RIPE NCC Database Documentation

A. M. R. Magee
RIPE NCC

Document: ripe-153
Date: 28th January, 1997.
Version: 1.3

ABSTRACT

This document is an introduction to the use of the RIPE Network Management Database (the "RIPE Database"). It contains details of how to query the database and how to create, change and delete information.

It also contains instructions on authorizing and authenticating changes to database objects.

Items marked with an "*" character are still in production.

Questions and comments about this document should be sent to `<ripe-dbm@ripe.net>`.

Acknowledgements

The author wishes to acknowledge the cooperation given by the following staff at the RIPE NCC: D. Karrenberg, C. Orange, M. Kühne, P. Caslav, C. Fletcher, E. McGuinness, R. Wilhelm and E. Willems.

Copyright Declaration on the RIPE Database

It should be stressed that this copyright is asserted to prevent the abuse and undue exploitation of the RIPE Network Management Database.

Outline of this document

Software Requirements

1. Background to the RIPE Network Management Database;

- Description of the objects in the database
- Management of the objects

2. Using the Database

- Querying the database
- Creating, Updating and Deleting Objects
- Authorization and authentication of changes
- Hierarchical object protection*
- Mirroring*

3. Use of the software to run a private database*

Appendix

- Attributes*
- Objects*
- TEST Database*
- Glossary*

- Mailing Lists
- A brief history of RIPE, the NCC and the Database.*

Software Requirements

To access the RIPE Network Management Database, you will need one of several TCP/IP based tools, namely :

- ‘whois’ client, RIPE ‘whois’ client
- WWW,
  http://www.ripe.net/db/
- WAIS
- Telnet
- E-mail

The database software uses a ‘whois’ (RFC954) server with some special RIPE specific extensions. Therefore, a ’whois’ client is generally the preferred tool for database look-ups, in particular, the RIPE version. The source code of the RIPE ’whois’ client program, which runs on UNIX, can be obtained from :

ftp://ftp.ripe.net/tools/ripe-whois-tools-VERSION.tar.gz

Changes to the RIPE Database can only be made by e-mail. They can not be made using the web-site.

The RIPE database has its own interface software. A public version of this software is available at

ftp://ftp.ripe.net/ripe/dbase/software

This software may be used by anyone to either set up a mirror of the RIPE database or to make a private database. This software is covered by the copyright below:

Copyright (c) 1997 The TERENA Association

All Rights Reserved
Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of the author not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS; IN NO EVENT SHALL THE

1.1 The RIPE Network Management Database

Purpose of the RIPE Database:

One of the activities of RIPE is to maintain a database of European IP networks, DNS domains, their contact persons and IP routing policies (i.e. the RIPE Routing Registry). This data is maintained by RIPE NCC along with various other kinds of network management information. This database is known as the RIPE Network Management Database or more usually, the "RIPE Database". The information in the database is available to the public for agreed Internet operational purposes. This supports NIC's/NOC's all over Europe and beyond to perform their respective tasks.

For example, if a user in network A cannot reach a machine in network B, then the network manager of network A could find out the technical contact person of network B and ask them to locate and solve the problem.

The RIPE database is not a "telephone directory" or "white pages" service for European Internet users.

1.2 Description and Management of the Database Objects

The RIPE Network Management Database (often called the "RIPE Database") contains records of

- allocations and assignments of IP address space (the IP address registry);
- domain names (the domain registry);
- routing policy information (the routing registry);
- contact information (details of people who are responsible for the operation of networks or routers, and those who are responsible for maintaining information in the RIPE Database).

Note: the information in the domain registry has no effect on operations; it is there as a convenient reference.

Records in the RIPE Database are known as "objects". Each object describes a single entity in internet operations. This implies that information about that entity should only be contained in the corresponding database object and should not be repeated in other objects.

The phrase "RIPE Database" is often used to refer to the interface software, rather than the information that is stored in the database. In ambiguous situations, it will made clear what is being discussed at any time.

Objects are composed of pairs of attributes and values e.g. the "person" object is composed of the following attributes:

person
address
phone
fax-no
e-mail
nic-hdl
remarks
notify
mnt-by
changed
source

Each of these attributes has a value, e.g.

```
person:      Ambrose Magee
address:     RIPE Network Co-ordination Centre (NCC)
address:     Kruislaan 409
address:     NL-1098 SJ   Amsterdam
address:     The Netherlands
phone:       +31 20 592 5065
fax-no:      +31 20 592 5090
e-mail:      ambrose@ripe.net
nic-hdl:     AMRM1-RIPE
notify:      ambrose@ripe.net
mnt-by:      AMRM1-RIPE-MNT
changed:     ambrose@ripe.net 970114
source:      RIPE
```

Similarly, the *domain* object, which represents a range of IP addresses, has the following attributes:

domain
descr
admin-c
tech-c
zone-c
nserver
sub-dom
dom-net
remarks
notify

mnt-by
mnt-lower
changed
source

An example of a domain object is:

```
domain:       over.ripe.net
descr:        Network for ripe-dbm use.
descr:        RIPE Network Coordination Centre
descr:        Kruislaan 409
descr:        NL-1098 SJ Amsterdam
descr:        The Netherlands
admin-c:      AMRM1-RIPE
tech-c:       AMRM1-RIPE
zone-c:       AMRM1-RIPE
mnt-by:       AMRM1-RIPE-MNT
changed:      ripe-dbm@ripe.net 970115
source:       RIPE
```

It may be seen that some attributes may appear more than once. Such an attribute is called "multiple". If an attribute can only appear once, it is called "single".

Some attributes must be in the object; these attributes are "mandatory"; otherwise they are "optional".

Some attributes are no longer used; they are known as "obsolete".

The RIPE Network Management Database is not a relational database (cf. SQL/ORACLE, MS Access, MS Excel). However, there is some cross-referencing of the records; e.g. if you do a look-up of a domain object, you will also get the details of the administrative contact person (the "admin-c") and the technical contact person (the "tech-c"). [These extra look-ups may be switched off, if preferred].

The objects and their description.

The types of objects stored in the RIPE database are summarized in the following table:

Rg = the registry involved

Object = the name of the object in RIPE Database

Describes = the entity in internet operations

References = indicates the relation between the objects;

IP = IP address allocation and assignment Registry;

D = Domain Registry

R = Routing Registry

All = all three registries

Objects in the RIPE Database			
Rg	Object	Describes	References
All	mntner	maintainer object	person, role, mailboxes
All	person	contact persons	mntner
All	role	role (e.g.help desk)	person, mntner
IP	inetnum	IP address space	person, role, mntner
IP	inet6num	IP version address space	person, role, mntner
D	domain	DNS domain	person, role, mntner
R	aut-num	autonomous system (AS) (aut-num,community)	person, role, mntner
R	as-macro	a group of autonomous systems mntner, aut-num	person, role,
R	community	a community of routes that cannot be represented by an AS or a group of AS's	person, role, mntner
R	route	a route being announced	aut-num, community, mntner
R	dom-prefix	CLNS address space and routing	person, role, mntner
R	inet-rtr	router name	person, role, mntner
	limerick	poetry	person, role, mntner

The first column indicates whether the object is part of the address registry (IP), the routing registry (R), the domain registry (D) or all three. The last column indicates the types of objects referenced in an object of this type. It can be seen that almost all objects reference contact persons.

The policies and procedures of address space assignment indicated in ripe-140 should be followed when an address space assignment is being entered. Similarly, whenever a routing policy is being entered into the routing registry, the policies of ripe-181 should be followed.

As stated earlier, the RIPE Database contains three registries and contact information. These will now be discussed.

The objects in the IP Registry and their functions will now be described.

The inetnum object:

Here is an example of an inetnum object:

```
inetnum:     193.0.0.0 - 193.0.0.255
netname:     RIPE-NCC
descr:       RIPE Network Coordination Centre
descr:       Amsterdam, Netherlands
country:     NL
admin-c:     DK143-RIPE
tech-c:      GJG1-RIPE
rev-srv:     ns.ripe.net
rev-srv:     ns.eu.net
notify:      ops@ripe.net
changed:     orange@ripe.net 960815
changed:     GeertJan.deGroot@ripe.net 970110
source:      RIPE
```

The inetnum attribute contains an address range specified in dotted quad notation, which is entered in the inetnum object. For example, if an organisation is assigned a /22 address set for 1024 network addresses, the assigned range [e.g. 193.1.192.0 - 193.1.195.255], is put in the "inetnum" field of the inetnum object. The precise syntax allowed is specified in detail in the Appendix*.

To help identify this assignment in the RIPE database, a short, but semantically meaningful name is entered in the netname field. A short description of the organisation that uses the assigned addresses is given. The information is specified using one or more `descr` fields in the Network Template. If, for example, the assigned addresses will be used by the Department of Neural Surgery at Catatonic State University, then the department and university names may be specified in two `descr` fields. The ISO 3166 country code should be specified in the `country` field. These country codes may be found at `ftp://ftp.ripe.net/iso3166-countrycodes`. The full country name can be used if the code is not known.

