Groupe de travail Réseau
Request for Comments : 3560
Catégorie : Sur la voie de la normalisation
Traduction Claude Brière de L'Isle

R. Housley, Vigil Security

juillet 2003

Utilisation de l'algorithme de transport de clé RSAES-OAEP dans la syntaxe de message cryptographique (CMS)

Résumé
Le présent document décrit les conventions pour l'utilisation de l'algorithme de transport de clés RSAES-OAEP avec la syntaxe de message cryptographique (CMS, *Cryptographic Message Syntax*). La CMS spécifie le type de contenu enveloped-data *(données enveloppées)* qui consiste en un contenu chiffré et des clés de chiffrement de contenu chiffrées pour un ou plusieurs receveurs. L'algorithme de transport de clés RSAES-OAEP peut être utilisé pour chiffrer des clés de chiffrement de contenu pour les receveurs prévus.

Table des matières

1. Introduction

PKCS n°1 version 1.5 [RFC2313] spécifie une variante largement déployée de l'algorithme de transport de clés RSA. Le transport de clé de PKCS n° 1 version 1.5 est vulnérable aux attaques adaptives de texte chiffré choisi, en particulier quand il est utilisé pour la gestion de clés dans des applications interactives. Cette attaque est souvent appelée "l'attaque du million de messages" et elle est expliquée dans [RSALABS] et [CRYPTO98]. L'exploitation de cette vulnérabilité, qui révèle le résultat d'un déchiffrement RSA, exige l'accès à un oracle qui va répondre à des centaines de milliers de textes chiffrés, qui sont construits de façon adaptative en réponse à des réponses reçues précédemment et fournit des informations sur les succès ou les échecs des opérations de déchiffrement tentées.

L'attaque est significativement moins faisable dans les environnements de remise différée *(store-and-forward)*, comme S/MIME. La [RFC3218] discute des contre-mesures à cette attaque disponibles quand le transport de clé de PKCS n° 1 version 1.5 est utilisé en conjonction avec la syntaxe de message cryptographique (CMS) [RFC3369].

Quand le transport de clé de PKCS n° 1 version 1.5 est appliqué comme couche de chiffrement intermédiaire au sein d'un environnement de communication demande-réponse interactif, l'exploitation pourrait être plus faisable. Cependant, les mises en œuvre des protocoles Couche de connexion sécurisée (SSL, *Secure Sockets Layer*) [SSL] et Sécurité de la couche transport (TLS, *Transport Layer Security*) [RFC2246] pourraient inclure des contre-mesures qui détectent et empêchent l'attaque du million de messages et autres attaques de texte chiffré choisi. Ces contre-mesures sont effectuées au niveau du protocole.

Dans l'intérêt d'une assurance de sécurité à long terme, il est prudent d'adopter une technique cryptographique améliorée plutôt que d'incorporer des contre-mesures dans les protocoles. À cette fin, une version mise à jour de PKCS n° 1 a été publiée. PKCS n° 1 version 2.1 [RFC3447] remplace la RFC 2313. Elle préserve la prise en charge du format de bourrage du chiffrement de PKCS n° 1 version 1.5, et en définit aussi une nouvelle. Pour résoudre la vulnérabilité au texte chiffré adaptif choisi, PKCS n° 1 version 2.1 spécifie et recommande l'utilisation du bourrage optimal de chiffrement asymétrique (OAEP, *Optimal Asymmetric Encryption Padding*) pour le transport de clé RSA.

Le présent document spécifie l'utilisation de l'algorithme de transport de clés RSAES-OAEP dans la CMS. La CMS peut être utilisée dans un environnement de remise différée ou de demandes-réponses interactives.

La CMS prend en charge diverses architectures pour la gestion de clé fondée sur le certificat, en particulier celle définie par le groupe de travail PKIX [RFC3280]. PKCS n° 1 version 1.5 et PKCS n° 1 version 2.1 exigent les mêmes informations de clé publique RSA. Donc, une clé publique RSA certifiée peut être utilisée avec l'une ou l'autres des techniques de transport de clés RSA.

La CMS utilise l'ASN.1 [X.208-88], les règles de codage de base (BER, *Basic Encoding Rules*) [X.209-88], et les règles de codage distinctives (DER, *Distinguished Encoding Rules*) [X.509-88].

Tout au long de ce document, quand les termes "DOIT", "NE DOIT PAS", "DEVRAIT", et "PEUT" sont utilisés en lettres majuscules, leur utilisation se conforme aux définitions de la [RFC2119]. Ces définitions de mots clés aident à rendre l'intention des documents de normalisation aussi claire que possible. Ces mots clés sont utilisés dans le présent document pour aider les mises en œuvre à réaliser l'interopérabilité.

