

TCG Trusted Network Connect TNC IF-M Security: Bindings to CMS

Specification Version 1.0
Revision 14
31 January 2008
Draft

Contact:

Paul Sangster – Paul_Sangster@symantec.com (Editor, TNC Co-Chair)

Steve Hanna - shanna@juniper.net (TNC Co-Chair)

Work In Progress

This document is an intermediate draft for comment only and is subject to change without notice. Readers should not design products based on this document.

TCG

TCG Confidential

Copyright © TCG 2005-2008

Copyright © 2005-2008 Trusted Computing Group, Incorporated.

Disclaimers, Notices, and License Terms

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Without limitation, TCG disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

This document is copyrighted by Trusted Computing Group (TCG), and no license, express or implied, is granted herein other than as follows: You may not copy or reproduce the document or distribute it to others without written permission from TCG, except that you may freely do so for the purposes of (a) examining or implementing TCG specifications or (b) developing, testing, or promoting information technology standards and best practices, so long as you distribute the document with these disclaimers, notices, and license terms.

Contact the Trusted Computing Group at www.trustedcomputinggroup.org for information on specification licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.

Revision History

12/23/2007	First draft with initial text describing the CMS Protected Attribute but lacking other attribute descriptions.
12/24/2007	Added ifmSecurityCapabilities and filled in text for key management requirements
12/31/2007	Updated to include steve's comments and initial draft of IF-M security errors
1/2/2008	Added partial security considerations section (shared only with reviewer team)
1/4/2008	More complete security considerations section plus updated for comments from TNC and IWG member reviews. Also closed a few option questions. Still need error message section.
1/8/2008	Initial complete draft including error message section and replay protection
1/10/2008	Minor edits from TNC review comments
1/13/2008	Many editorial clean ups and fixes to error discussions
1/14/2008	Final clean version for TNC approval
1/21/2008	Minor cleanup based on comments
1/31/2008	Updated to address comments from TC

Open Issues

This section tracks open issues for the duration of specification creation. This section will be removed prior to final draft.

#	Description	Comments
PS-001	ifmSecurityCapabilities CMS attribute is identical to the SMIMESecurityCapabilities CMS attribute. Should we use the same OID as was assigned to S/MIME or allocate a new one? See section 3.3.2.	Using the same OID might be a problem if SMIMESecurityCapabilities changes in a manner not desired by ifmSecurityCapabilities
PS-003	Should we require all signed information possibly except the IF-M attributes to be represented in DER? CMS doesn't require the encapsulated data in a signed-data content to be DER encoded so I think we can leave that field as just containing our attributes with no special encoding for use with CMS, but what about other fields included?	Needs more thought and maybe a consult with a CMS expert on when we can require DER.
PS-005	Signed-data and enveloped-data descriptions require sending certificates but use of certificates is optional for key management. Should these be lowered to MUST if used?	This impacts sections 3.3.2 and 3.3.3.

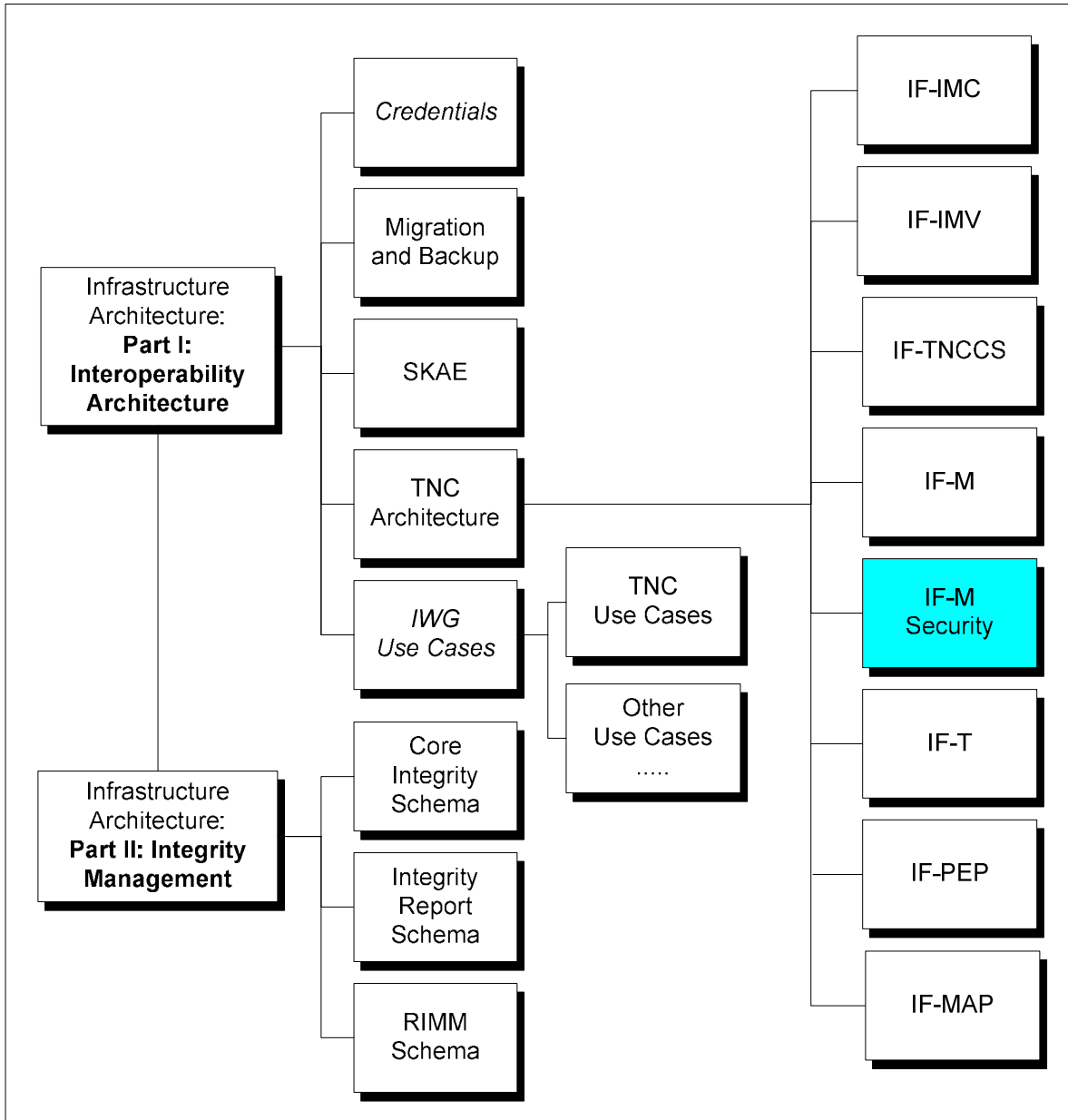
Closed Issues

This section will be removed prior to final draft.

#	Description	Comments
SR-001	Should we provide a minimum key length that MUST be supported for RSA in section 3.3.2.2?	This wasn't done for other CMS specs, but it makes sense to propose something here for interoperability.
SH-001	Add byte representations of signed-data and enveloped-data examples in an appendix. This will help improve interoperability.	Will wait for later revision due to time.

PS-004	TCG TC will need to approve OID hierarchy.	This can be removed once TC approves section 3.2 hierarchy. Moved to closed section assuming approval occurs.
PS-002	Need an RFC (stable) reference for SHA-256.	This is in the IETF editors queue. Will deal with this reference after the initial public review period when the I-D for SHA-256 is an approved RFC.

IWG TNC Document Roadmap



Acknowledgement

The TCG wishes to thank all those who contributed to this specification. This document builds on considerable work done in the various working groups in the TCG.