The admin-c and tech-c fields are used to specify the IR handle (NIC handle) for the administrative and technical contact persons, respectively. The administrative person specified in the admin-c field must be physically located at the site of the net- work. The technical person specified in the tech-c field may be a network support person located on site, but could also be a consultant that maintains the network for the organisation. In both cases, more than one person can be specified. The use of NIC handles to specify each contact person is required, as it assures each person has a unique entry in the database. If the person doesn't have an entry in the database, a unique NIC handle can be obtained using an automatic process; see Section 2.2.

For security purposes, a notify field can be filled out with an e-mail address to be notified when changes are made to the database object and a mnt-by field can reference a maintainer object which designates who can make changes to the object.

The `status` field can have one of these values:

ALLOCATED PA (the most common allocation type for Local IR's)

ALLOCATED PI (rare for Local IR's)

ALLOCATED UNSPECIFIED

ASSIGNED PA

ASSIGNED PI

These five values are explained in Appendix*.
Below is the template of the inetnum object:

```
inetnum:        [mandatory]       [single]
netname:        [mandatory]       [single]
descr:          [mandatory]       [multiple]
country:        [mandatory]       [multiple]
admin-c:        [mandatory]       [multiple]
tech-c:         [mandatory]       [multiple]
rev-srv:        [optional]        [multiple]
status:         [optional]        [single]
remarks:        [optional]        [multiple]
notify:         [optional]        [multiple]
mnt-by:         [optional]        [multiple]
mnt-lower:      [optional]        [multiple]
changed:        [mandatory]       [multiple]
source:         [mandatory]       [single]
aut-sys:        [obsoleted]
comm-list:      [obsoleted]
ias-int:        [obsoleted]
gateway:        [obsoleted]
```

The Domain Registry:

There is only one type of object in this registry; the domain object.

This object represents Top Level Domain (TLD) and other domain registrations. It is also used for Reverse Delegations. The domain name is written in fully qualified format, without trailing "." .

An example of a domain object is shown below:

```
domain:       over.ripe.net
descr:        RIPE Network Coordination Centre
descr:        Kruislaan 409
descr:        NL-1098 SJ Amsterdam
descr:        The Netherlands
admin-c:      HC56-RIPE
tech-c:       AMRM1-RIPE
zone-c:       AMRM1-RIPE
mnt-by:       AMRM-TEST
changed:      ripe-dbm@ripe.net 970115
source:       RIPE
```

The template of the domain object is given below:

```
domain:       [mandatory]  [single]
descr:        [mandatory]  [multiple]
admin-c:      [mandatory]  [multiple]
tech-c:       [mandatory]  [multiple]
zone-c:       [mandatory]  [multiple]
nserver:      [optional]   [multiple]
sub-dom:      [optional]   [multiple]
dom-net:      [optional]   [multiple]
remarks:      [optional]   [multiple]
notify:       [optional]   [multiple]
mnt-by:       [optional]   [multiple]
mnt-lower:    [optional]   [multiple]
changed:      [mandatory]  [multiple]
source:       [mandatory]  [single]
```

The Routing Registry:

The objects in the Routing Registry are:

aut-num (autonomous system), route, as-macro (autonomous system macro), community, dom-prefix (clns [connectionless network service] address space and routing object), inet-rtr (router name).

The RIPE Routing Registry was developed as part of the PRIDE project at the RIPE NCC and was based on the architecture of the RIPE database. It is, like the IP address allocation registry, a sub-set of the information in the RIPE database. The definitive document on the RIPE Routing Registry is Ripe-181. Ripe-181 became RFC1786.

The aut-num (Autonomous System, AS) object:

An Autonomous System (AS) is a group of IP networks operated by one or more network operators which has a single and clearly defined external routing policy.

An AS has a unique number associated with it which is used both in exchange of exterior routing information and as an identifier of the AS itself. Exterior routing protocols such as BGP and EGP are used to exchange routing information between AS's.

The term AS is often confused or even misused as a convenient way of grouping together a set of networks which belong under the same administrative umbrella even if within that group of networks there are various different routing policies. We provide the "community" concept for such use. AS's can strictly have only one single external routing policy.

An example of an aut-num object is:

```
aut-num:     AS3333
descr:       RIPE NCC
descr:       European Regional Internet Registry
as-in:       from AS286  120 accept ANY
as-in:       from AS1103 120 accept ANY
[.........stuff deleted.........]
as-out:      to AS286 announce AS3333
as-out:      to AS1103 announce AS3333
as-out:      to AS1104 announce AS3333
[.........stuff deleted.........]
admin-c:     DK143-RIPE
tech-c:      GJG1-RIPE
tech-c:      DK143-RIPE
mnt-by:      AS3333-MNT
changed:     GeertJan.deGroot@ripe.net 970114
source:      RIPE
```

The template of the aut-num object is:

```
aut-num:     [mandatory]  [single]
as-name:     [optional]   [single]
descr:       [mandatory]  [multiple]
as-in:       [optional]   [multiple]
as-out:      [optional]   [multiple]
interas-in:  [optional]   [multiple]
interas-out: [optional]   [multiple]
as-exclude:  [optional]   [multiple]
default:     [optional]   [multiple]
guardian:    [optional]   [single]
admin-c:     [mandatory]  [multiple]
tech-c:      [mandatory]  [multiple]
remarks:     [optional]   [multiple]
notify:      [optional]   [multiple]
mnt-by:      [mandatory]  [multiple]
changed:     [mandatory]  [multiple]
source:      [mandatory]  [single]
advisory:    [obsoleted]
```

The route object:

The route object is used to represent a single route injected into the Internet routing mesh. There are several important aspects of the attributes worthy of note.

The value of the route attribute is a classless address. It represents the exact route being injected into the routing mesh.

The value of the origin attribute will be an AS reference of the form AS1234 referring to an aut-num object. It represents the AS injecting this route into the routing mesh. The "aut-num" object (see above) thus referenced provides all the contact information for this route.

Special cases: There can only be a single originating AS in each route object. However in today's Internet sometimes a route is injected by more than one AS. This situation is potentially dangerous as it can create conflicting routing policies for that route and requires coordination between the originating AS's. In the routing registry this is represented by multiple route objects.

This is a departure from the one route (net), one AS principle of the ripe-181 routing registry.

An example of a route object is:

```
route:        193.0.0.0/24
descr:        RIPE-NCC
origin:       AS3333
notify:       ops@ripe.net
mnt-by:       RIPE-NCC-MNT
changed:      GeertJan.deGroot@ripe.net 960812
source:       RIPE
```

The template of the route object is:

```
route:        [mandatory]  [single]
descr:        [mandatory]  [multiple]
origin:       [mandatory]  [single]
hole:         [optional]   [multiple]
withdrawn:    [optional]   [single]
comm-list:    [optional]   [multiple]
advisory:     [optional]   [multiple]
remarks:      [optional]   [multiple]
notify:       [optional]   [multiple]
mnt-by:       [mandatory]  [multiple]
changed:      [mandatory]  [multiple]
source:       [mandatory]  [single]
```

The as-macro object:

It may be difficult to keep track of each and every new AS that is represented in the routing registry. A convenient way around this is to define an

‘AS Macro’ which essentially is a convenient way to group AS’s. This is done so the maintainer of an AS object does not have to add a new AS to it’s routing policy as described by the as- in and as-out attributes of their AS object.

An ‘AS Macro’ is a group of autonomous systems with the same routing policies.

An AS-Macro can be used in <routing policy expressions> for the "as-in" and "as-out" attributes in the aut-num object. The AS-Macro object is then used to derive the list or group of AS’s.

An example of an as-macro object is:

```
as-macro:    AS-EBONE
descr:       EBONE AS’s
as-list:     AS378  AS513  AS517  AS544  AS789  AS2060 AS33
[......................stuff deleted.....................
as-list:     AS6703 AS6740 AS6827 AS6867 AS6765 AS5608 AS67
as-list:     AS-TIPNET AS-BTGB AS-BTTRANSIT AS-ACONET AS-DE
as-list:     AS-TAIDE  AS-TELIANET AS-TMPEBONECWIX
guardian:    staff@ebone.net
tech-c:      BC83-RIPE
admin-c:     BE10
mnt-by:      EBONE-MNT
changed:     bc@sunet.se 970115
source:      RIPE
```

The template of the as-macro object is:

```
as-macro:    [mandatory]  [single]
descr:       [mandatory]  [multiple]
as-list:     [mandatory]  [multiple]
guardian:    [optional]   [single]
tech-c:      [mandatory]  [multiple]
admin-c:     [mandatory]  [multiple]
remarks:     [optional]   [multiple]
notify:      [optional]   [multiple]
mnt-by:      [mandatory]  [multiple]
changed:     [mandatory]  [multiple]
source:      [mandatory]  [single]
```

The community object:

A community is a group of routes that cannot be represented by an AS or a group of AS's. It is in some circumstances useful to define a group of routes that have something in common. This could be a special access policy to a supercomputer centre, a group of routes used for a specific mission, or a disciplinary group that is scattered among several autonomous systems. Also these communities could be useful to group routes for the purpose of network statistics.

Communities do not have a common routing policy.

