UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T X.843

(10/2000)

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

SÉRIE X: RÉSEAUX DE DONNÉES ET
COMMUNICATION ENTRE SYSTÈMES OUVERTS

Sécurité

Technologies de l'information – Techniques de sécurité – Spécification de services de tiers de confiance (TTP) pour la prise en charge des applications de signature numérique

Recommandation UIT-T X.843

(Antérieurement Recommandation du CCITT)

RECOMMANDATIONS UIT-T DE LA SÉRIE X

RÉSEAUX DE DONNÉES ET COMMUNICATION ENTRE SYSTÈMES OUVERTS

RÉSEAUX PUBLICS DE DONNÉES	
Services et fonctionnalités	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalisation et commutation	X.50–X.89
Aspects réseau	X.90–X.149
Maintenance	X.150–X.179
Dispositions administratives	X.180–X.199
INTERCONNEXION DES SYSTÈMES OUVERTS	
Modèle et notation	X.200–X.209
Définitions des services	X.210–X.219
Spécifications des protocoles en mode connexion	X.220–X.229
Spécifications des protocoles en mode sans connexion	X.230–X.239
Formulaires PICS	X.240–X.259
Identification des protocoles	X.260–X.269
Protocoles de sécurité	X.270–X.279
Objets gérés des couches	X.280–X.289
Tests de conformité	X.290–X.299
INTERFONCTIONNEMENT DES RÉSEAUX	
Généralités	X.300–X.349
Systèmes de transmission de données par satellite	X.350–X.369
Réseaux à protocole Internet	X.370–X.399
SYSTÈMES DE MESSAGERIE	X.400–X.499
ANNUAIRE	X.500–X.599
RÉSEAUTAGE OSI ET ASPECTS SYSTÈMES	
Réseautage	X.600–X.629
Efficacité	X.630–X.639
Qualité de service	X.640–X.649
Dénomination, adressage et enregistrement	X.650–X.679
Notation de syntaxe abstraite numéro un (ASN.1)	X.680–X.699
GESTION OSI	
Cadre général et architecture de la gestion-systèmes	X.700–X.709
Service et protocole de communication de gestion	X.710–X.719
Structure de l'information de gestion	X.720–X.729
Fonctions de gestion et fonctions ODMA	X.730–X.799
SÉCURITÉ	**X.800–X.849**
APPLICATIONS OSI	
Engagement, concomitance et rétablissement	X.850–X.859
Traitement transactionnel	X.860–X.879
Opérations distantes	X.880–X.899
TRAITEMENT RÉPARTI OUVERT	X.900–X.999

Pour plus de détails, voir la Liste des Recommandations de l'UIT-T.

NORME INTERNATIONALE ISO/CEI 15945

RECOMMANDATION UIT-T X.843

TECHNOLOGIES DE L'INFORMATION – TECHNIQUES DE SÉCURITÉ – SPÉCIFICATION DE SERVICES DE TIERS DE CONFIANCE (TTP) POUR LA PRISE EN CHARGE DES APPLICATIONS DE SIGNATURE NUMÉRIQUE

Résumé

La présente Recommandation | Norme internationale définit les services nécessaires à la prise en charge des applications de signature numérique pour la non-répudiation de création d'un document. Comme cette prise en charge suppose que le document est intègre et que le créateur est authentique, les services décrits peuvent également être couplés aux services responsables de l'intégrité et de l'authenticité.

Source

La Recommandation X.843 de l'UIT-T, élaborée par la Commission d'études 7 (1997-2000) de l'UIT-T, a été approuvée par l'AMNT (Montréal, 27 septembre 2000 – 6 octobre 2000). Un texte identique est publié comme Norme Internationale ISO/CEI 15945.

AVANT-PROPOS

L'UIT (Union internationale des télécommunications) est une institution spécialisée des Nations Unies dans le domaine des télécommunications. L'UIT-T (Secteur de la normalisation des télécommunications) est un organe permanent de l'UIT. Il est chargé de l'étude des questions techniques, d'exploitation et de tarification, et émet à ce sujet des Recommandations en vue de la normalisation des télécommunications à l'échelle mondiale.

L'Assemblée mondiale de normalisation des télécommunications (AMNT), qui se réunit tous les quatre ans, détermine les thèmes d'étude à traiter par les Commissions d'études de l'UIT-T, lesquelles élaborent en retour des Recommandations sur ces thèmes.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution 1 de l'AMNT.

Dans certains secteurs des technologies de l'information qui correspondent à la sphère de compétence de l'UIT-T, les normes nécessaires se préparent en collaboration avec l'ISO et la CEI.

NOTE

Dans la présente Recommandation, l'expression "Administration" est utilisée pour désigner de façon abrégée aussi bien une administration de télécommunications qu'une exploitation reconnue.

DROITS DE PROPRIÉTÉ INTELLECTUELLE

L'UIT attire l'attention sur la possibilité que l'application ou la mise en œuvre de la présente Recommandation puisse donner lieu à l'utilisation d'un droit de propriété intellectuelle. L'UIT ne prend pas position en ce qui concerne l'existence, la validité ou l'applicabilité des droits de propriété intellectuelle, qu'ils soient revendiqués par un Membre de l'UIT ou par une tierce partie étrangère à la procédure d'élaboration des Recommandations.

A la date d'approbation de la présente Recommandation, l'UIT n'avait pas été avisée de l'existence d'une propriété intellectuelle protégée par des brevets à acquérir pour mettre en œuvre la présente Recommandation. Toutefois, comme il ne s'agit peut-être pas de renseignements les plus récents, il est vivement recommandé aux responsables de la mise en œuvre de consulter la base de données des brevets du TSB.