2. Conventions de Enveloped-data

Le type de contenu de CMS enveloped-data consiste en un contenu chiffré et des clés de chiffrement de contenu enveloppées pour un ou plusieurs receveurs. L'algorithme de transport de clés RSAES-OAEP est utilisé pour envelopper la clé de chiffrement de contenu pour un receveur.

Un logiciel conforme DOIT satisfaire aux exigences pour la construction du type de contenu de données enveloppées déclarées dans la Section 6 de la [RFC3369] , "Type de contenu de données enveloppées".

Une clé de chiffrement de contenu DOIT être générée au hasard pour chaque instance de type de contenu de données enveloppées. La clé de chiffrement de contenu est utilisée pour chiffrer le contenu.

2.1 Champs EnvelopedData

Le type de contenu enveloped-data est codé en ASN.1 en utilisant la syntaxe EnvelopedData. Les champs de syntaxe EnvelopedData DOIT être remplis comme décrit dans ce paragraphe quand le transport de clés RSAES-OAEP est employé pour un ou plusieurs receveurs.

La version de EnvelopedData DOIT être 0, 2, ou 3.

Le champ originatorInfo *(informations sur l'origine)* de EnvelopedData n'est pas utilisé pour l'algorithme de transport de clés RSAES-OAEP. Cependant, ce champ PEUT être présent pour prendre en charge les receveurs qui utilisent d'autres algorithmes de gestion de clé.

Le CHOIX recipientInfos *(informations sur le receveur)* de EnvelopedData DOIT être KeyTransRecipientInfo *(Informations de receveur de transmission de clés)*. Voir au paragraphe 2.2 une discussion sur KeyTransRecipientInfo.

Le champ EnvelopedData encryptedContentInfo *(informations sur le contenu chiffré)* contentEncryptionAlgorithm *(algorithme de chiffrement de contenu)* DOIT être un identifiant d'algorithme de chiffrement symétrique.

Le champ EnvelopedData unprotectedAttrs *(attributs non protégés)* PEUT être présent.

2.2 Champs KeyTransRecipientInfo

Les champs de syntaxe KeyTransRecipientInfo DOIVENT être remplis comme décrit dans ce paragraphe quand le transport de clés RSAES-OAEP est employé pour un ou plusieurs receveurs.

La version de KeyTransRecipientInfo DOIT être 0 ou 2. Si l'identifiant de receveur *(RecipientIdentifier)* utilise la solution de remplacement issuerAndSerialNumber *(producteur et numéro de série)* , la version DOIT alors être 0. Si le RecipientIdentifier utilise la solution de remplacement subjectKeyIdentifier, la version DOIT alors être 2.

L'identifiant de receveur KeyTransRecipientInfo donne deux solutions alternatives pour spécifier le certificat du receveur, et par là la clé publique du receveur. Le certificat du receveur DOIT contenir une clé publique RSA. La clé de chiffrement de contenu est chiffrée avec la clé publique RSA du receveur. La solution issuerAndSerialNumber identifie le certificat du receveur par le nom distinctif du producteur et le numéro de série du certificat ; l'identifiant de clé sujette *(subjectKeyIdentifier)* identifie le certificat du receveur par la valeur d'extension de l'identifiant de clé sujette X.509.

L'algorithme de chiffrement de clé KeyTransRecipientInfo spécifie l'utilisation de l'algorithme RSAES-OAEP et ses paramètres associés, pour chiffrer la clé de chiffrement de contenu pour le receveur. Le processus de chiffrement de clé est décrit dans la [RFC3447]. Voir à la Section 3 du présent document la syntaxe d'identifiant d'algorithme et des paramètres.

La clé chiffrée KeyTransRecipientInfo est le résultat du chiffrement de la clé de chiffrement de contenu dans la clé publique RSA du receveur en utilisant l'algorithme RSAES-OAEP. La clé publique RSA DOIT être d'au moins 1024 bits. Quand elles utilisent une clé de chiffrement de contenu Triple-DES [3DES], les mises en œuvre DOIVENT ajuster les bits de parité pour assurer une parité impaire pour chaque octet de chaque clé DES composant la clé Triple-DES avant le chiffrement RSAES-OAEP.

