           X.509 Extensions for IP Addresses and AS Identifiers

Status of this Memo

Copyright Notice

Abstract

   This document defines two X.509 v3 certificate extensions.  The first
   binds a list of IP address blocks, or prefixes, to the subject of a
   certificate.  The second binds a list of autonomous system
   identifiers to the subject of a certificate.  These extensions may be
   used to convey the authorization of the subject to use the IP
   addresses and autonomous system identifiers contained in the
   extensions.

Table of Contents

1.  Introduction

   This document defines two X.509 v3 certificate extensions that
   authorize the transfer of the right-to-use for a set of IP addresses
   and autonomous system identifiers from IANA through the regional
   Internet registries (RIRs) to Internet service providers (ISPs) and
   user organizations.  The first binds a list of IP address blocks,
   often represented as IP address prefixes, to the subject (private key
   holder) of a certificate.  The second binds a list of autonomous
   system (AS) identifiers to the subject (private key holder) of a
   certificate.  The issuer of the certificate is an entity (e.g., the
   IANA, a regional Internet registry, or an ISP) that has the authority
   to transfer custodianship of ("allocate") the set of IP address
   blocks and AS identifiers to the subject of the certificate.  These
   certificates provide a scalable means of verifying the right-to-use
   for a set of IP address prefixes and AS identifiers.  They may be
   used by routing protocols, such as Secure BGP [S-BGP], to verify
   legitimacy and correctness of routing information, or by Internet
   routing registries to verify data that they receive.

   Sections 2 and 3 specify several rules about the encoding of the
   extensions defined in this specification that MUST be followed.
   These encoding rules serve the following purposes.  First, they
   result in a unique encoding of the extension's value; two instances
   of an extension can be compared for equality octet-by-octet.  Second,
   they achieve the minimal size encoding of the information.  Third,
   they allow relying parties to use one-pass algorithms when performing
   certification path validation; in particular, the relying parties do
   not need to sort the information, or to implement extra code in the
   subset checking algorithms to handle several boundary cases
   (adjacent, overlapping, or subsumed ranges).

1.1.  Terminology

   It is assumed that the reader is familiar with the terms and concepts
   described in "Internet X.509 Public Key Infrastructure Certificate
   and Certificate Revocation List (CRL) Profile" [RFC3280], "INTERNET
   PROTOCOL" [RFC791], "Internet Protocol Version 6 (IPv6) Addressing
   Architecture" [RFC3513], "INTERNET REGISTRY IP ALLOCATION GUIDELINES"
   [RFC2050], and related regional Internet registry address management
   policy documents.  Some relevant terms include:

   allocate - the transfer of custodianship of a resource to an
      intermediate organization (see [RFC2050]).

   assign - the transfer of custodianship of a resource to an end
      organization (see [RFC2050]).

Autonomous System (AS) - a set of routers under a single technical
    administration with a uniform policy, using one or more interior
    gateway protocols and metrics to determine how to route packets
    within the autonomous system, and using an exterior gateway
    protocol to determine how to route packets to other autonomous
    systems.

Autonomous System number - a 32-bit number that identifies an
    autonomous system.

delegate - transfer of custodianship (that is, the right-to-use) of
    an IP address block or AS identifier through issuance of a
    certificate to an entity.

initial octet - the first octet in the value of a DER encoded BIT
    STRING [X.690].

IP v4 address - a 32-bit identifier written as four decimal numbers,
    each in the range 0 to 255, separated by a ".".  10.5.0.5 is an
    example of an IPv4 address.

IP v6 address - a 128-bit identifier written as eight hexadecimal
    quantities, each in the range 0 to ffff, separated by a ":".
    2001:0:200:3:0:0:0:1 is an example of an IPv6 address.  One string
    of :0: fields may be replaced by "::", thus 2001:0:200:3::1
    represents the same address as the immediately preceding example.
    (See [RFC3513]).

prefix - a bit string that consists of some number of initial bits of
    an address, written as an address followed by a "/", and the
    number of initial bits.  10.5.0.0/16 and 2001:0:200:3:0:0:0:0/64
    (or 2001:0:200:3::/64) are examples of prefixes.  A prefix is
    often abbreviated by omitting the less-significant zero fields,
    but there should be enough fields to contain the indicated number
    of initial bits.  10.5/16 and 2001:0:200:3/64 are examples of
    abbreviated prefixes.

Regional Internet Registry (RIR) - any of the bodies recognized by
    IANA as the regional authorities for management of IP addresses
    and AS identifiers.  At the time of writing, these include
    AfriNIC, APNIC, ARIN, LACNIC, and RIPE NCC.

right-to-use - for an IP address prefix, being authorized to specify
    the AS that may originate advertisements of the prefix throughout
    the Internet.  For an autonomous system identifier, being
    authorized to operate a network(s) that identifies itself to other
    network operators using that autonomous system identifier.

   subsequent octets - the second through last octets in the value of a
      DER encoded BIT STRING [X.690].

   trust anchor - a certificate that is to be trusted when performing
      certification path validation (see [RFC3280]).

   The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD,
   SHOULD NOT, RECOMMENDED, and MAY, and OPTIONAL, when they appear in
   this document, are to be interpreted as described in [RFC2119].

2.  IP Address Delegation Extension

   This extension conveys the allocation of IP addresses to an entity by
   binding those addresses to a public key belonging to the entity.

2.1.  Context

   IP address space is currently managed by a hierarchy nominally rooted
   at IANA, but managed by the RIRs.  IANA allocates IP address space to
   the RIRs, who in turn allocate IP address space to Internet service
   providers (ISPs), who may allocate IP address space to down stream
   providers, customers, etc.  The RIRs also may assign IP address space
   to organizations who are end entities, i.e., organizations who will
   not be reassigning any of their space to other organizations.  (See
   [RFC2050] and related RIR policy documents for the guidelines on the
   allocation and assignment process).

