26 Type-19 Security Records

Type-19 logical records shall contain a Cryptographic Message Sequence (CMS) encoded in DER format.

The purpose of the Type-19 Record is to provide confidence that the separately taken Type-X records containing biometric, SMTA and biographical data have not been altered during transport and storage of the transaction. The mathematical digest of each record is recorded in this record and the responsible person creates a digital signature of that set of fields.

Well-known algorithms for digital signatures encapsulated in CMA have been in use since 1991 and used commonly in secure email since before 1999.

An Object Identifier (OID) is a common cryptographic and ASN.1 definition of a specific item of knowledge. When they appear in this specification, these strings are quoted. The quotes do not appear in the ITL-1 document itself.

26.1Fields of the Type-19 Logical Record

The Type-19 record consists of both ANSI text and binary data. The mathematical digest (henceforth termed digest) is a known length binary sequence of bytes. The CMS can also be treated as a binary sequence of bytes similar to the WSQ and JPG image types in the Type-4, Type-14 etc. records. Since the CMS ends the Type-19 record, the overall length of the Type-19 record defines its length.

26.1.1 Logical Record Length (LEN)

This mandatory 4-character field contains the length of the record in all bytes in the record. This field has a maximum size of 9999 and is zero padded to the left. This is a base 10 number represented in text.

26.1.2 Image Designation Character (IDC)

Mandatory two-character field zero padded to the left containing the IDC of this record matching the IDC in the Type-1 logical record.

26.1.3 OID of Signature Method (SSO)

This mandatory ANSI text string defines the method by which the signature was created. For signature type records this would be the string “1.1.2.840.113549.1.7.2” corresponding to the current version of CMS SignedData defined in RFC 3852.

26.1.4 Content Type (SCT)

A choice of the type of content this logical record contains. Choices are given in Table 1

Table 1 Content Type (SCT) of Type-19 Logical Record

Content Type	Value
CMS-SignedData	1

CMS-TimeStampToken	2

26.1.5 Digest OID (DGO)

This mandatory ANSI text string defines the method by which the other records in the document will be digested and therefore defines the specific size of that digest (also known as a hash). Truncated digest methods are not supported. Recommended digest algorithms are given in Table 26-2, but the specific requirements of which algorithm to use is beyond the scope of this document.

Algorithm	OID	Size in Bytes
SHA-1	1.3.14.3.2.26	20
SHA224	2.16.840.1.101.3.4.2.4	28
SHA256	2.16.840.1.101.3.4.2.1	32
SHA384	2.16.840.1.101.3.4.2.2	48

The Digest OID appears in standard “dotted” notation. For example the OID for SHA-1 is “1.3.14.3.2.26.”

26.1.6 Length of Signed Attributes (LAS)

This mandatory ANSI text string defines the length of all of the signed attributes of the record. The Signed Attributes begin at the 19.050 marker and end with the last character of this 19.096 <gs> marker. This length is always a 3-digit zero left padded ANSI text string of the length.

26.1.7 Count of Digests (CDI)

This mandatory 2 character string contains the count of the records that were hashed for this digital signature logical record. This string is zero padded to the left and decimal representation in ANS text.

26.1.8 List of Digest (LDI)

This mandatory sequence of text and binary data contains the list of the digest of each record. Each sub-subfield of this list is designated as a **IDI (Item of DIgest)**. Each IDI consists of three sub-fields, the IDC of the record to which the item corresponds, the Type of the record to which the item corresponds, and the binary digest of that record. The first two values are always each two character ANSI text strings describing the IDC and Type Number in base 10. These two subfields are separated by the <us> character. The last subfield is always a known length of bytes corresponding to the DGO or OID designated in 19.005. For example for SHA1, the length is 20 bytes. For SHA-2(224) it is 28 and for SHA256 it is 32. Thus for SHA1, the total length of a subfield item is 27 bytes.

Each subfield ends with an <rs> information separator except the last IDI, which ends with a <gs> information separator.

To create the digest of each record, the record is submitted to a standard hashing or digest algorithm as identified by its OID. The record is presented as an ordered array of bytes in transmission order starting with the very first character of the record designating its type and ending with the <fs> information separator if that field type has such a separator. If the record has no <fs> information separator then the digest ends with the last byte of binary data in the record.