Special thanks to the members of the TNC contributing to this document:

Scott Kelly	Aruba Networks
Mahalingam Mani	Avaya
Hidenobu Ito	Fujitsu Limited
Sung Lee	Fujitsu Limited
Kazuaki Nimura	Fujitsu Limited
Mauricio Sanchez	Hewlett-Packard
Han Yin	Huawei
Diana Arroyo	IBM
Stuart Bailey	Infoblox
Ravi Sahita	Intel Corporation
Ned Smith	Intel Corporation
Josh Howlett	JANET
Steve Hanna (TNC co-chair)	Juniper Networks, Inc.
John Jerrim	Lancope, Inc.
Ryan Hurst	Microsoft
Sandilya Garimella	Motorola
Meenakshi Kaushik	Nortel Networks
Paul Sangster (Editor, TNC co-chair)	Symantec
Greg Kazmierczak	Wave Systems
Thomas Hardjono	Wave Systems

Table of Contents

1	Scope and Audience	8
2	Background	9
2.1	Purpose of IF-M Security	9
2.2	Supported and Unsupported Use Cases	9
2.3	Requirements.....	9
2.4	Rationale for Use of CMS	10
2.5	Keywords	10
3	IF-M Attributes Supporting CMS	11
3.1	Linkage to IF-M Protocol.....	11
3.2	TCG TNC OID Hierarchy	11
3.3	CMS Protected Content Attribute.....	12
3.3.1	CMS Content Info and Content Types	12
3.3.2	CMS Signed-Data	12
3.3.3	CMS Enveloped-Data.....	15
3.4	Security Capabilities Attribute.....	18
3.4.1	ifmSecurityCapabilities Within Signed-Data.....	19
3.4.2	ifmSecurityCapabilities ASN.1.....	20
3.5	CMS Error Code Attribute	20
3.5.1	ifmErrorCode Within Signed-Data.....	20
3.5.2	ifmErrorCode ASN.1.....	21
3.5.3	TCG Standard ifmErrorCode Values.....	21
3.6	Nonce CMS Attribute	22
3.6.1	ifmNonce CMS Attribute ASN.1	23
3.6.2	ifmNonce CMS Attribute Example.....	24
4	Security Considerations	26
4.1	Countermeasures to IF-M Threats.....	26
4.1.1	Threats Addressed by Signed Attributes.....	26
4.1.2	Threats Addressed by Encrypted Attributes.....	26
4.2	Potential Threats Against IF-M use of CMS.....	27
4.2.1	Cryptography.....	27
4.2.2	Threats to Keys	27
4.2.3	Denial of Service	28
5	References	29
5.1	Normative References	29
5.2	Informative References.....	29

1 Scope and Audience

The Trusted Network Connect Work Group (TNC-WG) has defined an open solution architecture that enables network operators to enforce policies regarding the security state of endpoints in order to determine whether to grant access to a requested network infrastructure. This security assessment of each endpoint is performed using a set of asserted integrity measurements covering aspects of the operational environment of the endpoint. Part of the TNC architecture is IF-M, a standard protocol between Integrity Measurement Collectors (IMC) on the TNC Client to the Integrity Measurement Verifiers (IMV) on the TNC Server. This document defines and specifies a security mechanism for protecting attributes carried within the IF-M protocol, based on Cryptographic Message Syntax [RFC3852].

Architects, designers, developers and technologists who wish to implement, use, or understand IF-M security with CMS should read this document carefully. Before reading this document any further, the reader should review and understand the TNC architecture as described in [IF-ARCH] and the IF-M Protocol: Bindings to TLV [IF-M] and the CMS specification.

2 Background

2.1 Purpose of IF-M Security

The IF-M Security specification defines a set of IF-M attributes that enable IMC or IMV senders of IF-M messages to optionally protect the authenticity, integrity and confidentiality of one or more attributes' contents. This specification defines a layered security attribute protection scheme that leverages several defined attributes from the IF-M protocol specification. The encoding of each protected attribute follows the Cryptographic Message Syntax (CMS) format as defined in the IETF. This specification defines a profile of the CMS specification which is explicitly supported within the IF-M protocol in order to meet the security considerations described in the IF-M protocol specification.

2.2 Supported and Unsupported Use Cases

The IF-M specification describes several use cases that the IF-M protocol supports. Of these use cases, the remote IMV use case (section 2.2.3 of IF-M) discusses the need for security protection of attributes exchanged between the IMCs and IMVs. This specification defines an IF-M security protocol designed to address this use case. Therefore no additional use cases are required in this specification.

2.3 Requirements

The following are the requirements that an IF-M security protocol must meet in order to successfully protect IF-M exchanged attributes.

- Flexibility

The IF-M protocol defines a set of attributes that can be exchanged between IMCs and IMVs within the TNC architecture. This set of attributes is extensible allowing for both vendor-defined and standards-based attributes to exist without collisions or misinterpretation. The IF-M security protocol **MUST** be capable of protecting the authenticity, integrity and confidentiality of both vendor-defined and standards-based attributes from a wide variety of types of attacks (discussed in the IF-M security considerations section). The IF-M security protocol **SHOULD** be able to be flexible enough to protect the currently defined and future attribute values in an IF-M message. The IF-M security protocol **SHOULD** be capable of providing per attribute protection so that sensitive attribute can be given stronger protection if desired by a deployer.

- Secure

This specification is specifically about the security protection of IF-M message contents, so security requirements are described throughout the document. The IF-M security protocol provides authentication, integrity and optional security protections for one or more IF-M message attributes. These protections are offered to address the remote IMV use case described in the IF-M specification.

- Attribute Based

The IF-M protocol expects that each defined IF-M security protocol be designed to fit within the extensible attribute framework and/or component name framework provided by the IF-M specification. Therefore the IF-M security protocol **MUST** only define new IF-M attribute types to establish security context, perform any required security services, and to protect the IF-M message attributes transmitted. The IF-M security **MUST NOT** modify the underlying IF-M protocol to achieve its protections.

- Extensible

The IF-M security protocol **MUST** allow for use of alternative or new cryptographic algorithms as they become available in the future. The specification will define minimum required set of algorithms and allow for the future addition of new algorithms that can be unambiguously recognized by IF-M message

recipients. Because these algorithms will not initially be specified, some sets of IF-M message senders and recipients might support different algorithms. Therefore the IF-M security protocol SHOULD provide a method for senders of protected IF-M attributes to determine what algorithms are supported by the intended recipients.

- Efficiency

IF-M security protocol SHOULD be capable of supporting security protection of one or more attributes destined for a particular recipient. This enables a recipient of several security protected attributes to leverage a single set of cryptographic information (e.g. keys) that is used to protect multiple attributes. This capability allows more efficient creation and processing (e.g. signature verification, integrity hash and decryption) of the attribute set and reduces the amount of security negotiation that needs to occur prior to sending the protected attributes.

2.4 Rationale for Use of CMS

CMS was selected to protect IF-M attributes because of its suitability to provide security protections for a messaging oriented protocol. Messaging protocols may wish to avoid a potentially lengthy set of roundtrip message exchanges to setup a security association prior to being able to send protected messages. IF-M message senders may only wish to protect one of several attributes exchanged with another party. Such additional roundtrips can cause latency issues that could result in timeouts or other undesirable behavior in some underlying protocols (e.g. 802.1X).

It is envisioned that during an IF-M message dialog, several messages might be exchanged that do not need (or require different) security protections. For example, a deployment may not wish to protect messages requesting measurements, but may wish to protect the resulting measurement data and/or any final decision related attributes. In order to allow for each message's attributes to be protected independently, a more granular security mechanism was required. Note that the use of a protected session oriented protocol, such as TLS, could be provided by the IF-T protocol.

CMS has been used in the IETF to protect a number of messaging oriented protocols (e.g. MIME messages, firmware upgrades) so it was believed to be a good standards-based approach for protecting IF-M message attributes. This specification defines how CMS is applied to IF-M to provide origin authentication, integrity and optional confidentiality of one or more attributes. The use of other security protocols is plausible in the future; consequently this protocol will ensure that CMS protected attributes can be easily recognized by IMCs and IMVs.

2.5 Keywords

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 [KEYWORDS]. This specification does not distinguish blocks of informative comments and normative requirements. Therefore, for the sake of clarity, note that lower case instances of must, should, etc. do not indicate normative requirements.

