverse DNS delegations. It can also be used to represent Top Level Domain (TLD) and other forward domain registrations. However, the information about forward domain names is for reference only. It has no effect on operations and may not be complete or authoritative. The RIPE Database is not the same as the domain name registries run by the country code Top Level Domain (ccTLD) administrators. The IANA ccTLD Database provides a full list of the ccTLD administrators. [17]. You should contact these for authoritative information on forward domains.

You should write the domain name in fully qualified format, without a trailing dot. If included, the trailing dot is removed before the query is actioned. In this case a warning message will be reported in the data returned by the server.

1.2.5 filter-set

A **filter-set** object defines a set of routes that match the criteria that you specify in your ‘filter’ – in other words, it filters routes that you may or may not want to see.
The "filter-set:" attribute defines the name of your filter. It is an RPSL name that starts with "fltr-".

- The "filter:" attribute defines the policy filter of the set. This is a logical expression that, when applied to a set of routes, returns a subset of these routes. These are the ones that you want to filter in or out.
- The "mp-filter:" attribute extends the "filter:" attribute to allow you to specify IPv6 prefixes and prefix ranges.
- The "filter:" and "mp-filter:" attributes are optional. A filter-set object, must contain at least one of these two attributes. It cannot contain both within the same object.
- The name of a **filter-set** object can be hierarchical. A hierarchical filter-set name is a sequence of filter-set names and AS Numbers separated by colons. The first component of the name must be an actual filter-set name (that is, start with "fltr-").

1.2.6 inet6num

An **inet6num** object contains information on allocations and assignments of IPv6 address space. Only a single value for the "org:" attribute is allowed in the **inet6num** object. This is to ensure that only one organisation is responsible for this resource.

The "status:" attribute is used as an administrative tag to register the type of address space.

The "rev-srv:" attribute is for information only. It has no operational use.

1.2.7 inetnum

An **inetnum** object contains information on allocations and assignments of IPv4 address space. Only a single value for the "org:" attribute is allowed in the **inetnum** object. This is to ensure that only one organisation is responsible for this resource.

The "status:" attribute is used as an administrative tag to register the type of address space.

The "rev-srv:" attribute is for information only. It has no operational use.

1.2.8 inet-rtr

The **inet-rtr** object specifies routers.

- The "inet-rtr:" attribute is a valid DNS name for a router, without a trailing ".".
- Each "alias:" attribute, if present, is also a standard DNS name for the specified router.
- The "local-as:" attribute specifies the AS Number of the AS that owns or operates this router.
- The "ifaddr:" attribute specifies the interface address within an Internet router, as well as an optional action to set other parameters on this interface.
- The "interface:" attribute specifies a multi-protocol interface address within an Internet router, optional action and tunnel definition.
- The "peer:" attribute specifies the details of any interior or exterior router peering.
- The "mp-peer:" attribute extends the "peer:" attribute for IPv6 addresses.

1.2.9 irt

An **irt** object represents a Computer Security Incident Response Team (CSIRT). It includes contact and security information. It can be referenced from **inetnum** or **inet6num** objects to show which CSIRT is responsible for handling computer and network incidents for that address range. It is also used to link "abuse-mailbox:" attributes to **inetnum** and **inet6num** objects.

The **irt** object name starts with "irt-".

1.2.10 key-cert

A **key-cert** object is a database public key certificate that is stored in the RIPE Database. It is used with a **mntner** object for authentication when performing updates. Currently the RIPE Database supports two types of keys:

- For PGP key-cert objects, the value of the "key-cert:" attribute must be PGP-"key-id". These keys are compliant with the Open PGP Message Format [RFC 2440].
- For X.509 key-cert objects, the database software assigns this value as X.509-n. Here, ‘n’ is the next available number.

1.2.11 mntner

Using **mntner** objects protects objects in the RIPE Database. A **mntner** object contains the information needed to authorise creation, deletion or modification of any objects that it protects.

Objects are protected by a **mntner**, if they contain a reference to the **mntner** in the object. This is done by including a "mnt-by:" attribute. Other attributes offer hierarchical protection. The "mnt-by:" attribute is mandatory in all objects except **person** and **role** types. Most users set the "mnt-by:’ value in a mntner to reference itself.

The "referral-by:" attribute can refer to the **mntner** object itself. The database software does not currently use this attribute even though it is mandatory to include it.

1.2.12 organisation

The **organisation** object provides information about an organisation such as a company, charity or university that has a network resource stored in the RIPE Database. It was introduced as a means to link together all the resource and administration objects related to one organisation.

- "organisation:" specifies the ID of the **organisation** object. This is set by the database software. It is used as a label to allow other objects to reference it.
- "org-name:" attribute defines the name of the organisation.
- "org-type:" specifies the type of an organisation and can be IANA, RIR, NIR, LIR or OTHER. Users can only set their "org-type:" to OTHER. Only the Database Administrator uses all other types.

1.2.13 peering-set

A **peering-set** object defines the set of peerings that appear in the "peering:" or "mp-peering:" attribute.

The "peering-set:" attribute defines the name of the set. It is an RPSL name that starts with ‘prng-’.

