Network Working Group   C. Villamizar
Request for Comments: 2725   Avici
Category: Standards Track   C. Alaettinoglu
                                                                  ISI
                                                             D. Meyer
                                                                Cisco
                                                            S. Murphy
                                                                  TIS
                                                        December 1999

                     Routing Policy System Security

Abstract

   The RIPE database specifications and RPSL language define languages
   used as the basis for representing information in a routing policy
   system. A repository for routing policy system information is known
   as a routing registry. A routing registry provides a means of
   exchanging information needed to address many issues of importance to
   the operation of the Internet. The implementation and deployment of
   a routing policy system must maintain some degree of integrity to be
   of any operational use. This document addresses the need to assure
   integrity of the data by providing an authentication and
   authorization model.

Table of Contents

1 Overview

   The Internet Routing Registry (IRR) has evolved to meet a need for
   Internet-wide coordination. This need was described in RFC-1787, an
   informational RFC prepared on behalf of the IAB [14]. The following
   summary appears in Section 7 of RFC-1787.

      While ensuring Internet-wide coordination may be more and more
      difficult, as the Internet continues to grow, stability and
      consistency of the Internet-wide routing could significantly
      benefit if the information about routing requirements of various
      organizations could be shared across organizational boundaries.
      Such information could be used in a wide variety of situations
      ranging from troubleshooting to detecting and eliminating
      conflicting routing requirements. The scale of the Internet
      implies that the information should be distributed. Work is
      currently underway to establish depositories of this information
      (Routing Registries), as well as to develop tools that analyze, as
      well as utilize this information.

   A routing registry must maintain some degree of integrity to be of
   any use. The degree of integrity required depends on the usage of
   the routing policy system.

   An initial intended usage of routing policy systems such as the RIPE
   database had been in an advisory capacity, documenting the intended
   routing policies for the purpose of debugging. In this role a very
   weak form of authentication was deemed sufficient.

   The IRR is increasingly used for purposes that have a stronger
   requirement for data integrity and security. This document addresses
   issues of data integrity and security that is consistent with the
   usage of the IRR and which avoids compromising data integrity and
   security even if the IRR is distributed among less trusted
   repositories.

2 Background

   An early routing policy system used in the NSFNET, the policy routing
   database (PRDB), provided a means of determining who was authorized
   to announce specific prefixes to the NSFNET backbone. The need for a
   policy database was recognized as far back as 1989 [6, 4]. By 1991
   the database was in place [5]. Authentication was accomplished by
   requiring confirmation and was a manually intensive process. This
   solved the problem for the NSFNET, but was oriented toward holding
   the routing policy of a single organization.

The problem since has become more difficult. New requirements have
emerged.

1. There is a need to represent the routing policies of many
   organizations.

2. CIDR and overlapping prefixes and the increasing complexity of
   routing policies and the needs of aggregation have introduced new
   requirements.

3. There is a need to assure integrity of the data and delegate
   authority for the data representing specifically allocated
   resources to multiple persons or organizations.

4. There is a need to assure integrity of the data and distribute the
   storage of data subsets to multiple repositories.

The RIPE effort specificly focused on the first issue and needs of
the European community. Its predecessor, the PRDB, addressed the
needs of a single organization, the NSF. The RIPE database formats as
described in [2] were the basis of the original IRR.

Routing protocols themselves provide no assurance that the
origination of a route is legitimate and can actually reach the
stated destination. The nature of CIDR allows more specific prefixes
to override less specific prefixes [9, 15, 8]. Even with signed
route origination, there is no way to determine if a more specific
prefix is legitimate and should override a less specific route
announcement without a means of determining who is authorized to
announce specific prefixes. Failing to do so places no assurance of
integrity of global routing information and leaves an opportunity for
a very effective form of denial of service attack.

The Routing Policy System Language (RPSL) [1, 13] was a fairly
substantial evolutionary step in the data representation which was
largely targeted at addressing the second group of needs. The PRDB
accommodated CIDR in 1993 [12] and the RIPE database accommodated the
entry of CIDR prefixes from inception, but RPSL provides many needed
improvements including explicit support for aggregation.

This document addresses the third group of needs identified above.

While the current implementation supporting weak authentication
doesn't guarantee integrity of the data, it does provide extensive
mechanisms to make sure that all involved parties get notified when a
change is made to the database, whether the change was malicious or
intended. This provides inadequate protection against additions.
Since the software is increasingly used to configure the major parts

of the Internet infrastructure, it is not considered to be adequate
anymore to know about and have the ability roll back unintended
changes. Therefore, more active security mechanisms need to be
developed to prevent such problems before they happen.

A separate document will be needed to address the fourth group of
needs.

3 Implicit Policy Assumptions

The authorization model encodes certain policies for allocation of
address numbers, AS numbers, and for the announcement of routes.
Implicit to the authorization model is a very limited number of
policy assumptions.

1. Address numbers are allocated hierarchically. The IANA delegates
   portions of the address space to the regional registries
   (currently ARIN, APNIC and RIPE), which in turn delegate address
   space to their members, who can assign addresses to their
   customers.

2. AS numbers are allocated either singly or in small blocks by
   registries. Registries are allocated blocks of AS numbers,
   thereby making the allocation hierarchical.

3. Routes should only be announced with the consent of the holder of
   the origin AS number of the announcement and with the consent of
   the holder of the address space.

4. AS numbers and IP address registries may be different entities
   from routing registries.

For subsets of any of these three allocation spaces, network
addresses, AS numbers, and routes, these restrictions may be loosened
or disabled by specifying a very weak authorization method or an
authentication method of "none". However, even when no
authentication mechanism is used, all involved parties can be
notified about the changes that occurred through use of the existing
"notify" attribute.

4 Scope of Security Coverage

This document is intended only to provide an authentication and
authorization model to insure the integrity of the policy data in a
registry. Only authetication and authorization of additions,
deletions, and changes to the database are within the scope of this
document. Authentication and authorization of database queries is

explicitly out of scope. Mutual authentication of queries, that is
authenticating both the origin of the query and the repository from
which query results are obtained, is also out of scope.

5 Organization of this Document

Familiarity with RIPE-181 [2] and RPSL [1] is assumed throughout this
document. Goals are described in Section 6. Section 7 through
Section 9 provide descriptions of the changes and discussion.
Section 10 provides a concise summary of data formats and semantics.
Appendix C through Appendix E provide additional technical
discussion, examples, and deployment considerations.

   Goals and Requirements Section 6 provides a more detailed
   description of the issues and identifies specific problems that
   need to be solved, some of which require a degree of cooperation
   in the Internet community.

   Data Representation Section 7 provides some characteristics of
   RPSL and formats for external representations of information.

   Authentication Model Section 8 describes current practice,
   proposes additional authentication methods, and describes the
   extension mechanism if additional methods are needed in the
   future.

   Authorization Model Section 9 describes the means of determining
   whether a transaction contains the authorization needed to add,
   modify, or delete specific data objects, based on stated
   authentication requirements in related data objects.

   Data Format Summaries Section 10 provides a concise reference to
   the data formats and steps in transaction processing.

   Technical Discussion Section C contains some discussion of
   technical tradeoffs.

   Common Operational Cases Section D provides some examples drawn
   from past operational experience with the IRR.

   Deployment Considerations Section E describes some deployment
   issues and discusses possible means of resolution.

6 Goals and Requirements

The Internet is an open network. This openness and the large scale
of the Internet can present operational problems. Technical
weaknesses that allow misconfiguration or errant operation in part of

the network to propagate globally or which provide potentials for
simple denial of service attacks should be eliminated to the extent
that it is practical. The integrity of routing information is
critical in assuring that traffic goes where it is supposed to.

An accidental misconfiguration can direct traffic toward routers that
cannot reach a destination for which they are advertising
reachability. This is commonly caused by misconfigured static routes
though there are numerous other potential causes. Static routes are
often used to provide constant apparent reachability to single homed
destinations. Some of the largest ISPs literally have thousands of
static routes in their networks. These are often entered manually by
operators. Mistyping can divert traffic from a completely unrelated
destination to a router with no actual reachability to the advertised
destination. This can happen and does happen somewhat regularly. In
addition, implementation bugs or severe misconfigurations that result
in the loss of BGP AS path information or alteration of prefix length
can result in the advertisement of large sets of routes. Though
considerably more rare, on a few occasions where this has occurred
the results were catastrophic.