Communities do not exchange routing information, since they do not represent an autonomous system. More specifically, communities do not define routing policies, but access or use policies. However, they can de used as in conjunction with an AS's routing policy to define a set of routes the AS sets routing policy for.

An example of a community object is given below:

```
community:   HEPNET
descr:       High Energy Physics Network
authority:   HEPnet technical committee IP (HTC-IP)
guardian:    farrache@ccpnxt5.in2p3.fr
tech-c:      Gilles Farrache
admin-c:     Gilles Farrache
changed:     farrache@ccpnxt5.in2p3.fr 940222
source:      RIPE
```

The template of the community object is:

```
community:   [mandatory]  [single]
descr:       [mandatory]  [multiple]
authority:   [mandatory]  [single]
guardian:    [optional]   [single]
tech-c:      [mandatory]  [multiple]
admin-c:     [mandatory]  [multiple]
remarks:     [optional]   [multiple]
notify:      [optional]   [multiple]
mnt-by:      [mandatory]  [multiple]
changed:     [mandatory]  [multiple]
source:      [mandatory]  [single]
```

More details on community objects may be found in ripe-181.

The dom-prefix object:

This object is used for ConnectionLess Network Service (CLNS) representation.

The template of this object is:

```
dom-prefix:  [mandatory]  [single]
dom-name:    [mandatory]  [single]
descr:       [optional]   [multiple]
bis:         [optional]   [multiple]
dom-in:      [optional]   [multiple]
dom-out:     [optional]   [multiple]
default:     [optional]   [multiple]
admin-c:     [mandatory]  [multiple]
tech-c:      [mandatory]  [multiple]
guardian:    [optional]   [single]
remarks:     [optional]   [multiple]
notify:      [optional]   [multiple]
mnt-by:      [optional]   [multiple]
changed:     [mandatory]  [multiple]
source:      [mandatory]  [single]
maintainer:  [obsoleted]
```

The inet-rtr object:

This object represents an internet router.

Contact Information

The person object:

The person's name is specified in full in the person field. The full postal address is specified using multiple address fields. The international telephone number which can be used to reach the person at work is entered in the phone field, and the fax number is entered in the fax-no field. The NIC handle for this person is entered in the nic-hdl field to uniquely identify this person in the database. As with the network template, a notify field can be filled out with an e-mail address to be notified when changes are made to the database object and a mnt-by field can reference a maintainer object which designates who can make changes to the object. See Section 2.3 for further details on maintainer objects.

An example of a person object is:

```
person:       Ambrose Magee
address:      RIPE Network Co-ordination Centre (NCC)
address:      Kruislaan 409
address:      NL-1098 SJ   Amsterdam
address:      The Netherlands
phone:        +31 20 592 5065
fax-no:       +31 20 592 5090
e-mail:       ambrose@ripe.net
nic-hdl:      AMRM1-RIPE
notify:       ambrose@ripe.net
mnt-by:       AMRM1-RIPE-MNT
changed:      ambrose@ripe.net 970115
source:       RIPE
```

The template of the person object is:

```
person:       [mandatory]  [single]
address:      [mandatory]  [multiple]
phone:        [mandatory]  [multiple]
fax-no:       [optional]   [multiple]
e-mail:       [optional]   [multiple]
nic-hdl:      [optional]   [single]
remarks:      [optional]   [multiple]
notify:       [optional]   [multiple]
mnt-by:       [optional]   [multiple]
changed:      [mandatory]  [multiple]
source:       [mandatory]  [single]
```

The mntner object:

Objects in the RIPE database may be protected using mntner objects. The mntner (or maintainer) object contains details of who is authorised to make changes to objects and details of a process that actually verifies that the person who makes the changes is authorised to do so.

An example of a mntner object is:

```
mntner:      AMRM1-RIPE-MNT
descr:       Ambrose's mntner.
admin-c:     AMRM1-RIPE
tech-c:      AMRM1-RIPE
upd-to:      ambrose@ripe.net
mnt-nfy:     ambrose@ripe.net
mnt-nfy:     ripe-dbm@ripe.net
auth:        CRYPT-PW otpbdphttaoas
remarks:     This is a test mntner.
notify:      ambrose@ripe.net
mnt-by:      AMRM1-RIPE-MNT
changed:     ambrose@ripe.net 970115
source:      RIPE
```

Attributes used in all objects.

In all objects, space is reserved to identify the person submitting these entries to the registry database. Their e-mail address must be specified in the "changed" field together with the date the template is submitted.

Similarly, the "source" field is used to specify the registry database where the requester information can be found after an assignment is made. Usually, it is RIPE.

Details of attributes and objects may be found in the Appendix*. There are ways of obtaining the format or template of an object; these are discussed in Section 2.

1.1.2 Management of the database objects:

In principle, anyone can create, change or delete objects in the RIPE Database using electronic mail. E-mail is sent to `<auto-dbm@ripe.net>`, which is an "automatic" mailbox. The entire record or object must be in the body of the message, not only the attributes to be changed.

More than one object may be in the message, but objects must be separated by at least one blank line.

An acknowledgement message will be sent to you automatically once the update is complete. If a syntax error in your request is detected or if special authorization is required for a particular request, you will receive an error message.

You will always receive a reply. If you do not receive any reply, it implies that the mail has bounced or been lost.

E-mail updates are sent an automatic mailbox which handles update requests. When mail is sent to this address it is analysed, authorization checks are performed and finally the actual updates, if appropriate, are done.

In some cases, the request must be sent to a human mailbox. For security reasons, a request to create a mntner ("maintainer") object must be sent to `<ripe-dbm@ripe.net>`. However, requests to change or delete a mntner object should be sent to `<auto-dbm@ripe.net>`. If a request to create a mntner is sent to `<auto-dbm@ripe.net>`, it will be forwarded to `<ripe-dbm@ripe.net>`. This will be discussed further in Section 2.3.

Chapter 2; Database queries and updates:

2.1 Querying the RIPE Database

There are five ways of querying the RIPE database:
whois client, WWW, WAIS, telnet and e-mail.

The choice of which one to use depends on what resources are available at your site and which one your find the most appropriate.

Each way of querying the database will now be explained.

2.1.1 RIPE whois client:

Searching for the names of database objects;

If you wish to look up objects in the RIPE database, you must use special search keys. A full list of the search keys is given below:

OBJECT	KEYS
aut-num	AS number (e.g. AS3333)
as-macro	as-macro name (e.g. AS-EBONE)
community	community name (e.g. HEPNET)
domain	domain name (e.g. over.ripe.net)
inetnum	range of IP addresses e.g 193.0.0.0 - 193.0.0.255; network name e.g.RIPE-NCC
inet6num	range of IP version 6 addresses or network name
person	a person's name or NIC-handle or e-mail address in RFC822 format. e.g. Ambrose Magee or AMRM1-RIPE or ambrose@ripe.net
clns object	
domain-prefix	domain-prefix, domain-name
inet-rtr	internet router name (e.g.
limerick	name of limerick
mntner	name of mntner object e.g. AMRM1-RIPE-MNT
route	internet route e.g. 193.0.0.0/24
role	the name, the NIC-handle or the e-mail address (in RFC822 format) of a role object; e.g. RIPE NCC

If you also want to search for other strings in the objects, you can use the WAIS interface; however it doesn't support the special options that are provided in the RIPE 'whois' interface.

The RIPE whois client has several options, which may be used either alone or in combination.

The following is a list, in alphabetical order, of the available options, which are explained in more detail below.

Option	Function
-a	search all databases
-F	fast raw output (implies -Fr)
-h	search alternate server
-i	inverse look-up
-k	used with the telnet interface
-L	find all Less specific matches
-m	find first level More specific matches
-M	find all More specific matches
-p	connect to other port than the default whois port
-r	turn off recursive lookups
-s	search databases with source "source"
-S	tell server to leave out "syntactic sugar"
-t	requests template for object of type "type"
-T	only look for objects of type "type"
-HELP	gives a copy of the current ‘HELP & HOWTO’ document.

A very useful option is "-h", which allows you to connect directly to the server at the RIPE NCC or to a "mirror" of the RIPE database elsewhere.

- Example:

```
$ whois -h whois.ripe.net AMRM1-RIPE
```

-a option:

Example:

```
[an-lar] $ whois -h whois.ripe.net -a AMRM1-RIPE
```

The following databases may be queried in this way:

RIPE, RADB, INTERNIC, ANS, MCI, CANET, APNIC

-F option:

This option generates an output quickly, but with the abbreviated form of the attributes. Example:

```
[an-lar]$ whois -h whois.ripe.net AMRM1-RIPE
```

gives

```
person:       Ambrose Magee
address:      RIPE Network Co-ordination Centre (NCC)
address:      Kruislaan 409
address:      NL-1098 SJ   Amsterdam
address:      The Netherlands
phone:        +31 20 592 5065
fax-no:       +31 20 592 5090
e-mail:       ambrose@ripe.net
nic-hdl:      AMRM1-RIPE
notify:       ambrose@ripe.net
mnt-by:       AMRM-RIPE-MNT
changed:      ambrose@ripe.net 970115
source:       RIPE
```

However,

[an-lar]$ whois -F -h whois.ripe.net AMRM1-RIPE gives

```
*pn: Ambrose Magee
*ad: RIPE Network Co-ordination Centre (NCC)
*ad: Kruislaan 409
*ad: NL-1098 SJ   Amsterdam
*ad: The Netherlands
*ph: +31 20 592 5065
*fx: +31 20 592 5090
*em: ambrose@ripe.net
*nh: AMRM1-RIPE
*ny: ambrose@ripe.net
*mb: AMRM-RIPE-MNT
*ch: ambrose@ripe.net 970115
*so: RIPE
```

-i option:

This option allows you to do reverse or inverse look-ups of particular combinations of attributes and objects.