- The name of a peering-set object can be hierarchical. A hierarchical peering-set name is a sequence of peering-set names and AS Numbers separated by colons. At least one part of the name must be an actual **peering-set** name (that means that it should start with "prng-").
- The "peering:" attribute defines a peering that you can use to import or export routes.
- The "mp-peering:" attribute extends the "peering:" attribute to use IPv6 addresses.
- The "peering:" and "mp-peering:" attributes are optional. However, a **peering-set** object must contain at least one of these two attributes. It cannot contain both within the same object.

1.2.14 person

A **person** object contains information about the technical or administrative contact(s) responsible for an object. After it has been created, the "person:" attribute cannot be changed by users. It can be changed by the Database Administrator.

The person object is identified by the "nic-hdl:" attribute. This is a label, usually made up from the initials of the person’s name and the database "source:" (for example, DW-RIPE).

The "nic-hdl:" is used by other objects to reference the person.

1.2.15 poem

A **poem** object contains poems that users submit.

1.2.16 poetic-form

A **poetic-form** object defines the supported **poem** types.

1.2.17 role

A **role object** is similar to a **person** object. However, instead of describing a single person, it describes a role performed by one or more people. This might be a helpdesk, network monitoring centre, system administrator, etc. A **role** object is useful since often a person performing a specific job may change; however, the role itself remains. The "nic-hdl:" attributes of the **person** and **role** objects share the same name space. After it has been created, the "role:" attribute cannot be changed by users. It can be changed by the Database Administrator.

1.2.18 route

Each interAS route (also known as an interdomain route) originated by an AS can be specified by using a **route** object.

- The "route:" attribute is the address prefix of the route.
- The "origin:" attribute is the AS Number of the AS that originates the route into the interAS routing system.

1.2.19 route6

A **route6** object specifies an interAS IPv6 route, originated by an AS.

- The "route6:" attribute is the IPv6 address prefix of the route.
- The "origin:" attribute is the AS Number of the AS that originates the route into the interAS routing system.

1.2.20 route-set

A **route-set** object is a set of route prefixes, not of database **route** objects. The "route-set:" attribute defines the name of the set. It is an RPSL name that starts with "rs-". It defines a set of routes that can be represented by **route** objects or by address prefixes.

- In the first case, the set is populated by means of the "mbrs-by-ref:" attribute, in the latter, the members of the set are explicitly listed in the "members:" attribute.
- The "members:" attribute is a list of address prefixes or other route-set names.
- The "mp-members:" attribute is a list of IPv6 address prefixes or other route-set names.

- The name of a **route-set**object can be hierarchical. A hierarchical route-set name is a sequence of route-set names and AS numbers separated by colons. The first component of such a name must be an actual **route-set** name (that means that it should start with "rs-").

1.2.21 rtr-set

A **rtr-set** object defines a set of routers.

A set may be described by the "members:" attribute that is a list of inet-rtr names, IPv4 addresses or other rtr-set names. The "mp-members:" attribute extends the "members:" attribute to use IPv6 addresses.

A set may also be populated by means of the "mbrs-by-ref:" attribute, in which case it is represented by **inet-rtr** objects.

The "rtr-set:" attribute defines the name of the set. It is an RPSL name that starts with "rtrs-".

The name of a **rtr-set** object can be hierarchical. A hierarchical rtr-set name is a sequence of rtr-set names and AS Numbers separated by colons. The first component of such a name must be an actual rtr-set name (that means that it should start with "rtrs-").

2.0 Querying the RIPE Database

This section describes how you can query to find information in 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 three 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.

There is a set of general rules about server responses:

The same response will be returned from the server for each of the three methods of querying the database shown above.

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, whilst 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.

An empty line in the output is used to separate objects. 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.

When using the referral mechanism, the output of the referred server is passed to the client without modification.

The general format of a query is:

```
[optional query flags] <query argument>
```

The format of a query flag is:

```
-query_flag [optional query flag argument]
```

You may list each query flag separately as in:

```
-B –i tech-c –G ABC999-RIPE
```

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

```
-BGi tech-c 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 on the Internet, mainly in the RIPE 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 example193.0.0.0 - 193.0.3.255, this is the same range as in the above example).

The **inet6num** objects represent an 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 of length 1 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 notification 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 your 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. The exact match is returned, if found. If it cannot find an exact match, 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

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 four query flags.

These flags ("-M", "-m", "-L" and "-l") provide two generic types of queries known as more and less specific queries.

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.

- The "-M" flag 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.
- The "-m" flag 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.

- The "-L" flag 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.
- The "-l" flag 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 may be referenced from **inet(6)num** objects using the "mnt-irt:" attribute. This shows which CSIRT is responsible for handling computer and network incidents for that address range, as well as who is responsible for handling abuse complaints.

Not every **inet(6)num** object needs to contain a reference to the irt object that applies to its 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. 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

Support for 32 bit AS numbers was added in January 2007. This introduced a new, extended syntax for AS numbers. The general format for an AS number in the RIPE Database is:

```
ASx.y where x and y are 16 bit values.
```

For 16 bit numbers, where x=0, the high end value is omitted and it is represented as:

```
ASy
```

Queries for 32 bit numbers must be in 32 bit format. Queries for 16 bit numbers are accepted in either format. If a 16 bit number is entered as a 32 bit number (AS0.y) it is converted internally into the corresponding 16 bit number. In this case a warning message is returned with the query about changing the lookup key.

For as-block objects the same extended syntax rules apply. So any of these are acceptable as a query key:

```
AS123 - AS456
AS123 - AS0.456
AS0.123 - AS1.345
```

Warnings will be generated when 16bit numbers are entered as 32bit numbers.

2.4 Inverse Queries

Inverse queries are performed using an object's inverse key as an argument to a query. The query flag "-i" 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 arguments to the "-i" query flag and the corresponding inverse keys.

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 -i mnt-by,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 optional "abuse-mailbox:" attribute is available in the following objects:

- **person**
- **role**
- **irt**
- **organisation**
- **mntner**

This optional attribute contains at least one e-mail address. There is a "-b" query flag to make it easy to find the "abuse-mailbox:" attributes for any specific range.

This flag will instruct the whois server to first of all do a query as if it had been given a "-c" query flag (See Section 2.2.4, "Less Specific Queries Referencing irt Objects"). If a reference to an **irt** object is found, the "abuse-mailbox:" e-mail address will be taken from the **irt** object. The "abuse-mailbox:" e-mail address will also be taken from any of the other objects associated with the **inetnum** or **inet6num** object that references the **irt** object.

The output returned from the server will contain a brief summary. This summary will only include the **inetnum** or **inet6num** range of the queried object. Also the prefix of any corresponding **route** or **route6** objects, followed by the "abuse-mailbox:" attributes from all of the associated objects.

If no **irt** object reference is found in any encompassing objects then no object summaries will be returned.

If a summary is returned, it will include the correct e-mail address for complaints about spam and other network abuse. 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 any of these flags:

"-KFrGB"

2.6 Grouping the RIPE Database Output

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

One way is for the first part of the results to list the main objects like **inetnum** and **mntner**. Then the second part of the results lists all the objects associated with the main objects, like **organisation** and **person**. If any of these associated objects, like a **person** object, is 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 them. In this way, each of the main objects is followed immediately by all of its associated objects. The associated objects may appear more than once in this type of output. The default output is grouped. If you include the "-G" 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. Other attributes, also containing e-mail addresses, are filtered if one of the returned objects includes an "abuse-mailbox:" attribute.

The stricter filtering has slightly different behaviour depending on the use of grouping. If the output is grouped (See Section 2.6, "Grouping the RIPE Database Output"), then each

group is treated separately. If any object within a group of objects includes an "abuse-mailbox:" attribute, then the stricter filtering will apply to that group. In a group with no objects including an "abuse-mailbox:" attribute, the more basic filtering will apply to that group.

When the output is not grouped (using the "-G" query flag) the whole output is taken as a single group. If any one object includes an "abuse-mailbox:" attribute then the whole output is subject to the more strict filtering.

- The attributes that are always filtered out are "changed:" and "notify:"
- The additional attributes that are filtered out with the more strict case are "e-mail:" and "ref-nfy:"
- The default output is filtered. If you include the "-B" 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.

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 suite of routing policy analysis tools maintained by the Information Sciences Institute at the University of Southern California. Some of the tools in this set access Routing Registry servers through an authorisation whois interface.

Versions 3.0 and above of the RIPE Database Server include 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" 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" 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 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:" attribute of the **set** object.

2.10 More and Less Specific Lookups For Reverse Domains

Versions 3.0 and above of the RIPE Database support 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.

2.11 Referral Mechanism for Domains

The referral mechanism is for administrators of domain registries. It allows the whois server to reply to the user by fetching data from the domain registry database rather than from the local RIPE Database data. There are three elements to this mechanism. The "refer:" attribute, domain name stripping and the "-R" query flag.

2.11.1 The “refer:” Attribute

This attribute can be included in a **domain** object in the RIPE Database. When present, it allows the database software to pass the query on to the authoritative server of a ccTLD. This server can return information about a **domain** object to the RIPE Database server to pass back to the user.

This attribute specifies the hostname, port and referral type that the RIPE Database server should use to redirect the query. See Appendix A1, "Object Attributes" for the syntax of this attribute.

2.11.2 Domain Name Stripping

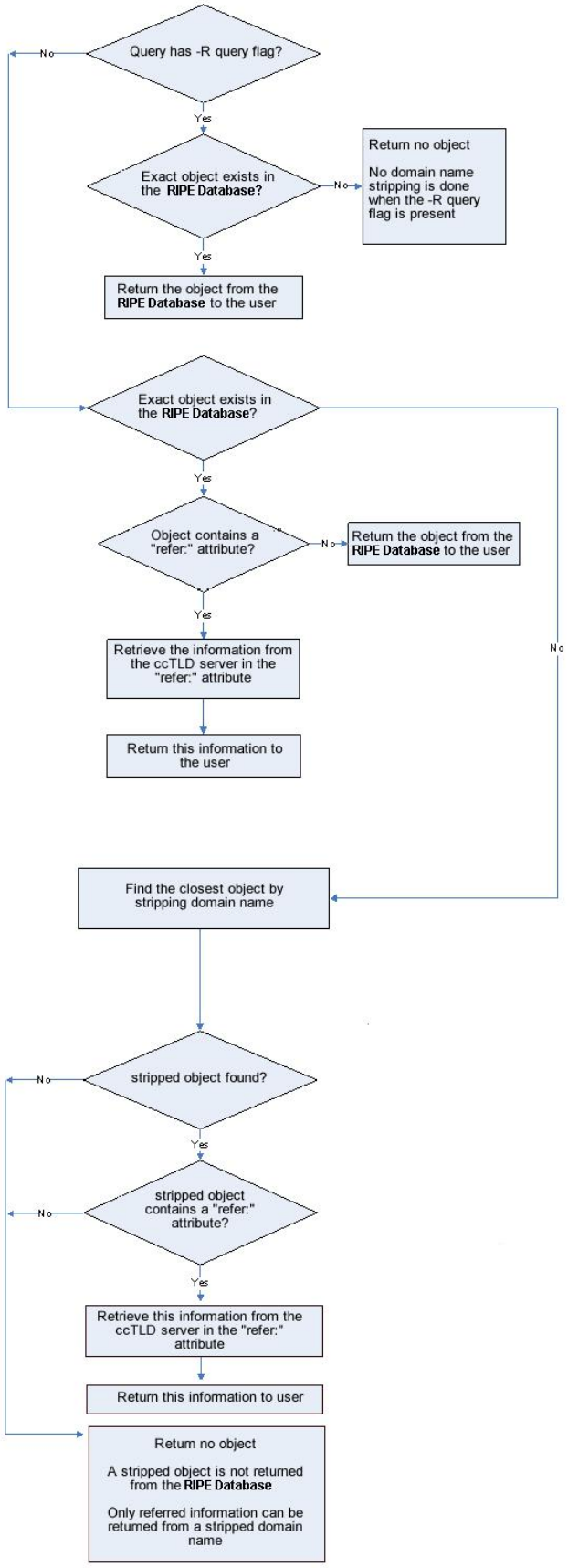
When no matching **domain** object is found in the RIPE Database with the name specified in the query argument, the domain name is stripped towards higher-level domains (xxx.yyy.zzz becoming yyy.zzz). The query is repeated with the shorter domain name. The cycle of stripping and repeating the query continues until a **domain** object is found or the search string becomes empty.

2.11.3 The “-R” Query Flag

You can force retrieval of an object from the RIPE Database without using the referral mechanism. This is done by including the "-R" query flag. When this flag is included in the query to the RIPE Database server, no referrals are made, not even if the object returned contains a "refer:" attribute.

2.11.4 The Referral Process

The software follows a specific process when checking the three elements of the referral mechanism. This process is best described by using a flow chart as follows:


Query has -R query flag?
No
Yes
Exact object exists in the RIPE Database?
No