3 IF-M Attributes Supporting CMS

This section discusses how CMS is used to protect IF-M message attributes. The IF-M protocol specification defines how IMCs and IMVs can exchange messages to perform an assessment. Each message is delivered to interested IMC(s) or IMV(s) based upon the component type (e.g. firewall) indicated in the PB-TNC message type.

Within each IF-M message is a set of one or more attributes expressed in a type-length-value (TLV) format. The attribute type indicates the format and semantics of the attribute's value. IF-M defines an extensible attribute type field allowing for both vendor defined and standard attributes to be included and easily identified by IF-M message recipients. For more information, see the IF-M specification. This specification defines the syntax and semantics of three new attribute types necessary to support CMS protection of IF-M message attributes. The following subsection summarizes each attribute and how it is linked to the IF-M protocol.

3.1 Linkage to IF-M Protocol

The IF-M protocol specification defines identifiers for several attribute types (see section 4.2 of the IF-M specification) that are expected to be specified and defined in IF-M security protocol specification. Specifically the following attributes are intended for use by the security protocol:

IF-M Attribute Type	Description
CMS Protected Content	Contains CMS content protecting a set of one or more IF-M attributes
Security Capabilities	Summarizes the cryptographic capabilities (e.g. algorithms) supported by the attribute sender
CMS Error Code	Includes protected error information specific to the IF-M security protocol. This error attribute is only used when errors occur while processing IF-M security attributes (protected with CMS). This attribute allows for signed and integrity protected error messages to be sent.

Through the use of these attributes, IMC and IMV can provide security protection to any attribute defined in the IF-M specification. This protection is provided by encapsulating one or more IF-M attributes within a CMS Protected Content attribute offering the cryptographic signatures and optionally encryption over the content. By using the Security Capabilities attribute, parties intending to send a CMS Protected Content attribute can proactively determine whether the recipient is capable of decrypting and verifying the CMS protected message before sending it.

3.2 TCG TNC OID Hierarchy

The IF-M security protocol makes uses of several OID values within the attributes discussed in section 3.1. For OIDs that reference TCG TNC values they fall under the following hierarchy:

-- OID Hierarchy for TNC IF-M Security

tcg OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) international-organization(23) tcg(133) }

tcg-tnc OBJECT IDENTIFIER ::= { tcg 16 }

tcg-tnc-ifm OBJECT IDENTIFIER ::= { tcg-tnc 1 }

tcg-tnc-ifm-cms OBJECT IDENTIFIER ::= { tcg-tnc-ifm 1 }

3.3 CMS Protected Content Attribute

The CMS Protected Content attribute allows an IMC or IMV to send one or more IF-M attributes protected within a CMS encapsulated object. This specification identifies the profile of CMS's capabilities that are necessary to provide authentication and integrity protection and optionally confidentiality protection of one or more IF-M message attributes. Some aspects of CMS are not required to achieve these security protections, and so for simplicity these are explicitly excluded from the IF-M security standard.

Because this specification describes a profile of CMS that directly applies to the protection of IF-M attributes, it does not attempt to repeat all the encoding and processing rules described by the CMS specification. Nonetheless these encoding and processing rules are required unless explicitly modified or excluded by this specification. The intention behind most of the profiling of CMS in this specification is to exclude portions of CMS or to alter (raise or remove) requirements for particular fields within the CMS structures to reflect their use in protecting IF-M attributes.

3.3.1 CMS Content Info and Content Types

Every CMS Protected Content attribute MUST begin with a ContentInfo structure. The ContentInfo structure encapsulates the top level ContentType identifier and the content itself. CMS allows nesting of content types so that other levels of content types may exist within the top level content field. The ContentInfo structure is described in section 3 of the CMS specification and is repeated below for the reader's convenience:

```
ContentInfo ::= SEQUENCE {  
    contentType ContentType,           -- signed-data or enveloped-data  
    content [0] EXPLICIT ANY DEFINED BY contentType } -- actual content of indicated type  
ContentType ::= OBJECT IDENTIFIER
```

Each contentType value is an OID that indicates the syntax and semantics of the associated content field. The CMS specification defines six different contentType values and formats while allowing more to be defined in other specifications. IF-M message security protection requires the support of only two ContentType values: signed-data and enveloped-data. The signed-data contentType provides origin authentication and integrity protection of the included set of one or more IF-M attributes. The signed-data protection MUST be present in all CMS Protected Content attributes.

Optionally, an IMC or IMV may wish also to protect the confidentiality of a signed set of attributes. This can be accomplished by encapsulating the signed-data content within an enveloped-data ContentType. The result is an encrypted version of the signed set of attributes being included in the CMS Protected Content. Therefore, all IMC or IMV supporting the CMS Protected Content MUST be capable of supporting the creation and/or processing of CMS Protected Content attributes containing either:

- o signed-data content (signed attributes)
- o signed-data content encapsulated within enveloped-data content (signed and encrypted attributes)

Other CMS contentType values MAY be supported but are outside the scope of this specification so are unlikely to offer interoperability. Implementations receiving a CMS Protected Content containing an unrecognized contentType MUST discard the attribute and SHOULD return a CMS Error Code attribute containing an errorCode of badContentType.

3.3.2 CMS Signed-Data

IF-M attributes that require authentication and integrity protection MUST use the signed-data CMS content type within a CMS Protected Content IF-M message attribute. This section defines the subset of the CMS signed-data features required for protection of IF-M message attributes. Readers should refer to section 5

of the CMS specification for background on the required CMS processing rules that form the basis for the profile discussed in this subsection.

The CMS signed-data content type has the following structure present in the content field of ContentInfo:

```
SignedData ::= SEQUENCE {
    version CMSVersion,                -- Indicates syntax of this structure
    digestAlgorithms DigestAlgorithmIdentifiers, -- Should be empty
    encapContentInfo EncapsulatedContentInfo, -- Signed IF-M attributes
    certificates [0] IMPLICIT CertificateSet OPTIONAL, -- Required for IF-M security
    crls [1] IMPLICIT RevocationInfoChoices OPTIONAL, -- Certificate revocation information
    signerInfos SignerInfos }         -- Single signature information
```

To simplify support and processing of CMS protected IF-M message attributes, the following signed-data field restrictions are introduced by this specification:

CMS signed-data Field	Restrictions from CMS Defined Functionality
CMSVersion	This field contains a value following the algorithm described in section 5.1 of the CMS specification. This profile does not support certificates and CRLs of type other nor attribute certificates, therefore it is expected that this value will normally be 3 or 1 (depending on the type of SignerIdentity used).
digestAlgorithms	This field SHOULD be empty indicating that recipients need to refer to the signerInfos field to determine the digest algorithm used by the signer. This field MAY contain a single DigestAlgorithmIdentifier OID corresponding to the digest algorithm used during the single signature computation included within the attribute. If present, this field MUST match the signerInfos's digestAlgorithms field described below.
encapContentInfo	This field contains another pair of content type and content (see section 5.2 of the CMS specification for details). The content type (referred to as eContentType) MUST be set to the id-data ContentType OID and the content field MUST only contain one or more IF-M message attribute(s) covered by the signature. The content field MUST be present so it is not optional as stated by CMS. The encoding of the IF-M message attributes within the content field will match their definition from the IF-M specification so does not require DER or BER encoding.
certificates	This field MUST contain the signer's X.509 version 3 identity certificate and SHOULD also contain the set of certificates leading from the signer's certificate to a recipient trusted certificate authority as discussed in the CMS specification. These certificate(s) enable the recipient(s) to perform path validation of the signer's certificate as part of its trust decision. It is expected that the recipient(s) of the message will have other methods for obtaining necessary certificates in the event that this field does not contain a sufficient set of certificates to complete validation. This field SHOULD NOT contain attribute certificates although they are allowable under standard CMS.
crls	No additional restrictions are placed on this field.

