Internet Engineering Task Force (IETF)
Request for Comments: 6488
Category: Standards Track
ISSN: 2070-1721

M. Lepinski
A. Chi
S. Kent
BBN

February 2012

Signed Object Template for the Resource Public Key Infrastructure (RPKI)

Abstract

This document defines a generic profile for signed objects used in the Resource Public Key Infrastructure (RPKI). These RPKI signed objects make use of Cryptographic Message Syntax (CMS) as a standard encapsulation format.

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/rfc6488.

Table of Contents

1. Introduction

The purpose of the Resource Public Key Infrastructure (RPKI) is to support assertions by current resource holders of IP (v4 and v6) address space and AS numbers, based on the records of organizations that act as Certification Authorities (CAs). IP address and AS number resource information is carried in X.509 certificates via RFC 3779 extensions [RFC6487]. Other information assertions about resources are expressed via digitally signed, non-X.509 data structures that are referred to as "signed objects" in the RPKI context [RFC6480]. This document standardizes a template for specifying signed objects that can be validated using the RPKI.

RPKI signed objects make use of Cryptographic Message Syntax (CMS) [RFC5652] as a standard encapsulation format. CMS was chosen to take advantage of existing open source software available for processing messages in this format. RPKI signed objects adhere to a profile (specified in Section 2) of the CMS signed-data object.

The template defined in this document for RPKI signed objects is not a complete specification for any particular type of signed object, and instead includes only the items that are common to all RPKI signed objects. That is, fully specifying a particular type of signed object requires an additional document that specifies the details specific to a particular type of signed object. Such details include Abstract Syntax Notation One (ASN.1) [X.208-88] for the object's payload and any additional steps required to validate the particular type of signed object. Section 4 describes in more detail the additional pieces that must be specified in order to define a new type of RPKI signed object that uses this template. Additionally, see [RFC6482] for an example of a document that uses this template to specify a particular type of signed object, the Route Origination Authorization (ROA).

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" [RFC5280], "X.509 Extensions for IP Addresses and AS Identifiers" [RFC3779], and "Cryptographic Message Syntax (CMS)" [RFC5652].

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. Note on Algorithms

CMS is a general format capable of accommodating a wide variety of signature and digest algorithms. The algorithms used in the RPKI (and associated key sizes) are specified in [RFC6485].

2. Signed Object Syntax

The RPKI signed object is a profile of the CMS [RFC5652] signed-data object, with the restriction that RPKI signed objects MUST be encoded using the ASN.1 Distinguished Encoding Rules (DER) [X.509-88].

The general format of a CMS object is:

```
ContentInfo ::= SEQUENCE {
  contentType ContentType,
  content [0] EXPLICIT ANY DEFINED BY contentType }

ContentType ::= OBJECT IDENTIFIER
```

The content-type is the signed-data type of id-data, namely the id-signedData OID [RFC5652], 1.2.840.113549.1.7.2.

2.1. Signed-Data Content Type

According to the CMS standard, the signed-data content type is the ASN.1 type SignedData:

```
SignedData ::= SEQUENCE {
  version CMSVersion,
  digestAlgorithms DigestAlgorithmIdentifiers,
  encapContentInfo EncapsulatedContentInfo,
  certificates [0] IMPLICIT CertificateSet OPTIONAL,
  crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,
  signerInfos SignerInfos }

DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier

SignerInfos ::= SET OF SignerInfo
```

Additionally, the SignerInfos set MUST contain only a single SignerInfo object.

2.1.1. version

The version is the syntax version number. It MUST be 3, corresponding to the signerInfo structure having version number 3.

2.1.2. digestAlgorithms

The digestAlgorithms set contains the OIDs of the digest algorithm(s) used in signing the encapsulated content. This set MUST contain exactly one digest algorithm OID, and the OID MUST be selected from those specified in [RFC6485].

2.1.3. encapContentInfo

encapContentInfo is the signed content, consisting of a content type identifier and the content itself. The encapContentInfo represents the payload of the RPKI signed object.

```
EncapsulatedContentInfo ::= SEQUENCE {
  eContentType ContentType,
  eContent [0] EXPLICIT OCTET STRING OPTIONAL }

ContentType ::= OBJECT IDENTIFIER
```

2.1.3.1. eContentType

This field is left undefined by this profile. The eContentType is an OID specifying the type of payload in this signed object and MUST be specified by the Internet Standards Track document that defines the object.

2.1.3.2. eContent

This field is left undefined by this profile. The eContent is the payload of the signed object and MUST be specified by the Internet Standards Track document that defines the RPKI object.

Note that the signed object profile does not provide version numbers for signed objects. Therefore, in order to facilitate transition to new versions of the signed objects over time, it is RECOMMENDED that each type of signed object defined using this profile include a version number within its eContent.

2.1.4. certificates

The certificates field MUST be included, and MUST contain exactly one certificate, the RPKI end-entity (EE) certificate needed to validate this signed object.

2.1.5. crls

The crls field MUST be omitted.

