Signer Info Algorithm Protection Attribute
draft-schaad-smime-algorithm-attribute-01

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 to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on June 20, 2010.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents.
carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the BSD License.

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 ................................................ 11
1. Introduction

In the current definition of [RFC3852] 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, lets look at each of the
fields for and discuss what is and is not protected by the signature.
The ASN.1 is included here for convience. (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 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. For RSA v 1.5 signatures, the fact that the RSA algorithm was known from the public key and the digest algorithm used is included in the formatted value over which the RSA computation is done. For DSA signature, the fact that a DSA public key was sufficient to know that SHA-1 was the digest algorithm as it was the only acceptable digest algorithm. With the advent of newer signature algorithms, especially those such as RSA-PSS which have parameters, this implicit protection of the signature algorithm is no longer possible.

signature is not directly protected by any other value unless a counter signature is present. However this represents the cryptographically computed value to protected 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 AlgorithmProtection:

```
  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) TBA }
```

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

```
CMSAlgorithmProtection ::= SEQUENCE {
  digestAlgorithm DigestAlgorithmIdentifier,
  signatureAlgorithm [1] SignatureAlgorithmIdentifier OPTIONAL,
}(WITH COMPONENTS { signatureAlgorithm PRESENT,
  macAlgorithm ABSENT } |
WITH COMPONENTS { signatureAlgorithm ABSENT,
  macAlgorithm PRESENT })
```

The fields are defined as follows:

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

signatureAlg 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 structure.

authenticationAlg contains a copy of the message authentication code
and any parameters associated with it. This field is only
populated if the attribute is placed in an authenticated data
structure.

Exactly one of signatureAlgorithm and macAlgorithm SHALL be present.
An algorithm protection 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 authentication 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 need not 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 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 can be modified by an attacker to change the state of the hashing algorithm used. (One example would be changing the initial parameter value for [I-D.schaad-smime-hash-experiment]

The attributes defined in this document are to be placed in locations that are protected by the signature. This attribute does not provide any additional security if placed in an un-signed or un-authenticated location.
5. References

5.1. Normative References


5.2. Informational References

Appendix A. ASN.1 Module

CMSAlgorithmAttribute
{ iso(1) member-body(2) us(840) rsadsi(113549)
  pkcs(1) pkcs-9(9) smime(16) modules(0) 989 }
DEFINITIONS IMPLICIT TAGS ::= BEGIN
IMPORTS
  -- Cryptographic Message Syntax (CMS) [RFC3852]
  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 [I-D.ietf-pkix-new-asn1]
  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 [RFC3852]
  -- Add this attribute to AuthAttributeSet in [RFC3852]

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

  id-aa-CMSAlgorithmProtection OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 989 }

  CMSAlgorithmProtection ::= SEQUENCE {
    digestAlgorithm DigestAlgorithmIdentifier,
    signatureAlgorithm [1] SignatureAlgorithmIdentifier 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