   The IP address delegation extension is intended to enable
   verification of the proper delegation of IP address blocks, i.e., of
   the authorization of an entity to use or sub-allocate IP address
   space.  Accordingly, it makes sense to take advantage of the inherent
   authoritativeness of the existing administrative framework for
   allocating IP address space.  As described in Section 1 above, this
   will be achieved by issuing certificates carrying the extension
   described in this section.  An example of one use of the information
   in this extension is an entity using it to verify the authorization
   of an organization to originate a BGP UPDATE advertising a path to a
   particular IP address block; see, e.g., [RFC1771], [S-BGP].

2.1.1.  Encoding of an IP Address or Prefix

   There are two families of IP addresses: IPv4 and IPv6.

   An IPv4 address is a 32-bit quantity that is written as four decimal
   numbers, each in the range 0 through 255, separated by a dot (".").
   10.5.0.5 is an example of an IPv4 address.

An IPv6 address is a 128-bit quantity that is written as eight
hexadecimal numbers, each in the range 0 through ffff, separated by a
semicolon (":"); 2001:0:200:3:0:0:0:1 is an example of an IPv6
address.  IPv6 addresses frequently have adjacent fields whose value
is 0.  One such group of 0 fields may be abbreviated by two
semicolons ("::").  The previous example may thus be represented by
2001:0:200:3::1.

An address prefix is a set of 2^k continuous addresses whose most-
significant bits are identical.  For example, the set of 512 IPv4
addresses from 10.5.0.0 through 10.5.1.255 all have the same 23
most-significant bits.  The set of addresses is written by appending
a slash ("/") and the number of constant bits to the lowest address
in the set.  The prefix for the example set is 10.5.0.0/23, and
contains 2^(32-23) = 2^9 addresses.  The set of IPv6 addresses
2001:0:200:0:0:0:0:0 through 2001:0:3ff:ffff:ffff:ffff:ffff:ffff
(2^89 addresses) is represented by 2001:0:200:0:0:0:0:0/39 or
equivalently 2001:0:200::/39.  A prefix may be abbreviated by
omitting the least-significant zero fields, but there should be
enough fields to contain the indicated number of constant bits.  The
abbreviated forms of the example IPv4 prefix is 10.5.0/23, and of the
example IPv6 prefix is 2001:0:200/39.

An IP address or prefix is encoded in the IP address delegation
extension as a DER-encoded ASN.1 BIT STRING containing the constant
most-significant bits.  Recall [X.690] that the DER encoding of a BIT
STRING consists of the BIT STRING type (0x03), followed by (an
encoding of) the number of value octets, followed by the value.  The
value consists of an "initial octet" that specifies the number of
unused bits in the last value octet, followed by the "subsequent
octets" that contain the octets of the bit string.  (For IP
addresses, the encoding of the length will be just the length.)

In the case of a single address, all the bits are constant, so the
bit string for an IPv4 address contains 32 bits.  The subsequent
octets in the DER-encoding of the address 10.5.0.4 are 0x0a 0x05 0x00
0x04.  Since all the bits in the last octet are used, the initial
octet is 0x00.  The octets in the DER-encoded BIT STRING is thus:

        Type Len  Unused Bits ...
        0x03 0x05  0x00  0x0a 0x05 0x00 0x04

Similarly, the DER-encoding of the prefix 10.5.0/23 is:

        Type Len  Unused Bits ...
        0x03 0x04  0x01  0x0a 0x05 0x00

In this case, the three subsequent octets contain 24 bits, but the
prefix only uses 23, so there is one unused bit in the last octet,
thus the initial octet is 1 (the DER require that all unused bits
MUST be set to zero-bits).

The DER-encoding of the IPv6 address 2001:0:200:3:0:0:0:1 is:

```
Type Len  Unused Bits ...
0x03 0x11  0x00  0x20 0x01 0x00 0x00 0x02 0x00 0x00 0x03
                       0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x01
```

and the DER-encoding of the prefix 2001:0:200/39, which has one
unused bit in the last octet, is:

```
Type Len  Unused Bits ...
0x03 0x06  0x01  0x20 0x01 0x00 0x00 0x02
```

## 2.1.2.  Encoding of a Range of IP Addresses

While any contiguous range of IP addresses can be represented by a
set of contiguous prefixes, a more concise representation is achieved
by encoding the range as a SEQUENCE containing the lowest address and
the highest address, where each address is encoded as a BIT STRING.
Within the SEQUENCE, the bit string representing the lowest address
in the range is formed by removing all the least-significant zero-
bits from the address, and the bit string representing the highest
address in the range is formed by removing all the least-significant
one-bits.  The DER BIT STRING encoding requires that all the unused
bits in the last octet MUST be set to zero-bits.  Note that a prefix
can always be expressed as a range, but a range cannot always be
expressed as a prefix.

