              Clearance Attribute and Authority Clearance Constraints
                           Certificate Extension

Abstract

   This document defines the syntax and semantics for the Clearance
   attribute and the Authority Clearance Constraints extension in X.509
   certificates. The Clearance attribute is used to indicate the
   clearance held by the subject. The Clearance attribute may appear in
   the subject directory attributes extension of a public key
   certificate or in the attributes field of an attribute certificate.
   The Authority Clearance Constraints certificate extension values in a
   Trust Anchor (TA), in Certification Authority (CA) public key
   certificates, and in an Attribute Authority (AA) public key
   certificate in a certification path for a given subject constrain the
   effective Clearance of the subject.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc5913.

to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1. Introduction

   Organizations that have implemented a security policy can issue
   certificates that include an indication of the clearance values held
   by the subject. The Clearance attribute indicates the security
   policy, the clearance levels held by the subject, and additional
   authorization information held by the subject. This specification
   makes use of the ASN.1 syntax for clearance from [RFC5912].

   The Clearance attribute may be placed in the subject directory
   attributes extension of a Public Key Certificate (PKC) or may be
   placed in a separate attribute certificate (AC).

   The placement of the Clearance attribute in PKCs is suitable 1) when
   the clearance information is relatively static and can be verified as
   part of the PKC issuance process (e.g., using local databases) or 2)
   when the credentials such as PKCs need to be revoked when the
   clearance information changes. The Clearance attribute may also be
   included to simplify the infrastructure, to reduce the infrastructure
   design cost, or to reduce the infrastructure operations cost. An
   example of placement of the Clearance attribute in PKCs in
   operational Public Key Infrastructure (PKI) is the Defense Messaging
   Service. An example of placement of attributes in PKCs is Qualified
   Certificates [RFC3739].

   The placement of Clearance attributes in ACs is desirable when the
   clearance information is relatively dynamic and changes in the
   clearance information do not require revocation of credentials such
   as PKCs, or the clearance information cannot be verified as part of
   the PKC issuance process.

   Since [RFC5755] does not permit a chain of ACs, the Authority
   Clearance Constraints extension may only appear in the PKCs of a
   Certification Authority (CA) or Attribute Authority (AA). The
   Authority Clearance Constraints extension may also appear in a trust
   anchor (TA) or may be associated with a TA.

   Some organizations have multiple TAs, CAs, and/or AAs, and these
   organizations may wish to indicate to relying parties which clearance
   values from a particular TA, CA, or AA should be accepted. For
   example, consider the security policies described in [RFC3114], where
   a security policy has been defined for Amoco with three security
   classification values (HIGHLY CONFIDENTIAL, CONFIDENTIAL, and
   GENERAL). To constrain a CA for just one security classification,
   the Authority Clearance Constraints certificate extension would be
   included in the CA's PKC.

Cross-certified domains can also make use of the Authority Clearance
Constraints certificate extension to indicate which clearance values
should be acceptable to relying parties.

This document augments the certification path validation rules for
PKCs (in [RFC5280]) and ACs (in [RFC5755]).

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in [RFC2119].

1.2. ASN.1 Syntax Notation

All X.509 PKC [RFC5280] extensions are defined using ASN.1 [X.680].
All X.509 AC [RFC5755] extensions are defined using ASN.1 [X.680].
Note that [X.680] is the 2002 version of ASN.1, which is the most
recent version with freeware compiler support.

2. Clearance Attribute

The Clearance attribute in a certificate indicates the clearances
held by the subject. It uses the clearance attribute syntax, whose
semantics are defined in [RFC5755], in the Attributes field. A
certificate MUST include either zero or one instance of the Clearance
attribute. If the Clearance attribute is present, it MUST contain a
single value.