2.1.6. signerInfos

SignerInfo is defined in CMS as:

```
SignerInfo ::= SEQUENCE {
  version CMSVersion,
  sid SignerIdentifier,
  digestAlgorithm DigestAlgorithmIdentifier,
  signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
  signatureAlgorithm SignatureAlgorithmIdentifier,
  signature SignatureValue,
  unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL }
```

2.1.6.1. version

The version number MUST be 3, corresponding with the choice of SubjectKeyIdentifier for the sid.

2.1.6.2. sid

The sid is defined as:

```
SignerIdentifier ::= CHOICE {
  issuerAndSerialNumber IssuerAndSerialNumber,
  subjectKeyIdentifier [0] SubjectKeyIdentifier }
```

For RPKI signed objects, the sid MUST be the SubjectKeyIdentifier that appears in the EE certificate carried in the CMS certificates field.

2.1.6.3. digestAlgorithm

The digestAlgorithm MUST consist of the OID of a digest algorithm that conforms to the RPKI Algorithms and Key Size Profile specification [RFC6485].

2.1.6.4. signedAttrs

The signedAttrs is defined as:

```
SignedAttributes ::= SET SIZE (1..MAX) OF Attribute

Attribute ::= SEQUENCE {
  attrType OBJECT IDENTIFIER,
  attrValues SET OF AttributeValue }

AttributeValue ::= ANY
```

The signedAttrs element MUST be present and MUST include the content-type and message-digest attributes [RFC5652]. The signer MAY also include the signing-time attribute [RFC5652], the binary-signing-time attribute [RFC6019], or both attributes. Other signed attributes MUST NOT be included.

The signedAttrs element MUST include only a single instance of any particular attribute. Additionally, even though the syntax allows for a SET OF AttributeValue, in an RPKI signed object, the attrValues MUST consist of only a single AttributeValue.

2.1.6.4.1. Content-Type Attribute

The content-type attribute MUST be present. The attrType OID for the content-type attribute is 1.2.840.113549.1.9.3.

The attrValues for the content-type attribute MUST match the eContentType in the EncapsulatedContentInfo. Thus, attrValues MUST contain the OID that specifies the payload type of the specific RPKI signed object carried in the CMS signed data structure.

2.1.6.4.2. Message-Digest Attribute

The message-digest attribute MUST be present. The attrType OID for the message-digest attribute is 1.2.840.113549.1.9.4.

The attrValues for the message-digest attribute contains the output of the digest algorithm applied to the content being signed, as specified in Section 5.4 of [RFC5652].

2.1.6.4.3. Signing-Time Attribute

The signing-time attribute MAY be present. Note that the presence or absence of the signing-time attribute MUST NOT affect the validity of the signed object (as specified in Section 3). The attrType OID for the signing-time attribute is 1.2.840.113549.1.9.5.

```
id-signingTime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 5 }
```

The attrValues for the signing-time attribute is defined as:

```
SigningTime ::= Time

Time ::= CHOICE {
  utcTime UTCTime,
  generalizedTime GeneralizedTime }
```

The Time element specifies the time, based on the local system clock, at which the digital signature was applied to the content.

The definition of Time matches the one specified in the 1997 version of X.509. Additional information regarding the use of UTCTime and GeneralizedTime can be found in [RFC5652].

2.1.6.4.4. Binary-Signing-Time Attribute

The binary-signing-time attribute MAY be present. Note that the presence or absence of the binary-signing-time attribute MUST NOT affect the validity of the signed object (as specified in Section 3). The attrType OID for the binary-signing-time attribute is 1.2.840.113549.1.9.16.2.46.

```
id-aa-binarySigningTime OBJECT IDENTIFIER ::= { iso(1)
    member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    smime(16) aa(2) 46 }
```

The attrValues for the signing-time attribute is defined as:

```
BinarySigningTime ::= BinaryTime

BinaryTime ::= INTEGER (0..MAX)
```

The BinaryTime element specifies the time, based on the local system clock, at which the digital signature was applied to the content. The precise definition of the BinaryTime element can be found in [RFC6019].

2.1.6.5. signatureAlgorithm

The signatureAlgorithm MUST conform to the RPKI Algorithms and Key Size Profile specification [RFC6485].

2.1.6.6. signature

The signature value is defined as:

```
SignatureValue ::= OCTET STRING
```

The signature characteristics are defined by the digest and signature algorithms.

2.1.6.7. unsignedAttrs

unsignedAttrs MUST be omitted.

3. Signed Object Validation

Before a relying party can use a signed object, the relying party MUST validate the signed object by verifying that all of the following conditions hold. A relying party may perform these checks in any order. Note that these checks are necessary, but not sufficient. In general, further validation MUST be performed based on the specific type of signed object.