The range of addresses represented by the prefix 10.5.0/23 is
10.5.0.0 through 10.5.1.255.  The lowest address ends in sixteen
zero-bits that are removed.  The DER-encoding of the resulting
sixteen-bit string is:

```
Type Len  Unused Bits ...
0x03 0x03  0x00  0x0a 0x05
```

The highest address ends in nine one-bits that are removed.  The DER-
encoding of the resulting twenty-three-bit string is:

```
Type Len  Unused Bits ...
0x03 0x04  0x01  0x0a 0x05 0x00
```

The prefix 2001:0:200/39 can be encoded as a range where the DER-
encoding of the lowest address (2001:0:200::) is:

         Type Len  Unused Bits ...
         0x03 0x06  0x01  0x20 0x01 0x00 0x00 0x02

and the largest address (2001:0:3ff:ffff:ffff:ffff:ffff:ffff), which,
after removal of the ninety least-significant one-bits leaves a
thirty-eight bit string, is encoded as:

         Type Len  Unused Bits ...
         0x03 0x06  0x02  0x20 0x01 0x00 0x00 0x00

The special case of all IP address blocks, i.e., a prefix of all
zero-bits -- "0/0", MUST be encoded per the DER with a length octet
of one, an initial octet of zero, and no subsequent octets:

         Type Len  Unused Bits ...
         0x03 0x01  0x00

Note that for IP addresses the number of trailing zero-bits is
significant.  For example, the DER-encoding of 10.64/12:

         Type Len  Unused Bits ...
         0x03 0x03  0x04  0x0a 0x40

is different than the DER-encoding of 10.64.0/20:

         Type Len  Unused Bits ...
         0x03 0x04  0x04  0x0a 0x40 0x00

## 2.2.  Specification

### 2.2.1.  OID

The OID for this extension is id-pe-ipAddrBlocks.

    id-pe-ipAddrBlocks  OBJECT IDENTIFIER ::= { id-pe 7 }

where [RFC3280] defines:

    id-pkix  OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
             dod(6) internet(1) security(5) mechanisms(5) pkix(7) }

    id-pe    OBJECT IDENTIFIER ::= { id-pkix 1 }

## 2.2.2.  Criticality

   This extension SHOULD be CRITICAL.  The intended use of this
   extension is to connote a right-to-use for the block(s) of IP
   addresses identified in the extension.  A CA marks the extension as
   CRITICAL to convey the notion that a relying party MUST understand
   the semantics of the extension to make use of the certificate for the
   purpose it was issued.  Newly created applications that use
   certificates containing this extension are expected to recognize the
   extension.

## 2.2.3.  Syntax

```
   id-pe-ipAddrBlocks       OBJECT IDENTIFIER ::= { id-pe 7 }

   IPAddrBlocks          ::= SEQUENCE OF IPAddressFamily

   IPAddressFamily       ::= SEQUENCE {     -- AFI & optional SAFI --
       addressFamily         OCTET STRING (SIZE (2..3)),
       ipAddressChoice       IPAddressChoice }

   IPAddressChoice       ::= CHOICE {
       inherit               NULL, -- inherit from issuer --
       addressesOrRanges     SEQUENCE OF IPAddressOrRange }

   IPAddressOrRange      ::= CHOICE {
       addressPrefix         IPAddress,
       addressRange          IPAddressRange }

   IPAddressRange        ::= SEQUENCE {
       min                   IPAddress,
       max                   IPAddress }

   IPAddress             ::= BIT STRING
```

### 2.2.3.1.  Type IPAddrBlocks

   The IPAddrBlocks type is a SEQUENCE OF IPAddressFamily types.

### 2.2.3.2.  Type IPAddressFamily

   The IPAddressFamily type is a SEQUENCE containing an addressFamily
   and ipAddressChoice element.

## 2.2.3.3.  Element addressFamily

The addressFamily element is an OCTET STRING containing a two-octet
Address Family Identifier (AFI), in network byte order, optionally
followed by a one-octet Subsequent Address Family Identifier (SAFI).
AFIs and SAFIs are specified in [IANA-AFI] and [IANA-SAFI],
respectively.

If no authorization is being granted for a particular AFI and
optional SAFI, then there MUST NOT be an IPAddressFamily member for
that AFI/SAFI in the IPAddrBlocks SEQUENCE.

There MUST be only one IPAddressFamily SEQUENCE per unique
combination of AFI and SAFI.  Each SEQUENCE MUST be ordered by
ascending addressFamily values (treating the octets as unsigned
quantities).  An addressFamily without a SAFI MUST precede one that
contains an SAFI.  When both IPv4 and IPv6 addresses are specified,
the IPv4 addresses MUST precede the IPv6 addresses (since the IPv4
AFI of 0001 is less than the IPv6 AFI of 0002).

## 2.2.3.4.  Element ipAddressChoice and Type IPAddressChoice

The ipAddressChoice element is of type IPAddressChoice.  The
IPAddressChoice type is a CHOICE of either an inherit or
addressesOrRanges element.

## 2.2.3.5.  Element inherit

If the IPAddressChoice CHOICE contains the inherit element, then the
set of authorized IP addresses for the specified AFI and optional
SAFI is taken from the issuer's certificate, or from the issuer's
issuer's certificate, recursively, until a certificate containing an
IPAddressChoice containing an addressesOrRanges element is located.

## 2.2.3.6.  Element addressesOrRanges

The addressesOrRanges element is a SEQUENCE OF IPAddressOrRange
types.  The addressPrefix and addressRange elements MUST be sorted
using the binary representation of:

    <lowest IP address in range> | <prefix length>

where "|" represents concatenation.  Note that the octets in this
representation (a.b.c.d | length for IPv4 or s:t:u:v:w:x:y:z | length
for IPv6) are not the octets that are in the DER-encoded BIT STRING
value.  For example, given two addressPrefix:

```
      IP addr | length  DER encoding
      ---------------  ------------------------
                       Type Len  Unused Bits...
      10.32.0.0 | 12    03   03    04    0a 20
      10.64.0.0 | 16    03   03    00    0a 40
```

the prefix 10.32.0.0/12 MUST come before the prefix 10.64.0.0/16
since 32 is less than 64; whereas if one were to sort by the DER BIT
STRINGs, the order would be reversed as the unused bits octet would
sort in the opposite order.  Any pair of IPAddressOrRange choices in
an extension MUST NOT overlap each other.  Any contiguous address
prefixes or ranges MUST be combined into a single range or, whenever
possible, a single prefix.