Where there is the potential for an accidental misconfiguration in a
remote part of the Internet affecting the global Internet there is
also the potential for malice. For example, it has been demonstrated
by accident that multiple hour outages at a major institution can be
caused by a laptop and a dial account if proper precautions are not
taken. The dial account need not be with the same provider used by
the major institution.

The potential for error is increased by the CIDR preference for more
specific routes [8]. If an institution advertises a single route of
a given length and a distant router advertises a more specific route
covering critical hosts, the more specific route, if accepted at all,
is preferred regardless of administrative weighting or any routing
protocol attributes.

There is a need to provide some form of checks on whether a route
advertisement is valid. Today checks are typically made against the
border AS advertising the route. This prevents accepting routes from
the set of border AS that could not legitimately advertise the route.
Theses checks rely on the use of information registered in the IRR to
generate lists of prefixes that could be advertised by a specific
border AS. Checks can also be made against the origin AS. If policy
information were sufficiently populated, checks could be made against
the entire AS path, but this is not yet feasible.

The use of a routing registry can also make it more difficult for
prefixes to be used without authorization such as unallocated
prefixes or prefixes allocated to another party.

In summary, some of the problems being addressed are:

o Localizing the impact of accidental misconfiguration made by
   Internet Providers to that provider's networks only.

o Eliminating the potential for an Internet provider's customer to
   use malicious misconfiguration of routing as a denial of service
   attack if the provider route filters their customers. Localizing
   the denial of service to that Internet provider only if the
   immediate Internet service provider does not route filter their
   customers but other providers route filter the route exchange at
   the interprovider peering.

o Eliminating the unauthorized use of address space.

If the data within a routing registry is critical, then the ability
to change the data must be controlled. Centralized authorities can
provide control but centralization can lead to scaling problems (and
is politically distasteful).

Address allocation and name allocation is already delegated. Since
delegation can be to outside registries it is at least somewhat
distributed [11]. Autonomous System (AS) numbers are allocated by
the same authorities. It makes sense to delegate the routing number
space in a manner similar to the address allocation and AS number
allocation. The need for this delegation of authority to numerous
registries increases the difficulty of maintaining the integrity of
the body of information as a whole.

As a first step, the database can be somewhat centrally administered
with authority granted to many parties to change the information.
This is the case with the current IRR. There are a very small number
of well trusted repositories and a very large number of parties
authorized to make changes. Control must be exercised over who can
make changes and what changes they can make. The distinction of who
vs what separates authentication from authorization.

o Authentication is the means to determine who is attempting to make
   a change.

o Authorization is the determination of whether a transaction
   passing a specific authentication check is allowed to perform a
   given operation.

Different portions of the database will require different methods of
authentication. Some applications will require authentication based
on strong encryption. In other cases software supporting strong
encryption may not be necessary or may not be legally available. For
this reason multiple authentication methods must be supported,
selected on a per object basis through the specification of
authentication methods in the maintainer object "auth" attribute.
The authentication methods may range from very weak data integrity
checks to cryptographicly strong signatures. The authorization model
must sure that the use of weak integrity checks in parts of the
database does not compromise the overall integrity of the database.

Additional requirements are placed on the authorization model if the
database is widely distributed with delegations made to parties that
may not be trustworthy or whose security practices may be lacking.
This problem must be addressed in the authorization model in order to
enable later evolution to a more distributed routing registry.

Autonomous system numbers can be delegated in blocks and subdelegated
as needed and then individual AS numbers assigned. Address
allocation is a simple numeric hierarchy. Route allocation is
somewhat more complicated. The key attributes in a route object (key
with regard to making it unique) contain both an address prefix and
an AS number, known as the origin AS. The addition of a route object
must be validated against the authorization criteria for both the AS
and the address prefix. Route objects may exist for the same prefix
with multiple origin AS values due to a common multihoming practice
that does not require a unique origin AS. There is often no
correlation between the origin AS of a prefix and the origin AS of
overlapping more specific prefixes.

There are numerous operational cases that must be accommodated. Some
of the more common are listed below. These are explored in greater
detail in Appendix D with discussion of technical tradeoffs in
Appendix C.

o   simple hierarchical address allocation and route allocation

o   aggregation and multihomed more specific routes

o   provider independent addresses and multiple origin AS

o   changing Internet service providers

o   renumbering grace periods

The authorization model must accommodate a variety of policies
regarding the allocation of address space and cannot mandate the use
of any one model. There is no standardization of address allocation
policies though guidelines do exist [11, 16]. Whether authorization
allows the recovery of address space must be selectable on a per
object basis and may differ in parts of the database. This issue is
discussed further in Appendix C.

7 Data Representation

RPSL provides a complete description of the contents of a routing
repository [1]. Many RPSL data objects remain unchanged from the
RIPE specifications and RPSL references the RIPE-181 specification as
recorded in RFC-1786 [2]. RPSL provides external data
representation. Data may be stored differently internal to a routing
registry.

Some database object types or database attributes must be added to
RPSL to record the delegation of authority and to improve the
authentication and authorization mechanisms. These additions are
very few and are described in Section 8 and Section 9.

Some form of encapsulation must be used to exchange data. The
defacto encapsulation has been the one which the RIPE tools accept, a
plain text file or plain text in the body of an RFC-822 formatted
mail message with information needed for authentication derived from
the mail headers or the body of the message. Merit has slightly
modified this using the PGP signed portion of a plain text file or
PGP signed portion of the body of a mail message. These very simple
forms of encapsulation are suitable for the initial submission of a
database transaction.

The encapsulation of registry transaction submissions, registry
queries and registry responses and exchanges between registries is
outside the scope of this document. The encapsulation of registry
transaction submissions and exchanges between registries is outside
the scope of this document.

8 Authentication Model

The maintainer objects serve as a container to hold authentication
filters. A reference to a maintainer within another object defines
authorization to perform operations on the object or on a set of
related objects. The maintainer is typically referenced by name in
mnt-by attributes of objects. Further details on the use of
maintainers are provided in Section 9.1.

The maintainer contains one or more "auth" attributes. Each "auth"
attribute begins with a keyword identifying the authentication method
followed by the authentication information needed to enforce that
method. The PGPKEY method is slightly syntactically different in
that the method PGPKEY is a substring.

Authentication methods currently supported include the following.
Note that pgp-from is being replaced by the pgpkey (see Section 10
and [18]).

mail-from This is a very weak authentication check and is
    discouraged. The authentication information is a regular
    expression over ASCII characters. The maintainer is authenticated
    if the from or reply-to fields in RFC-822 mail headers are matched
    by this regular expression. Since mail forgery is quite easy,
    this is a very weak form of authentication.

crypt-pw This is another weak form of authentication. The
    authentication information is a fixed encrypted password in UNIX
    crypt format. The maintainer is authenticated if the transaction
    contains the clear text password of the maintainer. Since the
    password is in clear text in transactions, it can be captured by
    snooping. Since the encrypted form of the password is exposed, it
    is subject to password guessing attacks.

pgp-from This format is being replaced by the "pgpkey" so that the
    public key certificate will be available to remote repositories.
    This is Merit's PGP extension. The authentication information is
    a signature identity pointing to an external public key ring. The
    maintainer is authenticated if the transaction (currently PGP
    signed portion of a mail message) is signed by the corresponding
    private key.

pgpkey This keyword takes the form "PGPKEY-hhhhhhhh", where
    "hhhhhhhh" is the hex representation of the four byte id of the
    PGP public key used for authentication. The public key
    certificate is stored in a separate object as described in [18].

Repositories may elect to disallow the addition of "auth" attributes
specifying weaker forms of authentication and/or disallow their use
in local transaction submissions. Repositories are encouraged to
disallow the addition of "auth" attributes with the deprecated "pgp-
from" method.

Any digital signature technique can in principle be used for
authentication. Transactions should be signed using multiple digital
signature techniques to allow repositories or mirrors that only use a

subset of the techniques to verify at least one of the signatures.
The selection of digital signature techniques is not within the scope
of this document.

9 Authorization Model

The authorization model must accommodate the requirements outlined in
Section 6. A key feature of the authorization model is the
recognition that authorization for the addition of certain types of
data objects must be derived from related data objects.

With multiple repositories, objects not found in RPSL are needed to
control AS delegations and new attributes are needed in existing
objects to control subdelegation. The definition of RPSL objects
used to implement a distrubuted routing registry system is not within
the scope of this document.

