Authorisation and Notification of Changes in the RIPE Database

Daniel Karrenberg
RIPE NCC

Document ID: ripe-96

ABSTRACT

Two new attributes are defined for all objects in the RIPE database in order to implement a generalised method for authorising changes and to notify interested parties of any changes made to a specific object. In addition the authorisation method provides a convenient way for distributed maintenance of the database.

The Notify Attribute

Each RIPE database object has an optional attribute called *notify*. The value of the *notify* attribute is one valid RFC822 e-mail address. There can be multiple *notify* attributes. Whenever the object concerned is changed in the database a notification message will be sent to each e-mail addresses appearing in a *notify* attribute.

This makes it straightforward to keep track of changes to specific objects and prevent changes from going unnoticed. Multiple *notify* attributes make it possible to notify a number of interested parties. This could be used to alert all contact persons for an object or the local contact persons as well as the relevant service provider. Although it may be tempting to put many *notify* attributes on database objects in order to notify everyone even remotely interested, this is not recommended. A very few key addresses should be sufficient. Prior to entering any mail address here, the explicit or implicit consent of the person responsible for that particular mailbox needs to be obtained.

The Maintainer Attribute

Each RIPE database has an optional attribute called *maintainer.* The value of the *maintainer* attribute is a registered maintainer name. There can only be one *maintainer* attribute per object. Whenever a change to the object concerned is attempted in a copy of the database the maintainer attribute of the current database object is examined.

If there is no maintainer attribute or the maintainer name is authorised to make changes in the copy of the database the update proceeds causing the necessary notifications as per the notify attribute.

If the maintainer name has no authorisation to change the local copy of the database, the update request is forwarded to the maintainer for processing. No notifications are performed in this case.

The following data will be maintained locally about each maintainer:

Maintainer name	
Authority	none change whole database change only own objects
Forwarding Info	mail/RFC822-address other/address
Authorisation	none mail/RFC-822-address other/key

Example 1: Regional Registries

In order to align the InterNIC and RIPE databases it has been agreed that European objects will be maintained in Europe. The RIPE NCC will provide the data for these objects to the InterNIC for inclusion in their database without further processing. The RIPE NCC will refer all updates for non-European objects to the InternNIC and the InterNIC will refer all updates for European objects to the RIPE NCC for processing.

This will be achieved by creating two maintainer names: INTERNIC and RIPE-NCC and tagging all European objects with RIPE-NCC and vice versa. The tags will be phased in slowly, avoiding a flag day with the associated massive consistency problems. Over time all objects in the RIPE database will be thus tagged.

Updates from third parties for objects with the maintainer attribute added can now be referred correctly. Updates from the other registry for objects it maintains can be accepted without further checking.

Example 2: Local Registries

Some European local registries keep their own copies[1] of the database containing the objects within their area. This leads to consistency problems as updates can be sent both to the RIPE NCC and to the local registry. Referrals are performed by ad hoc methods. Frequently only one of the databases is updated and alignment needs to be done manually.

By registering maintainer names for the local registries and tagging the appropriate objects this can be automated and made more reliable. The NCC would forward update requests for locally maintained objects to the local registry unless they come from that local registry itself.

Example 3: Guarded Objects

Some objects such as the autonomous-system object (see ripe-81) need to be protected against changes by anyone but a designated guardian since changes to these objects have a direct operational impact.

By registering appropriate maintainer names for the guardians and tagging the objects to be protected this functionality can be provided in a canonical way. Any change by third parties to such an object will not only be prevented but cause automatic notification of the guardian through the forwarding mechanism.

[1]: In fact some European local registries maintain their own database of registrations within their area. Selected fields of this database are sent on an ad-hoc or regular basis to RIPE to be included in the RIPE whois database. The selected fields may be subject to further processing before being sent to the RIPE database.