signerInfos	<p>A single SignerInfo structure MUST be included in the signerInfos field. Multiple signers MUST NOT be included. IF-M security does not support multiple signers so only a single SignerInfo can be present (not a set as described by CMS). The included digestAlgorithm MUST match the value included in the digestAlgorithms field above if one is present.</p> <p>IF-M recipients SHOULD fail and return a CMS Error Code IF-M message attribute containing a digestAlgorithmMismatch error code if the signerInfos's digestAlgorithm does not match the specified digestAlgorithm value.</p> <p>The unsignedAttrs field MUST NOT be used as they are not necessary to meet the requirements of IF-M security. The ifmNonce CMS attribute MUST be included to provide replay protection. Other signedAttrs field MAY be used to include additional supporting information about the protection on the CMS content such as SMIMEEncryptionKeyPreference and SMIMEEncryptionKeyPreference.</p>
-------------	---

3.3.2.1 CMS Signed-Data Example

This section provides an example of the contents of a Remediation Instructions IF-M attribute encapsulated within CMS signed-data content when sent by IF-M. For this example, the top level IF-M attribute is of type CMS Protected Content which indicates that CMS content is present in the Attribute Value field. The CMS content has a ContentType OID of signed-data and content field that provides a signature across a single encapsulated Remediation Instructions IF-M attribute. Due to the nesting of structures, this example is broken down into several tables each indicating where it encapsulates the subsequent table. The left column of each table shows the ASN.1 for the fields and the right column provides the example values.

Initially, each recipient of the example IF-M message would receive a message containing a single attribute of type Protected CMS Content. Inside the value portion of the Protected CMS Content TLV contains the following:

CMS ContentInfo Field	Example Field Value
ContentInfo ::= SEQUENCE {	
contentInfo ContentType,	id-signedData OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs7(7) 2 }
content[0] EXPLICIT ANY DEFINED BY contentInfo }	The contents of this field are described in table 2.

Table 1: CMS ContentInfo Header Example

ContentInfo's Content Field	Example Field Value
SignedData ::= SEQUENCE {	
version CMSVersion,	Set to 1
digestAlgorithms DigestAlgorithmIdentifiers,	Empty (0 length field)
encapContentInfo EncapsulatedContentInfo,	The contents of this field are described in table 3.
certificates [0] IMPLICIT CertificateSet	List of X.509 certificates including the sender's certificate

OPTIONAL,	and any parent CA certificates leading to a root trusted by the sender.
crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,	Revocation information for signer's certificate
signerInfos SignerInfos }	One set of signer information including the signer's identity and algorithms used in the signature. This field can also carry signed and unsigned CMS attributes. For this example, the SignerInfo instance uses issuerAndSerialNumber to denote the signer's certificate. The ifmNonce signed CMS attribute is included for replay protection. No unsigned attributes are included.

Table 2: CMS Content Field Example

SignedData's EncapsulatedContentInfo Field	Example Field Value
ContentInfo ::= SEQUENCE {	
contentType ContentType,	id-data OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs7(7) 1 }
content[0] EXPLICIT ANY DEFINED BY contentType }	This field contains the IF-M message attribute(s) included in the CMS signature. For this example, it contains the Remediation Instructions attribute as defined by the IF-M specification.

Table 3: Contents of SignedData's EncapsulatedContentInfo Example

3.3.2.2 Signed-Data Required Algorithms

In order to enable interoperability between independent implementations, this subsection defines the following set of algorithms that IF-M security compliant implementations are expected to support. Additional algorithms and key lengths MAY be supported.

Purpose	Algorithm (Length)	Requirements Level	Algorithm Definition
digestAlgorithm	SHA-1 (160)	MUST (May become SHOULD in future)	FIPS 180-1 RFC 3370, Section 2.1
	SHA-256 (256)	MUST	FIPS 180-2, IETF I-D[SHA-256]
signatureAlgorithm	RSA (2048)	MUST	RFC 3370, Section 3.2 PKCS #1 V1.5
	ECDSA (256)	SHOULD	FIPS 186-2, RFC 5008, Section 3

3.3.3 CMS Enveloped-Data

IF-M message attributes that require confidentiality protection MUST use the enveloped-data CMS content type to encapsulate and encrypt the signed-data content. IF-M security does not try to provide a confidentiality only security service, and therefore enveloped-data is used only in conjunction with signed-data (authentication and integrity protected) content. This section defines the subset of the CMS enveloped-data features required for the protection of IF-M message attributes already protected within a signed-data object. When a feature isn't specifically excluded or restricted by this specification, implementations MUST follow the processing rules defined in the CMS specification.

The CMS enveloped-data content type is defined in section 6.1 of the CMS specification as having the following structure:

```

EnvelopedData ::= SEQUENCE {
    version CMSVersion,                -- Indicates elements included
    originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL, -- Certificates for originator
    recipientInfos RecipientInfos,     -- Decryption keys
    encryptedContentInfo EncryptedContentInfo, -- Encrypted signed-data
    unprotectedAttrs [1] IMPLICIT UnprotectedAttributes OPTIONAL } -- Not used
    
```

To simplify support and processing of encrypted CMS protected IF-M message attributes, the following restrictions from full CMS are imposed on the enveloped-data content type is used:

CMS EnvelopedData Field	Limitation from CMS Defined Functionality
CMSVersion	This field contains a value derived from the algorithm described in section 6.1 of the CMS specification. Because this profile does not support certificates and CRLs of type other and unprotectedAttrs MUST NOT be used its expected that this value will normally be 0.
originatorInfo	<p>This field MUST contain the signer's X.509 version 3 identity certificate and SHOULD also contain the set of certificates leading from the signer's certificate to a recipient trusted certificate authority as discussed in the CMS specification. These certificates enable the recipient(s) to perform path validation of the signer's identity certificate. It is expected that the recipient(s) of the message will have other ways to obtain necessary certificates in the event that this field does not contain a insufficient set of certificates to complete validation. This field SHOULD NOT contain attribute certificates despite being allowable under standard CMS.</p> <p>Optionally this field may also include CRL information used to check the validity of the certificates presented by the originator. This specification does not change the CMS specification handling of CRLs.</p>
recipientInfos	This field contains a set of per recipient information necessary to process the encrypted content. This field contains the encrypted key destined for each recipient to be used to decrypt the encryptedContentInfo. This specification does not change the CMS processing of this field so readers should refer to section 6.2 of the CMS specification for details and information about handling of different key management techniques.
encryptedContentInfo	This field contains the encrypted version of the signed-data content together with information about the encryption algorithm used. The use of the encryptedContentInfo field is the same as specified in CMS except that this field MUST not be empty and MUST contain the encrypted signed-data (in a id-data content type). This field MUST NOT contain content that is not signed as it could be subject to undetectable integrity based attacks.
unprotectedAttrs	The CMS specification defines this field as optional. For IF-M security, this field MUST NOT be used.

3.3.3.1 CMS Enveloped-Data Example

This section shows an example of an encrypted and signed set of IF-M message attributes. Rather than duplicating the signed-data example from section 3.3.2.1, the following tables focus on the encrypted-data content and highlights where the signed-data is included. Initially each recipient of the example IF-M message would receive a message containing a single attribute of type Protected CMS Content. The value portion of the Protected CMS Content TLV would contain the following:

CMS ContentInfo Field	Example Field Value
ContentInfo ::= SEQUENCE {	
contentType ContentType,	id-envelopedData OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs7(7) 3 }
content[0] EXPLICIT ANY DEFINED BY contentType }	The contents of this field are described in table 5.

Table 4: CMS ContentInfo Header Example

ContentInfo's Content Field	Example Field Value
EnvelopedData ::= SEQUENCE {	
version CMSVersion,	Set to 0
originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,	List of X.509 certificates including the signer's certificate and potentially several parent CA certificates enabling the recipient to complete chain validation.
recipientInfos RecipientInfos,	This field contains encrypted versions of the keys associated with each recipient that are used to decrypt the encryptedContentInfo's content. Normally it is expected that a single recipient will be involved with a CMS protected message so this includes only one recipientInfo.
encryptedContentInfo EncryptedContentInfo,	The content of this field is and encrypted version of the IF-M message attributes as described in table 6.
unprotectedAttrs [1] IMPLICIT UnprotectedAttributes OPTIONAL }	This field is empty.