The subfield containing the digest of the Type-1 record contains the string "-1" in the IDC indicator sub-subfield. All other IDC designators should be positive numbers.

26.1.9 Authentication Material (AUT)

This mandatory field contains the authenticating signature. For the CMS defined by the sample OID in 19.003, "1.1.2.840.113549.1.7.2", this would be SignedData of RFC 3852.

26.2 Advantages of Signatures

Digital signatures have the advantage of providing

- Non-repudiation. The signer cannot deny the signature without admitting that another person has access to his or her private key.
- Reasonable proof that information has not been altered.
- Responsibility or traceability. Provides a deterrent to malicious behavior because the behavior is now traceable to an individual.

As such digital signatures rely on other cryptographic techniques, which are briefly explained below.

26.2.1 Digest Algorithms

Common known as hashes, digest algorithms such as SHA digest information into a smaller set of data. A typical digest such as SHA-1 results in a fixed size output regardless of how much data is "digested." They are designed with two primary factors in mind:

- They are reproducible. Given the same data, the digest will always be the same.
- Small changes to the data give rise to large changes in the digest.
- It is difficult to recreate the data given the digest, so the data itself remains private.
- It is difficult to create the same digest for two different sets of data.

26.2.2 Digital Signature

Digital signatures utilize the technique known as asymmetric cryptography. This is a cipher that utilizes a pair of keys, commonly known as the private key and the public key. The "owner" of a signature is the only person with access to the private key. This access

is granted by the system only after the owner has authenticated to that system. The owner uses the private key to create a cipher of the digest of the data. The public key can decrypt this cipher but it cannot create it. Therefore only the person with access to the private key can encrypt the data. Thus the encryption of the digest (hash) with the private key proves that the owner:

- had access to the data to create the hash.
- had access to the private key to encrypt the hash.

This technique has become know as a digital signature because it has the same effect as a written signature.

26.2.3 Certificates

The previously described signature is good as far as it goes. A private key that some other person acquires may or may not belong to the person we designated as the owner in the previous section.

A certificate is a well-defined set of data that is signed by a third party that everyone in the system agrees to trust. The trusted third-party is a Certificate Authority (CA). The CA has a private key, which is well protected, and it uses that key to sign the public key of the owner. Included in the signed document is information that uniquely identifies the owner of the key. Sometimes, there is a Registration Authority (RA) associated with a CA. The CA trusts the Registration Authority to actually verify the identity of the private key owner.

Certificates form what is known as a chain of trust. The Root Certificate of a CA is often what is published. Sometimes though, there may be a chain of certificates each signed by its predecessor. For example CA Company Root signs CA Company Email Root signs My Company Root signs John Does Certificate. This chain of signatures, all of which are traceable to some trusted root source is known as a certificate chain.

Databases containing certificates are often known as Certificate Stores.

26.2.4 Cryptographic Message Syntax (CMS)

RSA Laboratories first published cryptographic Message Syntax (CMS) in 1991 as the PKCS #7 specification. The abbreviation PKCS stands for Public Key Cryptographic System. The entire set of PCKS documents has withstood the test of time to become a number of RFC standards today.

As utilized here, the CMS provides a wrapper around a signature that contains enough information for the recipient to obtain the certificate, or it contains the certificate itself.

So when a signed set of data is received, one looks inside to find any one of the possible contents leading to information about the signer:

- The signer’s certificate, which contains information sufficient to search for the other certificates in the chain.
- The signers certificate and the certificate chain.

- Information about the signer and the CA of the signer so that the certificate chain can be found.

26.2.4.1 Abstract Syntax Notation

A variety of cryptographic providers exist which can parse and create a CMS binary object. The CMS binary object is defined in Abstract Syntax Notation commonly known as ASN.1. The Distinguished Encoding Rules (DER) are used to convert the information as defined by ASN.1 into digital bytes. The Distinguished Encoding Rules are a limitation of the specification of the Binary Encoding Rules (BER) such as produce the most compact an reproducible set of data for any given ASN.1 specification. The ASN.1 specification for CMS is contained in RFC 3852. The ASN.1 specification is itself contained in ISO/IEC 8824-1:2002

26.2.5 Time Stamp Token

A Time Stamp Token (TST) is a special type of CMS that is signed by a Time Stamp Authority (TSA). A hash of some data is presented to a TSA and the TSA signs the data along with its time of signing and returns that material known as a Time Stamp Token to the requestor. The requestor uses this material to prove that the data originated some time prior to the time in the TST.

