RIPE

# Support of Guarded fields within the RIPE Database

*Tony Bates*

*Document-ID: ripe-117*
*Obsoletes: ripe-108*

## 1. Introduction

The RIPE database contains several significant attributes which make it well suited for use as part of operational procedures and configuration. Most significantly are the attributes which make up the RIPE Routing Registry (RR) as specified in RIPE-81 [1][2], namely the *"aut-sys" and "comm-list"* attributes. For these attributes to be of use to service providers they must be:

- Properly authorised.
- Efficient for both maintainers of the attributes and the maintainers of the whole database.

This document describes an overview of the RIPE database attributes which are guarded, the procedure for updating these guarded attributes and the general use of "guarded" fields within the RIPE database. It should be noted that this document is an update of the original document, ripe-108 [5]. The significant change is section 4 which provides details of the exact matching algorithm now used for updating objects. This is a change from the original method used in ripe-108.

It should also be noted that with the database changing to a classless model and the stronger separation of the routing registry from the allocation registry guarded attributes will have a limited life. Significant investments in tools and procedures for guarded attributes is therefore discouraged.

## 2. The Database Guarded Attributes

All the guarded attributes currently supported in the RIPE database are contained within the "inet-num" or network object. However, the association corresponds to their relevant guarded database objects. If we look at a simple example this becomes clear:

```
inetnum:   192.87.45.0
netname:   RIPE-NCC
descr:     RIPE Network Coordination Centre
descr:     Amsterdam, Netherlands
country:   NL
admin-c:   Daniel Karrenberg
tech-c:    Marten Terpstra
connect:   RIPE NSF WCW
aut-sys:   AS1104
comm-list: SURFNET
ias-int:   192.87.45.80  AS1104
ias-int:   192.87.45.6   AS2122
ias-int:   192.87.45.254 AS2600
rev-srv:   ns.ripe.net
rev-srv:   ns.eu.net
notify:    ops@ripe.net
changed:   tony@ripe.net 940110
source:    RIPE
```

This shows that the RIPE-NCC network belongs to autonomous system 1104 and is in a community known as SURFNET. This is valuable information that could easily be used for example for routing policy purposes (as well as other operational uses). Currently support for the following set of guarded attributes is implemented:

aut-sys

The "aut-sys" attribute has a direct mapping to "aut-num" objects as defined in RIPE-81. That is the Autonomous System (AS) that the network number is a part of. As defined in RIPE-81, a network can only belong to one AS and hence the "aut-sys" attribute can only contain one AS number. The syntax of the "aut-sys" attribute is:

AS<positive integer between 1 and 65535> (1). i.e. AS1104

comm-list

The "comm-list" attribute has a direct mapping to "community" objects as defined in RIPE-81. A network can belong to more than one community. The syntax of "comm-list" is:

Multiple text strings which cannot start with "AS" or any of the <routing policy expression> KEYWORDS defined in RIPE-81.

As these attributes are tightly coupled to their associated objects it makes sense for these attributes to be updated not by the network maintainer but by the maintainer of the referenced

(1) This represents a change from RIPE-50 [3] where the "aut-sys" attribute was defined to be a positive integer only, not containing the string "AS" at the start. This change has been made to be consistent with the "aut-num" object syntax.

object(s). The basic premise behind this is that these attributes should be used for various operational procedures such as setting routing policy, accounting and so on. For these attributes to be used by network operators for day to day operations they need to be guarded in such a way that can be trusted and are guaranteed to be unique - with any conflicts quickly and easily resolved. The procedure for achieving this is detailed below.

## 3. The Basic Procedure

For each of the guarded attributes detailed above, a list of all networks having this attribute is kept separately from the general database itself. These lists (also called 'guarded files') will be maintained and be served as the 'only' source of membership information used in the database. Normal database updates 'never' change these attributes. If an update includes such an attribute and a discrepancy between the values in the update and those in the database is found, a diagnostic message will be sent to the originator of the update and the guarded value(s) will not be affected. The attributes as defined in these files are incorporated in the database once a day. To ensure proper control and authorisation, these lists will be maintained at the RIPE NCC on the same machine that contains the RIPE database. The "guardians" of the corresponding database objects will have to maintain their own guarded files. The guardians are provided with individually assigned login accounts at the RIPE NCC. The guardians can themselves decide in what manner they want to update their file. The NCC will offer interactive logins, ftp logins or any other means that might be deemed useful.