1. The signed object syntax complies with this specification. In particular, each of the following is true:

   a. The content-type of the CMS object is SignedData (OID 1.2.840.113549.1.7.2)

   b. The version of the SignedData object is 3.

   c. The certificates field in the SignedData object is present and contains one EE certificate, the SubjectKeyIdentifier field of which matches the sid field of the SignerInfo object.

   d. The crls field in the SignedData object is omitted.

   e. The version of the SignerInfo is 3.

   f. The signedAttrs field in the SignerInfo object is present and contains both the content-type attribute (OID 1.2.840.113549.1.9.3) and the message-digest attribute (OID 1.2.840.113549.1.9.4).

   g. The signedAttrs field in the SignerInfo object does not contain any attributes other than the following four: the content-type attribute (OID 1.2.840.113549.1.9.3), the message-digest attribute (OID 1.2.840.113549.1.9.4), the signing-time attribute (OID 1.2.840.113549.1.9.5), and the binary-signing-time attribute (OID 1.2.840.113549.1.9.16.2.46). Note that the signing-time and binary-signing-time attributes MAY be present, but they are not required.

   h. The eContentType in the EncapsulatedContentInfo is an OID that matches the attrValues in the content-type attribute.

   i. The unsignedAttrs field in the SignerInfo object is omitted.

   j. The digestAlgorithm in the SignedData and SignerInfo objects conforms to the RPKI Algorithms and Key Size Profile specification [RFC6485].

   k. The signatureAlgorithm in the SignerInfo object conforms to the RPKI Algorithms and Key Size Profile specification [RFC6485].

   l. The signed object is DER encoded.

2. The public key of the EE certificate (contained within the CMS signed-data object) can be used to successfully verify the signature on the signed object.

3. The EE certificate (contained within the CMS signed-data object) is a valid EE certificate in the RPKI as specified by [RFC6487]. In particular, a valid certification path from a trust anchor to this EE certificate exists.

If the above procedure indicates that the signed object is invalid, then the signed object MUST be discarded and treated as though no signed object were present. If all of the conditions above are true, then the signed object may be valid. The relying party MUST then perform any additional validation steps required for the particular type of signed object.

Note that a previously valid signed object will cease to be valid when the associated EE certificate ceases to be valid (for example, when the end of the certificate's validity period is reached, or when the certificate is revoked by the authority that issued it). See [RFC6487] for a complete specification of resource certificate validity.

4. Definition of Specific Signed Objects

Each RPKI signed object MUST be defined in an Internet Standards Track document based on this profile, by specifying the following data elements and validation procedure:

1. eContentType: A single OID to be used for both the eContentType field and the content-type attribute. This OID uniquely identifies the type of signed object.

2. eContent: Define the syntax for the eContent field in encapContentInfo. This is the payload that contains the data specific to a given type of signed object.

3. Additional Validation: Define a set of additional validation steps for the specific signed object. Before using this specific signed object, a relying party MUST perform both the generic validation steps in Section 3 above, as well as these additional steps.

5. Security Considerations

There is no assumption of confidentiality for the data in an RPKI signed object. The integrity and authenticity of each signed object is based on the verification of the object's digital signature, and

the validation of the EE certificate used to perform that verification. It is anticipated that signed objects will be stored in repositories that will be publicly accessible.

Since RPKI signed objects make use of CMS as an encapsulation format, the security considerations for CMS apply [RFC5652].

6. IANA Considerations

IANA has created a registry of "RPKI Signed Objects" types that utilize the template defined in this document. This registry contains three fields: an informal name for the signed object, the OID for the eContentType of the signed object, and a specification pointer that references the RFC in which the signed object is specified. The entries in this registry are managed by IETF Standards Action.

The registry has been initially populated with the following two entries.

Name	OID	Specification
ROA	1.2.840.113549.1.9.16.1.24	RFC 6482
Manifest	1.2.840.113549.1.9.16.1.26	RFC 6486

7. Acknowledgements

The authors wish to thank Charles Gardiner, Russ Housley, and Derek Kong for their help and contributions. Additionally, the authors would like to thank Rob Austein, Roque Gagliano, Danny McPherson, Sean Turner, and Sam Weiler for their careful reviews and helpful comments.

8. Normative References

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

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, June 2004.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, September 2009.

[RFC6485] Huston, G., "The Profile for Algorithms and Key Sizes for Use in the Resource Public Key Infrastructure (RPKI)", RFC 6485, February 2012.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, February 2012.

[X.208-88] CCITT. Recommendation X.208: Specification of Abstract Syntax Notation One (ASN.1), 1988.

[X.509-88] CCITT. Recommendation X.509: The Directory Authentication Framework, 1988.

9. Informative References

[RFC6019] Housley, R., "BinaryTime: An Alternate Format for Representing Date and Time in ASN.1", RFC 6019, September 2010.

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, February 2012.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, February 2012.

Authors' Addresses

Matt Lepinski
BBN Technologies
10 Moulton Street
Cambridge MA 02138

EMail: mlepinski@bbn.com

Andrew Chi
BBN Technologies
10 Moulton Street
Cambridge MA 02138

EMail: achi@bbn.com

Stephen Kent
BBN Technologies
10 Moulton Street
Cambridge MA 02138

EMail: kent@bbn.com