The attributes which can be used are:

admin-c, tech-c, zone-c;
notify;
origin;
mnt-by;
author.

The syntax of the usage is:

whois -i <attribute list> <object>

Examples of how the "-i" option is used are now given:

- `whois -i admin-c,tech-c,zone-c <nic-hdl|person or role name>`

[an-lar]$ whois -h whois.ripe.net -i admin-c,tech-c,zone-c AMRM1-RIPE

finds all objects which cite AMRM1-RIPE as an admin-c or a tech-c or a zone-c.

```
inetnum:      193.0.128.0 - 193.0.128.255
netname:      BORK-NET
descr:        Test Allocation
descr:        RIPE Network Coordination Centre
descr:        Amsterdam, Netherlands
country:      NL
admin-c:      AMRM1-RIPE
status:       ASSIGNED PA
notify:       orange@ripe.net
changed:      orange@ripe.net 960821
source:       RIPE
```

person: Ambrose Magee
[.......stuff deleted.......]

Any one or more of the three attributes admin-c, tech-c and zone-c may be used. Note that if "Ambrose Magee" was used, no objects would be found as "Ambrose Magee" is not used anywhere in an admin-c, tech-c or a zone-c attribute.

- `whois -i notify <RFC822 e-mail address>`

You can also find all objects which have a particular e-mail address in their "notify" attribute;

Example:

[an-lar]$ whois -h whois.ripe.net -i notify ambrose@ripe.net

finds all objects which have ambrose@ripe.net in their notify attribute.

```
mntner:      AMRM1-RIPE-MNT
descr:       Ambrose's mntner.
admin-c:     AMRM1-RIPE
tech-c:      AMRM1-RIPE
upd-to:      ambrose@ripe.net
mnt-nfy:     ambrose@ripe.net
mnt-nfy:     ripe-dbm@ripe.net
auth:        MAIL-FROM Ambrose.Magee@ripe.net
auth:        MAIL-FROM ripe-dbm@ripe.net
remarks:     This is a test mntner.
notify:      ambrose@ripe.net
mnt-by:      AMRM1-RIPE-MNT
changed:     ambrose@ripe.net 960815
changed:     ambrose@ripe.net 960930
source:      RIPE


person:      Ambrose Magee
[.......stuff deleted.......]
```

- `whois -i origin <AS number>`

Example:

[an-lar]$ whois -h whois.ripe.net -i origin AS3333

finds all route objects which specify AS3333 as their origin.

```
route:       193.0.0.0/24
descr:       RIPE-NCC
origin:      AS3333
notify:      ops@ripe.net
mnt-by:      RIPE-NCC-MNT
changed:     GeertJan.deGroot@ripe.net 960812
source:      RIPE


route:       193.0.0.0/23
descr:       RIPE-NCC
origin:      AS3333
notify:      ops@ripe.net
mnt-by:      RIPE-NCC-MNT
changed:     GeertJan.deGroot@ripe.net 960812
source:      RIPE
```

- `whois -i mnt-by <name of mntner-object>`

Example

[an-lar] $ whois -h whois.ripe.net -i mnt-by AMRM1-RIPE-MNT

finds all objects which are maintained by AMRM1-RIPE-MNT. (See Section 2.3 for details on mntner objects).

- `whois -i author <nic-hdl|person's name>`

Example:

[an-lar]$whois -h whois.ripe.net -i author AMRM1-RIPE
finds all limericks written by by AMRM1-RIPE;

The "-r" option:

All the above options do recursive look-ups i.e. they will look-up any person objects associated with admin-c, tech-c, zone-c or author attributes.

To disable this recursive look-up, use the "-r" option. This may be used with the other options, including "-i".

Example:

[an-lar]$ whois -h whois.ripe.net -r -i admin-c,tech-c,zone-c AMRM1-RIPE

Note that the "-r" flag must precede the "-i" flag and the arguments.

-k option:

This is used in the telnet interface. See Section 2.1.4(b)

-L option:

Sometimes when doing a look-up on an inetnum object, you may wish to find the "next objects up" in the hierarchy i.e. all less specific matches; e.g.

[an-lar]$ whois -h whois.ripe.net -r 193.1.0.0/16

```
inetnum:     193.1.0.0 - 193.1.255.255
netname:     IE-HEANET-193-1
descr:       DELEGATED BLOCK
descr:       Provider Local Registry
descr:       HEAnet
country:     IE
admin-c:     MN131
tech-c:      MN131
mnt-by:      RIPE-NCC-HM-MNT
changed:     roderik@ripe.net 950315
source:      RIPE
```

Now compare the above with the output of the following:

```
[an-lar]$ whois -h whois.ripe.net -r -L 193.1.0.0/16
```

gives

```
inetnum:     193.1.0.0 - 193.1.255.255
netname:     IE-HEANET-193-1
descr:       DELEGATED BLOCK
descr:       Provider Local Registry
descr:       HEAnet
country:     IE
admin-c:     MN131
tech-c:      MN131
mnt-by:      RIPE-NCC-HM-MNT
changed:     roderik@ripe.net 950315
source:      RIPE
inetnum:     193.0.0.0 - 193.255.255.255
netname:     EU-ZZ-193
descr:       European Regional Registry
descr:       Europe
country:     EU
admin-c:     DK58
tech-c:      DK13-RIPE
status:      ALLOCATED UNSPECIFIED
mnt-by:      RIPE-NCC-HM-MNT
changed:     hostmaster@ripe.net 960123
source:      RIPE
```

In other words, the inetnum object which "includes" IE-HEANET-193-1, is also found. The same process is implemented for route objects, but not for domain objects.

There is no "-l" option. The default behaviour of the database is to find exactly the object you want or first less specific match (for inetnum and route objects).

-m option:

This option allows you to look-up all those objects which are one level more specific than what you have given in your query. In the case of inetnum objects, this means that you can lookup all those inetnum objects which are the "children" of the inetnum object found by an exact match i.e. the first level more specific match.

Example:

```
[an-lar]$ whois -h whois.ripe.net -r 193.1.0.0/16
```

```
inetnum:      193.1.0.0 - 193.1.255.255
netname:      IE-HEANET-193-1
[.......stuff deleted.......]
```

But,

```
[an-lar]$ whois -h whois.ripe.net -r -m 193.1.0.0/16
```

finds 86 objects; i.e. there are 86 objects which are directly within the range of IP addresses given by 193.1.0.0 - 193.1.255.255.

-M option:

Some of the objects found by a "-m" look-up have "children" themselves; i.e. there are objects directly within the range of IP addresses specified by the objects found by a "-m" look-up.

These objects may be found by a "-M" look-up.

Example:

```
[an-lar]$ whois -h whois.ripe.net -r -M 193.1.0.0/16
```

finds 97 objects.

-p option:

Normally, when you do a "whois" query, you connect to port 43 of the server which you specified with the "-h" option. However, there are occasions when you wish to connect to a port different from the default port. E.g., if you had a private database, with a whois daemon of its own, you could connect to it.

```
[an-lar]$ whois -h localhost -p 4327
```

connects to port 4327 on the localhost.

-s option:

The RIPE database contains objects that are from other databases. This is indicated by the "source" attribute in each object. The sources used in the RIPE database are RIPE, RADB, INTERNIC, ANS, MCI, CANET and APNIC. Not every object type may be found using a source other than RIPE. Normally, only inetnum and aut-num objects may be found using non-RIPE sources.

Example:

[an-lar]$ whois -h whois.ripe.net -s INTERNIC AS3333

gives

```
aut-num:     AS3333
descr:       part of AS block AS3154 - AS3353
admin-c:     3353
remarks:     this AS number is assigned by the InterNIC
remarks:     please query whois.internic.net for more infor
changed:     ripe-dbm@ripe.net 970103
source:      INTERNIC
```

-S option:

This option tells the server to leave out text which is only added to make aut-num objects readable by humans. This text is called "Syntactic Sugar".

