Digital Signatures on Internet-Draft Documents

Status of This Memo

   This memo provides information for the Internet community. It does
   not specify an Internet standard of any kind. Distribution of this
   memo is unlimited.

Copyright Notice

Abstract

   This document specifies the conventions for digital signatures on
   Internet-Drafts. The Cryptographic Message Syntax (CMS) is used to
   create a detached signature, which is stored in a separate companion
   file so that no existing utilities are impacted by the addition of
   the digital signature.

1. Introduction

   This document specifies the conventions for storing a digital
   signature on Internet-Drafts. The Cryptographic Message Syntax (CMS)
   [CMS] is used to create a detached signature. The signature is
   stored in a separate companion file so that no existing utilities are
   impacted by the addition of the digital signature.

   Shortly after the IETF Secretariat posts the Internet-Draft in the
   repository, the digital signature is generated and posted as a
   companion file in the same repository. The digital signature allows
   anyone to confirm that the contents of the Internet-Draft have not
   been altered since the time that the document was posted in the
   repository.

   The signature of the IETF Secretariat is intended to provide a
   straightforward way for anyone to determine whether a particular file
   contains the document that was made available by the IETF
   Secretariat. The signing-time included by the IETF Secretariat
   provides the wall-clock time; it is not intended to provide a trusted
   timestamp.

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 RFC 2119 [STDWORDS].

1.2. ASN.1

   The CMS uses Abstract Syntax Notation One (ASN.1) [X.680]. ASN.1 is
   a formal notation used for describing data protocols, regardless of
   the programming language used by the implementation. Encoding rules
   describe how the values defined in ASN.1 will be represented for
   transmission. The Basic Encoding Rules (BER) [X.690] are the most
   widely employed rule set, but they offer more than one way to
   represent data structures. For example, both definite-length
   encoding and indefinite-length encoding are supported. This
   flexibility is not desirable when digital signatures are used. As a
   result, the Distinguished Encoding Rules (DER) [X.690] were invented.
   DER is a subset of BER that ensures a single way to represent a given
   value. For example, DER always employs definite-length encoding.

2. Internet-Draft Signature File

   All Internet-Draft file names begin with "draft-". The next portion
   of the file name depends on the source of the document. For example,
   documents from IETF working groups usually have "ietf-" followed by
   the working group abbreviation, and this is followed by a string that
   helps people figure out the subject of the document.

   All Internet-Draft file names end with a hyphen followed by a two-
   digit version number and a suffix. The suffix indicates the type of
   file. A plain text file with a suffix of ".txt" is required. Other
   formats may also be provided, and they employ the appropriate suffix
   for the file format.

   The companion signature file has exactly the same file name as the
   Internet-Draft, except that ".p7s" is added to the end. This file
   name suffix conforms to the conventions in [MSG]. Here are a few
   example names:

      Internet-Draft: draft-ietf-example-widgets-03.txt
      Signature File: draft-ietf-example-widgets-03.txt.p7s

      Internet-Draft: draft-ietf-example-widgets-03.ps
      Signature File: draft-ietf-example-widgets-03.ps.p7s

      Internet-Draft: draft-housley-internet-draft-sig-file-00.txt
      Signature File: draft-housley-internet-draft-sig-file-00.txt.p7s

   The IETF Secretariat will post the signature file in the repository
   shortly after the Internet-Draft is posted.

2.1. Need for Canonicalization

   In general, the content of the Internet-Draft is treated like a
   single octet string for the generation of the digital signature.
   Unfortunately, the plain text file requires canonicalization to avoid
   signature validation problems. The primary concern is the manner in
   which different operating systems indicate the end of a line of text.
   Some systems use a single new-line character, other systems use the
   combination of the carriage-return character followed by a line-feed
   character, and other systems use fixed-length records padded with
   space characters. For the digital signature to validate properly, a
   single convention must be employed.

2.2. Text File Canonicalization

   The canonicalization procedure follows the conventions used for text
   files in the File Transfer Protocol (FTP) [FTP]. Such files must be
   supported by FTP implementations, so code reuse seems likely.

   The canonicalization procedure converts the data from its internal
   character representation to the standard 8-bit NVT-ASCII
   representation (see TELNET [TELNET]). In accordance with the NVT
   standard, the <CRLF> sequence MUST be used to denote the end of a
   line of text. Using the standard NVT-ASCII representation means that
   data MUST be interpreted as 8-bit bytes.