TABLE DES MATIÈRES

Introduction

Actuellement le développement des technologies de l'information ainsi que celui de l'infrastructure mondiale de communication ouvrent la possibilité d'implémenter le commerce électronique dans des dimensions économiques appropriées. Les signatures numériques constituent une technique importante pour ajouter de la sécurité à ces applications commerciales et à d'autres champs d'application qui ont besoin de transactions électroniques légalement efficaces.

Les signatures numériques conviennent pour assurer l'intégrité des données et l'authentification des participants aux transactions. Elles peuvent constituer une analogie avec la signature manuscrite pour les commandes, offres et contrats numériques. La caractéristique la plus importante des signatures numériques dans ce contexte est qu'une personne ayant signé un document ne peut pas par la suite le nier. Cette caractéristique est appelée "non-répudiation de création" d'un document.

Dans de nombreux pays et dans le contexte international, la législation concernant les signatures numériques est mise en avant dans le but de prendre en charge le développement du commerce électronique et d'autres champs d'application qui ont besoin de transactions électroniques légalement efficaces.

Il existe un certain nombre de normes qui spécifient les signatures numériques ainsi que leur utilisation pour différents besoins tels que la non-répudiation ou l'authentification. Un certain nombre d'applications commerciales ainsi que des services d'offres TTP liés aux signatures numériques sont implémentés ou prévus. L'interopérabilité de ces tiers TTP entre eux et avec les applications commerciales est nécessaire pour une utilisation mondiale économiquement et légalement efficace des signatures numériques.

L'objectif de la présente Recommandation | Norme internationale est de définir les services nécessaires à la prise en charge des applications de signature numérique pour la non-répudiation de création. Comme l'utilisation des mécanismes pour la non-répudiation de création d'un document suppose l'intégrité du document et l'authenticité du créateur, les services décrits dans la présente Recommandation | Norme internationale peuvent également être combinés afin d'implémenter les services pour l'intégrité et l'authenticité. Ceci est réalisé d'une manière qui favorise l'interopérabilité entre tiers TTP ainsi qu'entre tiers TTP et applications commerciales.

NOTE – Il n'y a aucune raison inhérente que chaque tiers TTP prévoyant la prise en charge des applications de signature numérique doive offrir tous ces services. Il est possible que des tiers TTP offrant des services différents coopèrent pour prendre en charge l'utilisation de signatures numériques. Toutefois, du point de vue des applications commerciales potentielles, la totalité de la gamme des services peut être nécessaire et l'interopérabilité devient encore plus importante dans ce scénario. Il s'agit d'une raison supplémentaire de recueillir l'ensemble de tous ces services dans un document unique.

NORME INTERNATIONALE

RECOMMANDATION UIT-T

TECHNOLOGIES DE L'INFORMATION – TECHNIQUES DE SÉCURITÉ – SPÉCIFICATION DE SERVICES DE TIERS DE CONFIANCE (TTP) POUR LA PRISE EN CHARGE DES APPLICATIONS DE SIGNATURE NUMÉRIQUE

1 Domaine d'application

La présente Recommandation | Norme internationale définit les services TTP nécessaires à la prise en charge des applications de signature numérique aux fins de non-répudiation de création de documents.

La présente Recommandation | Norme internationale définit également les interfaces et protocoles permettant l'interopérabilité entre des entités associées à ces services TTP.

Les définitions de services techniques et de protocoles sont nécessaires pour permettre l'implémentation des services TTP et des applications commerciales associées.

La présente Recommandation | Norme internationale est centrée sur:

- l'implémentation et l'interopérabilité;
- les spécifications de services;
- les prescriptions techniques.

La présente Recommandation | Norme internationale ne décrit pas la gestion des tiers TTP ou d'autres questions organisationnelles, d'exploitation ou personnelles. Ces sujets sont principalement couverts dans la Rec. UIT-T X.842 | ISO/CEI TR 14516, *Technologies de l'information – Techniques de sécurité – Lignes directrices pour l'utilisation et la gestion des services de tiers de confiance*.

NOTE 1 – Puisque l'interopérabilité est le sujet principal de la présente Recommandation | Norme internationale, les restrictions suivantes s'appliquent:

i) ne sont pris en considération dans la présente Recommandation | Norme internationale que les services qui peuvent être fournis par un TTP soit aux entités terminales soit à un autre TTP;

ii) ne sont pris en considération que les services qui peuvent être demandés ou fournis au moyen de messages numériques normalisables;

iii) ne sont spécifiés en détail que les services pour lesquels des messages normalisés acceptables à grande échelle peuvent être adoptés au moment de la publication de la présente Recommandation | Norme internationale.

D'autres services seront spécifiés dans des documents distincts lorsque des messages normalisés acceptables à grande échelle seront disponibles pour eux. La définition des services horodateurs en particulier fera l'objet d'un document séparé.

NOTE 2 – La structure des données et les messages figurant dans la présente Recommandation | Norme internationale seront spécifiés conformément aux documents RFC 2510 et 2511 (pour les services de gestion de certificats) et RFC 2560 (pour les services OCSP). Le format de la demande de certificat permet aussi l'interopérabilité avec le système PKCS # 10. Voir l'Annexe C pour des références aux documents mentionnés dans la présente note.