## 2.2.3.7.  Type IPAddressOrRange

The IPAddressOrRange type is a CHOICE of either an addressPrefix (an
IP prefix or address) or an addressRange (an IP address range)
element.

This specification requires that any range of addresses that can be
encoded as a prefix MUST be encoded using an IPAddress element (a BIT
STRING), and any range that cannot be encoded as a prefix MUST be
encoded using an IPAddressRange (a SEQUENCE containing two BIT
STRINGs).  The following pseudo code illustrates how to select the
encoding of a given range of addresses.

```
      LET  N = the number of matching most-significant bits in the
               lowest and highest addresses of the range
      IF   all the remaining bits in the lowest address are zero-bits
       AND all the remaining bits in the highest address are one-bits
      THEN the range MUST be encoded as an N-bit IPAddress
      ELSE the range MUST be encoded as an IPAddressRange
```

## 2.2.3.8.  Element addressPrefix and Type IPAddress

The addressPrefix element is an IPAddress type.  The IPAddress type
defines a range of IP addresses in which the most-significant (left-
most) N bits of the address remain constant, while the remaining bits
(32 - N bits for IPv4, or 128 - N bits for IPv6) may be either zero
or one.  For example, the IPv4 prefix 10.64/12 corresponds to the
addresses 10.64.0.0 to 10.79.255.255, while 10.64/11 corresponds to
10.64.0.0 to 10.95.255.255.  The IPv6 prefix 2001:0:2/48 represents
addresses 2001:0:2:: to 2001:0:2:ffff:ffff:ffff:ffff:ffff.

An IP address prefix is encoded as a BIT STRING.  The DER encoding of
a BIT STRING uses the initial octet of the string to specify how many
of the least-significant bits of the last subsequent octet are

unused.  The DER encoding specifies that these unused bits MUST be
set to zero-bits.

Example:
        128.0.0.0        = 1000 0000.0000 0000.0000 0000.0000 0000
    to 143.255 255 255 = 1000 1111.1111 1111.1111 1111.1111 1111
  bit string to encode = 1000
        Type Len  Unused Bits ...
Encoding = 0x03 0x02  0x04  0x80

## 2.2.3.9.  Element addressRange and Type IPAddressRange

The addressRange element is of type IPAddressRange.  The
IPAddressRange type consists of a SEQUENCE containing a minimum
(element min) and maximum (element max) IP address.  Each IP address
is encoded as a BIT STRING.  The semantic interpretation of the
minimum address in an IPAddressRange is that all the unspecified bits
(for the full length of the IP address) are zero-bits.  The semantic
interpretation of the maximum address is that all the unspecified
bits are one-bits.  The BIT STRING for the minimum address results
from removing all the least-significant zero-bits from the minimum
address.  The BIT STRING for the maximum address results from
removing all the least-significant one-bits from the maximum address.

Example:
        129.64.0.0        = 1000 0001.0100 0000.0000 0000.0000 0000
    to 143.255.255.255  = 1000 1111.1111 1111.1111 1111.1111 1111
     minimum bit string = 1000 0001.01
     maximum bit string = 1000
Encoding = SEQUENCE {
        Type Len  Unused Bits ...
  min    0x03 0x03  0x06  0x81        0x40
  max    0x03 0x02  0x04  0x80
        }

To simplify the comparison of IP address blocks when performing
certification path validation, a maximum IP address MUST contain at
least one bit whose value is 1, i.e., the subsequent octets may not
be omitted nor all zero.

## 2.3.  IP Address Delegation Extension Certification Path Validation

Certification path validation of a certificate containing the IP
address delegation extension requires additional processing.  As each
certificate in a path is validated, the IP addresses in the IP
address delegation extension of that certificate MUST be subsumed by
IP addresses in the IP address delegation extension in the issuer's
certificate.  Validation MUST fail when this is not the case.  A

certificate that is a trust anchor for certification path validation
of certificates containing the IP address delegation extension, as
well as all certificates along the path, MUST each contain the IP
address delegation extension.  The initial set of allowed address
ranges is taken from the trust anchor certificate.

## 3.  Autonomous System Identifier Delegation Extension

This extension conveys the allocation of autonomous system (AS)
identifiers to an entity by binding those AS identifiers to a public
key belonging to the entity.

### 3.1.  Context

AS identifier delegation is currently managed by a hierarchy
nominally rooted at IANA, but managed by the RIRs.  IANA allocates AS
identifiers to the RIRs, who in turn assign AS identifiers to
organizations who are end entities, i.e., will not be re-allocating
any of their AS identifiers to other organizations.  The AS
identifier delegation extension is intended to enable verification of
the proper delegation of AS identifiers, i.e., of the authorization
of an entity to use these AS identifiers.  Accordingly, it makes
sense to take advantage of the inherent authoritativeness of the
existing administrative framework for management of AS identifiers.
As described in Section 1 above, this will be achieved by issuing
certificates carrying the extension described in this section.  An
example of one use of the information in this extension is an entity
using it to verify the authorization of an organization to manage the
AS identified by an AS identifier in the extension. The use of this
extension to represent assignment of AS identifiers is not intended
to alter the procedures by which AS identifiers are managed, or when
an AS should be used c.f., [RFC1930].