Example:

[an-lar]$ whois -h whois.ripe.net -r AS3333

gives

```
aut-num:    AS3333
descr:      RIPE NCC
descr:      European Regional Internet Registry
as-in:      from AS286  120 accept ANY
[........stuff deleted ........]
as-out:     to AS286 announce AS3333
[........stuff deleted ........]
```

Compare with the output of **[an-lar]$ whois -h whois.ripe.net -r -S AS3333**

```
aut-num:     AS3333
descr:       RIPE NCC
descr:       European Regional Internet Registry
as-in:       AS286  120 ANY
[........stuff deleted ........]
as-out:      AS286 AS3333
[........stuff deleted ........]
```

The "-t" option:

This option allows you to obtain a template of any type of object that is accepted by the RIPE database. You may specify either the full or the abbreviated name of the object as follows:

Full name	Abbreviation
aut-num	an
as-macro	am
community	cm
domain	dn
inetnum	in
inet6num	i6
persons	pn
dom-prefix	dp
inet-rtr	ir
limerick	li
mntner	mt
route	rt
role	ro

Example:

[an-lar]$ whois -h whois.ripe.net -t person

```
person:       [mandatory]  [single]
address:      [mandatory]  [multiple]
phone:        [mandatory]  [multiple]
fax-no:       [optional]   [multiple]
e-mail:       [optional]   [multiple]
nic-hdl:      [optional]   [single]
remarks:      [optional]   [multiple]
notify:       [optional]   [multiple]
mnt-by:       [optional]   [multiple]
changed:      [mandatory]  [multiple]
source:       [mandatory]  [single]
```

[an-lar]$ whois -h whois.ripe.net -t pn

gives the same output.

The "-T" option:

If you know the type of the object for which you are looking, you can restrict the search to that type only. This option is useful when different types of object have the same KEYS.

Example:

[an-lar]$ whois -h whois.ripe.net -r 193.0.0.0/24

gives

```
inetnum:     193.0.0.0 - 193.0.0.255
netname:     RIPE-MEETING
descr:       RIPE Meeting Terminal Room
descr:       Amsterdam, Netherlands
country:     NL
admin-c:     GJG1-RIPE
tech-c:      GJG1-RIPE
changed:     GeertJan.deGroot@ripe.net 970110
source:      RIPE


route:       193.0.0.0/24
descr:       RIPE-NCC
origin:      AS3333
notify:      ops@ripe.net
mnt-by:      RIPE-NCC-MNT
changed:     GeertJan.deGroot@ripe.net 960812
source:      RIPE
```

cf. the output of

[an-lar] $ whois -h whois.ripe.net -r -T inetnum 193.0.0.0/24

```
inetnum:     193.0.0.0 - 193.0.0.255
netname:     RIPE-MEETING
descr:       RIPE Meeting Terminal Room
descr:       Amsterdam, Netherlands
country:     NL
admin-c:     GJG1-RIPE
tech-c:      GJG1-RIPE
changed:     GeertJan.deGroot@ripe.net 970110
source:      RIPE
```

In this situation, only the inetnum object is found. Note that when using the "-T" option, the abbreviated form of the object names may be used; e,g, "in" can be used instead of "inetnum".

-HELP option:

The output of this option is an up-to-date copy of the "HELP & HOWTO" document for the RIPE database.

Example:

[an-lar]$ whois -h whois.ripe.net HELP

gives

`Document: HELP & HOWTO use the RIPE database`
`Version:  970106`

Author: David Kessens
E-mail: ripe-dbm@ripe.net

Introduction

This document describes how to access and create/change/delete your data in the RIPE (or test!) database. Also included is a section with hints and tips. At the end of the document is a chapter with pointers to more extensive documentation.
[...stuff deleted...]

To summarise, the syntax of the usage of the "whois" command is as follows:

Usage: whois [-aFrSv] [-h host] [-s source] [-T type] [-L|-m|-M] <search string>

Using non-RIPE ‘whois’ clients:

Please note that most of the options are NOT understood by non RIPE ’whois’ client programs. Sometimes the following work-around will work :

Instead of

```
[an-lar] $ whois -h whois.ripe.net -T person AMRM1-RIPE
```

you can use

```
[an-lar] $ whois -h whois.ripe.net "-T person AMRM1-RIPE"
```

2.1.2 WWW:

`http://www.ripe.net/db/`

This is a web-based interface to the whois server; therefore you can only do a search on the KEYS listed in Section 2.1.1

2.1.3 WAIS:

You should create a directory with a name like "wais-sources". In this directory, you can put all source files for WAIS applications.

You should then create a file with a name like "ripe-database.src". The contents of the file should be this:

```
(:source
    :version  3
        :ip-name "wais.ripe.net"
        :tcp-port 210
        :database-name "ripe-database"
        :cost 0
        :cost-unit :free
        :maintainer "ripe-dbm@ripe.net"
        :description "This WAIS database contains
        the RIPE Network Management
    Database which is also available at:
         ftp://ftp.ripe.net/ripe/dbase")
```

Now at your command line prompt, type "swais". You will now see a menu ("Source Selection") and a list of all the sources in your wais-sources directory. Follow the instructions. Note that you can search for any keyword, including addresses and telephone numbers.

The "q" character allows to close the process: "?" gives you a list of all the keywords.

2.1.4 telnet

There are currently two ways to telnet to the RIPE Database.

(a) The first is to simply telnet "whois.ripe.net" and enter one of the options listed in Section 2.1.1(b), followed by one of the KEYS listed in Section 2.1.1(a). You are not required to type "whois".

Example:

[an-lar]$ telnet whois.ripe.net

gives

```
Trying 193.0.0.194...
Connected to dbase.ripe.net.
Escape character is '^]'.
*******************************************************************
* RIPE NCC
*
* Telnet-Whois Interface to the RIPE Database
*
* Most frequently used keys are: IP address or prefix (classless),
```

```
* network name, persons last, first or complete name, NIC handle,
* and AS<number>.
*
* Use 'help' as key to get general help on the RIPE database.
* Contact <ncc@ripe.net> for further help or to report problems.
*********************************************************************

Enter search key [q to quit]: Ambrose Magee

person:    Ambrose Magee
address:   RIPE Network Co-ordination Centre (NCC)
[....stuff deleted....]

Enter search key [q to quit]: -T route 193.0.0.0/23

route:     193.0.0.0/23
[....stuff deleted....]

Enter search key [q to quit]: q
Connection closed by foreign host.
[an-lar]$
```

(b) The second is to telnet to "whois.ripe.net 43". This is a very basic interface to the whois server. You do not type "whois" in your query; you only type the whois option (e.g. "-r") and then the KEY.

Example:

```
[an-lar] $ telnet whois.ripe.net 43
```

gives

```
Trying 193.0.0.194...
Connected to dbase.ripe.net.
Escape character is '^]'.
-T rt 193.0.0.0/23

route:     193.0.0.0/23
[....stuff deleted....]

Connection closed by foreign host.
```

The connection is closed immediately after one query. If you wish to do more than one look-up during the same telnet session, you should use the "-k" option as shown:

Example:

```
Trying 193.0.0.194...
Connected to dbase.ripe.net.
Escape character is '^]'.
-k -T rt 193.0.0.0/24

% Server is running at low priority for -M, -m and -k queries
```

```
route:      193.0.0.0/24
descr:      RIPE-NCC
[....stuff deleted....]

-r -T in 193.0.0.0/24
inetnum:    193.0.0.0 - 193.0.0.255
netname:    RIPE-NCC
[....stuff deleted....]

-k

Connection closed by foreign host.
[an-lar] $
```

Note: "-k" is typed first, before any other characters. Typing "-k" alone closes the session.

2.1.5 E-mail:

(a) Send mail to `<whois@ripe.net>` with in the body text:

```
whois KEY,
```

where KEY is one of the KEYS from Section 2.1.1(a). "whois" must be at the beginning of a new line in the body of the message. Separate queries must be on separate lines. The "Subject:" line will not be interpreted.

Example:

An e-mail with the following in the body of the message

```
whois 193.0.0.0/24
whois AMRM1-RIPE
```

gets the following mail:

```
> Dear mail whois user,
>
> The requested RIPE database queries gave the following results:
>
> ------
> Query: 193.0.0.0/24
>
>
```

```
> inetnum:     193.0.0.0 - 193.0.0.255
> netname:     RIPE-NCC
> [.....stuff deleted.....]
>
> route:       193.0.0.0/24
> descr:       RIPE-NCC
> [.....stuff deleted.....]
>
> ------
> Query: AMRM1-RIPE
>
>
> person:      Ambrose Magee
> address:     RIPE Network Co-ordination Centre (NCC)
> [.....stuff deleted.....]
>
> Regards
> The RIPE NCC whois service (e-mail department)
```

Important !

- This method doesn't support the whois options !
- Separate queries must be on separate lines.

2.1.6 Means of querying other databases

To query the InterNIC database, use

```
[an-lar] $ whois -h whois.internic.net <search string>
```

All the databases that are mirrored by the RIPE NCC may be directly queried in this way.

The databases that may be mirrored are:
RIPE, RADB, INTERNIC, ANS, MCI, CANET, APNIC

2.2 Creating, Updating and Deleting an Object:

2.2.1 Creating a new object:

To create a new object, follow this procedure:

- get the template;
- modify the template;
- send the object to the appropriate mailbox i.e. `<hostmaster@ripe.net>` for new autonomous system objects, `<ripe-dbm@ripe.net>` for new mntner objects and `<auto-dbm@ripe.net>` for all other objects.