3. Identifiants et paramètres de l'algorithme RSAES-OAEP

L'algorithme de transport de clés RSAES-OAEP est le schéma de chiffrement RSA défini dans la [RFC3447], où le message à chiffrer est la clé de chiffrement de contenu. L'identifiant d'algorithme pour RSAES-OAEP est :

IDENTIFIANT D'OBJET id-RSAES-OAEP ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 7 }

Le champ de paramètres AlgorithmIdentifier DOIT être présent, et le champ paramètres DOIT contenir RSAES-OAEP-params. RSAES-OAEP-params a la syntaxe suivante :

```
RSAES-OAEP-params  ::=  SEQUENCE  {
   hashFunc        [0] AlgorithmIdentifier DEFAUT sha1Identifier,
   maskGenFunc [1] AlgorithmIdentifier DEFAUT mgf1SHA1Identifier,
   pSourceFunc   [2] AlgorithmIdentifier DEFAUT pSpecifiedEmptyIdentifier  }

sha1Identifier  AlgorithmIdentifier ::= { id-sha1, NULL }

mgf1SHA1Identifier  AlgorithmIdentifier ::= { id-mgf1, sha1Identifier }

pSpecifiedEmptyIdentifier  AlgorithmIdentifier ::= { id-pSpecified, nullOctetString }

nullOctetString  CHAINE D'OCTETS (TAILLE (0)) ::= { "H }

IDENTIFIANT D'OBJET id-sha1 ::= iso(1) identified-organization(3) oiw(14) secsig(3) algorithms(2) 26 }
```

IDENTIFIANT D'OBJET pkcs-1 ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) }

IDENTIFIANT D'OBJET id-mgf1 ::= { pkcs-1 8 }

IDENTIFIANT D'OBJET id-pSpecified ::= { pkcs-1 9 }

Les champs dans RSAES-OAEP-params ont la signification suivante :

hashFunc identifie la fonction de hachage unidirectionnelle. Les mises en œuvre DOIVENT prendre en charge SHA-1 [SHA1], et les mises en œuvre PEUVENT prendre en charge d'autres fonctions de hachage unidirectionnelles. L'identifiant d'algorithme SHA-1 est composé de l'identifiant d'objet id-sha1 et d'un paramètre de NULL. Les mises en œuvre qui effectuent le chiffrement DOIVENT omettre le champ hashFunc quand SHA-1 est utilisé, pour indiquer que l'algorithme par défaut a été utilisé. Les mises en œuvre qui effectuent le déchiffrement DOIVENT reconnaître l'identifiant d'objet id-sha1 et un champ absent hashFunc comme l'indication que SHA-1 a été utilisé.

maskGenFunc identifie la fonction de génération de gabarit. Les mises en œuvre DOIVENT prendre en charge MFG1 [RFC3447]. MFG1 exige une fonction de hachage unidirectionnelle, et il est identifié dans le champ Paramètre de l'identifiant d'algorithme MFG1. Les mises en œuvre DOIVENT prendre en charge SHA-1 [SHA1], et les mises en œuvre PEUVENT prendre en charge d'autres fonctions de hachage unidirectionnelles. L'identifiant d'algorithme MFG1 est composé de l'identifiant d'objet id-mgf1 et d'un paramètre qui contient l'identifiant d'algorithme de la fonction de hachage unidirectionnelle employée avec MFG1. L'identifiant d'algorithme SHA-1 se compose de l'identifiant d'objet id-sha1 et d'un paramètre de NULL. Les mises en œuvre qui effectuent le chiffrement DOIVENT omettre le champ maskGenFunc quand MFG1 avec SHA-1 est utilisé, ce qui indique l'utilisation de l'algorithme par défaut. Les mises en œuvre qui effectuent le déchiffrement DOIVENT reconnaître les deux identifiants d'objet id-mgf1 et id-sha1 ainsi qu'un champ maskGenFunc absent comme l'indication que MFG1 avec SHA-1 est utilisé.

pSourceFunc identifie la source (et éventuellement la valeur) des paramètres de codage, généralement appelée P. Les mises en œuvre DOIVENT représenter P par l'identifiant d'algorithme, id-pSpecified, indiquant que P est explicitement fourni comme une CHAINE D'OCTETS dans les paramètres. La valeur par défaut de P est une chaîne vide. Dans ce cas, pHash dans EME-OAEP contient le hachage d'une chaîne de longueur zéro. Les mises en œuvre DOIVENT prendre en charge une valeur de P de longueur zéro. Les mises en œuvre qui effectuent le chiffrement DOIVENT omettre le champ pSourceFunc quand une valeur de P de longueur zéro est utilisée, ce qui indique que la valeur par défaut a été utilisée. Les mises en œuvre qui effectuent le déchiffrement DOIVENT reconnaître l'identifiant d'objet id-pSpecified et un champ pSourceFunc absent comme l'indication qu'une valeur P de longueur zéro a été utilisée.