   Trailing space characters MUST NOT appear on a line of text. That
   is, the space character must not be followed by the <CRLF> sequence.
   Thus, a blank line is represented solely by the <CRLF> sequence.

   The form-feed nonprintable character (0x0C) is expected in Internet-
   Drafts. Other nonprintable characters, such as tab and backspace,
   are not expected, but they do occur. For robustness, any
   nonprintable or non-ASCII characters (ones outside the range 0x20 to
   0x7E) MUST NOT be changed in any way not covered by the rules for
   end-of-line handling in the previous paragraph.

   Trailing blank lines MUST NOT appear at the end of the file. That
   is, the file must not end with multiple consecutive <CRLF> sequences.

   Any end-of-file marker used by an operating system is not considered
   to be part of the file content. When present, such end-of-file
   markers MUST NOT be processed by the digital signature algorithm.

   Note: This text file canonicalization procedure is consistent with
   the ASCII NVT definition offered in Appendix B of RFC 5198 [UFNI].

2.3. XML File Canonicalization

   In accordance with the guidance of the World Wide Web Consortium
   (W3C) in Section 2.11 of [R20060816], a <LF> character MUST be used
   to denote the end of a line of text within an XML file. Any two-
   character <CRLF> sequence and any <CR> that is not followed by <LF>
   are to be translated to a single <LF> character.

2.4. Canonicalization of Other File Formats

   No canonicalization is needed for file formats currently used for
   Internet-Drafts other than plain text files and XML files. Other
   file formats are treated as a simple sequence of octets by the
   digital signature algorithm.

3. CMS Profile

   CMS is used to construct the detached signature of the Internet-
   Draft. The CMS ContentInfo content type MUST always be present, and
   it MUST encapsulate the CMS SignedData content type. Since a
   detached signature is being created, the CMS SignedData content type
   MUST NOT encapsulate the Internet-Draft. The CMS detached signature
   is summarized by:

```
  ContentInfo {
    contentType          id-signedData, -- (1.2.840.113549.1.7.2)
    content              SignedData
  }

  SignedData {
    version              CMSVersion, -- Always set to 3
    digestAlgorithms     DigestAlgorithmIdentifiers,
    encapContentInfo     EncapsulatedContentInfo,
    certificates         CertificateSet, -- Secretariat certificate(s)
    crls                 CertificateRevocationLists, -- Optional
    signerInfos          SET OF SignerInfo -- Only one signer
  }

  SignerInfo {
    version              CMSVersion, -- Always set to 3
    sid                  SignerIdentifier,
    digestAlgorithm      DigestAlgorithmIdentifier,
    signedAttrs          SignedAttributes, -- Always present
    signatureAlgorithm   SignatureAlgorithmIdentifier,
    signature            SignatureValue,
    unsignedAttrs        UnsignedAttributes -- Optional
  }

  EncapsulatedContentInfo {
    eContentType         id-ct-asciiTextWithCRLF,
                                -- (1.2.840.113549.1.9.16.1.27)
    eContent             OCTET STRING  -- Always absent
  }
```

3.1. ContentInfo

   The CMS requires the outer-most encapsulation to be ContentInfo
   [CMS]. The fields of ContentInfo are used as follows:

      contentType
         indicates the type of the associated content. For the detached
         Internet-Draft signature file, the encapsulated type is always
         SignedData, so the id-signedData (1.2.840.113549.1.7.2) object
         identifier MUST be present in this field.

      content
         holds the content. For the detached Internet-Draft signature
         file, the content is always a SignedData content.

3.2. SignedData

   The SignedData content type [CMS] contains the signature of the
   Internet-Draft and information to aid in the validation of that
   signature. The fields of SignedData are used as follows:

      version
         is the syntax version number. For this specification, the
         version number MUST be set to 3.

      digestAlgorithms
         is a collection of one-way hash function identifiers. It MUST
         contain the identifier used by the IETF Secretariat to generate
         the digital signature. See the discussion of digestAlgorithm
         in Section 3.2.1.

      encapContentInfo
         is the signed content, including a content type identifier.
         Since a detached signature is being created, it does not
         encapsulate the Internet-Draft. The use of the
         EncapsulatedContentInfo type is discussed further in Section
         3.2.2.

      certificates
         is an optional collection of certificates. It SHOULD include
         the X.509 certificate needed to validate the digital signature
         value. Certification Authority (CA) certificates and end
         entity certificates MUST conform to the certificate profile
         specified in [PKIX1].

   crls
      is an optional collection of certificate revocation lists
      (CRLs). It SHOULD NOT include any CRLs; however, any CRLs that
      are present MUST conform to the CRL profile specified in
      [PKIX1].

   signerInfos
      is a collection of per-signer information. For this
      specification, each item in the collection must represent the
      IETF Secretariat. More than one SignerInfo MAY appear to
      facilitate transitions between keys or algorithms. The use of
      the SignerInfo type is discussed further in Section 3.2.1.