### 3.2.  Specification

### 3.2.1.  OID

The OID for this extension is id-pe-autonomousSysIds.

    id-pe-autonomousSysIds  OBJECT IDENTIFIER ::= { id-pe 8 }

where [RFC3280] defines:

    id-pkix  OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
             dod(6) internet(1) security(5) mechanisms(5) pkix(7) }

    id-pe    OBJECT IDENTIFIER ::= { id-pkix 1 }

## 3.2.2.  Criticality

This extension SHOULD be CRITICAL.  The intended use of this
extension is to connote a right-to-use for the AS identifiers in the
extension.  A CA marks the extension as CRITICAL to convey the notion
that a relying party must understand the semantics of the extension
to make use of the certificate for the purpose it was issued.  Newly
created applications that use certificates containing this extension
are expected to recognize the extension.

## 3.2.3.  Syntax

```
id-pe-autonomousSysIds  OBJECT IDENTIFIER ::= { id-pe 8 }

ASIdentifiers          ::= SEQUENCE {
    asnum                  [0] EXPLICIT ASIdentifierChoice OPTIONAL,
    rdi                    [1] EXPLICIT ASIdentifierChoice OPTIONAL}

ASIdentifierChoice  ::= CHOICE {
    inherit                NULL, -- inherit from issuer --
    asIdsOrRanges          SEQUENCE OF ASIdOrRange }

ASIdOrRange            ::= CHOICE {
    id                     ASId,
    range                  ASRange }

ASRange                ::= SEQUENCE {
    min                    ASId,
    max                    ASId }

ASId                   ::= INTEGER
```

## 3.2.3.1.  Type ASIdentifiers

The ASIdentifiers type is a SEQUENCE containing one or more forms of
autonomous system identifiers -- AS numbers (in the asnum element) or
routing domain identifiers (in the rdi element).  When the
ASIdentifiers type contains multiple forms of identifiers, the asnum
entry MUST precede the rdi entry.  AS numbers are used by BGP, and
routing domain identifiers are specified in the IDRP [RFC1142].

## 3.2.3.2.  Elements asnum, rdi, and Type ASIdentifierChoice

The asnum and rdi elements are both of type ASIdentifierChoice.  The
ASIdentifierChoice type is a CHOICE of either the inherit or
asIdsOrRanges element.

### 3.2.3.3.  Element inherit

If the ASIdentifierChoice choice contains the inherit element, then
the set of authorized AS identifiers is taken from the issuer's
certificate, or from the issuer's issuer's certificate, recursively,
until a certificate containing an ASIdentifierChoice containing an
asIdsOrRanges element is located.  If no authorization is being
granted for a particular form of AS identifier, then there MUST NOT
be a corresponding asnum/rdi member in the ASIdentifiers sequence.

### 3.2.3.4.  Element asIdsOrRanges

The asIdsOrRanges element is a SEQUENCE of ASIdOrRange types.  Any
pair of items in the asIdsOrRanges SEQUENCE MUST NOT overlap.  Any
contiguous series of AS identifiers MUST be combined into a single
range whenever possible.  The AS identifiers in the asIdsOrRanges
element MUST be sorted by increasing numeric value.

### 3.2.3.5.  Type ASIdOrRange

The ASIdOrRange type is a CHOICE of either a single integer (ASId) or
a single sequence (ASRange).

### 3.2.3.6.  Element id

The id element has type ASId.

### 3.2.3.7.  Element range

The range element has type ASRange.

### 3.2.3.8.  Type ASRange

The ASRange type is a SEQUENCE consisting of a min and a max element,
and is used to specify a range of AS identifier values.

### 3.2.3.9.  Elements min and max

The min and max elements have type ASId.  The min element is used to
specify the value of the minimum AS identifier in the range, and the
max element specifies the value of the maximum AS identifier in the
range.

### 3.2.3.10.  Type ASId

The ASId type is an INTEGER.

3.3.  Autonomous System Identifier Delegation Extension Certification
      Path Validation

   Certification path validation of a certificate containing the
   autonomous system identifier delegation extension requires additional
   processing.  As each certificate in a path is validated, the AS
   identifiers in the autonomous system identifier delegation extension
   of that certificate MUST be subsumed by the AS identifiers in the
   autonomous system identifier delegation extension in the issuer's
   certificate.  Validation MUST fail when this is not the case.  A
   certificate that is a trust anchor for certification path validation
   of certificates containing the autonomous system identifier
   delegation extension, as well as all certificates along the path,
   MUST each contain the autonomous system identifier delegation
   extension.  The initial set of allowed AS identifiers is taken from
   the trust anchor certificate.

4.  Security Considerations

   This specification describes two X.509 extensions.  Since X.509
   certificates are digitally signed, no additional integrity service is
   necessary.  Certificates with these extensions need not be kept
   secret, and unrestricted and anonymous access to these certificates
   has no security implications.

   However, security factors outside the scope of this specification
   will affect the assurance provided to certificate users.  This
   section highlights critical issues that should be considered by
   implementors, administrators, and users.

   These extensions represent authorization information, i.e., a right-
   to-use for IP addresses or AS identifiers.  They were developed to
   support a secure version of BGP [S-BGP], but may be employed in other
   contexts.  In the secure BGP context, certificates containing these
   extensions function as capabilities: the certificate asserts that the
   holder of the private key (the Subject) is authorized to use the IP
   addresses or AS identifiers represented in the extension(s).  As a
   result of this capability model, the Subject field is largely
   irrelevant for security purposes, contrary to common PKI conventions.

5.  Acknowledgments

   The authors would like to acknowledge the contributions to this
   specification by Charles Gardiner, Russ Housley, James Manger, and
   Jim Schaad.

Appendix A -- ASN.1 Module

   This normative appendix describes the IP address and AS identifiers
   extensions used by conforming PKI components in ASN.1 syntax.