26.2.6 Authentication to the Private Key Store

As with certificates, databases containing private keys (or any other type of cryptographic key) are known as Key Stores. Sometimes and often times, the key store and certificate store are highly linked. In any case it is the responsibility of the system or key store sub-system to verify the authenticity of a user before using the private key on their behalf to create a signature. A private key that is not at least password protected could be utilized by anyone who knows of its existence to create a signed set of data.

A stronger authentication technique would utilize a smart card to hold the private key and the smart card would require biometric verification to allow the card-holder to sign some set of data.

26.3 Usage

26.3.1 Creating an Authentication Signature

The Type-19 logical record is probably the last record to be added to a transaction. To create the signature, the system or software should collect the records into byte arrays of known length. Each of these arrays is processed through the hashing or digesting algorithm and the result is turned into the subfield item, IDI. Then the Type-19 record is created up to and including 19.098.

The data between and inclusive of the 19.050: through the end of subfield 19.098 including the ending <gs> information separator is presented to the signature algorithm for signing.

The CMS algorithm should be configured to create a non-encapsulated signature. This specification also recommends that no certificates be attached to the signature. In general,

certificates should be transported through another means. The target body should be able to acquire the certificate from a local database by using the CA's identification and the certificate serial number contained within the CMS. That information is contained in the SignerInfo content of the SignedData of the CMS. The recommended

26.3.2 Types of Authentication Signatures

All of the signatures described in this section are for informational purposes only. It is left to the agencies utilizing this transaction to define the policy of its use. All of the following discussion assumes that reasonable authentication techniques are employed prior to allowing an individual to utilize the private key to sign any material.

The term modify, as used in the following text of this section refers to the fact that if any bit of the document is changed, then a system verifying the signature can observe that the change exists.

An authentication signature could be created using a self-signed certificate. An individual can be assured that if that individual's self-signed certificate was used to sign the material, then the material is the same as when it was signed.

A CA-signed certificate indicates that reasonable means were employed to verify the identity of the individual creating the signature. The signature indicates to any group of individuals that place trust the CA that the specific individual created the signature. If the individual (or some other authority) has reason to believe that the private key associated with the certificate is compromised (i.e. somebody else has access to it), then the CA can revoke the certificate by adding it to its Certificate-Revocation List (CRL). CRL's like certificates are available through a variety of proprietary and non-proprietary distribution techniques such as Online-Certificate Status Protocol (OCSP).

As described, the two above signatures do not prevent an individual from modifying their own material and signing the alteration with a date at an earlier time (or later time) than the original material.

The above signature, combined with a Time Stamp Signature (TSS) becomes stronger. A Time Stamp Signature firmly affixes a time to the information in question. A Time Stamp Authority (TSA) in which everyone in the system has trust provides the Time Stamp Signature. Once the Time Stamp Authority signs the document (including the officers Type 19 record) even the officer cannot modify the transaction.

CMS also has the concept of a counter-signature. A counter-signature could be applied to the material by the generating system with a private key to which only the system has access. The system would only do this during its normal processing and so it would make it difficult for the original signer to alter his or her own work after-the-fact.

Alternatively to a counter-signature, the system could also generate a separate Type-19 record, which signs the signature of the creator.

The drawback to a system private key is two fold. First if the software has access to the private key, then someone hacking the software might be able to get at the private key. Second, as the only key being used to sign material, any system private key does not provide any traceability beyond the system on which it resides.

References

[X.208-88]	CCITT. Recommendation X.208: Specification of Abstract Syntax Notation One (ASN.1). 1988.
[X.209-88]	CCITT. Recommendation X.209: Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1). 1988.
[RFC 3852]	Cryptographic Message Syntax (CMS) Network Working Group, 2004
[X9.95-2005]	ANSI X9 Trusted Time Stamp Management and Security, 2005
[RFC 3161]	X.509 Public Key Infrastructure Time-Stamp Protocol (TSP), Network Working Group, 2001