### 3.1. Some Details

As stated each guardian will be issued with an account on the central NCC machine known as **'guardian.ripe.net'.** This account will contain a 'restricted' environment which will allow the guardian of the relevant object to update their associated guardian file (2). Wherever possible the account name issued to the guardian will be the same as the object name.

For example, the guardian of the AS1104 aut-num object will receive an account known as "AS1104". With each guardian account the corresponding file will be parsed at each update run (once a day). This file will contain the list on networks associated with the object. See appendix A for details of the format and syntax of the guardian files.

A tool will also be provided within the restricted environment to syntax check the guarded file to avoid against possible typos and errors.

With each account, an electronic mail address (this is a mandatory attribute for all guarded objects) will be used by the NCC and database software. To make this flexible for the guardian a ".forward" file with the account which can be change when required. This will mean mail sent to **<guardian-name@guardian.ripe.net>** will go the to correct guardian.

---

(2) As stated, the mechanism for updating the guardian file will initially be by interactive login or file transfer. However, this doesn't preclude other mechanisms in the future.

### 3.2. How does it work ?

For each of the guarded files found on ‘guardian.ripe.net’ the database software will load any guarded attribute value(s) for the network object(s) listed in the guarded file. This will take place at the same time as the database is garbage collected (currently at 0500 MET). If a conflict is found (i.e if more than one entry exists for an attribute which can only contain one entry, currently only "aut-sys" contains this property), the current value will remain unchanged and all guardians involved in the conflict will be sent an electronic mail message informing them of the conflict. See Appendix B for an example.

If no conflict is found the attribute will be updated with the guarded value.

Correspondingly, to remove a guarded attribute just remove the network entry from the relevant guarded file and it will be deleted at the next update run. To be notified of this delete the "notify" attribute should be used.

If a guardian file contains an entry which is not in the database then the guardian will be notified as part of the conflict handling procedure.

If an update is sent to the database software using another mechanism (i.e. mail to auto-dbm@ripe.net) that contains a guarded attribute, this will not be allowed to change the guarded attribute. If the value of the attribute is the same as what is currently registered in the database then no warnings will be given. However, if the update contains a value for a guarded attribute that is different to that registered in the database, a warning will be sent to the originator and the guarded value will remain unchanged. Any changes of other (unguarded) fields in the update will be checked for syntactic correctness and if they pass will go through to the database irrespective of any conflicts for the guarded fields.

When through the normal database update procedure an object with guarded attributes is deleted, the guardians of these guarded attributes will be notified of this deletion. Only deletions will be notified in this way to guardians. For normal changes the "notify" attribute of the database should be used.

Although the guarded process will run once a day as part of the database garbage collection procedure it will also be possible to, "on request to the NCC", run an emergency guarded update process for a particular guarded object.

To have complete guardian accounts removed from the NCC machine, and thus all references to this guarded value, please contact the RIPE NCC at <ripe-dbm@ripe.net>. Removing an account and the guardian file that goes with it means that this guarded value will not be added any longer to any of the objects in the database.

## 4. Block Splitting

### 4.1. Current implementation

Whilst it was not directly stated in ripe-108. The implementation used for guarded attributes meant that if a value in a guardian file matched any value which was part of an object registered as a range in the database the object would be automatically split into separate objects by the guarded field mechanism. For example, if we had a database registered object of the following:

```
inetnum:   193.0.0.0 - 193.0.3.0
netname:   NCC-RS-TEST
descr:     Test networks
country:   NL
admin-c:   Tony Bates
tech-c:    Tony Bates
tech-c:    Marten Terpstra
connect:   RIPE
changed:   tony@ripe.net 930312
source:    RIPE
```