NOTE 3 – Il existe d'autres tentatives de normalisation des services TTP dans des environnements et des applications spécifiques, comme SET ou EDIFACT. Ils ne s'inscrivent pas dans le cadre de la présente Recommandation | Norme internationale.

NOTE 4 – La présente Recommandation | Norme internationale définit des spécifications techniques pour les services. Ces spécifications ne dépendent pas des politiques, des règlements juridiques particuliers ou des modèles organisationnels (qui pourraient par exemple définir comment sont réparties les tâches et les responsabilités entre les autorités de certification et les autorités d'enregistrement). Evidemment, la politique des TTP offrant les services décrits dans la présente Recommandation | Norme internationale devra spécifier comment les règlements juridiques et les autres aspects susmentionnés seront respectés par le TTP. La politique doit en particulier spécifier comment la validité des signatures et des certificats numériques est déterminée.

2 Références normatives

Les Recommandations et Normes internationales suivantes contiennent des dispositions qui, par suite de la référence qui y est faite, constituent des dispositions valables pour la présente Recommandation | Norme internationale. Au moment de la publication, les éditions indiquées étaient en vigueur. Toutes Recommandations et Normes sont sujettes à révision et les parties prenantes aux accords fondés sur la présente Recommandation | Norme internationale sont invitées à rechercher la possibilité d'appliquer les éditions les plus récentes des Recommandations et Normes indiquées ci-après. Les membres de la CEI et de l'ISO possèdent le registre des Normes internationales en vigueur. Le Bureau de la normalisation des télécommunications de l'UIT tient à jour une liste des Recommandations de l'UIT-T en vigueur.

2.1 Recommandations | Normes internationales identiques

- Recommandation UIT-T X.501 (1997) | ISO/CEI 9594-2:1998, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: les modèles.*
- Recommandation UIT-T X.509 (2000) | ISO/CEI 9594-8:2001, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: cadre général des certificats de clé publique et d'attribut*.
- Recommandation UIT-T X.520 (1997) | ISO/CEI 9594-6:1998, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: types d'attributs sélectionnés.*
- Recommandation UIT-T X.680 (1997) | ISO/CEI 8824-1:1998, *Technologies de l'information – Notation de syntaxe abstraite numéro un (ASN.1): spécification de la notation de base.*
- Recommandation UIT-T X.681 (1997) | ISO/CEI 8824-2:1998, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification des objets informationnels.*
- Recommandation UIT-T X.682 (1997) | ISO/CEI 8824-3:1998, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification des contraintes.*
- Recommandation UIT-T X.683 (1997) | ISO/CEI 8824-4:1998, *Technologies de l'information – Notation de syntaxe abstraite numéro un: paramétrage des spécifications de la notation de syntaxe abstraite numéro un.*
- Recommandation UIT-T X.690 (1997) | ISO/CEI 8825-1:1998, *Technologies de l'information – Règles de codage ASN.1: spécification des règles de codage de base, des règles de codage canoniques et des règles de codage distinctives.*
- Recommandation UIT-T X.810 (1995) | ISO/CEI 10181-1:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: aperçu général.*
- Recommandation UIT-T X.813 (1996) | ISO/CEI 10181-4:1997, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: non-répudiation.*

2.2 Autres références

- ISO/CEI 9796-2:1997, *Technologies de l'information – Techniques de sécurité – Schémas de signature numérique rétablissant le message – Partie 2: Mécanismes utilisant une fonction de hachage.*
- ISO/CEI 9796-3:2000, *Technologies de l'information – Techniques de sécurité – Schéma de signature numérique rétablissant le message – Partie 3: Mécanismes basés sur les logarithnmes discrets.*
- ISO/CEI 10118-1:1994, *Technologies de l'information – Techniques de sécurité – Fonctions de brouillage – Partie 1: Généralités.*
- ISO/CEI 10118-2:1994, *Technologies de l'information -Techniques de sécurité – Fonctions de brouillage – Partie 2: Fonctions de brouillage utilisant un algorithme de chiffrement par blocs de n bits.*
- ISO/CEI 10118-3:1998, *Technologies de l'information – Techniques de sécurité – Fonctions de brouillage – Partie 3: Fonctions de hachage dédiées.*
- ISO/CEI 11770-1:1996, *Technologies de l'information – Techniques de sécurité – Partie 1: Cadre général.*
- ISO/CEI 11770-2:1996, *Technologies de l'information – Techniques de sécurité – Gestion de clés – Partie 2: Mécanismes utilisant des techniques symétriques.*

- ISO/CEI 11770-3:1999, *Technologies de l'information – Techniques de sécurité – Gestion de clés – Partie 3: Mécanismes utilisant des techniques asymétriques.*
- ISO/CEI 13888-1:1997, *Technologies de l'information – Techniques de sécurité – Non-répudiation – Partie 1: Généralités.*
- ISO/CEI 13888-2:1998, *Technologies de l'information – Techniques de sécurité – Non-répudiation – Partie 2: Mécanismes utilisant des techniques symétriques.*
- ISO/CEI 13888-3:1997, *Technologies de l'information – Techniques de sécurité – Non-répudiation – Partie 3: Mécanismes utilisant des techniques asymétriques.*
- ISO/CEI 14888-1:1998, *Technologies de l'information – Techniques de sécurité – Signatures digitales avec appendice – Partie 1: Généralités.*
- ISO/CEI 14888-2:1999, *Technologies de l'information – Techniques de sécurité – Signatures digitales avec appendice – Partie 2: Mécanismes basés sur des identités.*
- ISO/CEI 14888-3:1998, *Technologies de l'information – Techniques de sécurité – Signatures digitales avec appendice – Partie 3: Mécanismes fondés sur certificat*.
- ISO/CEI 15946-2 (à publier), *Technologies de l'information – Techniques de sécurité – Techniques cryptographiques basées sur des courbes elliptiques – Partie 2: Signatures digitales*.