3.2.1. SignerInfo

   The IETF Secretariat is represented in the SignerInfo type. The
   fields of SignerInfo are used as follows:

   version
      is the syntax version number. In this specification, the
      version MUST be set to 3.

   sid
      identifies the IETF Secretariat's public key. In this
      specification, the subjectKeyIdentifier alternative is always
      used, which identifies the public key directly. This
      identifier MUST match the value included in the
      subjectKeyIdentifier certificate extension in the IETF
      Secretariat's X.509 certificate.

   digestAlgorithm
      identifies the one-way hash function, and any associated
      parameters, used by the IETF Secretariat to generate the
      digital signature.

   signedAttrs
      is an optional set of attributes that are signed along with the
      content. The signedAttrs are optional in the CMS, but
      signedAttrs is required by this specification. The SET OF
      Attribute must be encoded with the distinguished encoding rules
      (DER) [X.690]. Section 3.2.3 of this document lists the signed
      attributes that MUST be included in the collection. Other
      signed attributes MAY also be included.

   signatureAlgorithm
      identifies the digital signature algorithm, and any associated
      parameters, used by the IETF Secretariat to generate the
      digital signature.

   signature
      is the digital signature value generated by the IETF
      Secretariat.

   unsignedAttrs
      is an optional set of attributes that are not signed. Unsigned
      attributes are usually omitted; however, the unsigned
      attributes MAY hold a trusted timestamp generated in accordance
      with [TSP]. Appendix A of [TSP] provides more information
      about this unsigned attribute.

3.2.2. EncapsulatedContentInfo

   The EncapsulatedContentInfo structure contains a content type
   identifier. Since a detached signature is being created, it does not
   encapsulate the Internet-Draft. The fields of
   EncapsulatedContentInfo are used as follows:

   eContentType
      is an object identifier that uniquely specifies the content
      type. The content type associated with the plain text file
      MUST be id-ct-asciiTextWithCRLF. Other file formats may also
      be posted, and the appropriate content type for each format is
      discussed in Section 4. Additional file formats can be added
      if the Internet community chooses.

   eContent
      is optional. When an encapsulated signature is generated, the
      content to be signed is carried in this field. Since a
      detached signature is being created, eContent MUST be absent.

3.2.3. Signed Attributes

   The IETF Secretariat MUST digitally sign a collection of attributes
   along with the Internet-Draft. Each attribute in the collection MUST
   be DER-encoded. The syntax for attributes is defined in [X.501], and
   the X.500 Directory provides a rich attribute syntax. A very simple
   subset of this syntax is used extensively in [CMS], where
   ATTRIBUTE.&Type and ATTRIBUTE.&id are the only parts of the ATTRIBUTE
   class that are employed.

   Each of the attributes used with this CMS profile has a single
   attribute value. Even though the syntax is defined as a SET OF
   AttributeValue, there MUST be exactly one instance of AttributeValue
   present.

The SignedAttributes syntax within signerInfo is defined as a SET OF
Attribute. The SignedAttributes MUST include only one instance of
any particular attribute.

The IETF Secretariat MUST include the content-type, message-digest,
and signing-time attributes. The IETF Secretariat MAY also include
the binary-signing-time signed attribute as well as any other
attribute that is deemed appropriate. The intent is to allow
additional signed attributes to be included if a future need is
identified. This does not cause an interoperability concern because
unrecognized signed attributes are ignored at verification.

3.2.3.1. Content-Type Attribute

A content-type attribute is required to contain the same object
identifier as the content type contained in the
EncapsulatedContentInfo. The appropriate content type for each
format is discussed in Section 4. The IETF Secretariat MUST include
a content-type attribute containing the appropriate content type.
Section 11.1 of [CMS] defines the content-type attribute.

3.2.3.2. Message-Digest Attribute

The IETF Secretariat MUST include a message-digest attribute, having
as its value the output of a one-way hash function computed on the
Internet-Draft that is being signed. Section 11.2 of [CMS] defines
the message-digest attribute.