4. Conventions de certificat

La [RFC3280] spécifie le profil pour utiliser les certificats X.509 dans les applications de l'Internet. Quand une clé publique RSA va être utilisée pour le transport de clés RSAES-OAEP, les conventions spécifiées dans cette section augmentent la RFC 3280.

Traditionnellement, l'identifiant d'objet rsaEncryption est utilisé pour identifier les clés publiques RSA. Cependant, pour mettre en œuvre toutes les recommandations décrites dans la Section Considérations sur la sécurité de ce document (voir la Section 6) l'utilisateur de certificat doit être capable de déterminer la forme du transport de clés que le possesseur de la clé privée RSA associe à la clé publique.

L'identifiant d'objet rsaEncryption continue d'identifier la clé publique sujette quand le possesseur de la clé privée RSA ne souhaite pas limiter exclusivement l'utilisation de la clé publique à RSAES-OAEP. Dans ce cas, l'identifiant d'objet rsaEncryption DOIT être utilisé dans le champ Algorithme au sein des informations de la clé publique sujette, et le champ Paramètres DOIT contenir NULL.

IDENTIFIANT D'OBJET rsaEncryption := { pkcs-1 1 }

On trouvera une plus longue discussion des conventions associées à l'utilisation de l'identifiant d'objet rsaEncryption au paragraphe 2.3.1 de la [RFC3279].

Quand le possesseur de la clé privée RSA souhaite limiter l'usage de la clé publique exclusivement à RSAES-OAEP, l'identifiant d'objet id-RSAES-OAEP DOIT alors être utilisé dans le champ Algorithme au sein des informations de clé publique sujette, et le champ Paramètres DOIT contenir RSAES-OAEP-params. La valeur de l'identifiant d'objet id-

RSAES-OAEP et la syntaxe de RSAES-OAEP-params sont décrites entièrement à la Section 3 de ce document.

Sans considération de l'identifiant d'objet utilisé, la clé publique RSA est codée de la même manière dans les informations de clé publique sujette. La clé publique RSA DOIT être codée en utilisant le type RSAPublicKey :

```
RSAPublicKey ::= SEQUENCE {
    modulus            ENTIER,    -- n
    publicExponent     ENTIER  }  -- e
```

Ici, modulus est le module "n", et publicExponent est l'exposant public "e". La RSAPublicKey codée en DER est portée dans la CHAINE BINAIRE subjectPublicKey au sein des informations de clé publique sujette.

L'application prévue pour la clé PEUT être indiquée dans l'extension de certificat d'usage de clé (voir au paragraphe 4.2.1.3 de la [RFC3280]). Si l'extension keyUsage est présente dans un certificat qui porte une clé publique RSA avec l'identifiant d'objet id-RSAES-OAEP, alors l'extension d'usage de clé DOIT contenir une combinaison des valeurs suivantes :
keyEncipherment *(chiffrement de clé)* , et
dataEncipherment *(chiffrement des données)*.

Cependant, keyEncipherment et dataEncipherment NE DEVRAIENT PAS être tous les deux présentes.

Quand un certificat qui porte une clé publique RSA avec l'identifiant d'objet id-RSAES-OAEP, l'utilisateur du certificat DOIT utiliser la clé publique RSA certifiée seulement pour les opérations de RSAES-OAEP, et l'utilisateur du certificat DOIT effectuer ces opérations en utilisant les paramètres identifiés dans le certificat.

5. Conventions d'attribut SMIMECapabilities

Le paragraphe 2.5.2 de la [RFC2633] définit l'attribut signé SMIMECapabilities (défini comme une SEQUENCE de séquences de SMIMECapability) comme étant utilisé pour spécifier une liste partielle des algorithmes que le logiciel qui annonce les SMIMECapabilities peut prendre en charge. Lors de la construction d'un objet signedData, un logiciel conforme PEUT inclure l'attribut signé SMIMECapabilities qui annonce qu'il prend en charge l'algorithme RSAES-OAEP.