And we entered in the AS3333 file the entry 193.0.1.0, the guarded field mechanism would produce the following entries:

```
inetnum:   193.0.0.0
netname:   NCC-RS-TEST
descr:     Test networks
country:   NL
admin-c:   Tony Bates
tech-c:    Tony Bates
tech-c:    Marten Terpstra
connect:   RIPE
changed:   tony@ripe.net 930312
source:    RIPE
```

```
inetnum:   193.0.1.0
netname:   NCC-RS-TEST
descr:     Test networks
country:   NL
admin-c:   Tony Bates
tech-c:    Tony Bates
tech-c:    Marten Terpstra
connect:   RIPE
aut-sys:   AS3333
changed:   tony@ripe.net 930312
source:    RIPE
```

```
inetnum:   193.0.2.0 - 193.0.3.0
netname:   NCC-RS-TEST
descr:     Test networks
country:   NL
admin-c:   Tony Bates
tech-c:    Tony Bates
tech-c:    Marten Terpstra
connect:   RIPE
changed:   tony@ripe.net 930312
source:    RIPE
```

Here you can see the original object (193.0.0.0 - 193.0.3.0) has been split into three objects with the 193.0.1.0 object having the associated aut-sys attribute value AS3333 added.

### 4.2. A change in implementation - exact match algorithm

When moving the database to a classless paradigm, one part of the Internet address space can -and will- be described by multiple objects at different levels of granularity. Hence it becomes very difficult specify block splitting and to implement it correctly. Due to the clearer separation of the Routing Registry from the Allocation Registry on the other hand the lifetime of the guarded attribute mechanism is clearly finite. Once other authorisation mechanisms are used there will be no need for block splitting any more. The best solution therefore is not to invest in complex block splitting.

The current automatic splitting technique will not be used anymore. A guarded attribute will now only be updated (in accordance with the details in 3.2) if the object value and the value in the guarded file are EXACTLY the same. If they DO NOT match the guardian will be notified that the entry in the guarded file could not be found in the database.

One direct implication of this is that the guardian must now know the exact value of the object registered in database before he or she can update the object with their guarded attribute value. This will mean that some manual block splitting may need to take place before the attribute can be successfully added. This is an extremely simple procedure. This produce can be summarised as the following simple steps:

1) Delete the current object
2) Send in new objects split at correct values to provide exact match needed for adding relevant guarded attributes.

If we look at the above example of an object registered as 193.0.0.0 - 193.0.3.0 in the database and we would like to add the guarded aut-sys attribute to 193.0.1.0. We would need to send the following message to **`auto-dbm@ripe.net`**:

```
inetnum:    193.0.0.0 - 193.0.3.0
netname:    NCC-RS-TEST
descr:      Test networks
country:    NL
admin-c:    Tony Bates
tech-c:     Tony Bates
tech-c:     Marten Terpstra
connect:    RIPE
changed:    tony@ripe.net 930312
source:     RIPE
delete:     tony@ripe.net deleted for guarded split
```

```
inetnum:    193.0.0.0
netname:    NCC-RS-TEST
descr:      Test networks
country:    NL
admin-c:    Tony Bates
tech-c:     Tony Bates
tech-c:     Marten Terpstra
connect:    RIPE
changed:    tony@ripe.net 930312
source:     RIPE
```

```
inetnum:    193.0.1.0
netname:    NCC-RS-TEST
descr:      Test networks
country:    NL
admin-c:    Tony Bates
tech-c:     Tony Bates
tech-c:     Marten Terpstra
connect:    RIPE
changed:    tony@ripe.net 930312
source:     RIPE
```

```
inetnum:   193.0.2.0 - 193.0.3.0
netname:   NCC-RS-TEST
descr:     Test networks
country:   NL
admin-c:   Tony Bates
tech-c:    Tony Bates
tech-c:    Marten Terpstra
connect:   RIPE
changed:   tony@ripe.net 930312
source:    RIPE
```

The key point to note is the ordering of the update sent to auto-dbm@ripe.net. The delete of the original entry must take place first. The split objects can then be added without any problems. The objects can of course be sent in separate messages if preferred.

## 5. Conclusion

The update procedure as detailed above has the following advantages:

- Authorisation of adding/deleting is guaranteed.
- No need for mailing back and forth of authorisation messages.
- Simple procedure for both database maintainers and guardians.
- Guardians keep full control of their attribute.

It allows for the addition of any number of guarded attributes in the future. It describes a simple but effective procedure for maintaining the guarded files whilst not precluding alternate mechanisms in the future.