3 Définitions

Le terme suivant est utilisé comme défini dans l'ISO/CEI 11770-1:

gestion de clés

Le terme suivant est utilisé comme défini dans l'ISO/CEI 10181-1:

tiers de confiance (TTP, *trusted third party*)

Les termes ci-après sont utilisés tels que définis dans la Rec. UIT-T X.509 | ISO/CEI 9594-8:

certificat CA
autorité de certification (CA)
certificat de clé publique

NOTE – La dénomination abrégée "certificat" sera également employée dans la présente Recommandation | Norme internationale pour désigner un "certificat de clé publique".

politique de certification
liste d'annulation de certificats (CRL, *certificate revocation list*)
chemin de certification

Les termes suivants sont utilisés tels que définis dans l'ISO/CEI 10118-1:

fonction de hachage, fonction de brouillage
code de hachage; valeur de hachage

Le terme suivant est utilisé comme défini dans l'ISO/CEI 14888-1:

paramètre de domaine

Pour les besoins de la présente Recommandation | Norme internationale, les définitions suivantes s'appliquent:

3.1 services de gestion de certificats: tous les services nécessaires à la maintenance du cycle de vie des certificats, y compris l'enregistrement, certification, distribution et annulation de certificats.

3.2 service de certification: service de création et d'attribution de certificats assuré par une autorité de certification et décrit dans l'ISO/CEI 9594-8:1995.

3.3 signature numérique: transformation cryptographique d'une unité de donnée qui permet au destinataire de l'unité de donnée de prouver l'origine et l'intégrité de cette unité de donnée, qui protège l'émetteur et le destinateur de l'unité de donnée contre un faux fabriqué par un tiers et qui protège l'émetteur contre un faux fabriqué par le destinataire.

NOTE – Les signatures numériques peuvent être utilisées par des entités finales (voir ci-après) pour des besoins d'authentification, d'intégrité de données et de non-répudiation de création de données. L'utilisation aux fins de non-répudiation de création de données est le plus important de ces besoins de disposer de signatures numériques ayant force légale. La définition ci-dessus provient de l'ISO/CEI 9798-1.

3.4 autorité CA de confiance directe: une autorité CA de confiance directe est une autorité CA dont la clé publique a été obtenue et est en cours de mémorisation par une entité finale, en toute confiance et sécurité, et dont la clé publique est acceptée par cette entité finale dans le contexte d'une ou plusieurs applications.

3.5 clé de CA de confiance directe: une clé de CA de confiance directe est une clé publique d'une autorité CA de confiance directe. Elle a été obtenue et est en cours de mémorisation par une entité finale, en toute confiance et sécurité. Elle sert à vérifier des certificats sans être elle-même vérifiée au moyen d'un certificat créé par une autre autorité.

NOTE – Si, par exemple, les autorités CA de plusieurs organisations se certifient réciproquement les unes les autres (voir l'Annexe A), l'autorité CA de confiance directe pour une entité peut être l'autorité CA de l'organisation à laquelle appartient cette entité. Les autorités CA de confiance directe et les clés de CA de confiance directe peuvent varier d'une entité à l'autre. Une entité peut considérer plusieurs autorités CA comme des autorités CA de confiance directe.

3.6 service d'annuaire: service pour la recherche et la récupération d'informations à partir d'un catalogue d'objets bien définis, qui peut contenir des informations sur les certificats, numéros de téléphone, conditions d'accès, adresses, etc. Un exemple en est un service d'annuaire conforme à la Rec. UIT-T X.500 | ISO/CEI 9594-1.

3.7 service de distribution de clés: service de distribution de clés en toute sécurité à des entités autorisées, fourni par un Centre de distribution de clés et décrit dans l'ISO/CEI 11770-1.

3.8 non-répudiation de création: protection contre un démenti fallacieux par une entité du fait qu'elle a créé le contenu d'un message (c'est-à-dire, du fait qu'elle est responsable du contenu d'un message).

3.9 environnement de sécurité personnelle (PSE, *personal security environment*): stockage local sûr pour la clé privée d'une entité, pour la clé d'une autorité CA de confiance directe et pour d'autres données éventuelles. En fonction de la politique de sécurité appliquée par l'entité ou en fonction des prescriptions du système, il peut s'agir par exemple d'un fichier protégé par chiffrement ou d'un jeton de matériel inviolable.

3.10 service de personnalisation: service de stockage d'informations cryptographiques (spécialement des clés privées) dans un environnement PSE.

NOTE – Les mesures organisationnelles et de sécurité physique destinées à un tel service ne s'inscrivent pas dans le domaine d'application de la présente Recommandation | Norme internationale. Pour les mesures organisationnelles, se reporter à la Rec. UIT-T X.842 | ISO/CEI TR 14516, Lignes directrices pour l'utilisation et la gestion des services de tiers de confiance.

3.11 annuaire de clé publique (PKD, *public key directory*): annuaire contenant un (sous-)ensemble bien défini de certificats de clé publique. Cet annuaire peut contenir des certificats provenant de différentes autorités de certification.

3.12 infrastructure de clé publique (PKI, *public key infrastructure*): système constitué de tiers TTP, avec les services qu'ils fournissent pour la prise en charge de l'application (y compris la création et la validation) de signatures numériques, et des personnes ou composants techniques qui utilisent ces services.