3.2.3.3. Signing-Time Attribute

The IETF Secretariat MUST include a signing-time attribute,
specifying the time, based on the local system clock, at which the
digital signature was applied to the Internet-Draft. Since the IETF
Secretariat may choose to perform signatures in batches, the signing-
time may be several hours or days after the time that the Internet-
Draft was actually posted. Section 11.3 of [CMS] defines the
content-type attribute.

3.2.3.4. Binary-Signing-Time Attribute

The IETF Secretariat MAY include a binary-signing-time attribute,
specifying the time at which the digital signature was applied to the
Internet-Draft. If present, the time that is represented MUST match
the time represented in the signing-time attribute. The binary-
signing-time attribute is defined in [BinTime].

3.2.4. Unsigned Attributes

   Unsigned attributes are usually omitted. However, an unsigned
   attribute MAY hold a trusted timestamp generated in accordance with
   [TSP]. The idea is to timestamp the IETF Secretariat digital
   signature to prove that it was created before a given time. If the
   IETF Secretariat's certificate is revoked the timestamp allows a
   verifier to know whether the signature was created before or after
   the revocation date. Appendix A of [TSP] defines the signature time-
   stamp attribute that can be used to timestamp a digital signature.

4. Content Types

   This section lists the content types that are used in this
   specification. The eContentType field as described in Section 3.2.2
   contains a content type identifier, and the same value appears in the
   content-type attribute as described in Section 3.2.3.1.

   The following table lists the file formats and the associated content
   type.

       File Format   Content Type
       -----------   ------------
       Plain text   id-ct-asciiTextWithCRLF
       Extensible Markup Language (XML) id-ct-xml
       Portable Document Format (PDF)   id-ct-pdf
       PostScript   id-ct-postscript

   The object identifiers associated with the content types listed in
   the above table are:

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

       id-ct-asciiTextWithCRLF OBJECT IDENTIFIER ::= { id-ct 27 }

       id-ct-xml OBJECT IDENTIFIER ::= { id-ct 28 }

       id-ct-pdf OBJECT IDENTIFIER ::= { id-ct 29 }

       id-ct-postscript OBJECT IDENTIFIER ::= { id-ct 30 }

5. Security Considerations

   The IETF Secretariat MUST protect its private key. The use of a
   hardware security module (HSM) is strongly RECOMMENDED because
   compromise of the IETF Secretariat's private key permits masquerade.

The IETF Secretariat currently maintains servers at a primary
location and a backup location. This configuration requires two
HSMs, one at each location. However, the two HSMs do not need to use
the same signing key. Each HSM can have a different signing key, as
long as each one has their own certificate.

The generation of a public/private key pair for signature operations
relies on random number generation. The use of an inadequate pseudo-
random number generator (PRNG) can result in little or no security.
An attacker may find it much easier to reproduce the PRNG environment
that produced the key pair, searching the resulting small set of
possibilities, than to brute-force search the whole private key
space. The generation of quality random numbers is difficult, but
[RANDOM] offers important guidance in this area.

The IETF Secretariat should be aware that cryptographic algorithms
become weaker with time. As new cryptanalysis techniques are
developed and computing performance improves, the work factor to
break a particular digital signature algorithm or one-way hash
function will be reduced. Therefore, it SHOULD be possible to
migrate these algorithms. That is, the IETF Secretariat SHOULD be
prepared for the supported algorithms to change over time.

The IETF Secretariat must take care to use the correct time in
signing-time and binary-signing-time attributes. The inclusion of a
date within the Internet-Draft by the authors that is shortly before
the signing time attributes supplied by the IETF Secretariat provides
confidence about the date that the Internet-Draft was posted to the
repository. However, the IETF Secretariat may choose to perform
signatures in batches, and the signing-time may be several hours or
days after the time that the Internet-Draft was actually posted.

As stated above, the IETF Secretariat may choose to sign Internet-
Drafts in batches. This allows a single HSM to be used if multiple
servers are located in one geographic location, and it allows the HSM
to be off-line except when signatures are being generated. Further,
this allows the IETF Secretariat to include manual steps, such as
entering an HSM passphrase or inserting a smartcard, as part of the
signing procedure to improve operations security.

6. Deployment and Operational Considerations

The private key used to generate the IETF Secretariat signature ought
to be stored in an HSM to provide protection from unauthorized
disclosure. While the HSM will be operated by the IETF Secretariat,
it ought to be owned by the IETF Trust. Accordingly, the Trustees of
the IETF Trust will designate an appropriate certification authority

to issue a certificate to the IETF Secretariat, and they will approve
any procedures used by the IETF Secretariat for signing documents
consistent with this specification.