9.1 Maintainer Objects

The maintainer objects serve as a container to hold authentication
filters. The authentication methods are described in Section 8. The
maintainer can be referenced by name in other objects, most notably
in the mnt-by attributes of those objects.

Maintainers themselves contain mnt-by attributes. In some cases the
mnt-by in a maintainer will reference the maintainer itself. In this
case, authorization to modify the maintainer is provided to a
(usually very limited) set of identities. A good practice is to
create a maintainer containing a long list of identities authorized
to make specific types of changes but have the maintainer's mnt-by
attribute reference a far more restrictive maintainer more tightly
controlling changes to the maintainer object itself.

The mnt-by attribute is mandatory in all objects. Some data already
exists without mnt-by attributes. A missing mnt-by attribute is
interpreted as the absence of any control over changes. This is
highly inadvisable and most repositories will no longer allow this.

An additional maintainer reference can occur through a new attribute,
"mnt-routes", and is used in aut-num, inetnum and route objects. The
"mnt-routes" attribute is an extension to RPSL and is described in
detail in Section 10.

A mnt-routes attribute in an aut-num object allows addition of route
objects with that AS number as the origin to the maintainers listed.
A mnt-routes attribute in an inetnum object allows addition of route
objects with exact matching or more specific prefixes. A mnt-routes
attribute in a route object allows addition of route objects with

exact matching or more specific prefixes. A mnt-routes attribute
does not allow changes to the aut-num, inetnum, or route object where
it appears. A mnt-routes may optionally be constrained to only apply
to a subset of more specific routes.

Where "mnt-routes" or "mnt-lower" are applicable, any maintainer
referenced in the "mnt-by" still apply. The set of applicable
maintainers for whatever check is being made is the union of the
"mnt-routes" or "mnt-lower" and the "mnt-by". For example, when
authorizing a route object software would look at "mnt-routes", if it
does not exist, look at "mnt-lower", if that does not exist look at
"mnt-by".

9.2 as-block and aut-num objects

   An "as-block" object is needed to delegate a range of AS numbers to a
   given repository. This is needed for authorization and it is needed
   to avoid having to make an exhaustive search of all repositories to
   find a specific AS. This search would not be an issue now but would
   be if a more distributed routing repository is used. Distributed
   registry issues are not within the scope of this document.

   The "as-block" object also makes it possible to separate AS number
   allocation from registration of AS routing policy.

      as-block: AS1321 - AS1335

   The "aut-num" describes the routing policy for an AS and is critical
   for router configuration of that AS and for analysis performed by
   another AS. For the purpose of this document it is sufficient to
   consider the aut-num solely as a place holder identifying the
   existence of an AS and providing a means to associate authorization
   with that AS when adding "route" objects.

   The "as-block" object is proposed here solely as a means of recording
   the delegation of blocks of AS numbers to alternate registries and in
   doing so providing a means to direct queries and a means to support
   hierarchical authorization across multiple repositories.

9.3 inetnum objects

   The "inetnum" exists to support address allocation. For external
   number registries, such as those using "[r]whoisd[++]" the "inet-num"
   can serve as a secondary record that is added when an address
   allocation is made in the authoritative database. Such records could
   be added by a address registry such as ARIN as a courtesy to the
   corresponding routing registry.

```
      inetnum:           193.0.0.0 - 193.0.0.255
      source:            IANA
```

9.4 route objects

   Currently there are a quite few route objects in more than one
   registry. Quite a few are registered with an origin AS for which
   they have never been announced. There is a legitimate reason to be
   in more than one origin AS.

   The "route" object is used to record routes which may appear in the
   global routing table. Explicit support for aggregation is provided.
   Route objects exist both for the configuration of routing information
   filters used to isolate incidents of erroneous route announcements
   (Section 6) and to support network problem diagnosis.

9.5 reclaim and no-reclaim attributes

   A reclaim attribute is needed in as-block, inetnum and route objects.
   The reclaim attribute allows a control to be retained over more
   specific AS, IP address or route space by allowing modify and delete
   privileges regardless of the mnt-by in the object itself.

   The reclaim attribute provides the means to enforce address lending.
   It allows cleanup in cases where entities cease to exist or as a last
   presort means to correct errors such as parties locking themselves
   out of access to their own objects. To specify all more specific
   objects the reclaim attribute value should be "ALL". To allow finer
   control a set of prefixes can be specified.

   A no-reclaim attribute can be used to provide explicit exceptions. A
   reclaim attribute can only be added to an existing object if the
   addition of the reclaim attribute does not remove autonomy of
   existing more specific objects that are covered by the new reclaim
   attribute.

   1. A reclaim attribute can be added to an existing object if there
      are no existing exact matches or more specific objects overlapped
      by the new reclaim attribute, or

   2. if the submitter is listed in the maintainer pointed to by the
      mnt-by of the objects which are overlapped, or

   3. if any overlapped object is listed in a no-reclaim attribute in
      the object where the reclaim is being added.

   Similarly, a submitter may delete a no-reclaim attribute from an
   object only when that submitter is the only maintainer listed in the
   mnt-by attributes of any overlapped objects. If the submitter is not
   listed in any of the maintainers pointed to by the mnt-by attributes
   for one or more overlapped object, then the submitter is not
   permitted to delete the no-reclaim attribute.

   If neither a reclaim or no-reclaim attribute is present, then more
   specific objects of a given object cannot be modified by the
   maintainer of the less specified object unless the maintainer is also
   listed as a maintainer in the more specific object. However, the
   addition of a new route or inetnum object must pass authentication of
   the largest less specific prefix as part of the authentication check
   described in Section 9.9.

   See Section 10 for a full description of the reclaim and no-reclaim
   attributes.

9.6 Other Objects

   Many of the RPSL ancillary objects have no natural hierarchy the way
   AS numbers, Internet addresses and routes do have a numeric
   hierarchy. Some examples are "maintainers", "people" and "role"
   objects. For these objects, lack of any hierarchy leads to two
   problems.

   1. There is no hierarchy that can be exploited to direct queries to
      alternate registries. At some point the query strategy of
      searching all known registries becomes impractical.

   2. There is no hierarchy on which authorizations of additions can be
      based.

   The first problem can be addressed by considering the name space for
   each of the ancillary objects to be unique only within the local
   database and to use explicit references to an external repository
   where needed. To specify an external repository reference, the
   object key is preceded by the name of the repository and the
   delimiter "::". For example a NIC handle may take the form
   "RIPE::CO19". Currently there is a desire to keep NIC handles unique
   so the naming convention of appending a dash and the repository name
   is used. Prepending the repository name provides the unique name
   space since an object in the RIPE database referencing "CO19" would
   be interpreted as "RIPE::CO19" by default, but it would still be
   possible to query or reference "IANA::CO19". There is no possibility
   of accidentally forgetting to adhere to the conventions when making
   an addition and the existing objects are accommodated, including
   cases where name conflicts have already occurred.

The second problem can be partially addressed by using a referral
system for the addition of maintainers and requiring that any other
object be submitted by a registered maintainer or by IANA. The
referral system would allow any existing maintainer to add another
maintainer. This can be used in parallel with the addition of other
object types to support the maintenance of those objects. For
example, when adding a subdomain to the "domain" hierarchy (in the
RIPE repository where domains are also handled), even when adding a
new domain to a relatively flat domain such as "com", there is
already a maintainer for the existing domain. The existing
maintainer can add the maintainer that will be needed for the new
domain in addition to adding the new domain and giving the new
maintainer the right to modify it.

An organization gaining a presence on the Internet for the first time
would be given a maintainer. This maintainer may list a small number
of very trusted employees that are authorized to modify the
maintainer itself. The organization itself can then add another
maintainer listing a larger set of employees but listing the more
restrictive maintainer in the mnt-by attributes of the maintainers
themselves. The organization can then add people and role objects as
needed and any other objects as needed and as authorization permits.

9.7 Objects with AS Hierarchical Names

Many RPSL objects do not have a natural hierarchy of their own but
allow hierarchical names. Some examples are the object types "as-
set" and "route-set". An as-set may have a name corresponding to no
naming hierarchy such as "AS-Foo" or it may have a hierarchical name
of the form "AS1:AS-Bar".

When a hierarchical name is not used, authorization for objects such
as "as-set" and "route-set" correspond to the rules for objects with
no hierarchy described in Section 9.6.

