Abstract

PKIs are used to provide scalability and ease key management. One type of PKIs that is predominant for securing communications and data is based on the X.509 standard. Since the security of PKIs, ultimately, depends on the security of the cryptographic building blocks that are used for authentication and encryption, the standards community made algorithm agility a priority. Algorithm agility, in particular, enables upgrading to newly available algorithms when needed.

The CompositeCrypto (i.e., CompositeKey and CompositeSignature structures) described in this document provides an additional tool that enables the use of multiple algorithms to authenticate data without the need to use multiple certificates and more complex data structures.

This document provide the description of the definition and encoding rules for CompositeKey and CompositeSignature. A description of how to use these structures in main PKIX objects (e.g., X.509 certificates, CRLs, OCSP responses, etc.) is also provided.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 26, 2019.
Copyright Notice

Copyright (c) 2019 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 (https://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 Simplified BSD License.

Table of Contents

1. Requirements notation ................................. 2
2. Introduction and Scope ................................ 3
3. Composite Cryptography .................................. 3
   3.1. Composite Public Keys ................................. 4
   3.2. Composite Private Keys ................................. 4
   3.2.1. Encoding Rules ................................... 4
   3.2.2. Encrypted and Un-encrypted Local Storage .......... 4
   3.2.3. Asymmetric Key Packages ............................ 5
3.3. Composite Signatures ................................... 5
3.4. Generating Composite Signatures ....................... 6
3.5. Verifying Composite Signatures ....................... 6
4. Use of Composite Crypto in PKIX structures ................. 6
   4.1. Use in X.509 Certificates ......................... 6
   4.2. Use in X.509 CRLs ................................ 6
   4.3. Use in X.509 OCSP Messages ....................... 6
   4.4. Use in PKCS#7 .................................... 6
   4.5. Use in PKCS#8 .................................... 6
   4.6. Use in PKCS#12 ................................... 7
   4.7. Use in CMS ..................................... 7
5. Security Considerations .................................. 7
6. IANA considerations ..................................... 7
7. Acknowledgments ......................................... 7
8. Normative References .................................... 7
Appendix A. ASN.1 Module .................................... 7
Author’s Address .......................................... 8

1. Requirements 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. Introduction and Scope

With the definition of new algorithms (e.g. more efficient factoring techniques) and technologies (e.g., quantum-based computing devices) that might be available in the near future, the need to provide an easy-to-deploy and efficient solution capable of providing multi-algorithms authentication is paramount.

Today there are no complete or general solutions that allow the use of multiple public-key algorithms to authenticate PKIX data without using multiple X.509 certificates or complex data structures. For example, CRLs or OCSP responses cannot be protected via multiple algorithms without wrapping the OCSP responses' data via CMS or other signed containers.

We define two new building blocks, i.e. compositeKey and compositeSignature, that can be used in many environments where Public Key authentication is used - i.e., from the generation of certificates that are authenticated with multiple signatures (i.e., using multiple keys that may or may not use different cryptographic schemes or different number of security bits), to the possibility of specifying a composite key that combines multiple public keys together (instead of one) in a single certificate.

This document describes the encoding of the new building blocks and their application to different X.509 core data structures that are used in PKIs. In particular, this document focuses only on the definition of the composite keys and composite signatures definitions for X.509 based PKIs (PKIX) by providing the encoding rules and their usage in existing X.509 (PKIX) data structures.

3. Composite Cryptography

Composite Cryptography is defined as the possibility to combine different public keys and signatures in PKIX objects. The following OID is defined to identify the algorithm class:

```
id-pk-compositeCrypto OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) dod(6) internet(1) private(4)
    enterprise(1) OpenCA(18227) Algorithms(2) 1 }
```

Composite Cryptography provides three distinct building blocks: the compositePublicKey, the compositePrivateKey and the compositeSignature. The compositePublicKey is meant to carry all the public keys associated with an identity. The compositePrivateKey is meant to carry all the private keys associated with an identity. The compositeSignature, instead, carries a sequence of signatures that
are generated by using the different individual keys from a compositePrivateKey.

3.1. Composite Public Keys

This document defines a new Object Identifier and data structure for composite public keys as follows:

id-pk-compositePublicKey OBJECT IDENTIFIER ::= { id-kp-compositeCrypto 1 }

CompositePublicKey ::= SEQUENCE (1..MAX) OF SubjectPublicKeyInfo

3.2. Composite Private Keys

This section specifies a syntax and semantics for Composite Keys private key information. Composite private key information is built as a SEQUENCE of BIT STRINGs each of which contains the single private keys and parameters. Additionally, it may include the corresponding public keys.

The structure defined in this document allows for the distribution of the composite keys (public and private) and the associated domain parameters by using a sequence of OneAsymmetricKey as defined in [RFC5958].

The Algorithm Identifier and data structure associated for Composite Private Keys is defined as follows:

id-pk-compositePrivateKey OBJECT IDENTIFIER ::= { id-kp-compositeCrypto 2 }

CompositePrivateKey ::= SEQUENCE (1..MAX) OF OneAsymmetricKey

3.2.1. Encoding Rules

When encoding Composite Private Keys, generators SHOULD use Distinguished Encoding Rules (DER) [X.690] and receivers SHOULD be prepared to handle Basic Encoding Rules (BER) [X.690] and DER [X.690].