7. Design Rationale

   A detached signature is used for all file formats. Some file
   formats, such as PDF and XML, have file-format-specific ways of
   handling digital signatures. These file-format-specific approaches
   are not used for two reasons. First, a single way to sign Internet-
   Drafts will ease implementation by the IETF Secretariat. Second, if
   the author includes a signature using one of these file-format-
   specific approaches, the IETF Secretariat signature does not harm it
   in any way.

   File names are the means linking the detached signature to the signed
   document. A CMS signed attribute could have been specified to
   include another form of linkage, and this could be added in the
   future. At this point in time, it is important to support signature
   validation of expired Internet-Drafts that are obtained from non-IETF
   repositories. Therefore, the appropriate value for such a signed
   attribute is unclear. This specification allows an Internet-Draft
   and companion signature file to be stored anywhere without hindering
   signature validation.

8. Acknowledgments

   The idea for the Internet-Draft signature file came from a discussion
   with Scott Bradner at IETF 69 in Chicago. Many helpful suggestions
   came from Jim Schaad, Pasi Eronen, and Chris Newman.

9. References

9.1. Normative References

   [CMS]   Housley, R., "Cryptographic Message Syntax (CMS)", RFC
                3852, July 2004.

   [PKIX1]   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.

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

   [X.680]   ITU-T Recommendation X.680: ISO/IEC 8824-1:2002,
                Information technology - Abstract Syntax Notation One
                (ASN.1): Specification of basic notation, 2002.

   [X.690]   ITU-T Recommendation X.690: ISO/IEC 8825-1:2002,
                Information technology - ASN.1 encoding rules:
                Specification of Basic Encoding Rules (BER), Canonical
                Encoding Rules (CER) and Distinguished Encoding Rules
                (DER), 2002.

9.2. Informative References

   [BinTime]   Housley, R., "BinaryTime: An Alternate Format for
                Representing Date and Time in ASN.1", RFC 4049, April
                2005.

   [FTP]   Postel, J. and J. Reynolds, "File Transfer Protocol", STD
                9, RFC 959, October 1985.

   [MSG]   Ramsdell, B., Ed., "Secure/Multipurpose Internet Mail
                Extensions (S/MIME) Version 3.1 Message Specification",
                RFC 3851, July 2004.

   [OpenSSL]   "OpenSSL: The Open Source toolkit for SSL/TLS",
                http://www.openssl.org/.

   [R20060816] Bray, T., J. Paoli, C. M. Sperberg-McQueen, E. Maler, and
                F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fourth
                Edition)", W3C Recommendation, 16 August 2006,
                http://www.w3.org/TR/2006/REC-xml-20060816/.

   [RANDOM]   Eastlake, D., 3rd, Schiller, J., and S. Crocker,
                "Randomness Requirements for Security", BCP 106, RFC
                4086, June 2005.

   [TELNET]   Postel, J. and J. Reynolds, "Telnet Protocol
                Specification", STD 8, RFC 854, May 1983.

   [TSP]   Adams, C., Cain, P., Pinkas, D., and R. Zuccherato,
                "Internet X.509 Public Key Infrastructure Time-Stamp
                Protocol (TSP)", RFC 3161, August 2001.

   [UFNI]   Klensin, J. and M. Padlipsky, "Unicode Format for Network
                Interchange", RFC 5198, March 2008.

   [X.501]   ITU-T Recommendation X.501: Information Technology - Open
                Systems Interconnection - The Directory: Models, 1993.

Appendix: A

   OpenSSL 0.9.9 [OpenSSL] includes an implementation of CMS. The
   following command line can be used to verify an Internet-Draft
   signature:

     openssl cms -verify -CAfile <cert-file> -content <internet-draft> /
          -inform DER -in <p7s-file> -out /dev/null

   The arguments need to be provided as follows:

      <cert-file>
         the name of the file containing the trust anchor, which is
         typically the self-signed certificate of the certification
         authority that issued a certificate to the IETF Secretariat.

      <internet-draft>
         the name of the file containing the Internet-Draft after
         canonicalization.

      <p7s-file>
         the name of the file containing the detached signature that was
         generated in accordance with this specification.

Author's Address

   Russell Housley
   Vigil Security, LLC
   918 Spring Knoll Drive
   Herndon, VA 20170
   USA

   EMail: housley@vigilsec.com