If hierarchical names are used, then the addition of an object must
be authorized by the aut-num whose key is named by everything to the
left of the rightmost colon in the name of the object being added.
Authorization is determined by first using the mnt-lower maintainer
reference, or if absent, using the mnt-by reference.

9.8 Query Processing

A query may have to span multiple repositories. All queries should
be directed toward a local repository which may mirror the root
repository and others. Currently each IRR repository mirrors all
other repositories. In this way, the query may be answered by the
local repository but draw data from others.

The mechanism below when applied to multiple repositories assumes the
existence of an attribute for traversal of the repositories. The
definition of this attribute is considered a distributed registry
issue and is out of scope of this document.

For object types that have a natural hierarchy, such as aut-num,
inet-num, and route, the search begins at the root database and
follows the hierarchy. For objects types that have no natural
hierarchy, such as maintainer, person, and role objects, the search
is confined to a default database unless a database is specified.
The default database is the same database as an object from which a
reference is made if the query is launched through the need to follow
a reference. Otherwise the default is generally the local database or
a default set by the repository. The default can be specified in the
query itself as described in Section 9.7.

In the absense of attributes to traverse multiple registries a search
of all repositories is needed. With such attributes the search would
proceed as follows. In searching for an AS, the delegation attribute
in AS blocks can be consulted, moving the search to data from other
repositories. Eventually the AS is either found or the search fails.
The search for an inetnum is similar. Less specific inetnums may
refer the search to other databases. Eventually the most specific
inetnum is found and its status (assigned or not assigned) can be
determined. The definition of attributes for traversal of
repositories is considered a distrbiuted registry issue and is not
within the scope of this document.

The search for a route in the presence of attributes for the
traversal of multiple registries is similar except the search may
branch to more than one repository. The most specific route in one
repository may be more specific than the most specific in another.
In looking for a route object it makes sense to return the most
specific route that is not more specific than the query requests
regardless of which repository that route is in rather than return
one route from each repository that contains a less specific overlap.

9.9 Adding to the Database

The mechanism below when applied to multiple repositories assumes the
existence of an attribute for traversal of the repositories. The
definition of this attribute is considered a distributed registry
issue and is out of scope of this document.

The root repository must be initially populated at some epoch with a
few entries. An initial maintainer is needed to add more
maintainers. The referral-by attribute can be set to refer to itself
in this special case (Section 10 describes the referral-by). When

adding an inetnum or a route object an existing exact match or a less
specific overlap must exist. A route object may be added based on an
exact match or a less specific inetnum. The root repository must be
initially populated with the allocation of an inetnum covering the
prefix 0/0, indicating that some address allocation authority exists.
Similarly an initial as-block is needed covering the full AS number
range.

When adding an object with no natural hierarchy, the search for an
existing object follows the procedure outlined in Section 9.8.

When adding an aut-num (an AS), the same procedure used in a query is
used to determine the appropriate repository for the addition and to
determine which maintainer applies. The sequence of AS-block objects
and repository delegations is followed. If the aut-num does not
exist, then the submission must match the authentication specified in
the maintainer for the most specific AS-block in order to be added.

The procedure for adding an inetnum is similar. The sequence of
inet-num blocks is followed until the most specific is found. The
submission must match the authentication specified in the maintainer
for the most specific inetnum overlapping the addition.

Adding a route object is somewhat more complicated. The route object
submission must satisfy two authentication criteria. It must match
the authentication specified in the aut-num and the authentication
specified in either a route object or if no applicable route object
is found, then an inetnum.

An addition is submitted with an AS number and prefix as its key. If
the object already exists, then the submission is treated as a modify
(see Section 9.10). If the aut-num does not exist on a route add,
then the addition is rejected (see Section C for further discussion
of tradeoffs). If the aut-num exists then the submission is checked
against the applicable maintainer. A search is then done for the
prefix first looking for an exact match. If the search for an exact
match fails, a search is made for the longest prefix match that is
less specific than the prefix specified. If this search succeeds it
will return one or more route objects. The submission must match an
applicable maintainer in at least one of these route objects for the
addition to succeed. If the search for a route object fails, then a
search is performed for an inetnum that exactly matches the prefix or
for the most specific inetnum that is less specific than the route
object submission. The search for an inetnum should never fail but
it may return an unallocated or reserved range. The inetnum status
must be "allocated" and the submission must match the maintainer.

Having found the AS and either a route object or inetnum, the
authorization is taken from these two objects. The applicable
maintainer object is any referenced by the mnt-routes attributes. If
one or more mnt-routes attributes are present in an object, the mnt-
by attributes are not considered. In the absence of a mnt-routes
attribute in a given object, the mnt-by attributes are used for that
object. The authentication must match one of the authorizations in
each of the two objects.

If the addition of a route object or inetnum contains a reclaim
attribute, then any more specific objects of the same type must be
examined. The reclaim attribute can only be added if there are no
more specific overlaps or if the authentication on the addition is
present in the authorization of a less specific object that already
has a reclaim attribute covering the prefix range, or if the
authentication on the addition is authorized for the modification of
all existing more specific prefixes covered by the addition.

9.10 Modifying or Deleting Database Objects

When modifying or deleting any existing object a search for the
object is performed as described in Section 9.8. If the submission
matches an applicable maintainer for the object, then the operation
can proceed. An applicable maintainer for a modification is any
maintainer referenced by the mnt-by attribute in the object. For
route and inet-num objects an applicable maintainer may be listed in
a less specific object with a reclaim attribute.

If the submission is for a route object, a search is done for all
less specific route objects and inetnums. If the submission is for
an inetnum, a search is done for all less specific inetnums. If the
submission fails the authorization in the object itself but matches
the reclaim attribute in any of the less specific objects, then the
operation can proceed. Section C contains discussion of the
rationale behind the use of the reclaim attribute.

A modification to an inetnum object that adds a reclaim attribute or
removes a no-reclaim attribute must be checked against all existing
inetnums that are more specific. The same check of the reclaim
attribute that is made during addition must be made when a reclaim
attribute is added by a modification (see Section 9.9).

A deletion is considered a special case of the modify operation. The
deleted object may remain in the database with a "deleted" attribute
in which case the mnt-by can still be consulted to remove the
"deleted" attribute.

10 Data Format Summaries

   RIPE-181 [2] and RPSL [1] data is represented externally as ASCII
   text. Objects consist of a set of attributes. Attributes are name
   value pairs. A single attribute is represented as a single line with
   the name followed by a colon followed by whitespace characters
   (space, tab, or line continuation) and followed by the value. Within
   a value all whitespace is equivalent to a single space. Line
   continuation is supported by a backslash at the end of a line or the
   following line beginning with whitespace. When transferred,
   externally attributes are generally broken into shorter lines using
   line continuation though this is not a requirement. An object is
   externally represented as a series of attributes. Objects are
   separated by blank lines.

   There are about 80 attribute types in the current RIPE schema and
   about 15 object types. Some of the attributes are mandatory in
   certain objects. Some attributes may appear multiple times. One or
   more attributes may form a key. Some attributes or sets of
   attributes may be required to be unique across all repositories.
   Some of the attributes may reference a key field in an object type
   and may be required to be a valid reference. Some attributes may be
   used in inverse lookups.

   A review of the entire RIPE or RPSL schema would be too lengthy to
   include here. Only the differences in the schema are described.

10.1 Changes to the RIPE/RPSL Schema

   One new object type and several attributes are added to the RIPE/RPSL
   schema. There are significant changes to the rules which determine
   if the addition of an object is authorized.

   The new object type is listed below. The first attribute listed is
   the key attribute and also serves as the name of the object type.

       as-block   key mandatory single   unique
       descr   optional   multiple
       remarks   optional   multiple
       admin-c   mandatory multiple
       tech-c   mandatory multiple
       notify   optional   multiple
       mnt-by   mandatory multiple
       changed   mandatory multiple
       source   mandatory single

In the above object type only the key attribute "as-block" is new:

as-block This attribute provides the AS number range for an "as-
   block" object. The format is two AS numbers including the sub-
   string "AS" separated by a "-" delimiter and optional whitespace
   before and after the delimiter.

In order to support stronger authentication, the following keywords
are added to the "auth" attribute:

pgp-from The remainder of the attribute gives the string identifying
   a PGP identity whose public key is held in an external keyring.
   The use of this method is deprecated in favor of the "pgpkey"
   method.

pgpkey See [18].

