Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 26, 2011

J. Schaad
Soaring Hawk Consulting

November 22, 2010

Signer Info Algorithm Protection Attribute
draft-schaad-smime-algorithm-attribute-02

Abstract

A new attribute is defined that allows for protection of the digest and signature algorithm structures in an authenticated data or a signer info structure. Using the attribute includes the algorithm definition information in the integrity protection process.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 26, 2011.

Table of Contents

1. Introduction 3
1.1. Notation 4
2. Attribute Structure 5
3. Verification Process 7
3.1. Signed Data Verification Changes 7
3.2. Authenticated Data Verification Changes 7
4. Security Considerations 8
5. References 9
5.1. Normative References 9
5.2. Informational References 9
Appendix A. ASN.1 Module 10
Author's Address 12

1. Introduction

In the current definition of [CMS], there are some fields that are not protected in the process of doing either a signature validation or an authentication validation. In this document a new signed or authenticated attribute is defined which permits these fields to be validated.

Taking the SignerInfo structure from CMS, let's look at each of the fields and discuss what is and is not protected by the signature. The ASN.1 is included here for convenience. (The analysis of AuthenticatedData is similar.)

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

version is not protected by the signature. Many implementations of CMS today actually ignore the value of this field. If the structure decodes then this is considered sufficient to continue processing. Using most decoders on the market the value of this field does not control how the decoding is actually processed.

sid can be protected by the use of either version of the signing certificate authenticated attribute. SigningCertificateV2 is defined in [RFC5035]. SigningCertificate is defined in [RFC2634]. In addition to allowing for the protection of the signer identifier, the specific certificate is protected by including a hash of the certificate to be used for validation.

digestAlgorithm the digest algorithm used has been implicitly protected by the fact that CMS has only defined one digest algorithm for each hash value length. (The algorithm RIPEM-160 was never standardized). If newer digest algorithms are defined where there are multiple algorithms for a given hash length, or where parameters are defined for a specific algorithm, this implicit protection will no longer exist.

signedAttributes are directly protected by the signature when they are present. The DER encoding of this value is what is actually hashed for the signature computation.

signatureAlgorithm has been protected by implication in the past. The use of an RSA public key implied that the RSA v 1.5 signature algorithm was being used. The hash algorithm and this fact could be checked by the internal padding defined. This is no longer true with the addition of the RSA-PSS signature algorithms. The use of a DSA public key implied the SHA-1 hash algorithm as that was the only possible hash algorithm and the DSA was the public signature algorithm. This is longer true with the addition of the SHA2 signature algorithms.

signature is not directly protected by any other value unless a counter signature is present. However this represents the cryptographically computed value that protects the rest of the signature information.

unsignedAttrs is not protected by the signature value. It is also explicitly designed not to be protected by the signature value.

As can be seen above, the digestAlgorithm and signatureAlgorithm fields have been indirectly rather than explicitly protected in the past. With new algorithms that have been or are being defined this will no longer be the case. This document defines and describes a new attribute that will explicitly protect these fields along with the macAlgorithm field of the AuthenticatedData structure.

1.1. Notation

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].

2. Attribute Structure

The following defines the algorithm protection attribute:

The algorithm-protection attribute has the ASN.1 type CMSAlgorithmProtection:

```
    aa-cmsAlgorithmProtection ATTRIBUTE ::= {
        TYPE CMSAlgorithmProtection
        IDENTIFIED BY { id-aa-CMSAlgorithmProtection }
    }
```

The following object identifier identifies the algorithm-protection attribute:

```
    id-aa-CMSAlgorithmProtection OBJECT IDENTIFIER ::= { iso(1)
         member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9) 52 }
```

The algorithm-protection attribute uses the following ASN.1 type:

```
   CMSAlgorithmProtection ::= SEQUENCE {
```

```
        digestAlgorithm         DigestAlgorithmIdentifier,
        signatureAlgorithm  [1] SignatureAlgorithmIdentifier OPTIONAL,
        macAlgorithm        [2] MessageAuthenticationCodeAlgorithm
                                    OPTIONAL
    }
    (WITH COMPONENTS { signatureAlgorithm PRESENT,
                       macAlgorithm ABSENT } |
     WITH COMPONENTS { signatureAlgorithm ABSENT,
                       macAlgorithm PRESENT })
```

The fields are defined as follows:

digestAlgorithm contains a copy of the SignerInfo.digestAlgorithm field or the AuthenticatedData.digestAlgorithm field including any parameters associated with it.

signatureAlgorithm contains a copy of the signature algorithm identifier and any parameters associated with it. This field is only populated if the attribute is placed in a SignerInfo.signedAttrs sequence.

macAlgorithm contains a copy of the message authentication code algorithm identifier and any parameters associated with it. This field is only populated if the attribute is placed in an AuthenticatedData.authAttrs sequence.

Exactly one of signatureAlgorithm and macAlgorithm SHALL be present.

An algorithm-protection 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 algorithm-protection attribute MUST be a signed attribute or an authenticated attribute; it MUST NOT be an unsigned attribute, an unauthenticated attribute or an unprotected attribute.