   IPAddrAndASCertExtn { iso(1) identified-organization(3) dod(6)
           internet(1) security(5) mechanisms(5) pkix(7) mod(0)
           id-mod-ip-addr-and-as-ident(30) }
      DEFINITIONS EXPLICIT TAGS ::=
   BEGIN
           -- Copyright (C) The Internet Society (2004). This    --
           -- version of this ASN.1 module is part of RFC 3779;  --
           -- see the RFC itself for full legal notices.         --

   -- EXPORTS ALL --

   IMPORTS

   -- PKIX specific OIDs and arcs --
       id-pe FROM PKIX1Explicit88 { iso(1) identified-organization(3)
                 dod(6) internet(1) security(5) mechanisms(5) pkix(7)
                 id-mod(0) id-pkix1-explicit(18) };

   -- IP Address Delegation Extension OID --

   id-pe-ipAddrBlocks  OBJECT IDENTIFIER ::= { id-pe 7 }

   -- IP Address Delegation Extension Syntax --

   IPAddrBlocks        ::= SEQUENCE OF IPAddressFamily

   IPAddressFamily     ::= SEQUENCE { -- AFI & opt SAFI --
      addressFamily        OCTET STRING (SIZE (2..3)),
      ipAddressChoice      IPAddressChoice }

   IPAddressChoice     ::= CHOICE {
      inherit              NULL, -- inherit from issuer --
      addressesOrRanges    SEQUENCE OF IPAddressOrRange }

   IPAddressOrRange    ::= CHOICE {
      addressPrefix        IPAddress,
      addressRange         IPAddressRange }

   IPAddressRange      ::= SEQUENCE {
      min                  IPAddress,
      max                  IPAddress }

   IPAddress           ::= BIT STRING

```
   -- Autonomous System Identifier Delegation Extension OID --

   id-pe-autonomousSysIds  OBJECT IDENTIFIER ::= { id-pe 8 }

   -- Autonomous System Identifier Delegation Extension Syntax --

   ASIdentifiers        ::= SEQUENCE {
       asnum                [0] ASIdentifierChoice OPTIONAL,
       rdi                  [1] ASIdentifierChoice OPTIONAL }

   ASIdentifierChoice  ::= CHOICE {
       inherit              NULL, -- inherit from issuer --
       asIdsOrRanges        SEQUENCE OF ASIdOrRange }

   ASIdOrRange          ::= CHOICE {
       id                   ASId,
       range                ASRange }

   ASRange              ::= SEQUENCE {
       min                  ASId,
       max                  ASId }

   ASId                 ::= INTEGER

   END
```

Appendix B -- Examples of IP Address Delegation Extensions

    A critical X.509 v3 certificate extension that specifies:
    IPv4 unicast address prefixes
        1)  10.0.32/20    i.e., 10.0.32.0 to 10.0.47.255
        2)  10.0.64/24    i.e., 10.0.64.0 to 10.0.64.255
        3)  10.1/16       i.e., 10.1.0.0  to 10.1.255.255
        4)  10.2.48/20    i.e., 10.2.48.0 to 10.2.63.255
        5)  10.2.64/24    i.e., 10.2.64.0 to 10.2.64.255
        6)  10.3/16       i.e., 10.3.0.0  to 10.3.255.255, and
        7)  inherits all IPv6 addresses from the issuer's certificate
    would be (in hexadecimal):

```
30 46                          Extension {
   06 08 2b06010505070107      extnID          1.3.6.1.5.5.7.1.7
   01 01 ff                    critical
   04 37                       extnValue {
      30 35                       IPAddrBlocks {
         30 2b                       IPAddressFamily {
            04 03 0001  01              addressFamily: IPv4 Unicast
                                        IPAddressChoice
            30 24                          addressesOrRanges {
```

```
                                        IPAddressOrRange
              03 04 04 0a0020             addressPrefix 10.0.32/20
                                        IPAddressOrRange
              03 04 00 0a0040             addressPrefix 10.0.64/24
                                        IPAddressOrRange
              03 03 00 0a01              addressPrefix     10.1/16
                                        IPAddressOrRange
              30 0c                       addressRange {
                 03 04 04 0a0230            min         10.2.48.0
                 03 04 00 0a0240            max         10.2.64.255
                                          } -- addressRange
                                        IPAddressOrRange
              03 03 00 0a03              addressPrefix     10.3/16
                                     } -- addressesOrRanges
                                   } -- IPAddressFamily
         30 06                      IPAddressFamily {
            04 02 0002                addressFamily: IPv6
                                      IPAddressChoice
            05 00                       inherit from issuer
                                     } -- IPAddressFamily
                                 } -- IPAddrBlocks
                             } -- extnValue
                          } -- Extension
```

   This example illustrates how the prefixes and ranges are sorted.

   +  Prefix 1 MUST precede prefix 2, even though the number of unused
      bits (4) in prefix 1 is larger than the number of unused bits (0)
      in prefix 2.

   +  Prefix 2 MUST precede prefix 3 even though the number of octets
      (4) in the BIT STRING encoding of prefix 2 is larger than the
      number of octets (3) in the BIT STRING encoding of prefix 3.

   +  Prefixes 4 and 5 are adjacent (representing the range of addresses
      from 10.2.48.0 to 10.2.64.255), so MUST be combined into a range
      (since the range cannot be encoded by a single prefix).

   +  Note that the six trailing zero bits in the max element of the
      range are significant to the semantic interpretation of the value
      (as all unused bits are interpreted to be 1's, not 0's).  The four
      trailing zero bits in the min element are not significant and MUST
      be removed (thus the (4) unused bits in the encoding of the min
      element).  (DER encoding requires that any unused bits in the last
      subsequent octet MUST be set to zero.)

+  The range formed by prefixes 4 and 5 MUST precede prefix 6 even
   though the SEQUENCE tag for a range (30) is larger than the tag
   for the BIT STRING (03) used to encode prefix 6.