NOTE – Parfois on appelle entités finales les personnes et composants techniques qui participent à une infrastructure PKI en utilisant les services de tiers TTP, mais sans être eux-mêmes des tiers TTP. Un exemple d'équipement technique utilisé par une entité finale est une carte à puce qui peut être utilisée comme mémoire de stockage et/ou dispositif de traitement.

3.13 autorité d'enregistrement (RA, *registration authority*): autorité habilitée et chargée en toute confiance d'exécuter un service d'enregistrement tel que décrit ci-dessous.

3.14 service d'enregistrement: service d'identification d'entités et de leur enregistrement d'une manière qui permet l'attribution sûre de certificats à ces entités.

3.15 service d'horodatage: service attestant de l'existence d'une donnée électronique à un moment précis dans le temps.

NOTE – Les services d'horodatage sont utiles et probablement indispensables pour prendre à charge la validation à long terme de signatures. Ils seront définis dans une norme distincte.

4 Abréviations

Pour les besoins de la présente Recommandation | Norme internationale, les abréviations suivantes s'appliquent:

CA Autorité de certification (*certification authority*)

CRL Liste d'annulation de certificat (*certificate revocation list*)

EE Entité finale (*end entity*)

OCSP Protocole de statut de certificat en ligne (*on-line certificate status protocol*)

PKD Annuaire de clé publique (*public key directory*)

PKI Infrastructure de clé publique (*public key infrastructure*)

PSE Environnement de sécurité personnelle (*personal security environment*)

RA Autorité d'enregistrement (*registration authority*)

TTP Tiers de confiance (*trusted third party*)

5 Classification descriptive des services

Le présent article décrit des services qui peuvent être utilisés dans le contexte de services TTP pour des signatures numériques. Une description de haut niveau de ces services y est donnée, indépendante des formats de données, d'algorithmes, de langages de descriptions, etc.

Une spécification détaillée d'un certain nombre de ces services est fournie dans les articles ci-après.

Il n'est pas nécessaire que chaque tiers TTP prévoyant de prendre en charge l'application de signatures numériques offre la totalité de ces services. Il est possible que plusieurs tiers TTP offrant des services différents collaborent dans la prise en charge de l'utilisation de signatures numériques.

NOTE 1 – Sachant que l'interopérabilité est la question principale de la présente Recommandation | Norme internationale, seuls sont décrits ici les services qui sont offerts par un tiers TTP soit à des entités finales soit à un autre tiers TTP. En outre, seuls sont couverts les services qui peuvent être demandés ou fournis à l'aide de messages numériques normalisables. (Cela n'implique pas cependant que les messages normalisés soient effectivement définis pour tous les services mentionnés dans la présente Recommandation | Norme internationale.)

Les exemples suivants illustrent des services qui **ne sont pas** couverts:

1) enregistrement d'événements liés à la sécurité. Compte tenu de l'infrastructure PKI d'une signature numérique, il s'agit d'un service interne de tiers TTP mais qui n'est pas offert à des entités;

2) services cryptographiques généraux (par exemple service de chiffrement). Les procédés tels que le chiffrement font partie de certains services mais ne sont pas pertinents en tant que service autonome dans le contexte des signatures numériques;

3) archivage ou récupération de clés. Il peut s'agir d'un service interne pour les clés de CA de confiance directe. Ces actions ne sont pas généralement exécutées pour les clés de signatures numériques d'entités finales.

NOTE 2 – Les services d'horodatage seront définis dans un document distinct.

5.1 Services de gestion de certificats

Le présent paragraphe contient une description des services suivants qui font partie du cycle de vie des certificats:

- enregistrement;
- certification de clé publique;
- annulation de certificats;
- mise à jour de certificats;
- mise à jour de clé.

Une spécification détaillée du flux de messages en ligne de ces services (à l'exception de la détermination du statut du certificat) est fournie à l'article 7 et la spécification ASN.1 des structures de données nécessaires à ces messages est fournie à l'article 8. Les spécifications analogues pour la détermination du statut de certification en ligne (voir § 5.1.3.3, deuxième méthode) sont fournies à l'article 9.

Les protocoles d'accès à l'annuaire qui sont utilisés pour rendre publiquement disponibles les certificats et listes CRL ne sont pas spécifiés ici car les spécifications pour ces protocoles existent déjà dans les Rec. UIT-T X.511 | ISO/CEI 9594-3 et Rec. UIT-T X.519 | ISO/CEI 9594-5.

NOTE – D'autres protocoles d'accès à l'annuaire sont LDAP (LDAPv2 1777, 2555 et 2587 des RFC) ou l'accès WEB (RFC 2585).

La Figure 1 ci-après fournit un aperçu général de l'architecture d'une infrastructure PKI avec un certain nombre d'exemples de services.