Return no object
No domain name stripping is done when the -R query flag is present
Yes
Return the object from the RIPE Database to the user
Exact object exists in the RIPE Database?
Yes
Object contains a "refer:" attribute?
No
Return the object from the RIPE Database to the user
Yes
Retrieve the information from the ccTLD server in the "refer:" attribute
Return this information to the user
No
Find the closest object by stripping domain name
stripped object found?
No
Yes
stripped object contains a "refer:" attribute?
No
Yes
Retrieve this information from the ccTLD server in the "refer:" attribute
Return this information to user
Return no object
A stripped object is not returned from the RIPE Database
Only referred information can be returned from a stripped domain name

2.12 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 queries made for an IP address. Access is determined by 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 running cgi interfaces 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. Before you can, 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. A counting algorithm with defined limits temporarily blocks access when limits are reached within certain periods of time. This block will be automatically released as time goes on to allow querying to continue. The count is reduced by half every 12 hours. 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.

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" flag to prevent the 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. The 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 decreases. When it reaches zero, the query execution is aborted and the connection is terminated, displaying an error message to the client (see "Access errors" in Appendix A2, "RIPE Database Query Server Response Codes and Messages"), also a count of denials increases. Retrieving any other object type does not affect these counters.

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.

2.13 Other Server Features

2.13.1 RIPE NCC Mirrors of Other Databases

The RIPE NCC mirrors several other databases, in near real time, that are considered to be of interest to users of the RIPE Database. 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. Periodically, updates are requested 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 2.13.2.

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 the RIPE Database syntax before submission to the local database copy. This means that the data returned to a query of 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 RIPE NCC mirrored databases is not possible. The Access Control mechanism described in Section 2.12 applies to all data returned, regardless of the source.

2.13.2 The "–q" Query Flag

The RIPE Database server (Versions 3.0 and higher) supports the retrieval of certain information about itself and the data sets served, using a "-q" query flag.

The "-q" query flag tells 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. The information returned will contain one line for each available source in the following format:
  <source>:<NRTM_protocol_version_#>:<mirroring>:<first>-<last>
  where:
  - <source> is the string that identifies the database (for example RIPE)
  - <NRTM_protocol_version_#> identifies the version of the mirroring protocol (1 or 2)
  - <mirroring> can take one of the following values:
    -- Y/N/X – can mirror/cannot mirror/obscured (see table below)
  - <first> is the lowest serial number available
  - <last> is the most recent serial number available

The following semantics apply for –q sources <mirroring> state:

Y:<first>-<last>	Mirroring is allowed – serials from within range first-last available.
N:<first>-<last>	Mirroring not allowed for administrative reasons.
N:0-<last>	Mirroring physically not possible. (This may be a static snapshot of serial last)
X:<message>	No mirroring allowed. An explanation will be given in the <message>.

2.13.3. The “-t” Query Flag

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

2.13.4. The “-v” Query Flag

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

2.13.5. The “-F” 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.

2.13.6. The “-K” 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.

2.13.7. The “-T” Query Flag

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

2.13.8. The “-a” 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.

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-	**as-block** object whose primary key defines a range that matches or fully contains the range specified

number> separated by "-")	in the query argument.
<domain-name>	**domain** and **inet-rtr** objects whose primary keys match the query argument. For domains, a referral query may be performed. In such case, the actual query is performed by the referred server and the results are transparently passed to the client.
<irt-name>	**irt** object whose primary key matches the query argument.
<Person-name>	**person** and **role** objects whose "person:" or "role:" attributes contain the name specified in the query argument.
<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-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	<ip-lookup>	Only an exact match on the prefix. If no exact match is found, no objects are returned.
-M	<ip-lookup>	All level more specific **inetnum**, **inet6num, route** or **route6** objects, excluding exact matches.
-m	<ip-lookup>	First level more specific **inetnum**, **inet6num, route** or **route6** objects, excluding exact matches.
-L	<ip-lookup>	All level less specific **inetnum**, **inet6num, route** or **route6** objects, including exact matches.
-l	<ip-lookup>	First level less specific **inetnum**, **inet6num, route** or **route6** objects, excluding exact matches.
-d	<ip-lookup>	Enables lookups on reverse delegation domains. Can be used with "-x", "-M", "-m", "-L" and "-l" flags.
-c	<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 smallest, less specific **inetnum** or **inet6num** object found encompassing the previously returned **inetnum** or **inet6num** object.
-b		Provides a brief output of ranges with associated abuse contact information.

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

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> or <person-name>	Objects whose "admin-c:" attributes match the query argument.
ah (author)	<nic-handle> or <person-name>	**poem** objects whose "author:" attribute matches the query argument.