+  The IPv4 information MUST precede the IPv6 information since the
   address family identifier for IPv4 (0001) is less than the
   identifier for IPv6 (0002).

An extension specifying the IPv6 prefix 2001:0:2/48 and the IPv4
prefixes 10/8 and 172.16/12, and which inherits all IPv4 multicast
addresses from the issuer's certificate would be (in hexadecimal):

```
30 3d                           Extension {
   06 08 2b06010505070107          extnID          1.3.6.1.5.5.7.1.7
   01 01 ff                        critical
   04 2e                           extnValue {
      30 2c                           IPAddrBlocks {
         30 10                           IPAddressFamily {
            04 03 0001 01                  addressFamily: IPv4 Unicast
                                           IPAddressChoice
            30 09                            addressesOrRanges {
                                              IPAddressOrRange
               03 02 00 0a                     addressPrefix    10/8
                                              IPAddressOrRange
               03 03 04 b010                   addressPrefix    172.16/12
                                            } -- addressesOrRanges
                                         } -- IPAddressFamily
         30 07                           IPAddressFamily {
            04 03 0001 02                  addressFamily: IPv4 Multicast
                                           IPAddressChoice
            05 00                            inherit from issuer
                                         } -- IPAddressFamily
         30 0f                           IPAddressFamily {
            04 02 0002                     addressFamily: IPv6
                                           IPAddressChoice
            30 09                            addressesOrRanges {
                                              IPAddressOrRange
               03 07 00 200100000002          addressPrefix    2001:0:2/47
                                            } -- addressesOrRanges
                                         } -- IPAddressFamily
                                      } -- IPAddrBlocks
                                   } -- extnValue
                                } -- Extension
```

Appendix C -- Example of an AS Identifier Delegation Extension

   An extension that specifies AS numbers 135, 3000 to 3999, and 5001,
   and which inherits all routing domain identifiers from the issuer's
   certificate would be (in hexadecimal):

```
30 2b                              Extension {
   06 08 2b06010505070108    extnID         1.3.6.1.5.5.7.1.8
   01 01 ff                       critical
   04 1c                          extnValue {
      30 1a                          ASIdentifiers {
         a0 14                           asnum
                                           ASIdentifierChoice
            30 12                            asIdsOrRanges {
                                               ASIdOrRange
               02 02 0087                        ASId
                                               ASIdOrRange
               30 08                             ASRange {
                  02 02 0bb8                        min
                  02 02 0f9f                        max
                                                 } -- ASRange
                                               ASIdOrRange
               02 02 1389                        ASId
                                             } -- asIdsOrRanges
                                         } -- asnum
         a1 02                           rdi {
                                           ASIdentifierChoice
            05 00                            inherit from issuer
                                         } -- rdi
                                     } -- ASIdentifiers
                                 } -- extnValue
                             } -- Extension
```

Appendix D -- Use of X.509 Attribute Certificates

   This appendix discusses issues arising from a proposal to use
   attribute certificates (ACs, as specified in [RFC3281]) to convey,
   from the Regional Internet Registries (RIRs) to the end-user
   organizations, the "right-to-use" for IP address blocks or AS
   identifiers.

   The two resources, AS identifiers and IP address blocks, are
   currently managed differently.  All organizations with the right-to-
   use for an AS identifier receive the authorization directly from an
   RIR.  Organizations with a right-to-use for an IP address block
   receive the authorization either directly from an RIR, or indirectly,
   e.g., from a down stream service provider, who might receive its

authorization from an Internet service provider, who in turn gets its
authorization from a RIR.  Note that AS identifiers might be sub-
allocated in the future, so the mechanisms used should not rely upon
a three level hierarchy.

In section 1 of RFC 3281, two reasons are given for why the use of
ACs might be preferable to the use of public key certificates (PKCs)
with extensions that convey the authorization information:

   "Authorization information may be placed in a PKC extension or
   placed in a separate attribute certificate (AC).  The placement of
   authorization information in PKCs is usually undesirable for two
   reasons.  First, authorization information often does not have the
   same lifetime as the binding of the identity and the public key.
   When authorization information is placed in a PKC extension, the
   general result is the shortening of the PKC useful lifetime.
   Second, the PKC issuer is not usually authoritative for the
   authorization information.  This results in additional steps for
   the PKC issuer to obtain authorization information from the
   authoritative source."

   "For these reasons, it is often better to separate authorization
   information from the PKC.  Yet, authorization information also
   needs to be bound to an identity.  An AC provides this binding; it
   is simply a digitally signed (or certified) identity and set of
   attributes."

In the case of the IP address and AS identifier authorizations, these
reasons do not apply.  First, the public key certificates are issued
exclusively for authorization, so the certificate lifetime
corresponds exactly to the authorization lifetime, which is often
tied to a contractual relationship between the issuer and entity
receiving the authorization.  The Subject and Issuer names are only
used for chaining during certification path validation, and the names
need not correspond to any physical entity.  The Subject name in the
PKCs may actually be randomly assigned by the issuing CA, allowing
the resource holder limited anonymity.  Second, the certificate
hierarchy is constructed so that the certificate issuer is
authoritative for the authorization information.

Thus the two points in the first cited paragraph above are not true
in the case of AS number and IP address block allocations.  The point
of the second cited paragraph is also not applicable as the resources
are not being bound to an identity but to the holder of the private
key corresponding to the public key in the PKC.

RFC 3281 specifies several requirements that a conformant Attribute
Certificate must meet.  In relation to S-BGP, the more-significant
requirements are:

1   from section 1: "this specification does NOT RECOMMEND the use of
    AC chains.  Other (future) specifications may address the use of
    AC chains."