In order to disable authentication and give permission to anyone, the
authentication method "none" is added. It has no arguments.

An additional change is the "auth" attribute is allowed to exist in a
"person" or "role" object. The "auth" method "role" or "person" can
be used to refer to a role or person object and take the "auth"
fields from those objects. Care must be taken in implementations to
detect circular references and terminate expansion or the references
already visited.

A few attributes are added to the schema. These are:

mnt-routes The mnt-routes attribute may appear in an aut-num, inet-
   num, or route object. This attribute references a maintainer
   object which is used in determining authorization for the addition
   of route objects. After the reference to the maintainer, an
   optional list of prefix ranges (as defined in RPSL) inside of
   curly braces or the keyword "ANY" may follow. The default, when
   no additional set items are specified is "ANY" or all more
   specifics. The mnt-routes attribute is optional and multiple.
   See usage details in Section 9.1.

mnt-lower The mnt-lower attribute may appear in an inetnum, route,
   as-block or aut-num object. This attribute references a
   maintainer object. When used in an inetnum or route object the
   effect is the same as a "mnt-routes" but applies only to prefixes
   more specific than the prefix of the object in which it is
   contained. In an as block object, mnt-lower allows addition of
   more specific as-block objects or aut-num objects. In an aut-num

        object the mnt-lower attribute specifies a maintainer that can be
        used to add objects with hierarchical names as described in
        Section 9.7.

    reclaim The reclaim attribute may appear in as-block, aut-num,
        inet-num, or route objects. Any object of the same type below in
        the hierarchy may be modified or deleted by the maintainer of the
        object containing a reclaim attribute. The value of the attribute
        is a set or range of objects of the same type where the syntax of
        the set or range is as defined in RPSL. See Section 9.5 for
        restrictions on adding reclaim attributes.

    no-reclaim The no-reclaim attribute is used with the reclaim
        attribute. The no-reclaim attribute negates any reclaim attribute
        it overlaps. See Section 9.5 for restrictions on deleting no-
        reclaim attributes.

    referral-by This attribute is required in the maintainer object. It
        may never be altered after the addition of the maintainer. This
        attribute refers to the maintainer that created this maintainer.
        It may be multiple if more than one signature appeared on the
        transaction creating the object.

    auth-override An auth-override attribute can be added, deleted, or
        changed by a transaction submitted by maintainer listed in the
        referral-by. An auth-override can only be added to a maintainer
        if that maintainer has been inactive for the prior 60 days. The
        auth-override attribute itself contains only the date when the
        attribute will go into effect which must be at least 60 days from
        the current date unless there is already authorization to modify
        the maintainer. After the date in the auth-override is reached,
        those identified by the maintainer in the referral-by have
        authorization to modify the maintainer. This attribute exists as
        a means to clean up should the holder of a maintainer become
        unresponsive and can only take effect if that maintainer does not
        remove the auth-override in response to the automatic notification
        that occurs on changes.

    The existing "mnt-by" attribute references the "maintainer" object
    type. The "mnt-by" attribute is now mandatory in all object types.
    A new maintainer may be added by any existing maintainer. The
    "referral-by" attribute is now mandatory in the "maintainer" object
    to keep a record of which maintainer made the addition and can never
    be changed. Maintainers cannot be deleted as long as they are
    referenced by a "referral-by" attribute elsewhere.

A Core and Non-Core Functionality

   Most of the objects and attributes described in this document are
   essential to the authorization framework. These are referred to as
   being part of the "core" functionality. A few attributes listed here
   are considered "non-core".

   The "reclaim" and "no-reclaim" attributes are a convenience to
   support flexibility in the implementation of address lending.

   The "auth-override" attribute is a convenience to facilitate recovery
   in an environment where repository data is redistributed in any way.

   The "referal-by" attribute is a "core" feature. An individual
   registry may express its sutonomy by creating a self-referencing
   maintainer, one whose "referal-by" points to itslef. Other
   registries can decide on a case by case basis whether to consider
   such an entry valid. A registry may only allow the "referal-by" to
   refer to a specific maintainer under the control of the registry.
   This further restriction is an issue that is purely local to the
   registry.

B Examples

   The examples below leave out some required attributes that are not
   needed to illustrate the use of the objects and attributes described
   in this document. Missing are admin-c, tech-c, changed, source.
   Also missing are attributes such as mnt-nfy, whose use are a good
   practice but are not strictly required.

   To do anything at all a maintainer is needed. At some epoch a a
   single maintainer is populated in one repository and that maintianer
   has a referal-by pointing to itself. All others referal-by
   references can be traced back to that maintainer. At the epoch the
   as-block AS0- AS65535 and the inetnum 0.0.0.0-255.255.255.255 are
   also allocated. Other ancilliary object may also be needed to
   bootstrap.

```
    mntner:        ROOT-MAINTAINER
    auth:          pgpkey-12345678

    mnt-by:        ROOT-MAINTAINER
    referal-by:    ROOT-MAINTAINER
```

   This root maintainer might add a top level maintainer for some
   organization.

```
mntner:         WIZARDS
descr:          High level Technical Folks
auth:           pgpkey-23456789
auth:           pgpkey-3456789a
mnt-by:         WIZARDS
referal-by:     ROOT-MAINTAINER
```

That maintainer might add another who have more limited capabilities.

```
mntner:         MORTALS
descr:          Maintain day to day operations
auth:           pgpkey-456789ab
auth:           pgpkey-56789abc
auth:           pgpkey-6789abcd
mnt-by:         WIZARDS
referal-by:     WIZARDS
```

Note that the WIZARDS can change their own maintainer object and the
MORTALS maintainer object but MORTALS cannot.

At some point an as-block is allocated and broken down. In the
example below, private number space is used.

```
as-block:       AS65500-AS65510
mnt-by:         SOME-REGISTRY
mnt-lower:      WIZARDS
```

Note that a registry has control over the object that they have
created representing the allocation, but have given the party to
which the allocation was made the ability to create more specific
objects. Below this as-block, an aut-num is added. Note that
import and export are normally required for a aut-num but are not
shown here.

```
aut-num:        AS65501
mnt-by:         WIZARDS
mnt-lower:      MORTALS
```

In aut-num above the WIZARDS maintainer can modify the aut-num
itself. The MORTALS maintainer can add route objects using this AS
as the origin if they also have authorization for the IP number space
in a less specific route or inetnum.

We also need an inetnum allocation. In this example the inetnum is
allocated to a completely different organization. Again attributes
are omitted which would normally be needed in an inetnum.

```
inetnum:        192.168.144.0-192.168.151.255
mnt-by:         SOME-REGISTRY
mnt-lower:      ISP
reclaim:        ALL
```

The maintainer ISP can add more specific inetnums or routes with this
address space. Note that the registry has declared their ability to
reclaim the address space.

If ISP wished to reclaim all allocations but some suballocation of
theirs resisted, we might get something like the following in which
they will reclaim only the top half of an allocation (possibly if it
remains unused).

```
inetnum:        192.168.144.0-192.168.147.255
mnt-by:         ISP
mnt-lower:      EBG-COM
reclaim:        192.168.146/23+
```

If we assume that the maintainer EBG-COM and the maintainer MORTALS
want to add a route object, one way to do it is for both parties to
sign. If EBG-COM for some reason couldn't aggregate an allocate a
single top level route (which is inexcusable these days) or there was
a preference for some reason to avoid the joint signature approach on
a submission either party could give the other permission to make the
addition. A mnt-routes could be added to the aut-num or a mnt-lower
could be added to an inetnum.

```
aut-num:        AS65501
mnt-by:         WIZARDS
mnt-lower:      MORTALS
mnt-routes:     EBG-COM {192.168.144/23}
```

With this change to the aut-num the maintainer EBG-COM could add a
route with origin AS65501, but only with a limited address range.

```
route:          192.168.144/24
origin:         AS65501
descr:          These boneheads don't aggregate
mnt-by:         EBG-COM
mnt-by:         FICTION::MORTALS
```

Note that while the maintainer EBG-COM added the object they allowed
the maintainer MORTALS the ability to modify it.

If an object ended up in another repository, a single maintainer
could still be used. In the example above the notation
FICTION::MORTALS indicates that the route object is in a different
repository and rather than duplicate the maintainer, a reference is
made to the repository in which the MORTALS object resides.