at (auth)	<key-cert-id>	**mntner** objects whose "auth:" attribute matches the query argument. Please note that encrypted passwords cannot be inverse-searched, but only PGPKEY and X509 certificates.
fp (fingerprint)	<fingerprint>	**key-cert** objects whose "fingerpr:" attribute matches the query argument.
fr (form)	<poetic-form>	**poem** objects whose "form:" attribute matches the query argument.
pn (person)	<nic-handle> or <person-name>	Objects whose "admin-c:", "tech-c:", "zone-c:" 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-by:" 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> or <ip-lookup> (IPv4/IPv6 address)	**domain** objects whose "nserver:" attribute matches the query argument.
or (origin)	<as-name>	**route** and **route6** objects whose "origin:" attribute matches the query argument.
org	<org-id>	Objects whose "organisation:" attribute matches the query argument.
rb (referral-by)	<mntner-name>	**mntner** objects whose "referral-by:" attribute matches the query argument.
rz (rev-srv)	<domain-name> or <ip-lookup> (IPv4/IPv6 address)	**inetnum** and **inet6num** objects whose "rev-srv:" attribute matches the query argument.
sd (sub-dom)	<domain-name>	**domain** objects whose "sub-dom:" attribute matches the query argument.
tc (tech-c)	<nic-handle> or <person-name>	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> or <person-name>	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.
-K		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	(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.
-g	(mirroring request)	Request a NRTM stream from the server.

Table 2.5 Miscellaneous Queries

Flag	Flag Argument	Effect
-R		Switches off use of the referral mechanism for domain lookups, so that the database returns an object in the RIPE database with the exact match with the lookup argument, rather than doing a referral lookup.
-r		Switches off recursion for contact information after retrieving the objects that match the lookup key.
-B		Switches off filtering of objects.
-G		Switches off grouping of associated objects.
-T	Comma separated list of object types, no white space is allowed.	Restricts the types of objects to lookup in the query.
-a		Specifies that the server should perform lookups in all available sources. See also "-q sources" query.
-s	Comma separated list of sources, no white space is allowed.	Specifies which sources and in which order are to be looked up when performing a query.

Table 2.6 Informational Queries

The following notations are used in this table:
<object-type> means full or abbreviated name of a specific class;
<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	<object-type>	Requests a template for the specified object type.
-v	<object-type>	Requests a verbose template for the specified object type.
-V<client-tag>		Sends information about the client to the server.

Appendices

A1. Object Attributes

These are the definitions of the object attributes supported by the RIPE Database.

address:
Full postal address of a contact in free form.

admin-c:
References an on-site administrative contact.

aggr-bndry:
Defines a set of ASNs, which form the aggregation boundary.

aggr-mtd:
Specifies how the aggregate is generated. Please see [1] for more information.

alias:
Specifies a canonical DNS name for the router.

as-block:
Specifies the range of ASNs that the **as-block** object represents. Please see [2] for more information.

as-name:
A descriptive name associated with an AS.

as-set:
Defines the name of the set.

auth:
<auth-scheme> <scheme-info>
Defines an authentication scheme to be used.
<auth-scheme> and <scheme-info> can take the following values:

<auth-scheme>	<scheme-info>	Description
MD5-PW	Encrypted password based on MD5 hash.	This scheme is based on the MD5 hash algorithm. The authentication information stored in the database is a pass phrase encrypted using md5-crypt algorithm, which is a concatenation of the "1" string, the salt, and the 128-bit hash output. Because it uses eight-character salt and an almost unlimited pass phrase, this scheme is more stable against dictionary attacks. However, since the encrypted form is exposed it cannot be considered as a strong form of authentication.
PGPKEY	-<id>	Strong scheme of authentication. <id> is the PGP key ID to be used for authentication. This string is the same one that is used in the corresponding **key-cert** object's "key-cert:" attribute.
X.509	-<id>	Strongest scheme of authentication. <id> is the auto-generated ID of the X.509 certificate to be used for authentication. This string is the same one that is used in the corresponding **key-cert** object's "key-cert:" attribute.

author:
References a poem author.

aut-num:
The Autonomous System Number.

certif:
Contains the public key for a PGP Key or an X.509 certificate. The value of the public key should be supplied either using multiple "certif:" attributes, or in one "certif:" attribute. In the first case, this is easily done by exporting the key from your local key ring in ASCII armoured format or the certificate from your browser and starting each line of the key with the string "certif:". In the second case, line continuation should be used to represent the key. All the lines of the exported key must be included. For PGP, this includes the begin and end markers and the empty line which separates the header from the key body. For X.509 certificates, this includes the BEGIN CERTIFICATE and END CERTIFICATE lines.

changed:
Specifies who submitted the update, and when the object was updated. The format of the date is YYYYMMDD.

components:
The "components:" attribute defines what component routes are used to form the

aggregate.

Please refer to RFC 2622 [1] and RPSLng [14] for more information.

country:
Identifies the country.

default:
Specifies default routing policies. Please refer to RFC 2622 [1] for more information.

descr:
A short description related to the object in free form.

domain:
DNS name.

dom-net:
List of IP networks in a domain.

e-mail:
Specifies an e-mail address of a **person**, **role**, **organisation** or **irt** team.

encryption :
References a **key-cert** object representing a CSIRT public key used to encrypt correspondence sent to the CSIRT.

export:
Specifies an export policy expression. Please refer to RFC 2622 for more information.

export-comps:
Specifies an RPSL filter that matches the more specifics that need to be exported outside the aggregation boundary. Please refer to RFC 2622 [1] and RPSLng [14] for more information.