Quand tous les réglages par défaut sont choisis, la séquence SMIMECapability représentant RSAES-OAEP DOIT inclure l'identifiant d'objet id-RSAES-OAEP dans le champ capabilityID et DOIT inclure une séquence vide dans le champ Paramètres. Dans ce cas, RSAES-OAEP est représenté par rSAES-OAEP-Default-Identifier :

```
Identifiant d'algorithme rSAES-OAEP-Default-Identifier ::= { id-RSAES-OAEP, { sha1Identifier, mgf1SHA1Identifier,
                                                            pSpecifiedEmptyIdentifier } }
```

La séquence SMIMECapability qui représente rSAES-OAEP-Default-Identifier DOIT être codée en DER comme la chaîne hexadécimale suivante : 30 0D 06 09 2A 86 48 86 F7 0D 01 01 07 30 00

Lorsque des réglages autres que par défaut sont choisis, la séquence SMIMECapability représentant RSAES-OAEP DOIT inclure l'identifiant d'objet id-RSAES-OAEP dans le champ capabilityID et DOIT inclure la séquence RSAES-OAEP-params qui identifie les réglages non par défaut dans le champ Paramètres.

Quand SHA-256 est utilisé comme fonction de hachage et SHA-256 est utilisé avec MGF1 dans la fonction de génération de gabarit (maskGenFunc) la séquence SMIMECapability représentant RSAES-OAEP est l'identifiant rSAES-OAEP-SHA256-Identifier, comme spécifié à l'Appendice A. La séquence SMIMECapability représentant trSAES-OAEP-SHA256-Identifier DOIT être codée en DER comme la chaîne hexadécimale suivante :

```
30 38 06 09 2A 86 48 86 F7 0D 01 01 07 30 2B 30 0D 06 09 60 86 48 01 65 03 04 02 01 05 00 30 1A
06 09 2A 86 48 86 F7 0D 01 01 08 30 0D 06 09 60 86 48 01 65 03 04 02 01 05 00
```

Quand SHA-384 est utilisé comme fonction de hachage et SHA-384 est utilisé avec MGF1 dans la fonction de génération de gabarit (maskGenFunc) la séquence SMIMECapability représentant RSAES-OAEP est l'identifiant rSAES-OAEP-SHA384-Identifier, comme spécifié à l'Appendice A. La séquence SMIMECapability représentant trSAES-OAEP-SHA384-Identifier DOIT être codée en DER comme la chaîne hexadécimale suivante :

```
30 38 06 09 2A 86 48 86 F7 0D 01 01 07 30 2B 30 0D 06 09 60 86 48 01 65 03 04 02 02 05 00 30 1A
```

06 09 2A 86 48 86 F7 0D 01 01 08 30 0D 06 09 60 86 48 01 65 03 04 02 02 05 00

Quand SHA-512 est utilisé comme fonction de hachage et SHA-512 est utilisé avec MGF1 dans la fonction de génération de gabarit (maskGenFunc) la séquence SMIMECapability représentant RSAES-OAEP est l'identifiant rSAES-OAEP-SHA512-Identifier, comme spécifié à l'Appendice A. La séquence SMIMECapability représentant rSAES-OAEP-SHA512-Identifier DOIT être codée en DER comme la chaîne hexadécimale suivante :

30 38 06 09 2A 86 48 86 F7 0D 01 01 07 30 2B 30 0D 06 09 60 86 48 01 65 03 04 02 03 05 00 30 1A
06 09 2A 86 48 86 F7 0D 01 01 08 30 0D 06 09 60 86 48 01 65 03 04 02 03 05 00

6. Considérations sur la sécurité

Les mises en œuvre doivent protéger la clé privée RSA et la clé de chiffrement de contenu. La compromission de la clé privée RSA peut résulter en la divulgation de tous les messages protégés avec cette clé. La compromission de la clé de chiffrement de contenu peut résulter en la divulgation du contenu chiffré associé.

La génération de paires de clés publique/privée RSA s'appuie sur des nombres aléatoires. L'utilisation de générateurs de nombres pseudo aléatoires inadéquats (PRNG, *pseudo-random number generator*) pour générer des clés de chiffrement peut résulter en peu ou pas du tout de sécurité. Un attaquant peut trouver beaucoup plus facile de reproduire l'environnement du PRNG qui a produit les clés, cherchant dans le petit ensemble résultant de possibilités, plutôt qu'une recherche en force brute dans l'espace de clés total. La génération de nombres aléatoires de qualité est difficile. La [RFC1750] offres des lignes directrices importantes dans ce domaine.