The following object identifier identifies the Clearance attribute
(either in the subject directory attributes extension of a PKC or in
the Attributes field of an AC):

```
   id-at-clearance OBJECT IDENTIFIER ::= { joint-iso-ccitt(2)
     ds(5) attributeTypes(4) clearance(55) }
```

The ASN.1 syntax for the Clearance attribute is defined in [RFC5912]
and that RFC provides the normative definition. The ASN.1 syntax for
Clearance attribute is as follows:

```
   Clearance  ::=  SEQUENCE {
     policyId             OBJECT IDENTIFIER,
     classList            ClassList DEFAULT {unclassified},
     securityCategories   SET OF SecurityCategory
                               {{ SupportedSecurityCategories }} OPTIONAL
   }
```

```
     ClassList  ::=  BIT STRING {
       unmarked       (0),
       unclassified   (1),
       restricted     (2),
       confidential   (3),
       secret         (4),
       topSecret      (5)
     }

     SECURITY-CATEGORY ::= TYPE-IDENTIFIER

     SecurityCategory { SECURITY-CATEGORY:Supported }::= SEQUENCE {
       type  [0] IMPLICIT SECURITY-CATEGORY.&id({Supported}),
       value [1] EXPLICIT SECURITY-CATEGORY.&Type
                                    ({Supported}{@type})
     }
```

   NOTE: SecurityCategory is shown exactly as it is in [RFC5912]. That
   module is an EXPLICIT tagged module, whereas the module contained in
   this document is an IMPLICIT tagged module.

   The Clearance attribute takes its meaning from Section 4.4.6 of
   [RFC5755], which is repeated here for convenience:

     - policyId identifies the security policy to which the clearance
       relates. The policyId indicates the semantics of the classList
       and securityCategories fields.

     - classList identifies the security classifications. Six basic
       values are defined in bit positions 0 through 5, and more may be
       defined by an organizational security policy.

     - securityCategories provides additional authorization information.

   If a trust anchor's public key is used directly, then the Clearance
   associated with the trust anchor, if any, should be used as the
   effective clearance (also defined as effective-clearance for a
   certification path).

3. Authority Clearance Constraints Certificate Extension

   The Authority Clearance Constraints certificate extension indicates
   to the relying party what clearances should be acceptable for the
   subject of the AC or the subject of the last certificate in a PKC
   certification path. It is only meaningful in a trust anchor, a CA
   PKC, or an AA PKC. A trust anchor, CA PKC, or AA PKC MUST include

   either zero or one instance of the Authority Clearance Constraints
   certificate extension. The Authority Clearance Constraints
   certificate extension MAY be critical or non-critical.

   Absence of this certificate extension in a TA, a CA PKC, or an AA PKC
   indicates that clearance of the subject of the AC or the subject of
   the last certificate in a PKC certification path containing the TA,
   the CA, or the AA is not constrained by the respective TA, CA, or AA.

   The following object identifier identifies the Authority Clearance
   Constraints certificate extension:

      id-pe-authorityClearanceConstraints OBJECT IDENTIFIER ::= {
         iso(1) identified-organization(3) dod(6) internet(1) security(5)
         mechanisms(5) pkix(7) pe(1) 21 }

   The ASN.1 syntax for the Authority Clearance Constraints certificate
   extension is as follows:

      AuthorityClearanceConstraints ::= SEQUENCE SIZE (1..MAX) OF
                                               Clearance

   The syntax for the Authority Clearance Constraints certificate
   extension contains Clearances that the CA or the AA asserts. The
   sequence MUST NOT include more than one entry with the same policyId.
   This constraint is enforced during Clearance and Authority Clearance
   Constraints Processing as described below. If more than one entry
   with the same policyId is present in the Authority Clearance
   Constraints certificate extension, the certification path is
   rejected.

4. Processing of Clearance and Authority Clearance Constraints in a PKC

    This section describes the certification path processing when
    Clearance is asserted in the PKC under consideration.

