Internet Draft
Expires May 17, 1996
draft-ietf-rps-aspath-00.txt

Cengiz Alaettinoglu
USC/ISI
Jessica Yu
Merit Networking
July, 1995

Autonomous System Path Expression Extension to Ripe-181

Status of this Memo

This document extends Ripe-181 to enable the specification of AS path expressions in the aut-num objects.

This document is an Internet Draft, and can be found as draft-ietf-rps-aspath-00.txt in any standard internet drafts repository. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are draft documents valid for a maximum of six months. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material, or to cite them other than as a ``working draft'' or ``work in progress.''

Please check the I-D abstract listing contained in each Internet Draft directory to learn the current status of this or any other Internet Draft.

1 Introduction

Ripe-181 [Bates et al., 1994] is a language to specify routing policy constraints for inter-domain routing in the Internet. Ripe-181 allows the specification of both import policy constraints (by specifying as-in and interas-in attributes) and export policy constraints (by specifying as-out and interas-out attributes). There are limitations in the types of preference and access control policies that can be described by RIPE-181 and the limitations became evident when several enterprises tried to use RIPE-181 to describe their routing policies.

One such limitation is the lack of ability to specify autonomous-system path expressions, henceforth referred to as AS path expressions, in the policy attributes (i.e. as-in, interas-in, as-out and interas-out attributes). This document extends Ripe-181 to enable the specification of AS path expressions in the aut-num objects. It is assumed that the reader is

familiar with Ripe-181.

2 Extended Attributes

We define four new policy attributes for the aut-num object, extended-as-in, extended-as-out, extended-interas-in, and extended-interas-out to augment the as-in, as-out, interas-in, and interas-out attributes respectively. The short names for these attributes are as follows: xa for extended-as-in, xb for extended-as-out, xc for extended-interas-in, and xd for extended-interas-out.

The original attributes have the following forms [Bates et al., 1994]:

```
as-in:       from <aut-num> <cost>
             accept <routing policy expression>
as-out:      to <aut-num>
             announce <routing policy expression>
interas-in:  from <aut-num> <local-rid> <neighbor-rid> <preference>
             accept <routing policy expression>
interas-out: to <aut-num> <local-rid> <neighbor-rid> [<metric>]
             announce <routing policy expression>.
```

Please refer to RIPE-181 [Bates et al., 1994] for the details of their syntax and semantics.

The extended attributes have the same syntax and semantics as the corresponding original attributes except that the <routing policy expression> is extended. That is, we have the following new syntax for the extended attributes:

```
extended-as-in:       from <aut-num> <cost>
                      accept <extended routing policy expression>
extended-as-out:      to <aut-num>
                      announce <extended routing policy expression>
extended-interas-in:  from <aut-num> <local-rid> <neighbor-rid> <preference>
                      accept <extended routing policy expression>
extended-interas-out: to <aut-num> <local-rid> <neighbor-rid> [<metric>]
                      announce <extended routing policy expression>.
```

<extended routing policy expression> is a logical combination of AS numbers, AS macros, community names, route lists, predefined keywords (currently only the keyword ANY is defined), and AS path expressions. The syntax and semantics of <extended routing policy expression> are identical to the syntax and semantics of <routing policy expression> except that it can contain AS path expressions.

An AS path expression has the following syntax:

`<' <AS path regular expression> `>'.

The syntax and semantics of AS path regular expressions are given in Appendix A. An AS path expression represents the set of routes which traverses a sequence of autonomous systems matched by the AS path regular expression(1). Note that the AS path regular expression only needs to match a part of the complete AS path traversed. For example, AS path expression <AS1> matches the AS path ``1'' as well as the AS path ``2 1 3''.

The following are examples of extended attributes:

```
extended-as-in: from AS1 5 accept <^AS1 . * AS2$> AND NOT <AS3>
extended-as-out: to AS690 announce (<^AS1> OR <^AS2>) AND COMM_NSF_AUP.
```

The first example means ``accept the set of routes from AS1 with an AS_PATH to AS2 that do not transit AS3 with preference 5. The second example means announce the set of routes learned from AS1 and AS2 which are in COMM_NSF_AUP to AS690.

The same Ripe-181 rules that apply to original attributes also apply to the extended attributes. For example, in RIPE-181, one can only specify an interas-in/out attribute if a corresponding as-in/out attributed is specified for that peer AS. Hence, one can only specify an extended-interas-in/out attribute if a corresponding extended-as-in/out attribute is specified for that peer AS. Extended attributes are paired only with corresponding extended attributes.