Table 5: CMS Content Field Example

Content's EncryptedContentInfo Field	Example Field Value
ContentInfo ::= SEQUENCE {	
contentType ContentType,	id-signedData OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs7(7) 2 }
contentEncryptionAlgorithm ContentEncryptionAlgorithmIdentifier,	joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3)_nistAlgorithms(4) aes(1) aes128(2)
encryptedContent [0] IMPLICIT EncryptedContent OPTIONAL }	The contents of this field are an encrypted version of the information described in table 2. While this field is optional in CMS it is required for IF-M security (external signatures are not supported).

Table 6: Contents of EnvelopedData's EncryptedContentInfo Example

3.3.3.2 Enveloped-Data Required Key Management

This subsection discusses the required key management schemes as defined by the CMS specification. The key management scheme is used to establish a key that is shared by the communicating parties enabling them to perform cryptographic operations on their communications. For this specification, the exchanged content is signed-data containing one or more IF-M message attributes protected by a signature. In order to allow the end parties to use different types of credentials to protect this key negotiation, several key management schemes are defined. This specification follows the requirements from section 6.2 of the CMS specification which states:

“Implementations **MUST** support key transport, key agreement, and previously distributed symmetric key-encryption keys, as represented by ktri, kari, and kekri, respectively. Implementations **MAY** support the password-based key management as represented by pwri. Implementations **MAY** support any other key management technique as represented by ori.”

3.3.3.3 Enveloped-Data Required Algorithms

In order to enable interoperability between independent implementations, this subsection defines the mandatory to implement algorithms for each key management scheme and content encryption. The password-based key management scheme **MAY** be supported while the key transport, key agreement and previously distributed symmetric KEK schemes **MUST** be supported by compliant implementations. The following table defines the cryptographic algorithms that **MUST** be support for each key management scheme if an implementation wishes to be compliant. Other cryptographic algorithms, key lengths and key management technique **MAY** be supported.

Key Management Scheme	Algorithm (Length)	Requirements Level	Algorithm Definition
Key Transport	RSA wrapped AES CEK (2048)	MUST	RFC 3565, Section 2.2
Key Agreement	ESDH with AES KEK derivation (128 & 256)	MUST	RFC 3565, Section 2.3
Previously Distributed Symmetric Key-Encryption Key	AES key wrapping (128 & 256)	MUST	RFC 3565, Section 2.4
Passwords-based	Password derived AES KEK (128 & 256)	MUST (if supported)	RFC 3565, Section 2.5

The above described key management schemes are used to establish a symmetric content encryption key that protects the signed IF-M attributes. IF-M security compliant implementations **MUST** support the use of the following algorithms for content encryption:

Purpose	Algorithm (Length)	Requirements Level	Algorithm Definition
Content Encryption	AES (128 & 256)	MUST	RFC 3565, Section 2.1

3.4 Security Capabilities Attribute

The Security Capabilities attribute type allows an IMC or IMV to determine the supported set of cryptographic algorithms supported by the recipient(s) prior to creating a protected message. This provides a simple cryptographic algorithm discovery mechanism to assist the sender's selection of an algorithm consistent with the sender's policy and supported by the recipient. The algorithm list is encapsulated within a signed CMS message that the recipient can use to verify the authenticity and integrity of the algorithm list.

If confidentiality protection of the Security Capability attribute is desired, the sender may encapsulate it within an enveloped-data content type as described in section 3.2.3 of this specification. Note however that the sender of this attribute will not be aware of the cryptographic algorithms supported by the recipient (since its replying to a cryptographic discovery request). For this reason, implementations MAY support encryption of the security capabilities content using the enveloped-data; however, one of the mandatory encryption algorithms SHOULD be used to maximize the possibility that the recipient supports the algorithm.

In order for an IMC or IMV to determine the security capabilities supported by recipient(s), the IMC or IMV would include the Security Capabilities attribute type in an IF-M Attribute Request attribute (see section 4.2.2 of the IF-M specification). The Attribute Request MAY include other IF-M attribute types in the list if appropriate. The recipients of the Attribute Request attribute containing the Security Capabilities attribute type respond with the Security Capabilities attribute described in this section. NOTE that typically an IMC does not send an Attribute Request attribute to an IMV during an assessment as it normally is responding to requests for attributes. However if an IMC wishes to determine the security algorithms supported by recipient IMV(s), it may send a Request Attribute containing only the Security Capabilities attribute type.

The IF-M Security Capabilities attribute MUST consist of a single CMS signed-data content containing a single signed attribute in the signerInfo and an empty eContent (no other data) within the encapContentInfo. The Security Capabilities attribute SHOULD be signed with mandatory signature algorithm to ensure that the recipient will be able to verify the signature. Note that the CMS signature also includes a field called signed attribute (signedAttrs) that is information outside of the CMS content. This specification does not use unsigned CMS attributes but does use the signed CMS attribute to convey the supported security algorithms. For IF-M security, the attributes described in the IF-M specification are present in the data portion (eContent in encapContentInfo) of the CMS content; whereas the CMS defined attributes exist outside of the eContent section such as in the signerInfos field, and are represented by ASN.1 in this specification.

This specification defines a CMS attribute called ifmSecurityCapabilities that contains a prioritized list of the cryptographic algorithms supported for various purposes by the sender. The purpose of each algorithm is reflected by the OID definition and can include: signing, data encryption, key wrapping and digesting. The prioritized algorithm list MUST be grouped according to the algorithm's purpose to ease processing by the recipient. The CMS ifmSecurityCapabilities attribute is based on the SMIMECapabilities attribute defined in section 2.5.2 of the SMIME specification[RFC3851]. The processing rules for the SMIMECapabilities apply to the ifmSecurityCapabilities CMS attribute unless stated otherwise in this section.

3.4.1 ifmSecurityCapabilities Within Signed-Data

The ifmSecurityCapabilities CMS attribute exists within the signed-data content type described in section 3.2.2 of this specification. Rather than repeating all the detail of section 3.2.2, this section will focus on the differences between a signed set of IF-M attributes and an ifmSecurityCapabilities CMS attribute encapsulated within signed-data content.

The ifmSecurityCapabilities CMS attribute is present in the signedAttrs field; consequently it is included in the CMS signature. This enables recipients to detect modification of the sender's claimed security capabilities and to authenticate the sender's identity. Unlike normal signed-data content, the ifmSecurityCapabilities CMS attribute MUST exist in all Security Capabilities attributes and MUST be the only CMS attribute present in the signedAttrs list besides the required ifmNonce CMS (replay protection) attribute. This differs from normal signed-data content that is allowed to include other CMS attributes.

Another difference concerns the use of the encapContentInfo's eContent field. In the case of signed-data content, this field normally includes the IF-M message attributes being protected. For a Security Capabilities attribute, the eContent field MUST be empty. This is because the entire purpose of this attribute is to indicate the security capabilities of the sender and those capabilities are included in the signedAttrs field. No other IF-M message attributes are allowed to be encapsulated in this attribute. Note that an IF-M message can contain several attributes so other attributes could be sent in addition to the Security Capabilities attribute.