    User input, the Authority Clearance Constraints certificate
    extension, and Clearance attribute processing determines the
    effective clearance (henceforth called effective-clearance) for the
    end PKC. User input and the Authority Clearance Constraints
    certificate extension in the TA and in each PKC (up to but not
    including the end PKC) in a PKC certification path impact the
    effective-clearance. If there is more than one path to the end PKC,
    each path is processed independently. The process involves two
    steps:

         1) collecting the Authority Clearance Constraints; and

         2) using the Authority Clearance Constraints in the certification
            path and the Clearance in the end PKC to determine the
            effective-clearance for the subject of the end PKC.

   Assuming a certification path consisting of n PKCs, the effective-
   clearance for the subject of the end PKC is the intersection of 1)
   the Clearance attribute in the subject PKC, 2) the Authority
   Clearance Constraints, if present, in the trust anchor, 3) user
   input, and 4) all Authority Clearance Constraints present in n-1
   intermediate PKCs. Any effective-clearance calculation algorithm
   that performs this calculation and provides the same outcome as the
   one from the algorithm described herein is considered compliant with
   the requirements of this RFC.

   When processing a certification path, Authority Clearance Constraints
   are maintained in one state variable: permitted-clearances. When
   processing begins, permitted-clearances is initialized to the user
   input value or the special value all-clearances if Authority
   Clearance Constraints user input is not provided. The permitted-
   clearances state variable is updated by first processing Authority
   Clearance Constraints associated with the trust anchor, and then each
   time an intermediate PKC that contains an Authority Clearance
   Constraints certificate extension in the path is processed.

   When processing the end PKC, the value in the Clearance attribute in
   the end PKC is intersected with the permitted-clearances state
   variable.

   The output of Clearance attribute and Authority Clearance Constraint
   certificate extension processing is the effective-clearance (which
   could also be an empty list), and a status indicator of either
   success or failure. If the status indicator is failure, then the
   process also returns a reason code.

4.1. Collecting Constraints

   Authority Clearance Constraints are collected from the user input,
   the trust anchor, and the intermediate PKCs in a certification path.

4.1.1. Certification Path Processing

   When processing Authority Clearance Constraints certificate
   extensions for the purposes of validating a Clearance attribute in
   the end PKC, the processing described in this section or an
   equivalent algorithm MUST be performed in addition to the
   certification path validation.

The processing is presented as an addition to the certification path validation algorithm described in Section 6 of [RFC5280]. Note that this RFC is fully consistent with [RFC5280]; however, it augments [RFC5280] with the following steps:

   o Ability to provide and process Authority Clearance Constraints
      as an additional input to the certification path processing
      engine with Trust anchor information.

   o Requirement to process Authority Clearance Constraints present
      with trust anchor information.

4.1.1.1. Inputs

   User input may include an Authority Clearance Constraints structure
   or omit it.

   Trust anchor information may include the Authority Clearance
   Constraints structure to specify Authority Clearance Constraints for
   the trust anchor. In other words, the trust anchor may be
   constrained or unconstrained.

4.1.1.2. Initialization

   If the user input includes Authority Clearance Constraints, set
   permitted-clearances to the input value; otherwise, set permitted-
   clearances to the special value all-clearances.

   Examine the permitted-clearances for the same Policy ID appearing
   more then once. If a policyId appears more than once in the
   permitted-clearances state variable, set effective-clearance to an
   empty list, set error code to "multiple instances of same clearance",
   and exit with failure.

   If the trust anchor does not contain an Authority Clearance
   Constraints extension, continue at Section 4.1.1.3. Otherwise,
   execute the procedure described in Section 6 as an in-line macro by
   treating the trust anchor as a PKC.

4.1.1.3. Basic Certificate Processing

   If the PKC is the last PKC (i.e., certificate n), skip the steps
   listed in this section. Otherwise, execute the procedure described
   in Section 6 as an in-line macro.

4.1.1.4. Preparation for Certificate i+1

   No additional action associated with the Clearance attribute or the
   Authority Clearance Constraints certificate extensions is taken
   during this phase of certification path validation as described in
   Section 6 of [RFC5280].