3 Interaction Between Extended and Original Attributes

In specifying policies for a given peer AS, if an AS only uses the original attributes, or only uses the extended attributes, the semantics are clearly defined in the previous section. However, the semantics are not so clear when an AS uses both the original and extended attributes for a given peer AS. We allow only one kind of mixing of the original and the extended attributes as depicted by the following two rules: For a given peer AS, if an AS uses either an extended-as-in or extended-interas-in attribute, then for every extended-as-in policy it specifies, it also has to specify an as-in policy; and for every extended-interas-in policy it specifies, it also has to specify an interas-in policy. For a given peer AS, if an AS uses either an extended-as-out or extended-interas-out attribute, then for every extended-as-out policy it specifies, it also has to specify an as-out policy; and for every extended-interas-out policy it specifies, it also has to specify an interas-out policy. Note that the *-in attributes and *-out attributes are treated separately. When mixing, care should be taken that

the policies specified using the original and the matching extended attributes are as equivalent as possible and certainly not contradictory.

---

(1) A router can check this using the AS_PATH attribute in the Border Gateway Protocol [Rekhter and Li, 1994] or RD_PATH attribute in the Inter-Domain Routing Protocol [Rekhter, 1993] (other protocols have similar mechanisms).

When a domain mixes the extended and the original attributes for a peer AS, the tools that do not support the extended attributes use only the original attributes. The tools that support the extended attributes only consider the original as-in/interas-in attributes if there is no extended-as-in/extended-interas-in attributes, for a peer AS. Otherwise, they only consider the extended-as-in/extended-interas-in attributes. The tools that support the extended attributes only consider the original as-out/interas-out attributes if there is no extended-as-out/extended-interas-out attributes, for a peer AS. Otherwise, they only consider the extended-as-out/extended-interas-out attributes. We expect that all tools will be upgraded eventually to support the extended attributes, at which time, original attributes can be replaced by the extended attributes.

Here are some valid examples of mixing:

Eg. 1. (No mixing, original attributes only)
```
aut-num: AS1
as-in: from AS2 1 accept ANY
as-out: to AS2 announce ANY
```

Eg. 2. (No mixing, extended attributes only)
```
aut-num: AS1
extended-as-in: from AS2 1 accept ANY
extended-as-out: to AS2 announce ANY
```

Eg. 3. (Perfect mixing)
```
aut-num: AS1
as-in: from AS2 1 accept ANY
as-out: to AS2 announce ANY
extended-as-in: from AS2 1 accept ANY
extended-as-out: to AS2 announce ANY
```

And the following example is not valid because an extended-as-out attribute is used without using an as-out attribute.

```
Eg. 4. (Invalid mixing)
aut-num: AS1
as-in: from AS2 1 accept ANY
extended-as-in: from AS2 1 accept ANY
extended-as-out: to AS2 announce ANY AND NOT <AS1 . * AS5 AS6 .* AS10$>
```

Note that the following example, in which as-out line does not reflect the exact export policy of AS1, is valid.

```
Eg. 5. (Valid mixing)
aut-num: AS1
as-in: from AS2 1 accept ANY
as-out: to AS2 announce NOT AS10
extended-as-in: from AS2 1 accept NOT <AS5>
extended-as-out: to AS2 announce NOT <AS1 . * AS5 AS6 .* AS10>
```

We allow this so that the existing tools can function and produce approximate results (using the approximate policy constraints as illustrated in the example) at the discretion of the AS.

```
Eg. 6. (Valid mixing, as-out/extended-as-out only)
aut-num: AS1
as-in: from AS2 1 accept AS2
as-out: to AS2 announce AS1
extended-as-out: to AS2 announce <^[AS3 AS4 AS5]> OR AS1
```

In this example, as-out and extended-as-out attributes are mixed, and as-in attribute is not mixed with an extended attribute.

4 Suggestions

We reccomend administrators who register extended attributes to also register original attributes when possible as explained in Section 3.

We also caution that specifying AS path policy expressions which are too topology specific can have serious consequences (see [Bates et al., 1994]). This is because Internet's topology is dynamic and the changes in the topology can easily affect the set of routes matched by these expressions drastically.