3.4.2 ifmSecurityCapabilities ASN.1

The ifmSecurityCapabilities content mirrors the SMIMECapabilities attribute as described in section 2.5.2 of the SMIME specification. The ASN.1 defined for the ifmSecurityCapabilities attribute is as follows:

```
ifmSecurityCapabilitiesOID OBJECT IDENTIFIER ::= { tcn-tnc-ifm-cms 1 }  
  
ifmSecurityCapability ::= SEQUENCE {  
    capabilityID OBJECT IDENTIFIER,           -- OID of the algorithm supported  
    parameters ANY DEFINED BY capabilityID OPTIONAL } -- Optional parameters associated  
with algorithm  
  
ifmSecurityCapabilities ::= SEQUENCE OF ifmSecurityCapability
```

The ifmSecurityCapabilities CMS attribute is simply a prioritized (preference order) list of OIDs and associated cryptographic parameters of the algorithms supported by the sender. Ordering the list by preference provides another piece of information to those wishing to send protected information to the sender. This specification leverages the CMS Algorithms specification defined set of ASN.1 for the OIDs and parameters for the security algorithms represented in this list. For more information see section 7 of the CMS Algorithm specification and the AES algorithm specification. An in-depth discussion of the SMIMECapabilities CMS attribute that parallels the ifmSecurityCapabilities CMS attribute is included in the SMIME specification in section 2.5.2

3.5 CMS Error Code Attribute

This IF-M attribute allows a recipient of an invalid security protected IF-M message to send an integrity protected error response indicating the reason for the failure. In order to return protected error information related to the processing of CMS Protected Content attributes, IF-M security encapsulates the error code within a signed attribute, itself encapsulated within CMS signed-data content. Using a signed attribute allows recipients to verify the integrity and origin authentication of error status preventing spoofing and other related attacks. In some uncommon situations, recipients may not be able to verify the signature (e.g. the use of an unsupported digest algorithm) or establish trust in the sender (e.g. no common trust anchor) but at least the recipient can view the returned error code and decide whether to trust it and therefore how to act on it. Care should be taken when trusting information whose integrity can not be verified as it could leave the recipient open to various attacks.

All CMS processing errors MUST result in a response IF-M message containing a CMS Error Code Attribute. The CMS Error Code attribute MUST only contain a single CMS ContentInfo of content type signed-data. The Signed-Data element MUST contain an empty eContent and include only the ifmNonce CMS attribute and the ifmErrorCode CMS attribute in the signerInfo field. The CMS Error Code attribute SHOULD be signed with mandatory signature algorithm to ensure that the recipient will be able to verify the signature.

3.5.1 ifmErrorCode Within Signed-Data

The ifmErrorCode CMS attribute exists within the signed attributes portion of the signed-data content type. Rather than repeat all the detail of section 3.2.2, this section will describe the differences between a signed set of IF-M attributes and an ifmErrorCode CMS attribute housed within signed-data content.

The ifmErrorCode CMS attribute is an attribute that is present in the signedAttrs field so it is included in the CMS signature. This enables recipients to detect modifications of the error information and to authenticate the sender's identity. Unlike normal signed-data content, the paTncSecurityErrorCode CMS attribute MUST exist in all CMS Error Code attributes and MUST be the only CMS attribute present in the signedAttrs list

besides the required ifmNonce CMS (replay protection) attribute. This differs from normal signed-data content that is allowed to include other CMS attributes.

The other difference is the use of the encapContentInfo's eContent field. Normally in signed-data content, this field must include the IF-M message attributes being protected. For a CMS Error Code attribute, the eContent field MUST be empty. This is because the sole purpose of this attribute is to carry the error code related to an earlier IF-M message to the recipient and the error information is included in the signedAttrs field. No other IF-M message attributes (e.g. Request Attribute) are allowed to be encapsulated in this attribute. Note that an IF-M message can contain several attributes so other attributes could be sent in addition to the CMS Error Code attribute.

3.5.2 ifmErrorCode ASN.1

The ifmErrorCode CMS attribute is placed in the signerInfo's signedAttrs field. The signedAttrs field is included in the signature applied so that the recipient can verify the authenticity and integrity of the information before taking action. The following describes the syntax and semantics of the ifmErrorCode CMS attribute.

```
ifmErrorCode ::= SEQUENCE {
    vendorID OBJECT IDENTIFIER,      -- Name space of the error code
    status  errorCode,              -- CMS processing error code
    ContentInfo originalContent OPTIONAL } -- Copy of content that caused error
```

The following table describes each field within the ifmErrorCode

ifmErrorCode CMS Attribute Fields	Description
vendorID	This field MUST contain the OID of the vendor whose name space scopes the errorCode value included in this CMS signed attribute. This allows vendors to define their own error codes outside of the standard name space. The TCG TNC OID 'tcg-tnc' defined in section 3.2 MUST be used when TCG standard errorCode values are present.
errorCode	This field MUST contain the error code reflecting the error that occurred while processing the CMS message. The TCG TNC standard error codes are listed in section 3.5.3.
originalContent	This field SHOULD contain the contents of the CMS content that cause the error. If the original content is large and the deployment is bandwidth constrained this field MAY be empty.

3.5.3 TCG Standard ifmErrorCode Values

The following table lists the TCG TNC standard error codes for use in the errorCode field of an ifmErrorCode value. Other values MUST NOT be used in conjunction with the TCG TNC OID vendorID unless defined in a TCG specification. Vendors MUST use their assigned OID in the vendorID to use values not defined by TCG TNC. The following error codes were initially based on early work using CMS for Trust Anchor Management Protocol [TAMP].

```
ErrorCode ::= ENUMERATED {
    reserved          (0), -- This value MUST NOT be used
    decodeFailure    (1), -- Unable to decode content, content type doesn't match
    provided content
```

badContentInfo	(2),	-- Unknown or invalid ContentInfo syntax in provided content
badSignedData	(3),	-- Unknown, invalid or non-compliant signed-data format used
badEnvelopedContent	(4),	-- Unknown, invalid or non-compliant enveloped-data format
badCertificate	(5),	-- Invalid syntax used for included certificates
badSignerInfo	(6),	-- Invalid or unsupported SignerInfo syntax
badSignedAttrs attributes	(7),	-- Invalid, unsupported or non-compliant use of signed attributes
badUnsignedAttrs	(8),	-- Non-compliant use of unsigned attributes
missingContent	(9),	-- Non-compliant empty eContent in signed-data content
noTrustAnchor	(10),	-- Recipient lacks trust anchor associated with signer's cert
notAuthorized	(11),	-- Requestor's identity not authorized to perform operation
badDigestAlgorithm	(12),	-- digestAlgorithm used is unsupported or invalid
badSignatureAlgorithm	(13),	-- signatureAlgorithm used is unsupported or invalid
unsupportedKeySize	(14),	-- Key used for signature or encryption is an unsupported size
unsupportedParameters	(15),	-- Unsupported signature or encryption algorithm parameters
signatureFailure	(16),	-- Recipient computed signature does not match one provided
decryptionFailure	(17)	-- Unable to decrypt content using content encryption key
keyManageFailure	(18)	-- Unable to determine key encryption key
badKeyManage	(19)	-- Unknown or unsupported key management technique used
nonceMissing	(20)	-- Received content lacking required nonce attribute
invalidNonce	(21)	-- Unexpected or invalid nonce received
repeatedNonce	(22)	-- Nonce was recently used so it rejected as a potential replay
nonceOrdering	(23)	-- Non-initial nonce that was not one greater than prior nonce
badContentType	(24)	-- Invalid or unsupported contentType found
digestAlgorithmMismatch	(25)	-- SignerInfos and digestAlgorithm algorithm mismatch
missingSignature	(29),	-- Signed-data missing required signature value
resourcesBusy	(30),	-- Recipient unable to process content due to resource issues
versionNumberMismatch	(31),	-- Version in received message was invalid or unsupported
revokedCertificate	(33),	-- Certificate used was revoked by issuer
other	(127) }	-- Message could not be process for reason other than above.

3.6 Nonce CMS Attribute

Unlike the above three IF-M attributes, this attribute is a CMS attribute that is located in the signedAttrs field of the signed-data content within other IF-M security protected attributes. For example a CMS Protected Content attribute would include a Nonce CMS attribute in its signedAttrs field to detect replay attacks.

The Nonce CMS attribute allows the sender of an IF-M protected message to include a nonce that can be used by the recipient to detect a replay attack. The Nonce CMS attribute MUST be used in all IF-M security messages as defined within this specification. Recipients of CMS signed-data protected attributes lacking a

Nonce CMS attribute MUST create an error and not process the CMS message. The Nonce CMS attribute is a signed attribute that MUST exist within any signed-data content type including the IFM Security Capabilities, CMS Error Code, and CMS Protected Content attributes. The Nonce CMS attribute MUST NOT be used in the enveloped-data content type to simplify processing of such messages because the enveloped-data will encapsulate signed-data content that must include the nonce anyway.