fax-no:
The fax number of a contact.

filter:
Defines the **set**'s policy filter, a logical expression which when applied to a **set** of routes returns a subset of these routes. Please refer to RFC 2622 [1] for more information.

filter-set:
Defines the name of the filter. Please refer to RFC 2622 [1] for more information.

fingerpr:
A fingerprint of a key certificate generated by the database. Please refer to RFC 2726 [9] for a detailed description of this attribute

form:
Specifies the identifier of a registered **poem** type.

holes:
Lists the component address prefixes that are not reachable through the aggregate route (perhaps that part of the address space is unallocated). Please refer to RFC 2622 [1] and RPSLng [14] for more information.

ifaddr:
Specifies an interface address within an Internet router. Please refer to RFC 2622 [1] for more information.

import:
Specifies import policy expression. Please refer to RFC 2622 [1] for more information.

inetnum:
Specifies a range of IPv4 addresses.

inet6num:
Specifies a range of IPv6 addresses in prefix notation.

inet-rtr:
Fully qualified DNS name of the **inet-rtr** without trailing ".". Please refer to RFC 2622 [1] for more information.

inject:
Specifies which routers perform the aggregation and when they perform it. In **route** objects, the router expression can contain only IPv4 expressions, and in **route6** objects, it can only contain IPv6 expressions. Please refer to RFC 2622 [1] and RPSLng [14] for more information.

interface:
Specifies a multiprotocol interface address within an Internet router. Please refer to RPSLng [14] for more information.

irt:
A unique identifier of an **irt** object.

irt-nfy:
Specifies the e-mail address to be notified when a reference to the **irt** object is added or removed.

key-cert:
Defines the public key stored in the database.

local-as:
Specifies the autonomous system that operates the router. Please refer to RFC 2622 [1] for more information.

method:
Defines the type of the public key. Currently the only methods that are supported are "PGP" and "X.509". Please refer to RFC 2726 [9] for detailed description of this attribute.

member-of:
This attribute can be used in the **route**, **route6**, **aut-num** and **inet-rtr** classes. The value of the "member-of:" attribute identifies a **set** object that this object wants to be a member of. This claim, however, should be acknowledged by a respective "mbrs-by-ref:" attribute in the referenced object. Please refer to RFC 2622 [1] for more information.

members:
Lists the members of the **set**. Please refer to RFC-2622 [1] for more information.

mbrs-by-ref:
This attribute can be used in all **set** objects; it allows indirect population of a set. If this attribute is used, the **set** also includes objects of the corresponding type (**aut-num** objects for **as-set**, for example) that are protected by one of these maintainers and whose "member-of:" attributes refer to the name of the **set**. If the value of a "mbrs-by-ref:" attribute is ANY, any object of the corresponding type referring to the **set** is a member of the **set**. If the "mbrs-by-ref:" attribute is missing, the **set** is defined explicitly by the "members:" attribute.

mntner:
A unique identifier of the **mntner** object.

mnt-by:
Specifies the identifier of a registered **mntner** object used for authorisation of operations performed with the object that contains this attribute.

mnt-domains:
Specifies the identifier of a registered **mntner** object used for reverse **domain** authorisation. Protects **domain** objects. The authentication method of this **maintainer** object will be used for any encompassing reverse **domain** object.

mnt-irt:
May appear in an **inetnum** or **inet6num** object. It points to an existing **irt** object representing CSIRT that handles security incidents for the address space specified by the **inetnum** or **inet6num** object.

mnt-lower:
Specifies the identifier of a registered **mntner** object used for hierarchical authorisation. Protects creation of objects directly (one level) below in the hierarchy of an object type (only for **inetnum**, **inet6num**, **as-block**, **aut-num**, **route**, **route6** or **domain** objects). The authentication method of this **mntner** object will then be used upon creation of any object directly below the object that contains the "mnt-lower:" attribute.

mnt-nfy:
Specifies the e-mail address to be notified when an object protected by a **mntner** is successfully updated.

mnt- ref:
Specifies the **mntner** objects that are entitled to add references to the **organisation** object from other objects.

mnt-routes:
May appear in an **aut-num**, **inetnum**, **inet6num**, **route** or **route6** object. This attribute references a **maintainer** object that is used in determining authorisation for the creation of **route** and **route6** objects. After the reference to the maintainer, an optional list of prefix ranges inside of curly braces or the keyword "ANY" may follow. The default, when no additional set items are specified, is "ANY" or all more specifics. Please refer to RFC-2622 [1] and RPSLng [14] for more information.

mp- default :
Specifies default multiprotocol routing policies. Please refer to RPSLng [14] for more information.

mp- export:
Specifies a multiprotocol export policy expression. Please refer to RPSLng [14] for more information.

mp-filter:
Defines the **set**'s multiprotocol policy filter. Please refer to RPSLng [14] for more information.

mp-import:
Specifies multiprotocol import policy expression. Please refer to RPSLng [14] for more information.

mp-members :
Lists the multiprotocol members of the set. Refer to RPSLng [14] for more information.

mp-peer:
Specifies the details of any (interior or exterior) multiprotocol router peerings. Please refer to RPSLng [14] for more information.

mp-peering:
Defines a multiprotocol peering that can be used for importing or exporting routes. Please see RPSLng [14] for more information.

netname:
Specifies the name of a range of IP address space.

nic-hdl:
Specifies the NIC handle of a **role** or **person** object.

notify:
Specifies the e-mail address to which notifications of changes to an object should be sent.

nserver:
Specifies the nameservers of the domain.

org:
This attribute may appear in any object type. It points to an existing organisation object representing the entity that holds the resource, (in the cases where the RIPE Database object represents an Internet resource). In other objects, it can be used to specify the business relations. The value of this attribute is the ID of the organisation object.

The "org:" attribute is used to specify the holder of a resource in inetnum, inet6num and aut-num objects. In other objects, it specifies business relations (such as a person object, where it can be used to specify whom the person works for).

org-name:
Specifies the name of the organisation that this organisation object represents in the RIPE Database.

org-type:
Specifies the type of the organisation. The possible values are IANA for Internet Assigned Numbers Authority, RIR for Regional Internet Registries, NIR for National Internet Registries, LIR for Local Internet Registries, and NON-REGISTRY for all other organisations. Note that in the RIPE NCC service region there are no National Internet Registries, therefore in the RIPE Database there will not be any organisation object with this value in the "org-type:" attribute.

organisation :
Specifies the ID of an **organisation** object.

origin:
Specifies the AS that originates the route. The corresponding **aut-num** object should be registered in the database.

owner:
Specifies the owner of the public key. Please refer to RFC 2726 [9] for detailed description of this attribute.

peer:
May appear in an **inet-rtr** object. Specifies a protocol peering with another router. Please refer to RFC 2622 [1] for more information.

peering:
Defines a peering that can be used for importing or exporting routes. Please refer to RFC 2622 [1] for more information.

peering-set:
Specifies the name of the peering-set. Please refer to RFC 2622 [1] for more information.

person:
Specifies the full name of an administrative, technical or zone contact person for other objects in the database.

peering-set:
Specifies the name of the peering-set. Please refer to RFC 2622 [1] for more information.

phone:
Specifies a telephone number of the contact.

poem:
Specifies the title of a poem.

poetic-form:
Specifies the poem type.

ref- nfy:
Specifies the e-mail address to be notified when a reference to the **organisation** object is added or removed. An e-mail address as defined in RFC 2822 [1].

refer:
<type> <hostname> [<port>]
Specifies the referral type, hostname and port that the server should use to redirect the query when using referral mechanism for lookups for **domain** objects. Please see Section 2.11, "Referral Mechanism for Domains" for more information.

<type> specifies the type of referral to be used. Please see the table below for the supported types.
<hostname> is the DNS name or <ipv4 address> of the referred host.
<port> is an integer specifying TCP port number at which queries are accepted by the referred host. If <port> is omitted, the default number of 43 is used.

Referral type	Description
SIMPLE	Only lookup key (domain name) is passed to the referred server. All query flags are stripped.
INTERNIC	Same as SIMPLE. Supported for backward compatibility.
RIPE	Used when the referred server understands RIPE query flags. With this type of referral, all query flags specified by the client will be passed to the referred server unmodified.
CLIENTADDRESS	Same as SIMPLE, but the server will add "-V <version>, <ipv4 address>" flag to the query, where <version> is the version number of the server and <ipv4 address> is the IP address of the client that made this query. This referral type allows the referred host to perform accounting and implement an access control for clients using the RIPE Database server as a proxy.

referral-by:
This attribute is required in the **mntnr** object. It is not currently used by the database software.

remarks:
Contains remarks in free form.

role:
Specifies the full name of a role entity, e.g. RIPE DBM.

rev-srv:
Specifies a DNS nameserver for a range of IP addresses represented by the **inetnum** object that contains this attribute. It is for documentation only and has no effect on reverse delegation.

route:
Specifies the prefix of the interAS route. Together with the "origin:" attribute, makes up a primary key of the **route** object.

route6:
Specifies an IPv6 prefix. This is the prefix of the interAS route. Together with the "origin:" attribute, makes up a primary key of the **route6** object.

route-set:
Specifies the name of the route set. It is a primary key for the **route-set** object. Please refer to RFC 2622 [1] for more information.

rtr-set:
Defines the name of the **rtr-set**. Please refer to RFC 2622 [1] for more information.

signature:
References a **key-cert** object representing a CSIRT public key used by the team to sign their correspondence.

source:
Specifies the registry where the object is registered. Should be "RIPE" for the RIPE Database.

status:
<status>
Specifies the status of the address range represented by **inetnum** or **inet6num** object. For an **inetnum** object <status> must have one of these values:

ALLOCATED PA
ALLOCATED PI
ALLOCATED UNSPECIFIED
LIR-PARTITIONED PA
LIR-PARTITIONED PI
SUB-ALLOCATED PA
ASSIGNED PA
ASSIGNED PI
NOT-SET
EARLY-REGISTRATION

Please refer to the RIPE Document "IPv4 Address Allocation and Assignment Policies in the RIPE NCC Service Region" for further information. Please refer to [10] regarding usage of the LIR-PARTITIONED status value.

For **inet6num**, <status> can have one of the following values:

ALLOCATED-BY-RIR - For allocations made by an RIR to an LIR.
ALLOCATED-BY-LIR - For allocations made by an LIR or an LIR's downstream customer to another downstream organisation.
ASSIGNED - For assignments made to End User sites.

Please refer to [13] regarding usage of the status value for **inet6num** objects.

sub-dom:
Specifies list of sub-domains of a domain. Domain names are relative to the domain represented by the **domain** object that contains this attribute.

tech-c:
References a technical contact.

text:
Contains text of the poem. Must be humorous, but not malicious or insulting.

upd-to:
Specifies the e-mail address to be notified when an object protected by a **mntner** is unsuccessfully updated.

zone-c:
References a zone contact.

A2. 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.
% Rights restricted by copyright.
% See http://www.ripe.net/db/copyright.html
%ERROR:101: no entries found
%
% No entries found in the selected source(s).
```