4.1.1.5. Wrap-up Procedure

   To complete the processing, perform the following steps for the last
   PKC (i.e., certificate n).

   Examine the PKC and verify that it does not contain more than one
   instance of the Clearance attribute. If the PKC contains more than
   one instance of the Clearance attribute, set effective-clearance to
   an empty list, set the error code to "multiple instances of an
   attribute", and exit with failure.

   If the Clearance attribute is not present in the end PKC, set
   effective-clearance to an empty list and exit with success.

   Set effective-clearance to the Clearance attribute in the end PKC.

4.1.1.5.1. Wrap Up Clearance

   Examine effective-clearance and verify that it does not contain more
   than one value. If effective-clearance contains more than one value,
   set effective-clearance to an empty list, set error code to "multiple
   values", and exit with failure.

   If permitted-clearances is an empty list, set effective-clearance to
   an empty list and exit with success.

   If permitted-clearances has the special value all-clearances, exit
   with success.

   Let us say policyId in effective-clearance is X.

   If the policyId X in effective-clearance is absent from the
   permitted-clearances, set effective-clearance to an empty list and
   exit with success.

   Assign those classList bits in effective-clearance a value of one (1)
   that have a value of one (1) both in effective-clearance and in the
   clearance structure in permitted-clearances associated with policyId
   X. Assign all other classList bits in effective-clearance a value of
   zero (0).

   If none of the classList bits have a value of one (1) in effective-
   clearance, set effective-clearance to an empty list and exit with
   success.

   Set the securityCategories in effective-clearance to the intersection
   of securityCategories in effective-clearance and securityCategories
   for policyId X in permitted-clearances using the algorithm described
   in Section 7. Note that an empty SET is represented by simply
   omitting the SET.

   Exit with success.

4.1.1.6. Outputs

   If certification path validation processing succeeds, effective-
   clearance contains the subject's effective clearance for this
   certification path. Processing also returns success or failure
   indication and reason for failure, if applicable.

5. Clearance and Authority Clearance Constraints Processing in AC

   This section describes the certification path processing when
   Clearance is asserted in an AC. Relevant to processing are: one TA;
   0 or more CA PKCs; 0 or 1 AA PKC; and 1 AC.

   User input, Authority Clearance Constraints certificate extension,
   and Clearance attribute processing determine the effective clearance
   (henceforth called effective-clearance) for the subject of the AC.
   User input and the Authority Clearance Constraints certificate
   extensions in the TA and in each PKC (up to and including the AA PKC)
   in a certification path impact the effective-clearance. If there is
   more than one path to the AA PKC, each path is processed
   independently. The process involves two steps:

     1) collecting the Authority Clearance Constraints; and

     2) using the Authority Clearance Constraints in the PKC
        certification path and the Clearance in the AC to determine the
        effective-clearance for the subject of the AC.

   The effective-clearance for the subject of the AC is the intersection
   of 1) the Clearance attribute in the subject AC, 2) the Authority
   Clearance Constraints, if present, in trust anchor, 3) user input,
   and 4) all Authority Clearance Constraints present in the PKC
   certification path from the TA to the AA. Any effective-clearance
   calculation algorithm that performs this calculation and provides the
   same outcome as the one from the algorithm described herein is
   considered compliant with the requirements of this RFC.

The Authority Clearance Constraints are maintained in one state variable: permitted-clearances. When processing begins, permitted-clearances is initialized to user input or the special value all-clearances if Authority Clearance Constraints user input is not provided. The permitted-clearances state variable is updated by first processing the Authority Clearance Constraints associated with the trust anchor, and then each time a PKC (other than AC holder PKC) that contains an Authority Clearance Constraints certificate extension in the path is processed.

When processing the AC, the value in the Clearance attribute in the AC is intersected with the permitted-clearances state variable.

The output of Clearance attribute and Authority Clearance Constraint certificate extension processing is the effective-clearance, which could also be an empty list; and success or failure with a reason code for failure.