Généralement, une bonne pratique de la cryptographie emploie une certaine paire de clés RSA dans un seul schéma. Cette pratique évite le risque que des vulnérabilités dans un schéma puissent compromettre la sécurité de l'autre, et peut être essentielle pour conserver une sécurité démontrable. Bien que PKCS n° 1 version 1.5 [RFC2313] ait été employé pour le transport de clés et la signature numérique sans aucune mauvaise interaction connue, une telle utilisation combinée d'une paire de clés RSA n'est pas recommandée à l'avenir. Donc, une paire de clés RSA utilisée pour le transport de clés RSAES-OAEP ne devrait pas être aussi utilisée pour d'autres objets. Pour des raisons similaires, une paire de clés RSA devrait toujours être utilisée avec les mêmes paramètres RSAES-OAEP.

La présente spécification exige des mises en œuvre qu'elles prennent en charge la fonction de hachage unidirectionnelle SHA-1 pour l'interopérabilité, mais la prise en charge d'autres fonctions de hachage unidirectionnelles est permise. Au moment de la rédaction du présent mémoire, les meilleures attaques (connues) de collision contre SHA-1 sont des attaques génériques avec une complexité de 2^80, où 80 est la moitié de la longueur en bits de la valeur du hachage. Quand une fonction de hachage unidirectionnelle est utilisée avec un schéma de signature numérique, une attaque de collision est facilement traduite en une fausse signature. Donc, l'utilisation de SHA-1 dans un schéma de signature numérique donne un niveau de sécurité de pas plus de 80 bits. Si un niveau de sécurité supérieur est désiré, une fonction de hachage unidirectionnelle sûre avec une plus longue valeur de hachage est nécessaire. SHA-256, SHA-384, et SHA-512 sont des candidats probables [SHA2].

Les métriques pour les choix d'une fonction de hachage unidirectionnelle à utiliser dans les signatures numériques ne s'appliquent pas directement à l'algorithme de transport de clés RSAES-OAEP, car une attaque de collision sur la fonction de hachage unidirectionnelle ne se traduit pas directement en une attaque sur l'algorithme de transport de clés, sauf si le paramètre de codage P varie (dans ce cas une collision de la valeur de hachage pour les différents paramètres de codage peut être exploitée).

Néanmoins, pour la cohérence avec la pratique des schémas de signature numérique, et au cas où le paramètre de codage P n'est pas la chaîne vide, il est recommandé que la même règle d'approximation soit appliquée au choix d'une fonction de hachage unidirectionnelle à utiliser avec RSAES-OAEP. C'est-à-dire que la fonction de hachage unidirectionnelle devrait être choisie de façon telle que la longueur en bits de la valeur de hachage soit au moins deux fois celle du niveau de sécurité désiré en bits.

Une clé publique RSA de 1024 bits et SHA-1 fournissent tous deux un niveau de sécurité d'environ 80 bits. Dans [NISTGUIDE], l'Institut National des Normes et Technologies suggère qu'un niveau de sécurité de 80 bits est adéquat pour la plupart des applications jusqu'en 2015. Si un niveau de sécurité supérieur à 80 bits est nécessaire, une clé publique RSA plus longue et une fonction de hachage unidirectionnelle sûre avec une plus longue valeur de hachage sont nécessaires. Là encore, SHA-256, SHA-384, et SHA-512 sont des candidats probables pour une telle fonction de hachage unidirectionnelle. Pour cette raison, les identifiants d'algorithme pour ces fonctions de hachage unidirectionnelle sont inclus

dans le module ASN.1 de l'Appendice A.

La même fonction de hachage unidirectionnelle devrait être employée pour la fonction de hachage et la fonction de génération de gabarit (maskGenFunc), mais cela n'est pas exigé. Utiliser la même fonction de hachage unidirectionnelle réduit le potentiel d'erreurs de mise en œuvre.

7. Considérations relatives à l'IANA

Dans la CMS, les algorithmes sont identifiés par des identifiants d'objet (OID). Tous les OID utilisés dans le présent document ont été alloués dans des documents de normes de chiffrement à clés publiques (PKCS, *Public-Key Cryptography Standards*) ou par l'Institut National des Normes et Technologies (NIST). Aucune autre action de l'IANA n'est nécessaire pour le présent document ou ses mises à jour prévues.

8. Déclaration de droits de propriété intellectuelle

L'IETF ne prend pas position sur la validité et la portée de tout droit de propriété intellectuelle ou autres droits qui pourraient être revendiqués au titre de la mise en œuvre ou l'utilisation de la technologie décrite dans le présent document ou sur la mesure dans laquelle toute licence sur de tels droits pourrait être ou n'être pas disponible ; pas plus qu'elle ne prétend avoir accompli aucun effort pour identifier de tels droits. Les informations sur les procédures de l'ISOC au sujet des droits dans les documents de l'ISOC figurent dans les BCP 78 et BCP 79.