The following three <extended routing policy expression> represent the same set of routes, however the last one can be translated into a finte state automota much more efficiently than the second one, and the second one can be translated much more efficiently than the first one:

- o <AS1$> OR < AS2$> OR < AS3$>

- o <(AS1 | AS2 | AS3)$>

- o <[AS1 AS2 AS3]$>

We recommend the use of the last form whenever possible.

5 Acknowledgments

We thank Elise Gerich, Curtis Villamizar, Daniel Karrenberg, Dale Johnson, Laurent Joncheray and various members of the RA team for various discussions and their suggestions.

References

[Bates et al., 1994] T. Bates, E. Gerich, L. Joncheray, J-M. Jouanigot, D. Karrenberg, M. Terpstra, and J. Yu. Representation of IP Routing Policies in a Routing Registry. Technical Report ripe-181, RIPE, RIPE

NCC, Amsterdam, Netherlands, October 1994.

[Rekhter and Li, 1994] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). Request for Comment RFC-1654, Network Information Center, July 1994.

[Rekhter, 1993] Y. Rekhter. Inter-Domain Routing Protocol (IDRP). Journal of Internetworking Research and Experience, 4:61--80, 1993.

A AS Path Regular Expressions

AS Path Regular Expressions are POSIX compliant regular expressions over the alphabet of AS numbers. The regular expression constructs are informally summarized below:

ASN where ASN is an AS number in Ripe-181 notation (e.g. AS1, AS690). ASN matches the AS path that is of length 1 and contains the corresponding AS number (e.g. AS path regular expression AS1 matches the AS path ``1'').

ASMacro where ASMacro is an AS macro name (see [Bates et al., 1994]).

ASMacro matches any AS path that is matched by one of the AS's in the expansion of ASMacro.

. matches any AS path matched by any AS number.

[...] is an AS number set. It matches any AS path matched by any AS number listed between the brackets. If a `-' is used between two AS numbers in this set, all AS numbers between the two AS numbers are included in the set. If an AS macro ASMacro is used in this set, all AS numbers in the expansion of ASMacro are included in the set.

[^...] is a complemented AS number set. It matches any AS path which is not matched by the AS numbers in the set.

^ Matches the empty string at the beginning of an AS path.

$ Matches the empty string at the end of an AS path.

We next list the regular expression operators in the decreasing order of evaluation. These operators are left associative, i.e. performed left to right.

Unary postfix operators * + ? For a regular expression A, A* matches zero or more occurrence of A; A+ matches one or more occurrence of A; A? matches zero or one occurrence of A.

Binary catanation operator This is an implicit operator and exists between two regular expressions A and B when no other explicit operator is specified. The resulting expression AB matches

---

an AS path if A matches some amount of the beginning of that AS path and B matches the rest of the AS path.

Binary alternative (or) operator | For a regular expressions A and B, A | B matches any AS path that is matched by A or B.

Paranthesis can be used to override the default order of evaluation. Space, tab, and underscore characters can be used to increase readibility and they are ignored.

The BNF that parses the AS path regular expressions is as follows:

```
aspathre: aspathre_term
  | aspathre '|' aspathre_term
```

```
aspathre_term: aspathre_closure
  | aspathre_term aspathre_closure

aspathre_closure: aspathre_factor
  | aspathre_closure '*'
  | aspathre_closure '?'
  | aspathre_closure '+'

aspathre_factor: aspathre_no
  | '^'
  | '$'
  | '(' aspathre ')'

aspathre_no: ASNUM_TOK
  | ASMACRO_TOK
  | '.'
  | '[' aspathre_range ']'
  | '[' '^' aspathre_range ']'

aspathre_range: aspathre_subrange
  | aspathre_range aspathre_subrange

aspathre_subrange: ASNUM_TOK
  | ASMACRO_TOK
  | '.'
  | ASNUM_TOK '-' ASNUM_TOK
```

where ASNUM_TOK is a string whose first two characters are ``AS'' and the rest of the characters form a valid positive integer and ASMACRO_TOK is a string whose first three characters are ``AS-'' and the rest of the characters are uppercase letters from the English alphabet.

Author's Present Addresses

Cengiz Alaettinoglu
Information Sciences Institute
University of Southern California
Marina del Rey, CA 90292
e-mail: cengiz@isi.edu

Jessica Yu
Merit Networking
University of Michigan
Ann Arbor, MI 48105
e-mail: jyy@merit.edu