The SignedAttributes and AuthAttributes syntax are each defined as a SET of Attributes. The SignedAttributes in a signerInfo MUST include only one instance of the algorithm protection attribute. Similarly, the AuthAttributes in an AuthenticatedData MUST include only one instance of the algorithm protection attribute.

3. Verification Process

The exact verification process depends on the structure being dealt with.

When doing comparisons of the fields, a field whose value is a default value and one which is explicitly provided MUST compare as equivalent. It is not required that a field which is absent in one

case and present in another case be compared as equivalent. (This means that an algorithm identifier with absent parameters and one with NULL parameters are not expected to compare as equivalent.)

3.1. Signed Data Verification Changes

If a CMS validator supports this attribute, the following additional verification steps MUST be performed:

1. The SignerInfo.digestAlgorithm field MUST be compared to the digestAlgorithm field in the attribute. If the fields are not the same (modulo encoding) then signature validation MUST fail.

2. The SignerInfo.signatureAlgorithm field MUST be compared to the signatureAlgorithm field in the attribute. If the fields are not the same (modulo encoding) then the signature validation MUST fail.

3.2. Authenticated Data Verification Changes

If a CMS validator supports this attribute, the following additional verification steps MUST be performed:

1. The AuthenticatedData.digestAlgorithm field MUST be compared to the digestAlgorithm field in the attribute. If the fields are not same (modulo encoding) then signature validation MUST fail.

2. The AuthenticatedData.macAlgorithm field MUST be compared to the macAlgorithm field in the attribute. If the fields are not the same (modulo encoding) then the signature validation MUST fail.

4. Security Considerations

This document is designed to address the security issue of algorithm substitutions of the algorithms used by the validator. At this time there is no known method to exploit this type of attack. If the attack could be successful, then either a weaker algorithm could be substituted for a stronger algorithm or the parameters could be modified by an attacker to change the behavior of the hashing algorithm used. (One example would be changing the initial parameter value for [I-D.schaad-smime-hash-experiment].)

The attribute defined in this document is to be placed in a location that is protected by the signature or message authentication code. This attribute does not provide any additional security if placed in an un-signed or un-authenticated location.

5. References

5.1. Normative References

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

[RFC2634] Hoffman, P., "Enhanced Security Services for S/MIME", RFC 2634, June 1999.

[RFC5035] Schaad, J., "Enhanced Security Services (ESS) Update: Adding CertID Algorithm Agility", RFC 5035, August 2007.

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

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

5.2. Informational References

[I-D.schaad-smime-hash-experiment] Schaad, J., "Experiment: Hash functions with parameters in CMS and S/MIME", draft-schaad-smime-hash-experiment-01 (work in progress), December 2009.

Appendix A. ASN.1 Module

```
  CMSAlgorithmProtectionAttribute
    { iso(1) member-body(2) us(840) rsadsi(113549)
      pkcs(1) pkcs-9(9) smime(16) modules(0)
      id-mod-cms-algorithmProtect(52) }
  DEFINITIONS IMPLICIT TAGS ::=
  BEGIN
   IMPORTS

     -- Cryptographic Message Syntax (CMS) [RFC5652]

     DigestAlgorithmIdentifier, MessageAuthenticationCodeAlgorithm,
     SignatureAlgorithmIdentifier
     FROM  CryptographicMessageSyntax-2009
       { iso(1) member-body(2) us(840) rsadsi(113549)
         pkcs(1) pkcs-9(9) smime(16) modules(0) id-mod-cms-2004-02(41) }

     -- Common PKIX structures [RFC5912]

     ATTRIBUTE
     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)};

     --
     --  The CMS Algorithm Protection attribute is a Signed Attribute or
     --  an Authenticated Attribute.
     --
     --  Add this attribute to SignedAttributesSet in [RFC5652]
     --  Add this attribute to AuthAttriuteSet in [RFC5652]
     --

     aa-cmsAlgorithmProtection ATTRIBUTE ::= {
        TYPE CMSAlgorithmProtection
        IDENTIFIED BY { id-aa-cmsAlgorithmProtect }
     }
```

```
   id-aa-cmsAlgorithmProtect OBJECT IDENTIFIER ::= {
      iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
      pkcs9(9) 52 }

   CMSAlgorithmProtection ::= SEQUENCE {
      digestAlgorithm         DigestAlgorithmIdentifier,
      signatureAlgorithm  [1] SignatureAlgorithmIdentifier OPTIONAL,
      macAlgorithm        [2] MessageAuthenticationCodeAlgorithm
                                      OPTIONAL
```



```
   }
   (WITH COMPONENTS { signatureAlgorithm PRESENT,
                      macAlgorithm ABSENT } |
    WITH COMPONENTS { signatureAlgorithm ABSENT,
                      macAlgorithm PRESENT })

  END
```

Author's Address

Jim Schaad
Soaring Hawk Consulting
PO Box 675
Gold Bar, WA 98251

Email: ietf@augustcellars.com