(1) Getting the template:

Use `whois -t <type of object>`.
See Section 2.1.1 for details on how to obtain the template of any object.
example
[spoor] $ whois -h whois.ripe.net -t person
gives

```
person:       [mandatory]  [single]
address:      [mandatory]  [multiple]
phone:        [mandatory]  [multiple]
fax-no:       [optional]   [multiple]
e-mail:       [optional]   [multiple]
nic-hdl:      [optional]   [single]
remarks:      [optional]   [multiple]
notify:       [optional]   [multiple]
mnt-by:       [optional]   [multiple]
changed:      [mandatory]  [multiple]
source:       [mandatory]  [single]
```

(2) modification of the template;

(i) Now fill in the appropriate details in the template. Do not leave attributes with values such as "[optional]", or "[multiple]".

(ii) Attributes which are mandatory should .B not .R be empty. Otherwise, the update will be rejected by the database software.

(iii) Attributes which are optional may be left empty, but they will be removed by the database software.

Example:

```
person:      Ambrose Magee
address:RIPE NCC Network Co-ordination Centre
[.....stuff deleted.....]
phone:         +31 20 592 5065
fax:
e-mail:      ambrose@ripe.net
[.....stuff deleted.....]
```

The empty attribute "fax" will be deleted by the database software.

Attributes which are described in the template as "single" must be on only one line.

Attributes which are described as "multiple" may be on more than one line, but the attribute name must be at the beginning of each line.

The "address" attribute in the person object is an example of an attribute that may be multiple.

```
person:      Ambrose Magee
address:     RIPE Network Co-ordination Centre (NCC)
address:     Kruislaan 409
address:     NL-1098 SJ   Amsterdam
address:     The Netherlands
[.....stuff deleted.....]
```

(iv) If you are completing a person template or want to reference a person in an another object, you should use a RIPE handle ('nic-hdl:'). You can also use NIC handles from other registries such as the InterNIC if you are also registered there. NIC-handles will give you a unique identifier attached to a person which you can use as a reference. This avoids problems with different persons having the same name. You can get yourself a RIPE NIC-handle by putting the following in the NIC handle field:

nic-hdl: AUTO-1

OR

nic-hdl: AUTO-1YourInitials

The second case advises the database software to use YourInitials (no more then 4 characters) for building the NIC handle while the first case asks the database software to find the initials itself.

You can use the same identifiers (AUTO-1 or AUTO-1YourInitials) in the same update message in other objects as a reference. The database software will then fill in the freshly assigned NIC handles in the objects. Note that you can also use other numbers (example: AUTO-2) so that you can update more person objects and objects that reference the persons in one E-mail message.

Example:

```
domain:  perl.com
admin-c: AUTO-1
tech-c:  AUTO-2
[ ... stuff deleted ...]

person:  Ambrose Magee
nic-hdl: AUTO-1
[ ... stuff deleted ...]

person:  Larry Wall
nic-hdl: AUTO-2
[ ... stuff deleted ...]
```

N.B. The same rules apply when obtaining a NIC-handle for a role object.

(iv) Put today's date and your e-mail address in the "changed" attribute. The format of the date must be YYMMDD.

(v) use of "mnt-by" attribute;

You can protect your objects with maintainer objects by adding a ‘mnt-by:’ attribute. For some objects this is even mandatory. Put the name of the maintainer object in the "mnt-by" field

Example:

```
person:      Ambrose Magee
[....stuff deleted....]
mnt-by:      AMRM-MNT
[.....stuff deleted.....]
```

More information on maintainer objects may be found in Section 2.3.

(vi) use of the keywords "NEW" and "ASSIGN";

You can ensure that the database will only accept your object if it is not already in the database by using the keywords "NEW" and "ASSIGN".

Put "NEW" in the ‘Subject’ line of your e-mail if you want the database to only accept new objects. Use the keyword "ASSIGN" if you want the database to only accept new inetnum objects.

N.B. at the moment, these keywords are case insensitive. Avoid putting strings such as "Old person object with new address" or "Re-assign address space" in your ‘Subject’ line.

(3) send the object to the appropriate mailbox.

`<auto-dbm@ripe.net>`:

`<auto-dbm@ripe.net>` is an automatic mailbox and all e-mails that create, update or delete objects (see above for exceptions) should be sent to this address.

If you require a detailed acknowledgement, put the keyword "LONGACK" in the ‘Subject’ line of your e-mail.

You will always receive an acknowledgement, even if "LONGACK" is not in the ‘Subject’ line. If you do not receive an acknowledgement, the reason could be that your e-mail address is unknown and therefore the acknowledgement mail has bounced or has been lost.

`<ripe-dbm@ripe.net>`:

Questions and bug reports should be sent to `<ripe-dbm@ripe.net>`. You are recommended to send as much information as possible.

2.2.2 Updating an existing object:

How to update an existing object

This is done by sending in the new object. Probably the easiest way to do this is to get a copy of the old object, change or add the fields you want to change or add, and then put a new ’changed:’ attribute in the object.

Here are the steps in more detail:

(1) Get a copy of the existing object; you can do this using

whois -h whois.ripe.net <search string> >TemporaryFile

Example:

[spoor] $ whois -h whois.ripe.net AMRM1-RIPE > update-file

(2) Load the TemporaryFile into your favorite editor and make your changes or additions to the object. It is recommended to use NIC handles instead of the person names for references to person objects to guarantee that your object points to a single person. See Section 2.2.1 for more information.

If you are updating a person or role object with a new or different NIC handle, or if you are updating a route object with a new or different origin, please note:

it is wise to delete the old person, role or route object before adding the new objects for the following reasons: (a) the database will treat two person

objects with the same name but with different NIC handles, as different objects;
(b) the database will treat two person objects with the same name as different objects, if one object has a NIC-handle, but the other does not;
(c) the database will treat route objects with the same "prefix length" representation, but with different origins as different route objects;
(d) the database handles role objects in the same way as it handles person objects.

If you do not delete the old objects first, then the database will see the old and new objects as different objects and the update request will be treated as the creation of a new object instead of an update of the old object. The database will however recognize that a person object is an update if the only difference between the old and new object is that the old object did not have a NIC handle.

(3) Add a "changed" attribute. This attribute has the following syntax :

```
changed: E-mailAddress Date
```

where 'E-mailAddress' is an RFC822 E-mail address specifying who made the change and 'Date' is the date of the change in either YYYYMMDD or YYMMDD format.

(4) Send your message to `<auto-dbm@ripe.net>`.

(5) You will receive an acknowledgement. If you send in an object that is identical to one that is already in the database, the acknowledgement message will say "NOOP" meaning "NO OPeration"; i.e. the object in the database is not modified.

2.2.3 Deleting an object:

This is done by sending in the whole object just like an update and adding a special attribute, 'delete', to it:

delete: <free text>

It is recommended that the reason for deleting the object be put as the value of the delete attribute.

Example:

```
person:     Ambrose Magee
address:    RIPE Network Coordination Centre (NCC)
address:    Kruislaan 409
address:    NL-1098 SJ Amsterdam
address:    Netherlands
phone:      +31 20 592 5065
fax-no:     +31 20 592 5090
e-mail:     ambrose@ripe.net
nic-hdl:    AMRM2-RIPE
notify:     ambrose@ripe.net
changed:    ambrose@ripe.net 970110
source:     RIPE
delete:     Duplicate person object.
```

The deletion is only accepted if the object in the message is .B exactly very careful when deleting person or role objects unless you are sure that the person or role object is not referenced by other objects. You can find objects that reference a certain person by doing a '-i' query; see Section 2.1.1.

Example:

```
[spoor] $ whois -h whois.ripe.net -r -i tech-c,admin-c,zone-c AMRM1-RIPE
```

Some people might have used their name as a reference instead of their NIC-handle, so you should also do a look-up on that.

If you're not sure that you were the only one referencing a certain person object, do NOT delete that person object. It will not affect the database very much if there are some obsolete person objects. Authorization, notification and acknowledgement messages for deletes are handled exactly the same way as for ordinary updates.

2.2.4 Warning and Error messages:

The syntax of all objects sent to <auto-dbm@ripe.net> is rigorously checked. Acknowledgement messages are generated by these syntax checks.

(a) You will always receive an acknowledgement that your update has been received.

(b) If there are no mistakes in your update, you will receive an acknowledgement message saying "No errors were found in your update".

(c) Minor, but not fatal mistakes will be corrected and the object will be accepted into the database. The acknowledgement message will indicate what correction was made.

Example: suppose this object is sent to `<auto-dbm@ripe.net>`.

```
person:      Ambrose Magee
[.....stuff deleted.....]
changed: ambrose@ripe.net
```

The following warning message would be generated:

WARNING: added current date to "changed" field.

(d) Fatal mistakes will .B not .R be corrected and the object will .B not .R be accepted into the database. An acknowledgement message will be sent, which will identify the error.