    Allocation from IANA to RIRs to ISPs to DSPs and assignment to end
    organizations would require the use of chains, at least for IP
    address blocks.  A description of how the superior's AC should be
    located and how it should be processed would have to be provided.
    Readers of this document are encouraged to propose ways the
    chaining might be avoided.

2   from section 4.2.9: "section 4.3 defines the extensions that MAY
    be used with this profile, and whether or not they may be marked
    critical.  If any other critical extension is used, the AC does
    not conform to this profile.  However, if any other non-critical
    extension is used, the AC does conform to this profile."

    This means that the delegation extensions defined in this
    specification, which are critical, could not be simply placed into
    an AC.  They could be used if not marked critical, but the
    intended use requires that the extensions be critical so that the
    certificates containing them cannot be used as identity
    certificates by an unsuspecting application.

3   from section 4.5: "an AC issuer, MUST NOT also be a PKC issuer.
    That is, an AC issuer cannot be a CA as well."

    This means that for each AC issuer there would need to be a
    separate CA to issue the PKC that contains the public key of the
    AC holder.  The AC issuer cannot issue the PKC of the holder, and
    the PKC issuer cannot sign the AC.  Thus, each entity in the PKI
    would need to operate an AC issuer in addition to its CA.  There
    would be twice as many certificate issuers and CRLs to process to
    support Attribute certificates than are needed if PKCs are used.
    The possibility of mis-alignment also arises when there are two
    issuers issuing certificates for a single purpose.

    The AC model of RFC 3281 implies that the AC holder presents the
    AC to the AC verifier when the holder wants to substantiate an
    attribute or authorization.  The intended usage for the extensions
    defined herein does not have a direct interaction between an AC
    verifier (a NOC) and the AC issuers (all RIRs and NOCs).  Given a

signature on a claimed right-to-use object, the "AC verifier" can
locate the AC holder's PKC, but there is no direct way to locate
the Subject's AC(s).

4   from section 5: "4.  The AC issuer MUST be directly trusted as an
    AC issuer (by configuration or otherwise)."

    This is not true in the case of a right-to-use for an IP address
    block, which is allocated through a hierarchy.  Certification path
    validation of the AC will require chaining up through the
    delegation hierarchy.  Having to configure each relying party
    (NOC) to "trust" every other NOC does not scale, and such "trust"
    has resulted in failures that the proposed security mechanisms are
    designed to prevent.  A single PKI with a trusted root is used,
    not thousands of individually trusted per-ISP AC issuers.

    The amount of work that would be required to properly validate an
    AC is larger than for the mechanism that places the certificate
    extensions defined in this document in the PKCs.  There would be
    twice as many certificates to be validated, in addition to the
    ACs.  There could be a considerable increase in the management
    burden required to support ACs.

References

Normative References

   [IANA-AFI]   http://www.iana.org/assignments/address-family-numbers.

   [IANA-SAFI] http://www.iana.org/assignments/safi-namespace.

   [RFC2119]    Bradner, S., "Key words for use in RFCs to Indicate
                Requirement Level", BCP 14, RFC 2119, March 1997.

   [RFC3280]    Housley, R., Polk, W., Ford, W. and D. Solo, "Internet
                X.509 Public Key Infrastructure Certificate and
                Certificate Revocation List (CRL) Profile", RFC 3280,
                April 2002.

   [X.690]      ITU-T Recommendation X.690 (1997) | ISO/IEC 8825-1:1998,
                "Information Technology - ASN.1 Encoding Rules:
                Specification of Basic Encoding Rules (BER), Canonical
                Encoding Rules (CER) and Distinguished Encoding Rules
                (DER)".

Informational References

    [RFC791]        Postel, J., "Internet Protocol -- DARPA Internet Program
                    Protocol Specification", RFC 791, September 1981.

    [RFC1142]       D. Oran, Ed., "OSI IS-IS Intra-domain Routing Protocol",
                    RFC 1142, February 1990.

    [RFC1771]       Rekhter, Y. and T. Li, Eds., "A Border Gateway Protocol 4
                    (BGP-4)", RFC 1771, March 1995.

    [RFC1930]       Hawkinson, J. and T. Bates, "Guidelines for creation,
                    selection, and registration of an Autonomous System
                    (AS)", BCP 6, RFC 1930, March 1996.

    [RFC2050]       Hubbard, K., Kosters, M., Conrad, D., Karrenberg, D. and
                    J. Postel, "Internet Registry IP Allocation Guidelines",
                    BCP 12, RFC 2050, November 1996.

    [RFC3513]       Hinden, R. and S. Deering, "Internet Protocol Version 6
                    (IPv6) Addressing Architecture", RFC 3513, April 2003.

    [RFC3281]       Farrell, S. and R. Housley, "An Internet Attribute
                    Certificate Profile for Authorization", RFC 3281, April
                    2002.

    [S-BGP]         S. Kent, C. Lynn, and K. Seo, "Secure Border Gateway
                    Protocol (S-BGP)," IEEE JSAC Special Issue on Network
                    Security, April 2000.

Authors' Address

    Charles Lynn
    BBN Technologies
    10 Moulton St.
    Cambridge, MA 02138
    USA

    Phone: +1 (617) 873-3367
    EMail: CLynn@BBN.Com


    Stephen Kent
    BBN Technologies
    10 Moulton St.
    Cambridge, MA 02138
    USA

    Phone: +1 (617) 873-3988
    EMail: Kent@BBN.Com


    Karen Seo
    BBN Technologies
    10 Moulton St.
    Cambridge, MA 02138
    USA

    Phone: +1 (617) 873-3152
    EMail: KSeo@BBN.Com

Full Copyright Statement

Intellectual Property

Acknowledgement