5.1. Collecting Constraints

Authority Clearance Constraints are collected from the user input, the trust anchor, and all the PKCs in the AA PKC certification path.

5.1.1. Certification Path Processing

When processing Authority Clearance Constraints certificate extensions for the purpose of validating a Clearance attribute in the AC, the processing described in this section or an equivalent algorithm MUST be performed in addition to the certification path validation. The processing is presented as an addition to the PKC certification path validation algorithm described in Section 6 of [RFC5280] for the AA PKC certification path and the algorithm described in Section 5 of [RFC5755] for the AC validation. Also see the note related to [RFC5280] augmentation in Section 4.1.1.

5.1.1.1. Inputs

Same as Section 4.1.1.1.

In addition, let us assume that the PKC certification path for the AA consists of n certificates.

5.1.1.2. Initialization

Same as Section 4.1.1.2.

5.1.1.3. Basic PKC Processing

   Same as Section 4.1.1.3 except that the logic is applied to all n
   PKCs.

5.1.1.4. Preparation for Certificate i+1

   Same as Section 4.1.1.4.

5.1.1.5. Wrap-up Procedure

   To complete the processing, perform the following steps for the AC.

   Examine the AC and verify that it does not contain more than one
   instance of the Clearance attribute. If the AC contains more than
   one instance of the Clearance attribute, set effective-clearance to
   an empty list, set the error code to "multiple instances of an
   attribute", and exit with failure.

   If the Clearance attribute is not present in the AC, set effective-
   clearance to an empty list and exit with success.

   Set effective-clearance to the Clearance attribute in the AC.

5.1.1.5.1. Wrap Up Clearance

   Same as Section 4.1.1.5.1.

5.1.1.6. Outputs

   Same as Section 4.1.1.6.

   In addition, apply AC processing rules described in Section 5 of
   [RFC5755].

6. Computing the Intersection of permitted-clearances and Authority
    Clearance Constraints Extension

   Examine the PKC and verify that it does not contain more than one
   instance of the Authority Clearance Constraints extension. If the
   PKC contains more than one instance of Authority Clearance
   Constraints extension, set effective-clearance to an empty list, set
   error code to "multiple extension instances", and exit with failure.

   If the Authority Clearance Constraints certificate extension is not
   present in the PKC, no action is taken, and the permitted-clearances
   value is unchanged.

If the Authority Clearance Constraints certificate extension is
present in the PKC, set the variable temp-clearances to the value of
the Authority Clearance Constraints certificate extension. Examine
the temp-clearances for the same Policy ID appearing more then once.
If a policyId appears more than once in the temp-clearances state
variable, set effective-clearance to an empty list, set error code to
"multiple instances of same clearance", and exit with failure.

If the Authority Clearance Constraints certificate extension is
present in the PKC and permitted-clearances contains the all-
clearances special value, then assign permitted-clearances the value
of temp-clearances.

If the Authority Clearance Constraints certificate extension is
present in the PKC and permitted-clearances does not contain the all-
clearances special value, take the intersection of temp-clearances
and permitted-clearances by repeating the following steps for each
clearance in the permitted-clearances state variable:

  - If the policyId associated with the clearance is absent in the
    temp-clearances, delete the clearance structure associated with
    the policyID from the permitted-clearances state variable.

  - If the policyId is present in temp-clearances:

    -- For every classList bit, assign the classList bit a value of
       one (1) for the policyId in the permitted-clearances state
       variable if the bit is one (1) in both the permitted-
       clearances state variable and the temp-clearances for that
       policyId; otherwise, assign the bit a value of zero (0).

    -- If no bits are one (1) for the classList, delete the clearance
       structure associated with the policyId from the permitted-
       clearances state variable and skip the next step of processing
       securityCategories.

    -- For the policyId in permitted-clearances, set the
       securityCategories to the intersection of securityCategories
       for the policyId in permitted-clearances and in temp-
       clearances using the algorithm described in Section 7. Note
       that an empty SET is represented by simply omitting the SET.