3.2.2. Encrypted and Un-encrypted Local Storage

The compositePrivateKey format as defined in the previous subsection can also be used for local storage of an unencrypted compositePrivateKey binary object. The compositePrivateKey can also be formatted in PEM format that uses the (".pem") file extension and which is encoded as the the Base64 encoding (see Section 4 of [RFC4648]), of the DER-encoded compositePrivateKey object with the following armour:
Local storage of an encrypted CompositePrivateKeys object is out of scope of this document. However, CompositePrivateKeys should be the format for the plaintext key being encrypted. DER [X.690] encoding the CompositePrivateKeys will promote interoperability if the key is encrypted for transport to another party. PEM encoding the DER-encoded CompositePrivateKeys is common; "Proc-Type:" and "DEK-INFO:" fields [RFC1421] followed by the DER-encoded CompositePrivateKeys. The following armour used in this case is as follows:

-----BEGIN COMPOSITE PRIVATE KEY-----
-----END COMPOSITE PRIVATE KEY-----

3.2.3. Asymmetric Key Packages

The Cryptographic Message Syntax (CMS), as defined in RFC 5652, can be used to digitally sign, digest, authenticate, or encrypt the asymmetric key format content type.

When encoding Composite Private Keys, the privateKeyAlgorithm in the OneAsymmetricKey SHALL be set to id-kp-compositePrivateKey.

The parameters of the privateKeyAlgorithm SHALL be a sequence of AlgorithmIdentifier objects, each of which are encoded according to the rules defined for each of the different keys in the Composite Private Key.

The value of the privateKey field in the OneAsymmetricKey SHALL be set to the DER encoding of the SEQUENCE of private key values that make up the composite key. The number and order of elements in the sequence SHALL be the same as identified in the sequence of parameters in the privateKeyAlgorithm.

The value of the the publicKey (if present) SHALL be set to the DER encoding of the SEQUENCE of publicKey values. If this field is present, the number and order of elements SHALL be the same as identified in the sequence of parameters in the privateKeyAlgorithm.

The value of the attributes is encoded as usual.

3.3. Composite Signatures

The use of composite signatures allows for the use of multiple algorithms for authentication.
id-pk-compositeSignature OBJECT IDENTIFIER ::= { compositeCrypto 3 }

CompositeSignature OBJECT IDENTIFIER ::= SEQUENCE (1..MAX) OF BITSTRING

3.4. Generating Composite Signatures

When generating a CompositeSignature, the signing entity MUST generate one signature per key that is in the corresponding compositePrivateKey set.

The value of the compositeSignature is the DER encoding of the SEQUENCE of BIT STRING where each BIT STRING is the DER representation of the signature generated with one of the private key in the compositePrivateKey set.

When signing, the order of the signature MUST respect the order of keys in the compositePrivateKey and compositePublicKey sets.

3.5. Verifying Composite Signatures

When validating a compositeSignature, the relying party MUST verify at least one of the signatures in the compositeSignature structure and SHOULD verify all of them.

The process of validating composite signatures starts with going through the sequence of the signatures and if the inner signature algorithm is supported, the relying party MUST verify the signature with the corresponding public key in the compositePrivateKey.

The order of the signatures MUST respect the order of keys in the compositePrivateKey and compositePublicKey sets.

4. Use of Composite Crypto in PKIX structures

4.1. Use in X.509 Certificates

4.2. Use in X.509 CRLs

4.3. Use in X.509 OCSP Messages

4.4. Use in PKCS#7

4.5. Use in PKCS#8
4.6. Use in PKCS#12

4.7. Use in CMS

5. Security Considerations

This structures described in this document do not protect the private keys information in any way unless combined with a security protocol or encryption properties of the objects (if any) where the CompositePrivateKey is used (see next Section).

Protection of the private key information is vital to public key cryptography. The consequences of disclosure depend on the purpose of the private key. If a private key is used for signature, then the disclosure allows unauthorized signing. If a private key is used for key management, then disclosure allows unauthorized parties to access the managed keying material. The encryption algorithm used in the encryption process must be as ‘strong’ as the key it is protecting.

6. IANA considerations

The CMS content type OID is registered in a DoD arc. The ASN.1 module OID is TBD. The id-pk-compositeCrypto, id-pk-compositePrivateKey, id-pk-compositePublicKey, and id-pk-compositeSignature OIDs are to be assigned by IANA. The authors suggest to use the id-pkix arc for this usage.

7. Acknowledgments

The authors would like to thank everybody who provided insightful comments and helped in any form. This document uses a lot of text from similar documents ([RFC3279] and [RFC8410]) as well as [I-D.ietf-lamps-cms-hash-sig]. Thanks go to the authors of those documents. "Copying always makes things easier and less error prone" - [RFC8411].

8. Normative References


Appendix A. ASN.1 Module
CompositeCrypto-2009 { iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) TBD }

DEFINITIONS IMPLICIT TAGS ::= BEGIN

EXPORTS ALL;

IMPORTS
PUBLIC-KEY, SIGNATURE-ALGORITHM
FROM AlgorithmInformation-2009 -- RFC 5911 [CMSASN1]

--

-- Object Identifiers
--

id-pk-compositeCrypto OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-mod-algorithmInformation-02(58) }

id-pk-compositePublicKey OBJECT IDENTIFIER ::= { id-kp-compositeCrypto 1 }

id-pk-compositePrivateKey OBJECT IDENTIFIER ::= { id-kp-compositeCrypto 2 }

id-pk-compositeSignature OBJECT IDENTIFIER ::= { id-kp-compositeCrypto 3 }

END

Author’s Address

Massimiliano Pala
CableLabs
858 Coal Creek Cir
Louisville, CO  80027
USA

Email: director@openca.org
URI:  http://www.linkedin.com/in/mpala