In the example below, a pair of route-sets are added and hierarchical
names are used.

```
route-set:      AS65501:Customers
mnt-by:         WIZARDS
mnt-lower:      MORTALS

route-set:      AS65501:Customers:EBG-COM
mnt-by:         MORTALS
mnt-lower:      EBG-COM
```

Suppose in the 192.168.144/24 object above, only the EBG-COM
maintainer is listed. If EBG-COM goes bankrupt, no longer needs
address space, and stops responding, it could be difficult to delete
this object. The maintainer listed in the EBG-COM referral-by
attribute could be contacted. They could add a auth-override
attribute to the EBG-COM object. Later they could modify the EBG-COM
object and then any objects with EBG-COM in the mnt-by.

```
mntner:         EBG-COM
mnt-by:         EBG-COM
auth-override:  19990401
```

The examples above stray significantly from realism. They do provide
simple illustrations of the usage of the objects type and attributes
described in this document and hopefully in doing some are of some
value.

C Technical Discussion

A few design tradeoffs exist. Some of these tradeoffs, the selected
solution, and the alternatives are discussed here. Some of the
issues are listed below.

1. Whether to err on the side of permissiveness and weaken
   authorization controls or risk the possibility of erecting
   barriers to registering information.

2. Whether to support enforcible address lending or provide the
   smaller or end user with ultimate control over the registration of
   the prefixes they are using.

   3. What to do with older objects that either don't conform to newer
      requirements regarding minimum authorization, authentication, and
      accountability, or are of questionable validity.

C.1 Relaxing requirements for ease of registry

   If the requirement that an aut-num exists is relaxed, then it is
   possible for anyone to make use of an unassigned AS number or make
   use of an assigned AS number for which the aut-num has not been
   entered. Placing requirements on the entry of aut-num presumes
   cooperation of the Internet address allocation authority (if separate
   from the routing registry). The address allocation authority must be
   willing to field requests to populate skeleton aut-nums from the
   party for which the allocation has been made. These aut-num must
   include a reference to a maintainer. A request to the address
   allocation authority must therefore include a reference to an
   existing maintainer.

   The ability to add route objects is also tied to the existence of
   less specific route objects or inetnums. The Internet address
   allocation authority (if separate from the routing registry) must
   also be willing to field requests to add inetnum records for the
   party already allocated the address space.

   The Internet address allocation authority should also add inetnums
   and aut-nums for new allocations. In order to do so, a maintainer
   must exist. If a party is going to connect to the Internet, they can
   get a maintainer by making a request to the Internet service provider
   they will be connecting to. Once they have a maintainer they can
   make a request for address space or an AS number. The maintainer can
   contain a public key for a cryptographicly strong authorization
   method or could contain a "crypt-key" or "mail-to" authorization
   check if that is considered adequate by the registering party.
   Furthermore an address allocation authority should verify that the
   request for an AS number or for address space matches the
   authorization criteria in the maintainer.

   Currently only the registries themselves may add maintainers. This
   becomes a problem for the registry, particularly in verifying public
   keys. This requirement is relaxed by allowing existing maintainers
   to add maintainers. Unfortunately the accountability trail does not
   exist for existing maintainers. The requirement then should be
   relaxed such that existing maintainers may remain but only existing
   maintainers that have a "referral-by" attribute can add maintainers.
   The "referral-by" cannot be modified. This requirement can be
   relaxed slightly so that a "referral-by" can be added to a maintainer

by an existing maintainer with a "referral-by". This will allow the
accountability trail to be added to existing maintainers and these
maintainers can then add new maintainers.

Verifying that a party is who they claim to be on initial addition,
is one of the problems that currently falls upon the AS number and
address registry. This problem is reduced by allowing existing
maintainers to add maintainers. This may actually make it easier to
get maintainers and therefore easier to register. The number
authority still must verify that the AS or address space is actually
needed by the party making a request.

Authorization checks made during the addition of route objects that
refer to AS objects and inetnums strongly rely on the cooperation of
the Internet address allocation authorities. The number authorities
must register as-blocks, aut-nums, or inetnums as AS numbers or
address space is allocated. If only a subset of the number
authorities cooperate, then either an inetnum or as-block can be
created covering the space that registry allocates and essentially
requiring null allocation (for example a "crypt-pw" authentication
where the password is given in the remarks in the object or its
maintainer) or those obtaining addresses from that number authority
will have trouble registering in the routing registry. The
authorization model supports either option, though it would be
preferable if the number authorities cooperated and the issue never
surfaced in practice.

The maintainer requirements can be relaxed slightly for existing
maintainers making it easier to register. Relaxing requirements on
other objects may defeat the authorization model, hence is not an
option.

C.2 The address lending issue

The issue of whether lending contracts should be enforcible is an
issue of who should ultimately be able to exercise control over
allocations of address space. The routing registry would be wise to
stay as neutral as possible with regard to disputes between third
parties. The "reclaim" and "no-reclaim" are designed to allow either
outcome to the decision as to whether the holder of a less specific
inetnum or route object can exercise control over suballocations in
the registry. The routing registry itself must decide whether to
retain control themselves and if so, should very clearly state under
what conditions the registry would intervene. A registry could even
go to the extreme of stating that they will intervene in such a
dispute only after the dispute has been resolved in court and a court
order has been issued.

When an allocation is made by a registry, the registry should keep a
"reclaim" attribute in the less specific object and make a strong
policy statement that the reclaim privilege will not be used except
under very clearly defined special circumstances (which at the very
minimum would include a court order). If the allocation is further
subdivided the party subdividing the allocation and the party
accepting the suballocation must decide whether a "reclaim" can be
kept by the holder of the less specific allocation or whether a "no-
reclaim" must be added transferring control to the holder of the more
specific. The registry is not involved in that decision. Different
pairs of third parties may reach different decisions regarding the
"reclaim" and any contractual restrictions on its use that may be
expressed outside of the registry in the form of a legal contract and
ultimately resolved by the courts in the event of a bitter dispute.

By retaining "reclaim" rights the registry retains the ability to
abide by a court order. This may only truly become an issue in a
distributed registry environment where registries will be rechecking
the authorization of transactions made elsewhere and may fail to
process the attempt of another registry to abide by a court order by
overriding normal authorization to change the registry contents if a
reclaim is not present.

C.3 Dealing with non-conformant or questionable older data

Some of the newer requirements include requiring that all objects
reference a maintainer object responsible for the integrity of the
object and requiring accountability for the creation of maintainers
to be recorded in the maintainer objects so that accountability can
be traced back from an unresponsive maintainer. In the event that
contact information is absent or incorrect from objects and there is
any question regarding the validity of the objects, the maintainer
can be contacted. If the maintainer is unresponsive, the maintainer
that authorized the addition of that maintainer can be contacted to
either update the contact information on the maintainer or confirm
that the entity no longer exists or is no longer actively using the
Internet or the registry.

Many route objects exist for which there are no maintainers and for
which inetnum and AS objects do not exist. Some contain the now
obsoleted guardian attribute rather than a mnt-by.

It is not practical to unconditionally purge old data that does not
have maintainers or does not conform to the authorization hierarchy.
New additions must be required to conform to the new requirements
(otherwise the requirements are meaningless). New requirements can
be phased in by requiring modifications to conform to the new
requirements.

A great deal of questionable data exists in the current registry.
The requirement that all objects have maintainers and the
requirements for improved accountability in the maintainers
themselves may make it easier to determine contact information even
where the objects are not updated to reflect contact information
changes.

It is not unreasonable to require valid contact information on
existing data. A great deal of data appears to be unused, such as
route objects for which no announcement has been seen in many months
or years. An attempt should be made to contact the listed contacts
in the object, in the maintainer if there is one, then up the
maintainer referral-by chain if there is one, and using the number
registry or origin AS contact information if there is no maintainer
accountability trail to follow. Experience so far indicates that the
vast majority of deletions identified by comparing registered
prefixes against route dumps will be positively confirmed (allowing
the deletion) or there will be no response due to invalid contact
information (in many cases the IRR contact information points to
nsfnet-admin@merit.edu).

By allowing the registry to modify (or delete) any objects which are
disconnected from the maintainer accountability trail, cleanup can be
made possible (though mail header forging could in many cases have
the same effect it is preferable to record the fact that the registry
itself made the cleanup). Similarly, a mechanism may be needed in
the future to allow the maintainer in the referral-by to override
maintainer privileges in a referred maintainer if all contacts have
become unresponsive for a maintainer. The referral-by maintainer is
allowed to add an "auth-override" attribute which becomes usable as
an "auth" within 60 days from the time of addition. The maintainer
themselves would be notified of the change and could remove the
"auth-override" attribute before it becomes effective and inquire as
to why it was added and correct whatever problem existed. This can
be supported immediately or added later if needed.