7. Computing the Intersection of securityCategories

   The algorithm described here has the idempotent, associative, and
   commutative properties.

This section describes how to compute the intersection of
securityCategories A and B. It uses the state variable temp-set. It
also uses temporary variables X and Y.

Set the SET temp-set to empty.

Set X = A and Y = B.

If SET X is empty (i.e., securityCategories is absent), return temp-
set.

If SET Y is empty (i.e., securityCategories is absent), return temp-
set.

For each type OID in X, if all the elements for the type OID in X and
if and only if all the elements for that type OID in Y are identical,
add those elements to temp-set and delete those elements from X and
Y. Note: identical means that if the element with the type OID and
given value is present in X, it is also present in Y with the same
type OID and given value and vice versa. Delete the elements from X
and from Y.

If SET X is empty (i.e., securityCategories is absent), return temp-
set.

If SET Y is empty (i.e., securityCategories is absent), return temp-
set.

For every element (i.e., SecurityCategory) in the SET X, carry out
the following steps:

   1. If there is no element in SET Y with the same type OID as the
      type OID in the element from SET X, go to step 5.

   2. If there is an element in SET Y with the same type OID and value
      as in the element in SET X, carry out the following steps:

      a) If the element is not present in the SET temp-set, add an
         element containing the type OID and the value to the SET
         temp-set.

   3. If the processing semantics of type OID in the element in SET X
      is not known, go to step 5.

   4. For each element in SET Y, do the following:

      a) If the type OID of the element in SET Y is not the same as
         the element in SET X being processed, go to step 4.d.

       b) Perform type-OID-specific intersection of the value in the
          element in SET X with the value in the element in SET Y.

       c) If the intersection is not empty, and the element
          representing the type OID and intersection value is not
          already present in temp-set, add the element containing the
          type OID and intersection value as an element to temp-set.

       d) Continue to the next element in SET Y.

     5. If more elements remain in SET X, process the next element
        starting with step 1.

   Return temp-set.

8. Recommended securityCategories

   This RFC also includes a recommended securityCategories object as
   follows:

   recommended-category SECURITY-CATEGORY ::=
     { BIT STRING IDENTIFIED BY OID }

   The above structure is provided as an example. To use this
   structure, the object identifier (OID) needs to be registered and the
   semantics of the bits in the bit string need to be enumerated.

   Note that type-specific intersection of two values for this type will
   be simply setting the bits that are set in both values. If the
   resulting intersection has none of the bits set, the intersection is
   considered empty.

9. Security Considerations

   Certificate issuers must recognize that absence of the Authority
   Clearance Constraints in a TA, in a CA certificate, or in an AA
   certificate means that in terms of the clearance, the subject
   Authority is not constrained.

   Absence of the Clearance attribute in a certificate means that the
   subject has not been assigned any clearance.

   If there is no Clearance associated with a TA, it means that the TA
   has not been assigned any clearance.

   If the local security policy considers the clearance held by a
   subject or those supported by a CA or AA to be sensitive, then the
   Clearance attribute or Authority Clearance Constraints should only be

included if the subject's and Authority's certificates can be privacy
protected. Also in this case, distribution of trust anchors and
associated Authority Clearance Constraints extension or Clearance
must also be privacy protected.

10. References

10.1. Normative References

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

   [RFC5280] Cooper, D. et. al., "Internet X.509 Public Key
              Infrastructure Certificate and Certification Revocation
              List (CRL) Profile", RFC 5280, May 2008.

   [RFC5755] Farrell, S., Housley, R., and S. Turner, "An Internet
              Attribute Certificate Profile for Authorization", RFC
              5755, January 2010.

   [RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the
              Public Key Infrastructure Using X.509 (PKIX) RFC 5912,
              June 2010.

   [X.680]   ITU-T Recommendation X.680 (2002) | ISO/IEC 8824-1:2002.
              Information Technology - Abstract Syntax Notation One.