Example: suppose this object is sent to `<auto-dbm@ripe.net>`.

```
route:       193.0.0.139/32
descr:       x1.ripe.net
origin:      AS3333
changed:     ambrose@ripe.net 961221
source:      RIPE
The following warning message would be generated:
--------------------------------------------------------
Update FAILED: [route] 193.0.0.139/32
route:       193.0.0.139/32
descr:       x1.ripe.net
origin:      AS3333
notify:      ambrose@ripe.net
changed:     ambrose@ripe.net 961221
source:      RIPE
*ERROR*:     mandatory field "mnt-by" missing


Objects that just generated a WARNING have been updated as
shown.

Objects that generated an *ERROR* have NOT been updated as
requested.

Please re-submit corrected objects.
-----------------------------------------------------------
```

(e) If you have any questions about any error messages, you may send e-mail to `<ripe-dbm@ripe.net>`. This mailbox is checked regularly. Please send as much information as possible.

2.3 Object Protection:

Authorization and authentication:

"Authorization" indicates only who has the authority to change database objects, but does not check that it is that person who is making the changes.

"Authentication" actually verifies that the person who is making the changes is authorised to do so.

2.3.1. The "mntner" object:

(a) This is an object which can be used to notify you of any changes to your objects and to allow only certain individuals to change your objects. The mntner object contains information on contact persons and on authentication and notification details.

The mntner object is used every time a database object with a "mnt-by" attribute is created, updated or deleted to determine whether or not the originator of the update request is authorised to make the update.

Each mntner object should have an unique name.

From now on, "mntner" and "maintainer" will be used interchangeably.

Sometimes, the word "maintainer" refers to the human being referenced in the "mnt-nfy" attribute of the "mntner" object. In this document, "mntner" refers to the object and "maintainer" refers to the person.

Adding a new mntner object must be authorised manually by RIPE NCC staff. Details are given later.

2.3.2 Functionality:

(i) Individual Object Protection:

You can protect any object with a mntner object. All you have to do is include a "mnt-by" attribute in your object. For some objects, a "mnt-by" attribute is mandatory.

The value of the "mnt-by" attribute is a mntner object which details of how changes to the object are to be authorised.

Example:

```
person:       Ambrose Magee
address:      RIPE Network Co-ordination Centre (NCC)
address:      Kruislaan 409
address:      NL-1098 SJ   Amsterdam
address:      Kingdom of the Netherlands
phone:        +31 20 592 5065
fax-no:       +31 20 592 5090
e-mail:       ambrose@ripe.net
nic-hdl:      AMRM1-RIPE
notify:       ambrose@ripe.net
mnt-by:       AMRM1-RIPE-MNT
changed:      ambrose@ripe.net 961220
source:       RIPE
```

The "mnt-by" attribute in the above object refers to this mntner object:

```
mntner:       AMRM1-RIPE-MNT
descr:        Ambrose's mntner.
admin-c:      AMRM1-RIPE
tech-c:       AMRM1-RIPE
upd-to:       ambrose@ripe.net
mnt-nfy:      ambrose@ripe.net
mnt-nfy:      ripe-dbm@ripe.net
auth:         MAIL-FROM Ambrose.Magee@ripe.net
remarks:      This is a test mntner.
notify:       ambrose@ripe.net
mnt-by:       AMRM1-RIPE-MNT
changed:      ambrose@ripe.net 960815
changed:      ambrose@ripe.net 960930
source:       RIPE
```

You can have more than one "mnt-nfy" attribute in a mntner object. Multiple mntner objects can be referenced in objects by separating them with blank spaces or by using multiple mnt-by attributes per object.

Example:

```
person:     Ambrose Magee
[.....stuff deleted.....]
mnt-by: AMRM1-RIPE-MNT RIPE-DBM-MNT
[.....stuff deleted.....]
```

or

```
person:     Ambrose Magee
[.....stuff deleted.....]
mnt-by: AMRM1-RIPE-MNT
mnt-by: RIPE-DBM-MNT
[.....stuff deleted.....]
```

In this case all maintainers referenced are equally authorised to make changes to the object.

When responding to queries, the RIPE whois server will not automatically give the associated mntner object that is referenced in the object for which you asked. i.e. you must look up the mntner object separately.

Whenever a change to an object is attempted, the mnt-by attribute of the current database object is examined. The object may only be updated or deleted according to the authorisation scheme described in the associated mntner object. See Section 2.3.3.

(ii) Notification of (Successful) Changes to objects that are protected by a mntner object:

The "notify" attribute can be included in every object. It contains an e-mail address to which notifications of changes to the object itself will be sent.

It is possible to put a "notify" attribute in all your objects. However, if the notify e-mail address is changed, then every object with a notify attribute that references that e-mail address will have to be updated.

A more efficient approach is to include a "mnt-by" attribute in all your objects. This attribute will reference a mntner object, which itself has a "mnt-nfy" (maintainer notify) attribute. The value of this attribute is an e-mail address to which a notification will be sent if any object maintained by this mntner object is added, updated or deleted. The functionality is exactly the same as if a "notify" attribute had been defined in the object. Specifying it here has the advantage that any changes of the address affect only one object.

Whenever a change to an object is attempted the mnt-by attribute of the current database object is examined.

If there is no mnt-by attribute, the update always proceeds causing any notifications specified in notify attributes of the object. It is also a perfectly legitimate authorisation model for those objects that do not need sophisticated authorisation. Also we would like to stress that using stronger authorisation requires timely processing of update requests. An unresponsive maintainer preventing others from making updates frequently is a worse solution than having no authorisation at all.

If the update is originated by a maintainer referenced in a mnt-by attribute, the update also proceeds causing the necessary notifications. There are various methods to authenticate the origin of an update request. These can be configured in the mntner object as described in Section 2.3.3.

If a new object with a mnt-by attribute is added to the database or a mnt-by attribute is added to an object that previously had no such attribute, the authorisation step is performed on the maintainer `to be added`.

If a database object has both a "notify" attribute and a "mnt-by" attribute, then two notification messages will be generated. If both notifications are to be sent to the same e-mail address, then only one e-mail is sent, but it will contain two identical parts.

E.g.

`Dear Colleague,`

This is to notify you that some object(s) in the RIPE Database which you either maintain or are listed as to-be-notified have been added, deleted or changed.

The objects below are the old and new entries for these objects in the database. In case of DELETIONS, the deleted object is displayed. NOOPs are not reported.

The update causing these changes had the following mail headers:
- From: Ambrose Magee <Ambrose.Magee@ripe.net>
- Subject: LONGACK
- Date: Tue, 17 Dec 1996 10:36:47 +0100 (MET)
- Msg-Id: <199612170936.KAA10058@kantoor.ripe.net>

RIPE Database Notification Department
\---

PREVIOUS OBJECT:

```
person:     Ambrose Magee
address:    RIPE Network Co-ordination Centre (NCC)
address:    Kruislaan 409
address:    NL-1098 SJ   Amsterdam
address:    The Netherlands
phone:      +31 20 592 5065
fax-no:     +31 20 592 5090
e-mail:     ambrose.magee@ripe.net
nic-hdl:    AMRM1-RIPE
notify:     ambrose@ripe.net
mnt-by:     AMRM1-RIPE-MNT
changed:    ambrose@ripe.net 960920
changed:    ambrose@ripe.net 961001
changed:    ambrose@ripe.net 961217
source:     RIPE
```

REPLACED BY:

```
person:     Ambrose Magee
address:    RIPE Network Co-ordination Centre (NCC)
address:    Kruislaan 409
address:    NL-1098 SJ   Amsterdam
address:    Kingdom of the Netherlands
phone:      +31 20 592 5065
fax-no:     +31 20 592 5090
e-mail:     ambrose.magee@ripe.net
nic-hdl:    AMRM1-RIPE
notify:     ambrose@ripe.net
mnt-by:     AMRM1-RIPE-MNT
changed:    ambrose@ripe.net 960920
changed:    ambrose@ripe.net 961001
changed:    ambrose@ripe.net 961217
source:     RIPE
---
```

PREVIOUS OBJECT:

```
person:     Ambrose Magee
address:    RIPE Network Co-ordination Centre (NCC)
address:    Kruislaan 409
address:    NL-1098 SJ   Amsterdam
address:    The Netherlands
phone:      +31 20 592 5065
fax-no:     +31 20 592 5090
```

e-mail: ambrose.magee@ripe.net
nic-hdl: AMRM1-RIPE
notify: ambrose@ripe.net
mnt-by: AMRM1-RIPE-MNT
changed: ambrose@ripe.net 960920
changed: ambrose@ripe.net 961001
changed: ambrose@ripe.net 961217
source: RIPE

REPLACED BY:
person: Ambrose Magee
address: Ambrose Magee
address: RIPE Network Co-ordination Centre (NCC)
address: Kruislaan 409
address: NL-1098 SJ Amsterdam
address: Kingdom of the Netherlands
phone: +31 20 592 5065
fax-no: +31 20 592 5090
e-mail: ambrose.magee@ripe.net
nic-hdl: AMRM1-RIPE
notify: ambrose@ripe.net
mnt-by: AMRM1-RIPE-MNT
changed: ambrose@ripe.net 960920
changed: ambrose@ripe.net 961001
changed: ambrose@ripe.net 961217
source: RIPE

NOOP means "NO OPeration" i.e. the object was not changed.