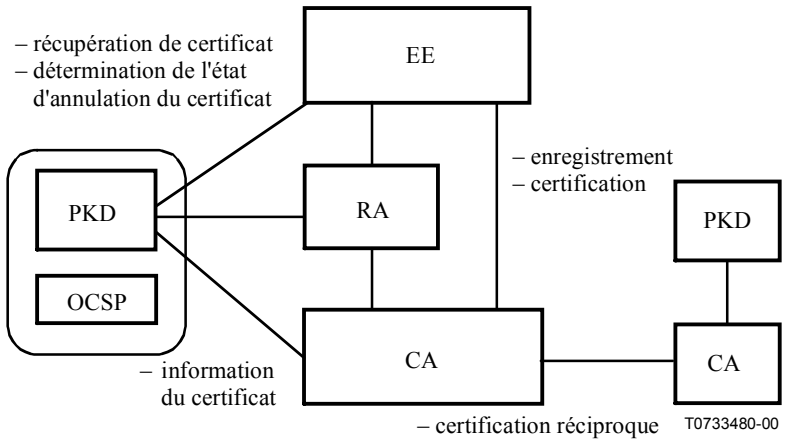

Figure 1 – Aperçu général des services de gestion de certificats

5.1.1 Enregistrement

La véracité d'une infrastructure de clé publique s'appuie sur l'identification et l'enregistrement corrects des entités.

Pour toutes les entités, une autorité RA (qui peut être également une autorité de certification) doit vérifier l'identité des entités finales par des moyens appropriés et doit attribuer un nom unique non ambigu à chaque entité finale à l'intérieur de son domaine. La politique de certification détermine la nature des moyens à utiliser pour effectuer cette identification (par exemple des documents d'identité) et la question de savoir si l'entité finale doit être présente en personne. Une dénomination des noms d'entité finale peut également être nécessaire. Les entités finales qui sont des composants techniques sont enregistrées conformément à la politique de sécurité de l'application, par exemple la gestion de réseau. Il convient que les applications dans lesquelles la différence entre entités finales humaines et non humaines est importante établissent clairement cette distinction dans le certificat. Par exemple, un "nom" pourrait être "dispositif de type X", numéro "Y" situé dans "Z", la distinction pourrait être établie conformément à la politique ou une extension peut indiquer la différence.

Un formulaire d'enregistrement peut être utilisé pour recueillir les données pertinentes, par exemple le nom, l'unité d'entreprise, l'adresse d'entreprise et l'adresse de livraison du matériel de codage.

EXEMPLE: un scénario dans une entreprise peut consister en ce que chaque employé doit être présent en personne au bureau du personnel approprié et montrer sa carte d'identité en cours de validité. Le responsable du personnel en fonction atteste de l'identité et envoie un formulaire d'enregistrement signé à l'autorité de certification.

5.1.2 Certification de clé publique

L'autorité de certification est responsable du processus de certification, du lien d'un nom d'entité ou d'un pseudonyme avec les clés publiques. Un format de certificat est décrit dans la Rec. UIT-T X.509 | ISO/CEI 9594-8.

La présente Recommandation | Norme internationale exige que la preuve de la possession de la clé privée correspondant à la clé publique incluse dans le certificat soit apportée au cours du service de certification. En fonction de la politique, une autorité de certification peut garantir d'autres propriétés de la clé publique de l'entité finale, en incluant par exemple l'un ou les deux services décrits aux § 5.3.2 et 5.3.3.

5.1.3 Annulation de certificats

5.1.3.1 Généralités

Un souci majeur avec les certificats de clés publiques est qu'ils peuvent devoir être *annulés* avant le moment prévu de leur expiration à cause de circonstances diverses. L'annulation peut être effectuée par l'autorité CA émettrice ou par une autre autorité en fonction de la politique.

L'annulation est obligatoire si la clé privée a été altérée. D'autres raisons d'annulation peuvent être un changement de nom, la fin d'une période d'embauche, etc.

Il convient que la politique de certification établisse toutes les raisons d'annulation. On peut trouver un certain nombre de ces raisons dans la Rec. UIT-T X.509 | ISO/CEI 9594-8. Il convient que la politique définisse qui peut faire les demandes d'annulation et, aussi, si un jeton spécifique peut être utilisé pour identifier des entités autorisées à annuler des certificats tels que les mots de passe d'annulation à usage unique.

5.1.3.2 Méthodes d'annulation

On peut utiliser deux méthodes pour annuler des certificats:

1) Emission d'une liste d'annulation de certificats (CRL)

   Une liste CRL est une liste émise périodiquement et signée par l'autorité de certification et elle identifie tous les certificats qui ont été annulés.

   L'annulation entraîne une saisie immédiate dans la liste CRL, qui inclut également le moment de l'annulation du certificat concerné. En outre, la politique de l'autorité de certification peut exiger de retirer le certificat de l'annuaire des clés publiques PKD.

   NOTE – Selon la politique, il peut être indiqué de conserver le certificat afin de vérifier la validité des signatures apposées avant la date d'annulation (par exemple, lorsque le motif d'annulation n'altère pas la clé).

   Outre la publication périodique, la liste CRL mise à jour peut être publiée immédiatement.

   L'ISO/CEI 11770-1 indique deux horodatages d'annulation différents:

   – le moment de l'altération connue ou suspectée; et
   – le moment auquel l'autorité de certification a été avisée par l'entité de l'altération.

   En fonction de la politique, la saisie du certificat dans la liste CRL peut être effacée lorsque le certificat expire (comparer avec le § 5.1.3.3, Détermination de l'état d'annulation de certificat).

2) Le stockage du statut du certificat dans une base de données interne de confiance de l'autorité de certification et l'offre aux entités d'informations en ligne sur le statut des certificats (comparer avec le § 5.1.3.3. Détermination de l'état d'annulation de certificat).

Ces deux méthodes peuvent être combinées.

Si une clé est altérée, la procédure normale est d'annuler le certificat correspondant, de réinitialiser l'entité finale, de créer une nouvelle paire de clés publique/privée et de certifier la nouvelle clé publique avec les attributs précédents.