10.2. Informative References

   [RFC3114] Nicolls, W., "Implementing Company Classification Policy
              with the S/MIME Security Label", RFC 3114, May 2002.

   [RFC3739] Santesson, S., Nystrom, M., and T. Polk, "Internet X.509
              Public Key Infrastructure: Qualified Certificates
              Profile", RFC 3739, March 2004.

Appendix A. ASN.1 Module

   This appendix provides the normative ASN.1 definitions for the
   structures described in this specification using ASN.1 as defined in
   X.680.

   ClearanceConstraints { iso(1) identified-organization(3) dod(6)
   internet(1) security(5) mechanisms(5) pkix(7) mod(0) 46 }

   DEFINITIONS IMPLICIT TAGS ::=

   BEGIN

   -- EXPORTS ALL --

   IMPORTS

   -- IMPORTS from [RFC5912]

   id-at-clearance, Clearance
      FROM PKIXAttributeCertificate-2009
      { iso(1) identified-organization(3) dod(6) internet(1)
        security(5) mechanisms(5) pkix(7) id-mod(0)
        id-mod-attribute-cert-02(47)
      }

   -- IMPORTS from [RFC5912]

   EXTENSION, SECURITY-CATEGORY
     FROM PKIX-CommonTypes-2009
      { iso(1) identified-organization(3) dod(6) internet(1)
        security(5) mechanisms(5) pkix(7) id-mod(0)
        id-mod-pkixCommon-02(57)
      }
   ;

   -- Clearance attribute OID and syntax

   -- The following is a 2002 ASN.1 version for clearance.
   -- It is included for convenience.

   -- id-at-clearance OBJECT IDENTIFIER ::=
   -- { joint-iso-ccitt(2) ds(5) attributeTypes(4) clearance (55) }

   -- Clearance ::= SEQUENCE {
   --   policyId   OBJECT IDENTIFIER,
   --   classList   ClassList DEFAULT {unclassified},
   --   securityCategories SET OF SecurityCategory

```
   --                              {{SupportSecurityCategories }} OPTIONAL
   -- }

   -- ClassList  ::=  BIT STRING {
   --   unmarked       (0),
   --   unclassified  (1),
   --   restricted     (2),
   --   confidential  (3),
   --   secret         (4),
   --   topSecret      (5)
   -- }

   -- SECURITY-CATEGORY ::= TYPE-IDENTIFIER

   -- NOTE that the module SecurityCategory is taken from a module
   -- that uses EXPLICIT tags [RFC5912].  If Clearance was not imported
   -- from [RFC5912] and the comments were removed from the ASN.1
   -- contained herein, then the IMPLICIT in type could also be removed
   -- with no impact on the encoding.

   -- SecurityCategory { SECURITY-CATEGORY:Supported } ::= SEQUENCE {
   --   type  [0] IMPLICIT SECURITY-CATEGORY.&id({Supported}),
   --   value [1] EXPLICIT SECURITY-CATEGORY.&Type
   --                                    ({Supported}{@type})
   -- }

   -- Authority Clearance Constraints certificate extension OID
   -- and syntax

   id-pe-clearanceConstraints OBJECT IDENTIFIER ::=
     { iso(1) identified-organization(3) dod(6) internet(1) security(5)
       mechanisms(5) pkix(7) pe(1) 21 }

   authorityClearanceConstraints EXTENSION ::= {
     SYNTAX          AuthorityClearanceConstraints
     IDENTIFIED BY   id-pe-clearanceConstraints
   }

   AuthorityClearanceConstraints ::= SEQUENCE SIZE (1..MAX) OF Clearance

   END
```

Authors' Addresses

    Sean Turner
    IECA, Inc.
    3057 Nutley Street, Suite 106
    Fairfax, VA 22031
    USA

    EMail: turners@ieca.com

    Santosh Chokhani
    CygnaCom Solutions, Inc.

    EMail: SChokhani@cygnacom.com