D Common Operational Cases

In principle, address allocation and route allocation should be
hierarchical with the hierarchy corresponding to the physical
topology. In practice, this is often not the case for numerous
reasons. The primary reasons are the topology is not strictly tree
structured and the topology can change. More specifically:

1. The Internet topology is not strictly tree structured.

   o At the top level the network more closely resembles a
      moderately dense mesh.

   o Near the bottom level many attachments to the Internet are
      multi-homed to more than one Internet provider.

2. The Internet topology can and does change.

   o Many attachments switch providers to obtain better service or
      terms.

   o Service providers may modify adjacencies to obtain better
      transit service or terms.

   o Service providers may disappear completely scattering
      attachments or they may merge.

Renumbering is viewed as a practical means to maintain a strict
numeric hierarchy [16]. It is also acknowledged that renumbering
IPv4 networks can be difficult [16, 3, 17]. We examine first the
simple case where hierarchy still exists. We then examine the
operational cases where either initial topology is not tree
structured or cases where topology changes.

D.1 simple hierarchical address allocation and route allocation

This is the simplest case. Large ranges of inetnums are assigned to
address registries. These registries in turn assign smaller ranges
for direct use or to topologically large entities where allocations
according to topology can reduce the amount of routing information
needed (promote better route aggregation).

AS objects are allocated as topology dictates the need for additional
AS [10]. Route objects can be registered by those with authorization
given by the AS and by the address owner. This is never an issue
where the maintainer of the AS and the inetnum are the same. Where
they differ, either the provider can give permission to add route
objects for their AS, or the party allocated the address space can
give the provider permission to add route objects for their address
space, or both parties can sign the transaction. Permission is
provided by adding to maintainer attributes.

D.2 aggregation and multihomed more specific routes

   Aggregation is normally not a problem if a provider is aggregating
   address space allocated to the provider and then suballocated
   internally and/or to customers. In fact, the provider would be
   expected to do so. This is not a problem even if the route object
   for the aggregation is added after the more specific route objects
   since only less specific objects are considered.

   Aggregation is potentially a problem if a provider or a set of
   providers plan to aggregate address space that was never explicitly
   allocated as a block to those providers but rather remains the
   allocation of a address registry. These large aggregations can be
   expected to be uncommon, but relatively easily dealt with.
   Superaggregates of this type will generally be formed by
   topologically close entities who have also managed to draw adjacent
   address allocations. In effect, the registry must give permission to
   form such a superaggregate by either giving permission to do so in
   the mnt-routes of an inetnum or by signing the submission along with
   the other parties.

D.3 provider independent addresses and multiple origin AS

   Provider independent addresses and multihoming arrangement using
   multiple origin AS present a similar problem to multihoming. The
   maintainer of the address space and the maintainer of the AS is not
   the same. Permission can be granted using mnt-routes or multiple
   signatures can appear on the submission.

D.4 change in Internet service provider

   A change in Internet service providers is similar to multihoming. A
   minor difference is that the AS for the more specific route will be
   the AS of the new provider rather than the AS of the multihomed
   customer. Permission can be granted using mnt-routes or multiple
   signatures can appear on the submission.

D.5 renumbering grace periods

   Renumbering grace periods allow a provider who wants to keep an
   address allocation intact to allow a customer who has chosen to go to
   another provider to renumber their network gradually and then return
   the address space after renumbering is completed. The issue of
   whether to require immediate renumbering or offer renumbering grace
   periods and how long they should be or whether they should be
   indefinite has been topic of bitter disputes. The authorization
   model can support no renumbering grace period, a finite renumbering

grace period, or an indefinite renumbering grace period. The
"reclaim" attribute described in Section 9.1 provides a means to end
the grace period.

E Deployment Considerations

   This section describes deployment considerations. The intention is
   to raise issues and discuss approaches rather than to provide a
   deployment plan.

   The use of routing registries is not yet universally accepted. There
   still remain Internet providers who see no reason to provide the
   added assurance of accurate routing information described in Section
   6. More accurately, these benefits are viewed as being insufficient
   to justify the cost. This has been largely caused an inability of a
   very major router vendor up until recently to handle prefix lists of
   the size needed to specify routing policy on a per prefix basis.

   Another reason cited is that filtering on a prefix basis in an
   environment where routing registry information is incomplete or
   inaccurate can interfere with connectivity.

   There clearly is a critical mass issue with regard to the use of
   routing registries. A minority of providers use the existing IRR to
   filter on a per prefix basis. Another minority of providers do not
   support the IRR and generally fail to register prefixes until
   connectivity problems are reported. The majority of providers
   register prefixes but do not implement strict prefix filtering.

   Deploying new authentication mechanisms has no adverse consequences.
   This has been proven with Merit's deployment of PGP.

   In deploying new authorization mechanisms, a major issue is dealing
   with existing data of very questionable origin. A very large number
   of route objects refer to prefixes that have not been announced for
   many years. Other route objects refer to prefixes that are no longer
   announced with the origin AS that they are registered with (some were
   incorrectly registered to start with). There are many causes for
   this.

   1. During the transition from the NSFNET PRDB to the RADB a large
      number of prefixes were registered with an origin AS corresponding
      to the border AS at which the NSFNET had once heard the route
      announcements. The PRDB did not support origin AS, so border AS
      was used. Many of these routes were no longer in use at the time
      and are now routed with a submitter listed as "nsfnet-
      admin@merit.edu".

   2. As CIDR was deployed, aggregates replaced previously separately
      announced more specific prefixes. The route objects for the more
      specific prefixes were never withdrawn from the routing
      registries.

   3. Some prefixes are simply no longer in use. Some networks have
      been renumbered. Some network no longer exist. Often the routing
      registry information is not withdrawn.

   4. As provider AS adjacencies changed and as end customers switched
      providers often the actual origin AS changed. This was often not
      reflected by a change in the routing registry.

Inaccuracies will continue to occur due to the reasons above, except
the first. The hierarchical authorization provides greater
accountability. In the event that the contacts for specific objects
become unresponsive traversal up the authorization hierarchy should
help identify the parties having previous provided authorization.
These contacts may still have sufficient authorization to perform the
necessary cleanup. This issue is discussed in Section C.

A great deal of information is currently missing in the IRR. Quite a
few AS have no aut-num. Quite a lot of data has no maintainer and
the vast majority of maintainers use only the weakest of
authentication methods. Very little can be done by the registries to
correct this. The defaults in the cases of missing objects needed
for authorization has to be to make no authentication checks at all.

The transition can be staged as follows:

1. Add and make use of stronger authorization models.

2. Make schema modifications necessary to support delegations.

3. Add delegation attributes needed for query traversal.
4. Base query traversal on delegations rather than a search of all
   known registries.

5. Obtain the cooperation of the address registries for the purpose
   of populating the "inetnum" entries on an ongoing basis.

6. Add hierarchical authorization support for critical object types,
   "aut-num", "inetnum" and "route".

7. Add the requirement that database object either be in use or have
   valid contact information and if queries are made by the registry
   a response from a contact indicating that the object serves a
   purpose if it is not clear what its use is.

   8. Begin to purge data which is clearly not in use and for which
      there is no valid contact information or no response from the
      contacts.

   Deployment of hierarchical authorization requires cooperation among
   the existing routing registries. New code will have to be deployed.
   In some cases minimal development resources are available and
   substantial inertia exists due to the reliance on the current
   repository and the need to avoid disruption.

   If hierarchical authorization of route objects depends on the
   existence of address registration information, minimal cooperation of
   the currently separate address registries is required. The extent of
   the cooperation amounts to sending cryptographically signed
   transactions from the address registry to the number registry as
   address allocations are made or providing equivalent access to new
   address allocations.

   Currently most registries return query results from all of the known
   repositories using their mirrored copies. Cross registry
   authorizations are not yet implemented. Minimal schema changes have
   to be made to support the ability to delegate objects for which there
   is an authorization hierarchy and to support queries and references
   to other repositories. In the case of AS delegations, "as-block"
   need to be created solely for the purpose of traversal.

F Route Object Authorization Pseudocode

   The following list provides a brief review of basic concepts.