En fonction de la politique, un certificat peut:

– être annulé de manière permanente;
– être suspendu. La saisie dans la liste CRL inclut un fanion qui indique l'état "en suspens" ("on hold").

La politique de l'autorité de certification doit spécifier la signification de l'état "en suspens" relativement au niveau de confiance qu'il représente pour une entité et la manière dont il convient qu'une entité traite cette situation. Par exemple, un certificat pourrait être suspendu à la suite d'une demande d'annulation non autorisée. Certaines politiques ne permettent pas du tout l'état "en suspens".

EXEMPLE: une politique peut établir ce qui suit: lorsqu'une vérification est effectuée, elle doit être rejetée tant que le certificat est suspendu. Lorsqu'une preuve est vérifiable à l'aide d'un certificat suspendu, le résultat de la vérification doit être négatif si un résultat est immédiatement nécessaire mais il peut également être interprété comme une validité conditionnelle. Si la suspension est suivie d'une annulation, la preuve devient non valide et la date d'annulation doit être la date du début de la suspension (et non la date de la fin de la suspension).

5.1.3.3 Détermination de l'état d'annulation de certificat

Ce service permet à une entité de déterminer si un certificat est annulé. On peut le faire suivant différentes méthodes qui correspondent aux méthodes d'annulation telles que décrites au § 5.1.3.2:

1) Méthode 1: vérification de l'annuaire PKD et de la liste CRL.
2) Méthode 2: demande en ligne du statut auprès d'un tiers TTP en qui on a confiance pour ce besoin. La réponse du tiers TTP doit être acheminée d'une manière authentique à l'entité.

5.1.3.4 Annulation d'un certificat d'une autorité CA

Un scénario spécial est constitué par l'altération de la clé privée d'une autorité de certification. Si cela se produit:

– le certificat de la clé publique correspondant à la clé altérée de l'autorité CA doit être annulé.

  La confiance en les certificats calculés avec la clé altérée de l'autorité CA n'est plus assurée par la signature de l'autorité CA qui est contenue dans ces certificats. Il est toutefois possible de garantir la validité de ces certificats par d'autres moyens (par exemple, lorsque les certificats sont stockés d'une manière digne de confiance par un TTP qui assure un service OCSP). Dans tous les cas, les entités doivent obtenir un nouveau certificat et une nouvelle clé de l'autorité CA non altérée pour des signatures ultérieures.

En fonction de la politique, les certificats calculés avec la clé altérée sont annulés, ce qui constitue un moyen d'informer les entités finales de la nécessité d'obtenir de nouveaux certificats.

Dans un certain nombre de situations dépendant de la politique de certification et de l'architecture du système, il convient de créer de nouvelles paires de clés destinées aux entités.

Il est important de remarquer que dans un tel scénario, toute signature qui a été émise avant que ne se produise l'altération et qui est sécurisée par des moyens supplémentaires (par exemple horodatée par une autorité d'horodatage dont le certificat est toujours valide) peut être toujours considérée valide en fonction de la politique, alors que toute signature qui a été émise après le moment de l'altération déterminée ou qui n'a pas été sécurisée par des moyens supplémentaires sera considérée non valide;

– si la clé publique correspondant à la clé privée altérée de l'autorité CA est certifiée de manière réciproque avec d'autres autorités CA, un message d'alerte doit être envoyé à ces autorités de certification. Ce message d'alerte leur notifie d'annuler le certificat des autorités de certification qui a été l'objet de certification réciproque.

5.1.4 Mise à jour de certificat

Dans le cas où un certificat expire, il peut être mis à jour en émettant un nouveau certificat pour la clé publique de l'entité qui était déjà contenue dans l'ancien certificat.

On ne doit pas utiliser cette méthode:

– si la paire de clés de l'entité est altérée; ou

– si l'état de cryptographie indique que l'algorithme de clé publique en rapport avec les paramètres de la paire de clés ne peut pas garantir la sécurité des signatures qui ont été créées avec cette paire de clés pour la période de validité du nouveau certificat; ou

– si le nouveau certificat présente des différences substantielles en termes de politique, extensions ou attributs par rapport à l'ancien certificat.

La politique de certification d'une autorité CA peut spécifier qu'une procédure d'enregistrement simplifiée est acceptable pour l'émission d'un nouveau certificat.

EXEMPLE: si l'ancien certificat n'est pas annulé, on peut accepter ce fait comme suffisant pour émettre un nouveau certificat.

La modification d'attributs non critiques dans un certificat pendant sa période de validité, tels que le nom ou l'affiliation (par exemple par un déplacement à un autre département) peut également conduire à la prescription pour une recertification des attributs modifiés, avec les mêmes clés qu'auparavant. Néanmoins, le certificat précédent doit être annulé dans ce cas.

5.1.5 Mise à jour de clés

Une nouvelle paire de clés est créée soit par l'entité elle-même, soit par le tiers TTP et un certificat est émis pour la clé publique de cette nouvelle paire.

Cette méthode doit être choisie dans le cas d'expiration d'un certificat si la mise à jour du certificat n'est pas acceptable pour l'une des raisons fournies au § 5.1.4. Elle peut également être utilisée dans d'autres situations en fonction de la politique de l'autorité de certification.

La politique de certification de l'autorité CA peut spécifier qu'une procédure d'enregistrement simplifiée est acceptable pour l'émission d'un certificat destiné à une clé mise à jour.

5.2 Services de gestion de clés

Des descriptions générales des services de gestion de clés sont fournies dans l'ISO/CEI 11770 (dans toutes les parties).