The value of the nonce MUST be unpredictable to third parties so MUST NOT be based on network observable information. Use of good sources of entropy is highly desired however implementations may use persistently stored sequence numbers that do not repeat even across reboots and other disruptive events). The Nonce CMS attribute contains two separate values each under the control of an IMC or IMV. This allows both sides of the message exchange to provide entropy and receive replay protection.

The initial sender of a CMS message generates its nonce and includes it in the Nonce CMS attribute with a zero value for the other party. When initially responding to a CMS protected message containing a zero value nonce, the responder generates its nonce and includes it in the reply together with a copy of the nonce sent by the other party. If the initial sender wishes to send another signed-data message to the other party it creates a Nonce CMS attribute by copying the other party's nonce and by incrementing its own nonce by one. If the resulting value is 2^{32} then it should randomly generate a new nonce. This process continues until the completion of an assessment. Implementations unable to generate a good nonce value MAY use persistent sequence numbers providing that it can ensure that no repeated values are used in a predictable manner.

When a CMS message recipient receives a message, it must check the message's nonce attribute to ensure that its nonce matches the value of the nonce that it sent or contains a zero. Similarly it must also check that the nonce created by its peer is one greater than the last received assessment message nonce if it is not the first CMS protected message of the assessment.

3.6.1 ifmNonce CMS Attribute ASN.1

The ifmNonce CMS attribute is included in every signed-data content (in the signedAttr field) exchanged during a CMS protected attribute exchange during an assessment including the Security Capabilities attribute. The following ASN.1 shows the syntax of the ifmNonce CMS attribute:

```

NonceType ::= INTEGER (0 .. 4294967295)    -- Max is (2^32)-1 since all 1s causes new random
value selection

ifmNonce ::= SEQUENCE {
    imcNonce NonceType,                    -- IMC selected nonce, zero if unknown
    imvNonce NonceType }                  -- IMV selected nonce, zero if unknown
    
```

The following table describes each field within the ifmNonce:

ifmNonce CMS Attribute Fields	Description
imcNonce	<p>This field contains an unpredictable 32 bit unsigned integer of the IMC's choosing. The selection of this value MUST be consistent with the following rules:</p> <p>Initial value during assessment:</p> <ul style="list-style-type: none"> • If IMC is sending an initial CMS protected attribute during an assessment, IMC MUST select an unpredictable, non-zero nonce value for this field. • If IMV is sending an initial CMS protected attribute during an assessment, the IMV MUST set this field to zero. Zero indicates that the IMC has not yet had an opportunity to establish an initial nonce

	<p>value.</p> <p>Non-initial value during assessment:</p> <ul style="list-style-type: none"> • IMC MUST increment by one the prior imcNonce value used during this assessment and if $<2^{32}$ include this value in this field. If 2^{32} is reached, a new unpredictable, non-zero value MUST be selected. The selected value SHOULD be compared against a list of those recently used to avoid causing the recipients to consider this a replay and sending an error. Use of the prior imcNonce + one approach to new nonce selection was done to ease nonce create and replay table maintenance. • IMV MUST copy imcNonce from most recent valid CMS protected message from IMC during the assessment • Recipients MUST verify appropriate nonce used in both fields to detect replay attempts. Recipients SHOULD maintain a table of recently used nonce ranges for each peer.
<p>imvNonce</p>	<p>This field contains an unpredictable 32 bit unsigned integer of the IMC's choosing. The selection of this value MUST be consistent with the following rules:</p> <p>Initial value during assessment:</p> <ul style="list-style-type: none"> • If IMV is sending an initial CMS protected attribute during an assessment, the IMV MUST create an unpredictable, non-zero nonce value for this field. • If IMC is sending an initial CMS protected attribute during an assessment, the IMC MUST set this field to zero. Zero indicates that the IMV has not yet had an opportunity to establish an initial nonce value. <p>Non-initial value during assessment:</p> <ul style="list-style-type: none"> • IMV MUST increment the prior imcNonce value used during this assessment and if $<2^{32}$ include the value in this field. If 2^{32} is reached, a new unpredictable, non-zero value MUST be selected. The selected value SHOULD be compared against a list of those recently used to avoid causing the recipients to consider this a replay and sending an error. • IMC MUST copy imcNonce from most recent valid CMS protected message from IMV during the assessment. • Recipients MUST verify appropriate nonce used in both fields to detect replay attempts. Recipients SHOULD maintain a table of recently used nonce ranges for each peer.

3.6.2 ifmNonce CMS Attribute Example

This section provides a simple example of a nonce exchange. In this example, a single IMV and IMC will participate in a two roundtrip exchange including three CMS protected attribute messages. The sub-bullet in each step describes the pcNonce and pvNonce.

1. IMV sends an unprotected IF-M Request Attribute containing the Security Capabilities attribute type
 - o No nonces involved with this message (unprotected).
2. IMC responds with an IF-M Security Capabilities attribute

- $imcNonce = \text{initial value } X; imvNonce = 0$
- 3. IMV sends an IF-M CMS Protected Content attribute containing an IF-M Request Attribute requesting Product Information about endpoint's operating system
 - Verify X wasn't recently used by IMC
 - $imcNonce = X; imvNonce = \text{initial value } Y$
- 4. IMC responds with an IF-M Product Information attribute encapsulated within a CMS Protected Content attribute
 - Verify Y wasn't recently used by IMV
 - $imcNonce = X+1; imvNonce = Y$
- 5. IMV sends an IMV Assessment Result attribute in a CMS Protected Content attribute
 - Verify $imcNonce$ is last received nonce + 1
 - $imcNonce = X+1; imvNonce = Y+1$

Note that this example does not involve X or Y reaching 2^{32} so no new unpredictable values were required. If this was required the recipient would need to verify that the last nonce value was $2^{32}-1$ and the new value had not been used recently.

Using this algorithm both parties can detect replayed messages from the other party (or an attacking imposter). One further benefit is that the loss of a message during a CMS exchange can be detected by the recipient who can respond to this failure by sending an error message (nonceOrdering) to the sender, who could resend the prior message if appropriate.

4 Security Considerations

This section discusses how the security countermeasures provided by the IF-M security protocol address the threats to IF-M messages as discussed in section 5 of the IF-M specification. This section also discusses some potential threats directly against the IF-M security protocol.

4.1 Countermeasures to IF-M Threats

The IF-M specification discusses a range of potential threats to the IF-M protocol. Some deployment environments may have mitigating controls already in place on the network or have a threat model that accepts the risks. For example, many deployments may deploy cryptographically protected IF-T protocols and trust the TNCC and TNCS not to compromise the attributes exchanged. For deployments that require security protections of the attributes sent between the IMC and IMV, the following sections discuss how the use of CMS can provide the necessary protections.

The following subsections are organized along the capabilities of CMS protected IF-M attributes. This allows a single discussion of the cryptographic protection provided by the countermeasure and a summary of the threats addressed by the countermeasure. The IF-M security binding to CMS leverages the signed-data and enveloped-data content types to provide different levels of protection for one or more attributes. The following subsections discuss how each content type's protections address the IF-M threats.

4.1.1 Threats Addressed by Signed Attributes

The signed-data content type of CMS provides a cryptographic signature around the set of one or more attributes. This cryptographic protection enables the recipient of an IF-M message to detect any changes to the content of the protected attributes that occurred after the data was signed. Similarly the recipient can authenticate the identity of the sender of the attributes, and so is able to detect adversaries attempting to masquerade as a trustworthy origin of the attribute contents.