Des copies des dépôts d'IPR faites au secrétariat de l'IETF et toutes assurances de disponibilité de licences, ou le résultat de tentatives faites pour obtenir une licence ou permission générale d'utilisation de tels droits de propriété par ceux qui mettent en œuvre ou utilisent la présente spécification peuvent être obtenues sur le répertoire en ligne des IPR de l'IETF à http://www.ietf.org/ipr.

L'IETF invite toute partie intéressée à porter son attention sur tous copyrights, licences ou applications de licence, ou autres droits de propriété qui pourraient couvrir les technologies qui peuvent être nécessaires pour mettre en œuvre la présente norme. Prière d'adresser les informations à l'IETF à ietf-ipr@ietf.org.

9. Remerciements

Le présent document est le résultat de contributions de nombreux professionnels. Merci de leur dur labeur à tous les membres du groupe de travail S/MIME de l'IETF. Des remerciements tout particuliers à Burt Kaliski, Jakob Jonsson, François Rousseau, et Jim Schaad.

10. Références

Cette section donne les références normatives et pour information.

10.1 Références normatives

[3DES] American National Standards Institute. ANSI X9.52-1998, "Triple Data Encryption Algorithm Modes of Operation", 1998.

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. *(MàJ par* RFC8174)

[RFC2633] B. Rmasdell, "Spécification de message S/MIME version 3", juin 1999. (*Obsolète, voir* RFC3851) *(P.S.)*

[RFC3280] R. Housley, W. Polk, W. Ford et D. Solo, "Profil de certificat d'infrastructure de clé publique X.509 et de liste de révocation de certificat (CRL) pour l'Internet", avril 2002. *(Obsolète, voir* RFC5280)

[RFC3369] R. Housley, "Syntaxe de message cryptographique (CMS)", août 2002. (*Obsolète, voir* RFC3852) *(P.S.)*

[RFC3447] J. Jonsson et B. Kaliski, "Normes de cryptographie à clés publiques (PKCS) n° 1 : Spécifications de la cryptographie RSA version 2.1", février 2003. *(Obsolète, remplacée par* RFC8017) *(Information)*

[SHA1] National Institute of Standards and Technology. FIPS Pub 180-1: "Secure Hash Standard", avril 1995.

[X.208-88] Recommandation UIT-T X.208 "Spécification de la notation n° 1 de syntaxe abstraite (ASN.1)", 1988.

[X.209-88] Recommandation UIT-T X.209, "Spécification des règles de codage de base pour la notation n° 1 de syntaxe abstraite (ASN.1)", 1988.

[X.509-88] Recommandation UIT-T X.509, "L'annuaire - cadre d'authentification", 1988.

10.2 Références pour information

[CRYPTO98] Bleichenbacher, D. "Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1", dans H. Krawczyk (editor), "Advances in Cryptology - CRYPTO '98 Proceedings", notes de lecture dans Computer Science 1462 (1998), Springer-Verlag, pp. 1-12.

[NISTGUIDE] National Institute of Standards et Technology, "Key Management Guideline, Part 1: General Guidance", juin 2002. [http://csrc.nist.gov/encryption/kms/guideline-1.pdf]

[RFC1750] D. Eastlake 3rd et autres, "Recommandations d'aléa pour la sécurité", décembre 1994. *(Info., remplacée par* RFC4086)

[RFC2246] T. Dierks et C. Allen, "Protocole TLS version 1.0", janvier 1999. *(P.S. ; MàJ par* RFC7919)

[RFC2313] B. Kaliski, "PKCS n° 1 : Chiffrement RSA version 1.5", mars 1998.

[RFC3218] E. Rescorla, "Empêcher l'attaque du million de messages sur la syntaxe de message cryptographique", janvier 2002. *(Information)*

[RFC3279] L. Bassham, W. Polk et R. Housley, "Algorithmes et identifiants pour le profil de certificat d'infrastructure de clé publique X.509 et de liste de révocation de certificat (CRL) pour l'Internet", avril 2002.