(iv) Notification of unauthorised attempts to change objects protected by a mntner:

Any request to change an object protected by a mntner must first satisfy the authorisation scheme given in the mntner. If the request does not satisfy these scheme, then it is forwarded to the mailbox given in the "upd-to" attribute of the mntner object. Whoever attempted the update is also notified that this has occurred. No other notifications are sent in this case.

There is another use of the mailbox specified in the "upd-to" attribute. If a person wishes to change an object, but is not authorised to do so, then the update should be sent to this mailbox.

If an update is not authorised and no appropriate maintainer can be identified the request will be forwarded to the RIPE NCC for action.

2.3.3 Authorization schemes:

This is specified using the "auth" attribute of the mntner object. The required scheme is specified here.

Example:

```
mntner:      AMRM1-RIPE-MNT
descr:       Ambrose's mntner.
admin-c:     AMRM1-RIPE
tech-c:      AMRM1-RIPE
upd-to:      ambrose@ripe.net
mnt-nfy:     ambrose@ripe.net
auth:        CRYPT-PW qlfichzrkkwgm
remarks:     This is a test mntner.
notify:      ambrose@ripe.net
mnt-by:      AMRM1-RIPE-MNT
changed:     ambrose@ripe.net 970115
source:      RIPE
```

There are three schemes;

NONE, MAIL-FROM and CRYPT-PW

The format is:

<scheme-id> <auth-info>

The scheme-ids currently defined are: NONE, MAIL- FROM and CRYPT-PW. The auth-info is additional information required by a particular scheme. When more than one auth attribute is specified any of them can be used alternatively for authentication of updates, i.e. specifying NONE and others does not make much sense. The auth attribute is not protected information in any way; it is returned normally like any attribute by the whois server and available in stored copies of the database. The strength of an authentication scheme thus has to lie in the scheme itself and cannot be based on the obscurity of the auth attribute or on keeping this information secret. It is very difficult to keep the information confidential and thus the RIPE NCC cannot take this responsibility.

The "NONE" scheme:

With this scheme, any changes made are notified to the mailbox specified in the "mnt-nfy" field. There is no authentication i.e. anyone can make changes.

However, if you are in doubt whether or not to change the object maintained (i.e. protected) by the mntner, you should send your request to the mailbox specified in the "upd-to" attribute.

This is the simplest authentication scheme which is entirely backwards compatible with the one previously used. No authentication is provided, i.e. it is assumed that all update requests originate from an authorised maintainer or are at least coordinated with one. Anyone in doubt whether it is OK to issue update requests should check with the maintainer concerned first, preferably at the mailbox specified in the upd-to attribute. When making any changes the mnt-by attribute should not be changed without explicit consent from the current maintainer.

This scheme is for those who want to give everyone the possibility to make changes while at the same time using the mnt-by attribute for its notification and documentation features. A good reason to use this scheme is when the maintainer cannot guarantee timely processing of updates themselves.

The "MAIL-FROM" scheme:

This scheme checks the "From" line in the e-mail which contains the update. If it is the same as what is specified here, the update is allowed.

This authentication method checks the content of the RFC822 From: header of an update request against the regular expression specified as <auth-info>. If the regular expression matches the content of the From: header the update request is authenticated success- fully. The regular expressions supported are described in POSIX 1003.2 section 2.8. As it is expected that most regular expressions will either be literals or of a form similar to .*@some.domain.or.other an extensive description of the possibilities will not be given. Note that the matching is applied to the whole content of the From: header including comments if present. No attempt is made to isolate the mailbox part.

It should be stressed that this authentication scheme is very weak. Forging RFC822 headers does not take much effort or ingenuity. The reason for the scheme's existence is that it easily prevents accidental updates rather than allowing them first and fixing them later when notified.

This scheme is for those who want to prevent accidental updates by others and are able to process update requests in a timely manner.

The "CRYPT-PW" scheme:

This uses an encryption process which is similar to that used to make and check login passwords in UNIX.

A password is chosen by the user; this is encrypted and put in the database. The encrypted password can be seen by everyone. The user sends the clear password in the same mail as the request to change an object.

It is difficult to get the original, clear password from the encrypted one; that is why the encrypted one can be put in the public database.

However, passwords based on words which may be found in a dictionary should be avoided; encrypted passwords based on such words may be broken by the use of lots of fast machines running for a sufficiently long time.

This scheme uses the Unix crypt(3) routine, which is also used for login passwords under Unix. This routine provides a so called "trap door" function, the inverse of which is somewhat hard to calculate. The password provided by the user is encrypted with this function and stored in its encrypted form only. When the user later provides the password again for authentication, the encryption is repeated and the results are compared. Since the original (cleartext) password cannot easily be computed from the encrypted version the encrypted password does not have to be kept secret.

The <auth-info> is the encrypted password. This can either be obtained locally with the appropriate Unix tools or on e-mail request from <ripe-dbm@ripe.net>. When sending in update requests the cleartext password has to be provided in the message body by specifying

password: cleartext-password

at the beginning of a line and preceding any update requests to be thus authenticated. The password will remain valid for all requests following it in the same e-mail message or until another password is specified.

This scheme is slightly stronger than the MAIL-FROM scheme. It is by no means meant to keep out a determined malicious attacker. The crypt function is vulnerable to exhaustive search by (lots of) fast machines and programs to do the searching are widely available. For this reason it is strongly discouraged to use encrypted passwords also used for other purposes such as Unix login accounts in this scheme. As you are publishing the encrypted password in the database it is open to attack. The usual caveats about crypt passwords apply, so is not very wise to use words or combinations nations of words found in any dictionary of any language. See [R. Morris, K. Thompson: Password Security: A Case History] for more details about the scheme and its vulnerabilities.

Example of use:

- auth: NONE
- auth: CRYPT-PW dhjsdfhruewf
- auth: MAIL-FROM .*@ripe.net

The "auth" field must be included in a mntner object, even if no authorisation scheme other than ‘NONE’ is required. Furthermore, several "auth"fields may be included in the same mntner object.

Example:

```
mntner:      AMRM1-RIPE-MNT
descr:       Ambrose's mntner object.
admin-c:     AMRM1-RIPE
tech-c:      AMRM1-RIPE
upd-to:      ambrose@ripe.net
mnt-nfy:     ambrose@ripe.net
auth:        MAIL-FROM Ambrose.Magee@ripe.net
auth:        MAIL-FROM ripe-dbm@ripe.net
auth:        CRYPT-PW  eiktegisvfxg
notify:      ambrose@ripe.net
mnt-by:      AMRM1-RIPE-MNT
changed:     ambrose@ripe.net 961011
source:       RIPE
```

Multiple authentication methods can be specified in the same mntner object. They all have equal status i.e. any one of the authenticators is sufficient to authenticate the update request from the maintainer.

In the above example, objects protected by this mntner object may be changed if:

- (1) the e-mail message is from Ambrose.Magee@ripe.net;
- (2) the e-mail message is from ripe-dbm@ripe.net;
- (3) the e-mail message contains the unencrypted form of the password eiktegisvfxg.

If multiple maintainers maintain an object this feature should not be used. Multiple maintainers should be represented by multiple mntner objects referenced in the mnt-by attribute. There is no way in the current model to require a combination of multiple authenticators to authenticate a request.

2.4 Obtaining a mntner object:

Unlike other objects, mntner objects are not created automatically; they are created on your behalf by the staff at RIPE NCC. To obtain a mntner object, do the following:

(i) First, obtain a template of a mntner object by using the "-t" option with the ‘whois’ command (see Section 2.1).

(ii) Fill in the template and send it to `<auto-dbm@ripe.net>`, from where it will be forwarded to `<ripe-dbm@ripe.net>`. You may also send it directly to `<ripe-dbm@ripe.net>`.

(iii) Your mntner object will be created as soon as possible, if your request satisfies the criteria mentioned below.

Criteria

(i) The request comes from someone who is responsible for the maintenance of objects that are related to public address space, domain names and routes.

(ii) Normally, individuals who request a mntner object already have objects in the RIPE database which they wish to protect.

(iii) Such objects should be related to public address space or domain names that have been assigned by the appropriate registry.

Appendix

Mailing Lists

1. Introduction

There are three mailing lists that are relevant to the use and development of the RIPE database:

- db-wg
- db-beta
- rr-impl

1.1. db-wg

This is the mailing list of the RIPE Database Working Group. This working group deals with all aspects of the RIPE network management database.

The current chair of this group is
`Wilfried Woeber <woeber@cc.univie.ac.at>`.

Membership of this list is open.
The list is archived since October 1992.
Send mail to `<db-wg@ripe.net>`.

1.2. rr-impl

This mailing list is to discuss Routing Registry Implementations.
Membership is open, but aimed at Routing Registry Implementors.
The list is archived since Feb 1994
Send mail to `<rr-impl@ripe.net>`.

1.3. info db-beta

This mailing list is used to exchange information about the current beta release of the RIPE database software.

The list membership is closed to beta testers.

The list is archived since October 1993.

Send mail to `<db-beta@ripe.net>`.

2. More Information

For more information about any of these lists, contact `<ncc@ripe.net>` or `<ripe-dbm@ripe.net>`.