A2.1 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: as-block range invalid
Querying for ASa – Asb where a>b.

A2.2 Access Errors

%ERROR:201: access denied
Access from the host has been 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 2.12, "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 2.12, "Access Control for Queries" for more information.

%ERROR:204: maximum referral lines exceeded
The referral query result exceeded a set maximum number of lines. Only the maximum number of lines is output and then, the whois server closes the connection.

%ERROR:205: multiple addresses passed by proxy
The query included more than one "-V" query flag.

%ERROR:208: connection refused
The maximum number of simultaneous connections from your host has been exceeded.

A2.3 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.

A2.4 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.

%ERROR:402: not authorised to mirror the database
See section 2.11 "Access control for queries" for more information. You may use "-q sources" query to get more information about the NRTM source.

%ERROR:403: unknown source
The database identified by the <source> is not served by the server. Use "-q sources" for a list of available sources.

A2.5 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.

%WARNING:904: useless no-referral flag passed
The "-R" query flag was used in a query that did not have a domain object as the argument.

%WARNING:905: fixed lookup key
The IP address has been changed to a different format for the query.

A2.6 Referral Text

```
% The object shown below is NOT in the RIPE Database.
% It has been obtained by querying a remote server:
% <server-name> at port 43.
% To see the object stored in the RIPE Database
% use the -R flag in your query
%
%REFERRAL START
<remote server output>
%REFERRAL END
```

The output from the remote server is returned between the lines "%REFERRAL START" and "%REFERRAL END".

A3. Copyright Information

A3.1 RIPE Database Copyright

The information in the RIPE Database is available to the public for agreed Internet operation purposes, but is under copyright. The copyright statement at the time of publishing this manual is:

"*Except for agreed Internet operational purposes, no part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without prior permission of the RIPE NCC on behalf of the copyright holders. Any use of this material to target advertising or similar activities is explicitly forbidden and may be prosecuted*."

You can find the copyright statement at:
http://www.ripe.net/db/copyright.html

A3.2 RIPE NCC Copyright

© RIPE NCC. All rights reserved.

Acknowledgements

The authors wish to acknowledge the effort done by the original developers of the version 3.0 of the RIPE Database at the RIPE NCC: Daniele Arena, Marek Bukowy, Engin Gunduz, Roman Karpiuk, Shane Kerr, A.M.R. Magee, Chris Ottrey and Filippo Portera. Those who have continued its development include Can Bican, Katie Petrusha, Denis Walker, Tiago Antao and Agoston Horvath.

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