Section 5.2.2 through 5.2.5 of the IF-M specification discusses potential attacks against the integrity of the attribute exchange by creating falsified attributes, modifying legitimate attributes, inserting attributes within an exchange or replaying prior attributes. The use of a digital signature covering the attributes' content allows each recipient to detect fabricated attributes that were claiming to come from a party other than the authenticated identity. The signer of a set of attributes must have the appropriate credentials in order to create a valid signature associated with a trusted sender. The digital signature includes a cryptographic digest of the contents of the attributes that enables the recipient to detect any alterations, additions or deletions to the signed content. Because the signature can cover multiple attributes, an attack can not remove one of the attributes without invalidating the hash value. The ifmNonce CMS attribute included in the signedAttrs field is also included in the CMS hash and signature. These CMS attribute also are protected from modification. Because the ifmNonce CMS attribute is mandated by this specification and includes freshness values from each party, attempts to replay previously valid attributes can be detected by the recipient using a replay cache. It's critical that all IMCs and IMVs check the nonce values prior to operating upon a received set of attributes to avoid replay attacks. This check includes validating that the nonce values are appropriate (incremented from prior values) and checking a cache of previously used initial nonce values. Finally, deployments could choose to also use enveloped-data encapsulation of the signed-data content. Enveloped-data provides encryption of the signed-data using per-session encryption keys that would not be known (or replayable) by network based intermediaries.

4.1.2 Threats Addressed by Encrypted Attributes

The CMS enveloped-data content type used by IF-M security provides an encrypted envelope around the signed-data content to protect the signed data from disclosure while traveling between the IMC and IMV. The encryption of the signed set of attributes allows the attributes to pass through untrustworthy intermediary devices and components while maintaining the confidentiality and privacy of the information.

Section 5.2.1 of the IF-M specification discusses the threat of information theft by adversaries capable of intercepting the attributes while traversing the network and TNC architecture components. Deployers wishing to protect the exchanged attributes without trusting or using other countermeasures to protect the attributes can use enveloped-data to establish private attribute exchanges between IMC and IMV. Malicious intermediaries would require knowledge of the encryption key (or indirectly via the key encrypting key) to obtain the attribute information.

4.2 Potential Threats Against IF-M use of CMS

The use of CMS with IF-M security provides security protections for the exchanged IF-M attributes but CMS itself may be directly attacked by adversaries. This section discusses some potential threats to CMS and their implications to deployers.

4.2.1 Cryptography

CMS protections are based on the use of cryptographic digests, signatures, and encryption (both content and key). Signing, encryption and key management keys must be protected from a variety of potential threats that would result in their discovery by adversaries. The encryption algorithms themselves become weaker over time and eventually may become vulnerable to various forms of attack including brute force. This risk is elevated as computing performance increases and new mathematical weaknesses are discovered allowing faster searching of the key space. Implementations should be agile enough to support protected dynamic negotiation and addition of new algorithms as necessary. IF-M security offers dynamic discovery of supported cryptographic capabilities; this allows senders to use newer and stronger algorithms when recipients are also deployed with those algorithms. IF-M message senders should use care not to send data using weak encryption algorithms that are no longer appropriate for the sensitivity of the attributes being protected.

4.2.2 Threats to Keys

Signed-data content makes use of X.509 certificates for communicating the signer's public key and associated meta-data, such as the holder's identity to recipients. These certificates are protected from alteration as long as recipients verify the content signature and properly inspect the signing certificate for validity, authenticity and trustworthiness prior to usage. Part of the validation process normally involves consulting one or more trust anchors typically manifested as a set of certificates associated with trusted certificate authorities. Implementations need to protect the trust anchor database from unauthorized modification, addition or deletion in order to ensure that only trusted certificate authorities are present. If an adversary is able to alter the trust anchor database then falsified certificates could pass validation and cause harm to the TNC deployment.

Enveloped-data content can make use of data encryption keys, initialization vectors and padding that are generated by the sender and must be unpredictable by third parties using entropy that can't be influenced or predicted by untrusted software[RFC1740]. The generated keys must be resilient to passive eavesdropping and active attacks that attempt to steal them for future use. Therefore, CMS encrypts these keys when sent between the IMC and IMV using a variety of types of key management algorithms discussed in the CMS specification. Any non-public key used to encrypt the content encryption keys must also be protected from prediction or disclosure on the network, TNC client or TNC server system. Key management schemes that make use of previously distributed key encrypting key require those keys are protected from unauthorized access while on persistent storage and in memory. Failure to do so could lead to the exposure of the content encryption keys and thus the protected attributes.

4.2.3 Denial of Service

IF-M security provides a protective CMS wrapper around a set of one or more IF-M attributes allowing the recipient to detect attacks on the IF-M message attributes. However while detection is possible, repair of the attribute is not, so recipients are forced to drop protected contents that have been altered. If an attacker can modify every protected attribute, this would result in the protected attributes being dropped and thus a denial of service (DoS) of the assessment. Implementations should provide proper audit logging facilities and alerting capabilities to enable deployers to become aware of when such attacks are in progress. These facilities may also be used to cause other DoS attacks, so the amount of logging and alerting should be able to be throttled by deployer controls (e.g. notify the admin at most once per hour). A similar DoS attack can be achieved by a malicious intermediary that just drops all TNC messages.

Another form of DoS against the CMS protected content involves sending a high rate of IF-M messages containing large falsified or replayed enveloped-data protected attributes. This will cause the recipients to spend CPU cycles decrypting the messages before finding out the content is falsified or replayed when the attributes signatures is verified. This threat may not be feasible when an authenticated IF-T transport is present.

Other forms of DoS attack target the CMS wrapper information for enveloped-data. This information is outside of the CMS signature so could be modified to cause problems for recipients processing the message after significant CPU time has occurred. For example an attacker might modify the recipientInfos structure to break the key management schemes used to exchange the content encryption keys. The result is that the encrypted content would no longer be able to decrypt and the message would be discarded.

Finally, DoS attacks are possible by hostile intermediaries modifying the ifmErrorCode, ifmSecurityCapabilities, or ifmTncNonce CMS attributes such that potential senders of protected information are unable to find common algorithms with their target recipients or pass the replay checks. Because the CMS signed attributes are contained in signedAttrs field, these modifications will be detected and thus the information discarded.

5 References

5.1 Normative References

- [KEYWORDS] S. Bradner, "Keywords for use in RFCs to Indicate Requirement Levels", <http://www.ietf.org/rfc/rfc2119.txt>, IETF, March 1997.
- [RFC3370] R. Housley, "Cryptographic Message Syntax (CMS) Algorithms", <http://www.ietf.org/rfc/rfc3370.txt>, IETF, August 2002.
- [RFC3565] J. Schaad, "Use of the Advanced Encryption Standard (AES) Encryption Algorithm in Cryptographic Message Syntax (CMS)", <http://www.ietf.org/rfc/rfc3565.txt>, IETF, July 2003.
- [RFC3851] B. Ramsdell, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", <http://www.ietf.org/rfc/rfc3851.txt>, IETF, July 2004.
- [RFC3852] R. Housley, "Cryptographic Message Syntax (CMS)", <http://www.ietf.org/rfc/rfc3852.txt>, IETF, July 2004.
- [RFC5008] R. Housley, J. Solinas, "Suite B in Secure/Multipurpose Internet Mail Extensions (S/MIME)", <http://www.ietf.org/rfc/rfc5008.txt>, IETF, September 2007.
- [CMS-SHA2] S. Turner, "Using SHA2 Algorithms with Cryptographic Message Syntax", IETF, Work in Progress.

5.2 Informative References

- [IF-ARCH] Trusted Computing Group, "TNC Architecture for Interoperability", https://www.trustedcomputinggroup.org/specs/TNC/TNC_Architecture_v1_2_r4.pdf, May 2007.
- [IF-M] Trusted Computing Group, "TNC IF-M: TLV Binding", Work in Progress.
- [RFC1740] D. Eastlake 3rd, S. Crocker, and J. Schiller, "Randomness Recommendations for Security", <http://www.ietf.org/rfc/rfc1740.txt>, IETF, December 1994
- [TAMP] R. Housley, et. al, "Trust Anchor Management Protocol (TAMP)", <http://www.ietf.org/internet-drafts/draft-housley-tamp-00.txt>, October 2007.