## 6. References

[1] Bates, T., Jouanigot, J-M., Karrenberg, D., Lothberg, P., Terpstra, M., "Representation of IP Routing Policies in the RIPE Database", RIPE-81, February 1993.

[2] Bates, T., Karrenberg, D., "Description of Inter-AS Networks in the RIPE Routing Registry", RIPE-103, December 1993.

[3] Karrenberg, D., "RIPE Database Template for Networks", RIPE-50, April 1992.

[4] J.-M. Jouanigot, "Policy based routing within RIPE", May 1992.

[5] Bates, T, "Support of Guarded fields within the RIPE Database", July 1994.

**Appendix A — Format of Guardian Files.**

We propose to keep the file format as simple as possible. The name of the file is identical to the name of guarded object. The format used within the file is kept simple. It allows lines to be either comments or the actual object entry that is to be guarded. A comment must contain either a semicolon (;) or hash (#) at the beginning of the comment line. The object name entries must be exactly the same as they are in the database. Currently, the only object containing guarded attributes is the "inetnum" object so the file can contain either the 'well-known' dotted quad network notation or RIPE dotted quad range notation. Here is a simple example of what the AS1104 guarded file would look like. The file would be stored in the home directory of the AS1104 account on guardian.ripe.net and be called AS1104 (told you it was simple). It would contain something like the following:

```
#
# File : AS1104
#
; An alternate comment format
;
; This file was updated by jan.dijkstra@gouda.nl
; on 940109
;
192.16.183.0
192.16.185.0 - 192.16.186.0
192.16.194.0
192.16.199.0
192.87.45.0
```

Empty lines in the file are also ignored but you are encouraged to keep the file as concise as possible.

As stated above, a tool known as 'checkguard' will be available to make it simple to check the syntax of the guarded file.

NOTE: the entry in the file must match exactly the object entry in the database. See section 4 for more details.

**Appendix B - Example of conflict handling**

If a conflict occurs (e.g. by listing the same network number in more than one AS guarded file), then each of the guardians involved will be notified of the conflict by electronic mail. Let's look at a simple example. Suppose the guardians for AS1104 and AS2122 update their relevant guardian files and create a conflict by having the same network in them. For this example he network in question is "192.16.183.0". Here is the AS1104 guardian file:

```
#
192.16.183.0
192.16.185.0
192.16.186.0
192.16.194.0
192.16.199.0
192.87.45.0
```

And here is the AS2122 guardian file:

```
#
192.16.183.0
193.0.0.0 - 193.0.7.0
```

As you can see "192.16.183.0 exists in both files.

At update time the following mails are generated. Firstly, to the guardian of AS2122.

```
Date: Fri, 14 Jan 1994 13:22:43 +0100
Message-Id: <9401141222.AA07125@ns.ripe.net>
From: RIPE Database Conflict Handler <ripe-dbm@ripe.net>
Subject: Guarded attributes conflicts found
To: as2122@ripe.net

Dear Guardian,

One or more conflicts have been found regarding guarded
attributes in the RIPE database. Some of the conflicts
concern the guarded values you are a guardian for.

Please verify and correct the conflicts below.
The guarded values for objects below have been set to
the value they had in the database before this guarded
attributes run.

Kind Regards,
RIPE Database Conflict Department

------

"192.16.183.0" also appears in guardian files: AS1104
```

And similarly to the AS1104 guardian.

```
Date: Fri, 14 Jan 1994 13:22:42 +0100
Message-Id: <9401141222.AA07121@ns.ripe.net>
From: RIPE Database Conflict Handler <ripe-dbm@ripe.net>
Subject: Guarded attributes conflicts found
To: as1104@ripe.net

Dear Guardian,

One or more conflicts have been found regarding guarded
attributes in the RIPE database. Some of the conflicts
concern the guarded values you are a guardian for.

Please verify and correct the conflicts below.
The guarded values for objects below have been set to
the value they had in the database before this guarded
attributes run.

Kind Regards,
RIPE Database Conflict Department

------

"192.16.183.0" also appears in guardian files: AS2122
```

From this you can see conflict can be quickly and easily resolved, assuming good collaboration between the guardians. The existing database entry will of course not be changed with regard to the guarded attribute) as long as there exists a conflict.