                   Compressed Data Content Type for
                   Cryptographic Message Syntax (CMS)

Status of this Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements. Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol. Distribution of this memo is unlimited.

Abstract

   This document defines a format for using compressed data as a
   Cryptographic Message Syntax (CMS) content type. Compressing data
   before transmission provides a number of advantages, including the
   elimination of data redundancy which could help an attacker, speeding
   up processing by reducing the amount of data to be processed by later
   steps (such as signing or encryption), and reducing overall message
   size. Although there have been proposals for adding compression at
   other levels (for example at the MIME or SSL level), these don't
   address the problem of compression of CMS content unless the
   compression is supplied by an external means (for example by
   intermixing MIME and CMS).

1. Introduction

   This document describes a compressed data content type for CMS. This
   is implemented as a new ContentInfo type and is an extension to the
   types currently defined in CMS [RFC2630]. CMS implementations SHOULD
   include support for the CompressedData content type.

   The format of the messages are described in ASN.1 [ASN1].

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

1.1 Compressed Data Content Type

The compressed-data content type consists of content of any type, compressed using a specified algorithm. The following object identifier identifies the compressed-data content type:

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

The compressed-data content type shall have ASN.1 type CompressedData:

```
   CompressedData ::= SEQUENCE {
     version CMSVersion,
     compressionAlgorithm CompressionAlgorithmIdentifier,
     encapContentInfo EncapsulatedContentInfo
     }
```

The fields of type CompressedData have the following meanings:

version is the syntax version number. It MUST be 0. Details of the CMSVersion type are discussed in CMS [RFC2630], section 10.2.5.

compressionAlgorithm is a compression algorithm identifier, as defined in section 2.

encapContentInfo is the content which is compressed. Details of the EncapsulatedContentInfo type are discussed in CMS [RFC2630], section 5.2.

Implementations SHOULD use the SMIMECapabilities attribute to indicate their ability to process compressed content types. Details of SMIMECapabilities are discussed in MSG [RFC2633], section 2.5.2.

A compression SMIMECapability consists of the AlgorithmIdentifier for the supported compression algorithm. In the case of the algorithm specified in this document, this is id-alg-zlibCompression, as specified in section 2. Alternatively, the use of compression may be handled by prior arrangement (for example as part of an interoperability profile).

   The SMIMECapability SEQUENCE representing the ability to process
   content compressed with the algorithm identified by id-alg-
   zlibCompression MUST be DER-encoded as the following hexadecimal
   string:

      30 0D 06 0B 2A 86 48 86 F7 0D 01 09 10 03 08

   (but see also the implementation note in section 2.1).

2. Compression Types

   CMS implementations that support the CompressedData content type MUST
   include support for the ZLIB compression algorithm [RFC1950]
   [RFC1951], which has a freely-available, portable and efficient
   reference implementation. The following object identifier identifies
   ZLIB:

      id-alg-zlibCompress OBJECT IDENTIFIER ::= { iso(1) member-body(2)
        us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) alg(3) 8 }

   This algorithm has no parameters. The parameters field SHOULD be
   encoded as omitted, but MAY be encoded as NULL (see the
   implementation note in section 2.1).

2.1. Implementation notes

   ZLIB allows for a number of compression levels ranging from good but
   slow compression, to less good but fast compression. The compression
   level is always compatible with the decompression algorithm, so there
   is no need to specify the compression level as an algorithm
   parameter.

   There are two possible encodings for the ZLIB null parameters field
   which arise from the fact that when the 1988 syntax for
   AlgorithmIdentifier was translated into the 1997 syntax, the OPTIONAL
   associated with the AlgorithmIdentifier parameters got lost. Later
   it was recovered via a defect report, but by then, everyone thought
   that algorithm parameters were mandatory. Because of this, some
   implementations will encode null parameters as an ASN.1 NULL element
   and some will omit them entirely (see for example section 12 of CMS
   [RFC2630]). Although the correct encoding is to omit the parameters
   field, implementations may encounter encodings which use an ASN.1
   NULL element for the parameters.

3. Security Considerations

   This RFC is not concerned with security, except for the fact that
   compressing data before encryption can enhance the security provided
   by other processing steps by reducing the quantity of known plaintext
   available to an attacker. However, implementations should be aware
   of possible security threats of combining security sensitive material
   with possibly untrusted data before the compression and encryption.
   This is because information about the sensitive data may be inferred
   from knowing the untrusted data and the compression ratio.

4. IANA Considerations

   The CompressedData content type and compression algorithms are
   identified by object identifiers (OIDs). OIDs were assigned from an
   arc contributed to the S/MIME Working Group by RSA Security. Should
   additional compression algorithms be introduced, the advocates for
   such algorithms are expected to assign the necessary OIDs from their
   own arcs. No action by the IANA is necessary for this document or
   any anticipated updates.

References

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

   [RFC2119] Bradner, S., "Key Words for Use in RFC's to Indicate
              Requirement Levels", BCP 14, RFC 2119, March 1997.

   [RFC1950] Deutsch, P. and J-L Gailly, "ZLIB Compressed Data Format
              Specification version 3.3", RFC 1950, May 1996.

   [RFC1951] Deutsch, P., "DEFLATE Compressed Data Format Specification
              version 1.3", RFC 1951, May 1996.

   [RFC2630] Housley, R., "Cryptographic Message Syntax", RFC 2630, June
              1999.

   [RFC2633] Rmasdell, B., "S/MIME Version 3 Message Specification", RFC
              2633, June 1999.

Appendix A: ASN.1 Module

```
   CompressedDataContent
     { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
       smime(16) modules(0) compress(11) }

   DEFINITIONS IMPLICIT TAGS ::=
   BEGIN

   IMPORTS
     CMSVersion, EncapsulatedContentInfo FROM CryptographicMessageSyntax
       { iso(1) member-body(2) us(840) rsadsi(113549)
         pkcs(1) pkcs-9(9) smime(16) modules(0) cms(1) }
     AlgorithmIdentifier FROM AuthenticationFramework
       { joint-iso-itu-t ds(5) module(1) authenticationFramework(7) 3 };

   CompressedData ::= SEQUENCE {
     version CMSVersion,        -- Always set to 0
     compressionAlgorithm CompressionAlgorithmIdentifier,
     encapContentInfo EncapsulatedContentInfo
     }

   CompressionAlgorithmIdentifier ::= AlgorithmIdentifier

   -- Algorithm Identifiers

   id-alg-zlibCompress OBJECT IDENTIFIER ::= { iso(1) member-body(2)
       us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) alg(3) 8 }

   -- Content Type Object Identifiers

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

   END
```

Author Address

   Peter Gutmann
   University of Auckland
   Private Bag 92019
   Auckland, New Zealand

   EMail: pgut001@cs.auckland.ac.nz

Full Copyright Statement

Acknowledgement