   1. The route object submission must satisfy two authentication
      criteria. It must match the authentication specified in the aut-
      num and the authentication specified in either a route object or
      if no applicable route object is found, then an inetnum.

   2. When checking for prefix authorization, an exact route object
      prefix match is checked for first. If there is not an exact match
      then a longest prefix match that is less specific than the prefix
      is searched for. If the route prefix search fails, then a search
      is performed for an inetnum that exactly matches the prefix or for
      the most specific inetnum that is less specific than the route
      object submission.

      The search for an inetnum should never fail but it may return an
      unallocated or reserved range. The inetnum status must be
      "allocated" and the submission must pass it's maintainer

      authorization in order to get authorization from an inetnum. So
      an unallocated or reserved range inetnum will cause the route
      object submission to fail.

   3. A route object must pass authorization from both the referenced
      aut-num object and the route or inetnum object. Authorization
      shall be tested using the maintainer(s) referenced in the "mnt-
      routes" attribute(s) first. If that check fails, the "mnt-lower"
      attributes are checked. If that check fails the "mnt-by"
      attributes are used for the authorization check.

   4. The "reclaim" attribute can appear in inetnum, route and as-block
      objects and provides a means to support address lending. "reclaim"
      gives authorization over more specific objects, regardless of the
      "mnt-by" in the object. The value of a "reclaim" attribute can be
      a list or set of objects to provide finer grain control.

      The "reclaim" attribute is important to this discussion since it
      affects prefix/origin authentication when a new route object is
      submitted.

      The "no-reclaim" attribute is used to provide explicit exceptions.

   The following pseudocode outlines the algorithm used to check for
   proper authorization of a route object submission.

    Case #1. Route object add
              (ie, no exact prefix/origin match exists).

     /* first check the aut-num authorization */

     if (the referenced aut-num object does not exist or
          the aut-num authorization fails)
       authorization fails

     /* next we check for prefix authorization */

     if (a less specific route(s) with the longest prefix is found) [
       if (authorization does not pass for at least one of the less
            specific route(s))
         authorization fails

     /* now check for a "reclaim" attr */

       if (the object has a "reclaim" attribute) [
         if (no more-specifics exist
              OR a less specific exists which passes
                 authorization and has a "reclaim" attribute

```
             OR all more specifics routess pass modify authorization )
          authorization passes
        else
          authorization fails
      ] else
        authorization passes
    ]

    /* there are no less specific routes to check for prefix
       authentication, so we need to try and get authorization from an
       inetnum object */

    if ( ( an inetnum is found that is an exact match
           OR is less specific and it's status is "allocated" )
         AND a maintainer referenced by the inetnum
             passes authorization )
      authorization succeeds

    /* if there is no inetnum or route object then then
       authorization fails.  This should never happen if
       the DB is initialized properly. */

    authorization fails.

    Case #2.  Route object modify/delete
              (ie, exact prefix/origin match exists).

    if ( the mnt-by passes authorization )
      authorization succeeds

    /* if the authorization did not pass from the matched object,
       we can still get authorization from a less specific route if it
       has a "reclaim" attribute and we pass authorization */

    if ( a less specific route or inetnum object passes authorization
         AND has a "reclaim" attribute applicable to
             the object to be modified )
      authorization succeeds
    else
      authorization fails
```

Acknowledgments

Gerald Winters provided the pseudocode in an email message to the
RIPE dbsec mailing list that was the basis of the pseudocode found in
appendix F. Susan Harris provided comments and numerous editorial
corrections.

Intellectual Property Notice

The IETF takes no position regarding the validity or scope of any
intellectual property or other rights that might be claimed to
pertain to the implementation or use of the technology described in
this document or the extent to which any license under such rights
might or might not be available; neither does it represent that it
has made any effort to identify any such rights. Information on the
IETF's procedures with respect to rights in standards-track and
standards-related documentation can be found in BCP-11. Copies of
claims of rights made available for publication and any assurances of
licenses to be made available, or the result of an attempt made to
obtain a general license or permission for the use of such
proprietary rights by implementors or users of this specification can
be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any
copyrights, patents or patent applications, or other proprietary
rights which may cover technology that may be required to practice
this standard. Please address the information to the IETF Executive
Director.

References

    [1]   Alaettinoglu, C., Bates, T., Gerich, E., Karrenberg, D., Meyer,
          D., Terpstra M. and C. Villamizar, "Routing Policy
          Specification Language (RPSL)", RFC 2280, January 1998.

    [2]   Bates, T., Gerich, E., Joncheray, L., Jouanigot, J-M.,
          Karrenberg, D., Terpstra, M. and J. Yu, "Representation of IP
          Routing Policies in a Routing Registry (ripe-81++)", RFC 1786,
          March 1995.

    [3]   Berkowitz, H., "Router Renumbering Guide", RFC 2072, January
          1997.

    [4]   Braun, H-W., "Models of policy based routing", RFC 1104, June
          1989.

    [5]   Braun, H-W. and Y. Rekhter, "Advancing the NSFNET routing
          architecture", RFC 1222, May 1991.

   [6]   Clark, D., "Policy routing in Internet protocols", RFC 1102,
         May 1989.

   [7]   Crocker, D., "Standard for the format of ARPA Internet text
         messages", STD 11, RFC 822, August 1982.

   [8]   Fuller, V., Li, T., Yu, J. and K. Varadhan, "Classless Inter-
         Domain Routing (CIDR): an Address Assignment and Aggregation
         Strategy", RFC 1519, September 1993.

   [9]   Internet Engineering Steering Group and R. Hinden,
         "Applicability Statement for the Implementation of Classless
         Inter-Domain Routing (CIDR)", RFC 1517, September 1993.

  [10]   Hawkinson, J. and T. Bates, "Guidelines for creation,
         selection, and registration of an Autonomous System (AS)", RFC
         1930, March 1996.

  [11]   Hubbard, K., Kosters, M., Conrad, D., Karrenberg, D. and J.
         Postel, "Internet Registry IP Allocation Guidelines", BCP 12,
         RFC 2050, November 1996.

  [12]   Knopper, M. and S. Richardson, "Aggregation Support in the
         NSFNET Policy-Based Routing Database", RFC 1482, June 1993.

  [13]   Meyer, D., Prior, M., Alaettinoglu, C., Schmitz, J. and Carol
         Orange, "Using RPSL in Practice", RFC 2650, August 1999.

  [14]   Rekhter, Y., "Routing in a Multi-provider Internet", RFC 1787,
         April 1995.

  [15]   Rekhter Y. and T. Li, "An Architecture for IP Address
         Allocation with CIDR", RFC 1518, September 1993.

  [16]   Rekhter Y. and T. Li, "Implications of Various Address
         Allocation Policies for Internet Routing", RFC 2008, October
         1996.

  [17]   Rekhter, Y., Lothberg, P., Hinden, R., Deering, S. and J.
         Postel, "An IPv6 Provider-Based Unicast Address Format", RFC
         2073, January 1997.

  [18]   Zsako, J., "PGP Authentication for RIPE Database Updates", RFC
         2726, December 1999.

Security Considerations

   This document primarily addresses authorization rules for making
   additions, deletions, and changes to routing policy information
   repositories. The authentication of these transactions through
   strong cryptographic means are addressed by [18], referenced
   thorughout this document. The authorization rules are designed such
   that the integrity of any transaction can be verified independently
   by any party mirroring a repository to insure that rules are adhered
   to. To accomplish this the mirror must contain data already known to
   be properly authorized. In other words, the mirror must be complete
   and authentication and authorization checks must be made continuously
   as changes to the repository are recieved and processed in order.

   Authentication alone does not provide a complete security model.
   Current practice specifies authorization for deletions and changes
   only, not for additions. The authorization rules provide here
   complete the security model for additions, deletions, and changes by
   very explicitly defining rules for addition and clarifying procedures
   for handling exception cases such as organizations which have ceased
   to exist and therefore become entirely unresponsive.

   Authentication and authorization of queries is explicitly stated to
   be out of scope of this document.

Authors' Addresses

   Curtis Villamizar
   Avici Systems
   EMail: curtis@avici.com

   Cengiz Alaettinoglu
   ISI
   EMail: cengiz@ISI.EDU

   David M. Meyer
   Cisco
   EMail: dmm@cisco.com

   Sandy Murphy
   Trusted Information Systems
   EMail: sandy@tis.com

Full Copyright Statement

Acknowledgement