Network Working Group
Request for Comments: 4049
Category: Experimental

R. Housley
Vigil Security

April 2005

BinaryTime: An Alternate Format for Representing Date and Time in ASN.1

Status of This Memo

Copyright Notice

Abstract

This document specifies a new ASN.1 type for representing time: BinaryTime. This document also specifies an alternate to the signing-time attribute for use with the Cryptographic Message Syntax (CMS) SignedData and AuthenticatedData content types; the binary-signing-time attribute uses BinaryTime. CMS and the signing-time attribute are defined in RFC 3852.

1. Introduction

This document specifies a new ASN.1 [ASN1] type for representing time: BinaryTime. This ASN.1 type can be used to represent date and time values.

This document also specifies an alternative to the signing-time attribute used with the Cryptographic Message Syntax (CMS) [CMS] SignedData and AuthenticatedData content types, allowing the BinaryTime type to be used instead of the traditional UTCTime and GeneralizedTime types.

1.1. BinaryTime

Many operating systems represent date and time as an integer. This document specifies an ASN.1 type for representing date and time in a manner that is also an integer. Although some conversion may be necessary due to the selection of a different epoch or a different granularity, an integer representation has several advantages over the UTCTime and GeneralizedTime types.

First, a BinaryTime value is smaller than either a UTCTime or a GeneralizedTime value.

Second, in some operating systems, the value can be used with little or no conversion. Conversion, when it is needed, requires only straightforward computation. If the endian ordering is different from the ASN.1 representation of an INTEGER, then straightforward manipulation is needed to obtain an equivalent integer value. If the epoch is different than the one chosen for BinaryTime, addition or subtraction is needed to compensate. If the granularity is something other than seconds, then multiplication or division is needed to compensate. Also, padding may be needed to convert the variable-length ASN.1 encoding of INTEGER to a fixed-length value used in the operating system.

Third, date comparison is very easy with BinaryTime. Integer comparison is easy, even when multi-precision integers are involved. Date comparison with UTCTime or GeneralizedTime can be complex when the two values to be compared are provided in different time zones.

This is a rare instance which both memory and processor cycles can be saved.

1.2. Binary Signing Time Attribute

The signing-time attribute is defined in [CMS]. The alternative binary-signing-time attribute is defined in this document in order to obtain the benefits of the BinaryTime type.

1.3. Terminology

In this document, the key words MUST, MUST NOT, REQUIRED, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL are to be interpreted as described in [STDWORDS].

2. BinaryTime Definition

The BinaryTime ASN.1 type is used to represent an absolute time and date. A positive integer value is used to represent time values based on coordinated universal time (UTC), which is also called Greenwich Mean Time (GMT) and ZULU clock time.

The syntax for BinaryTime is:

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

The integer value is the number of seconds, excluding leap seconds, after midnight UTC, January 1, 1970. This time format cannot represent time values prior to January 1, 1970. The latest UTC time value that can be represented by a four-octet integer value is 03:14:07 on January 19, 2038, which is represented by the hexadecimal value 7FFFFFFF. Time values beyond 03:14:07 on January 19, 2038, are represented by integer values that are longer than four octets, and a five-octet integer value is sufficient to represent dates covering the next seventeen millennia.

This specification uses a variable-length encoding of INTEGER. This permits any time value after midnight UTC, January 1, 1970, to be represented.

When encoding an integer value that consists of more than one octet, which includes almost all the time values of interest, the bits of the first octet and bit 8 of the second octet MUST NOT all be ones or all zeros. This rule ensures that an integer value is always encoded in the smallest possible number of octets. However, it means that implementations cannot assume a fixed length for the integer value.

3. Binary Signing Time Attribute Definition

The binary-signing-time attribute type specifies the time at which the signer (purportedly) performed the signing process. The binary-signing-time attribute type is intended for use in the CMS SignedData content type; however, the attribute can also be used with the AuthenticatedData content type.

The binary-signing-time attribute MUST be a signed attribute or an authenticated attribute; it MUST NOT be an unsigned attribute, unauthenticated attribute, or unprotected attribute.

The following object identifier identifies the binary-signing-time attribute:

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

The binary-signing-time attribute values have ASN.1 type BinarySigningTime:

```
BinarySigningTime ::= BinaryTime
```

In [CMS], the SignedAttributes syntax and the AuthAttributes syntax are each defined as a SET OF Attributes. However, the binary-signing-time attribute MUST have a single attribute value, even though the syntax is defined as a SET OF AttributeValue. There MUST NOT be zero or multiple instances of AttributeValue present.

The SignedAttributes contained in the signerInfo structure within SignedData MUST NOT include multiple instances of the binary-signing-time attribute. Similarly, the AuthAttributes in an AuthenticatedData MUST NOT include multiple instances of the binary-signing-time attribute.

No requirement is imposed concerning the correctness of the signing time itself, and acceptance of a purported signing time is a matter of a recipient's discretion. It is expected, however, that some signers, such as time-stamp servers, will be trusted implicitly.

4. References

This section provides normative and informative references.

4.1. Normative References

[ASN1] CCITT. Recommendation X.208: Specification of Abstract Syntax Notation One (ASN.1). 1988.

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

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

4.2. Informative References

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

5. Security Considerations

Use of the binary-signing-time attribute does not necessarily provide confidence in the time when the signature value was produced. Therefore, acceptance of a purported signing time is a matter of a recipient's discretion. RFC 3161 [TSP] specifies a protocol for obtaining time stamps from a trusted entity.

The original signing-time attribute defined in [CMS] has the same semantics as the binary-signing-time attribute specified in this document. Therefore, only one of these attributes SHOULD be present in the signedAttrs of a SignerInfo object or in the authAttrs of an AuthenticatedData object. However, if both of these attributes are present, they MUST provide the same date and time.

Appendix A: ASN.1 Module

The ASN.1 module contained in this appendix defines the structures that are needed to implement this specification. It is expected to be used in conjunction with the ASN.1 modules in [CMS].

```
BinarySigningTimeModule
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
    pkcs-9(9) smime(16) modules(0) 27 }

DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- BinaryTime Definition

BinaryTime ::= INTEGER (0..MAX)

-- Signing Binary Time Attribute

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

BinarySigningTime ::= BinaryTime

END
```

Author’s Address

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

EMail: housley@vigilsec.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.