[RSALABS] Bleichenbacher, D., B. Kaliski, and J. Staddon, "Recent Results on PKCS #1: RSA Encryption Standard". RSA Laboratories' Bulletin No. 7, 26 juin 1998. [http://www.rsasecurity.com/rsalabs/bulletins]

[SHA2] National Institute of Standards and Technology, Draft FIPS Pub 180-2: "Specifications for the Secure Hash Standard", mai 2001. [http://csrc.nist.gov/encryption/shs/dfips-180-2.pdf]

[SSL] Freier, A., P. Karlton, and P. Kocher, "The SSL Protocol, Version 3.0". Netscape Communications. novembre 1996. [http://wp.netscape.com/eng/ssl3/draft302.txt]

Appendice A. Module ASN.1

```
CMS-RSAES-OAEP
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) modules(0) cms-rsaes-oaep(20) }

ÉTIQUETTES IMPLICITES DE DEFINITIONS ::= DÉBUT

-- EXPORTE TOUT --

IMPORTE
AlgorithmIdentifier
```

```
 DE PKIX1Explicit88 -- RFC 3280
 { iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-pkix1-explicit(18) };

IDENTIFIANT D'OBJET pkcs-1 ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) }

IDENTIFIANT D'OBJET rsaEncryption ::=  { pkcs-1 1 }

IDENTIFIANT D'OBJET id-RSAES-OAEP ::=  { pkcs-1 7 }

RSAES-OAEP-params  ::=  SEQUENCE  {
     hashFunc        [0] AlgorithmIdentifier DEFAUT sha1Identifier,
     maskGenFunc [1] AlgorithmIdentifier DEFAUT mgf1SHA1Identifier,
     pSourceFunc    [2] AlgorithmIdentifier DEFAUT pSpecifiedEmptyIdentifier }

sha1Identifier  AlgorithmIdentifier  ::=  { id-sha1, NULL }
sha256Identifier  AlgorithmIdentifier  ::=  { id-sha256, NULL }
sha384Identifier  AlgorithmIdentifier  ::=  { id-sha384, NULL }
sha512Identifier  AlgorithmIdentifier  ::=  { id-sha512, NULL }
mgf1SHA1Identifier  AlgorithmIdentifier  ::= { id-mgf1, sha1Identifier }
mgf1SHA256Identifier  AlgorithmIdentifier  ::= { id-mgf1, sha256Identifier }
mgf1SHA384Identifier  AlgorithmIdentifier  ::= { id-mgf1, sha384Identifier }
mgf1SHA512Identifier  AlgorithmIdentifier  ::= { id-mgf1, sha512Identifier }
pSpecifiedEmptyIdentifier  AlgorithmIdentifier ::= { id-pSpecified, nullOctetString }

nullOctetString  CHAINE D'OCTETS (TAILLE (0))  ::=  { "H }

IDENTIFIANT D'OBJET id-sha1 ::=   iso(1) identified-organization(3) oiw(14) secsig(3) algorithms(2) 26 }

IDENTIFIANT D'OBJET id-sha256 ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3)
                                        nistalgorithm(4) hashalgs(2) 1 }

IDENTIFIANT D'OBJET id-sha384 ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3)
                                        nistalgorithm(4) hashalgs(2) 2 }

IDENTIFIANT D'OBJET id-sha512 ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3)
                                        nistalgorithm(4) hashalgs(2) 3 }

IDENTIFIANT D'OBJET id-mgf1 ::=  { pkcs-1 8 }

IDENTIFIANT D'OBJET id-pSpecified  IDENTIFIANT D'OBJET ::= { pkcs-1 9 }

rSAES-OAEP-Default-Identifier AlgorithmIdentifier ::= { id-RSAES-OAEP, { sha1Identifier, mgf1SHA1Identifier,
                                                          pSpecifiedEmptyIdentifier } }

rSAES-OAEP-SHA256-Identifier AlgorithmIdentifier ::= { id-RSAES-OAEP, { sha256Identifier, mgf1SHA256Identifier,
                                                          pSpecifiedEmptyIdentifier } }

rSAES-OAEP-SHA384-Identifier AlgorithmIdentifier ::= { id-RSAES-OAEP, { sha384Identifier, mgf1SHA384Identifier,
                                                          pSpecifiedEmptyIdentifier } }

rSAES-OAEP-SHA512-Identifier AlgorithmIdentifier ::= { id-RSAES-OAEP, { sha512Identifier, mgf1SHA512Identifier,
                                                          pSpecifiedEmptyIdentifier } }

FIN
```

Adresse de l'auteur

Russell Housley
Vigil Security, LLC
918 Spring Knoll Drive
Herndon, VA 20170
USA

mél : housley@vigilsec.com

Déclaration complète de droits de reproduction

Remerciement
Le financement de la fonction d'édition des RFC est actuellement fourni par l'Internet Society.