Le présent paragraphe contient seulement une description des services de gestion de clés qui peuvent être offerts comme faisant partie des services liés au cycle de vie de certificats (comparer avec le § 5.1). La spécification détaillée pour le flux de messages en ligne de ces services dans l'article 7 et la spécification ASN.1 des structures de données dans l'article 8 couvrent les services de gestion de clés tant qu'ils aboutissent à des messages en ligne entre une autorité de certification (CA), une autorité d'enregistrement (RA) ou une entité finale (EE).

5.2.1 Création de clés

Dans le contexte des signatures numériques, des tiers TTP peuvent créer des paires de clés privées/publiques, si les entités ne le font pas elles-mêmes. Bien que ce service puisse être offert par des tiers TTP indépendants, on admet en plus que ce service est fourni par une autorité CA ou une autorité RA en réponse à une demande de certification ou par une entité finale avant une demande de certification.

5.2.2 Distribution de clés

5.2.2.1 Distribution de clés privées

Pour la description du service "distribution de clé", on peut distinguer différents modes de transmission de clé (en ligne ou en mode non connecté) et le composant créateur de clé (tiers TTP ou entité). Dans le cas de la création centralisée de clés, le tiers TTP est responsable de la transmission sécurisée du certificat de clé privée et de clé publique de l'entité. De plus, il doit être garanti qu'une clé privée d'entité est envoyée de manière confidentielle. Il peut en être ainsi par chiffrement de la clé par une clé de transport spéciale (symétrique) connue seulement du tiers TTP et de l'entité correspondante. En variante, la clé privée peut être transmise à l'aide d'outils matériels sécurisés appropriés tels que les cartes à puce par exemple. La transmission de la clé privée n'est pas nécessaire si l'entité est à même de créer sa propre paire de clés asymétrique. Dans ce cas, le tiers TTP doit juste exécuter un certain nombre de vérifications de plausibilité (par exemple si l'entité peut signer un message avec une clé privée correspondant à la clé publique) afin de certifier la clé publique de l'entité et de rendre ce certificat disponible.

5.2.2.2 Distribution de clés publiques

Les clés publiques doivent être disponibles aux entités d'une manière qui garantisse leur authenticité. Dans le cas de clés publiques certifiées, la distribution de clés est réalisée par la distribution du certificat, l'authenticité étant garantie par la signature de l'autorité de certification qui a créé le certificat.

Dans le cas d'une clé de CA de confiance directe, on doit utiliser d'autres moyens de distribution sécurisée. Si des clés privées d'une entité sont distribuées à une entité en utilisant un jeton matériel sécurisé, ce jeton peut également être utilisé pour fournir la clé de l'autorité CA. Dans d'autres cas, un processus supplémentaire est nécessaire. Pour des méthodes à cet effet, consulter l'ISO/CEI 11770-3, § 8.1 Distribution de clés publiques sans tiers de confiance "*Public key distribution without a trusted third party*".

5.2.3 Personnalisation

Le stockage de clés privées et de données supplémentaires peut être obtenu en utilisant un jeton physique. Dans cette situation, la personnalisation d'un jeton doit être prise en charge par l'autorité CA, par l'autorité RA ou par les entités finales. Par exemple, la personnalisation de cartes à puce peut inclure des procédures de mise en place (par exemple la création de système de fichiers), la sélection d'un numéro PIN (Numéro d'identification personnel), au hasard ou un mot de passe, ainsi que la livraison et le stockage de toutes les données pertinentes dans une carte à puce.

5.3 Autres services

5.3.1 Certification réciproque

La certification réciproque est un service offert afin de permettre la vérification de signatures, provenant d'entités finales munies de certificats d'une autorité de certification donnée, par des entités finales munies de certificats d'une autre autorité de certification. Par exemple, une autorité CA1 émet un certificat pour une autorité CA2 appartenant à une autre infrastructure PKI avec l'effet que les entités qui font confiance à CA1 peuvent vérifier les certificats d'entités appartenant à l'autre infrastructure PKI par le biais d'un chemin de certification incluant ce nouveau certificat.

Une spécification détaillée pour le flux de messages en ligne de ce service est fournie dans l'article 7, la spécification ASN.1 des structures de données nécessaires à ces messages étant fournie dans l'article 8.

5.3.2 Validation de paramètres de domaine

La validation de paramètres de domaine est la validation d'un ensemble proposé de paramètres de domaine afin de garantir que chaque paramètre de l'ensemble satisfait à tous les attributs qui sont revendiqués pour ce paramètre.

EXEMPLES:

a) on peut exiger qu'un paramètre valide soit un nombre premier: pour cette validation, un test de nombre premier (peut-être probabiliste) est effectué pour s'assurer que le nombre premier proclamé est bien un nombre premier;

b) on peut exiger qu'un paramètre valide soit dans une certaine relation arithmétique avec un certain autre paramètre ou certains autres paramètres: pour cette validation, la relation arithmétique est soumise à l'essai afin de garantir qu'elle tient;

c) un cas de faiblesse spécifique (par exemple, sur une liste d'exclusion) peut être vérifié pour garantir qu'il ne s'applique pas à l'ensemble en question; ou

d) on peut exiger qu'un paramètre soit créé en utilisant un germe dans une fonction de hachage à germe canonique: pour cette validation, le germe est injecté en donnée d'entrée dans la fonction de hachage à germe canonique afin de garantir qu'elle crée effectivement le paramètre.

Il convient que le créateur d'un ensemble de paramètres de domaine garantisse qu'ils passent avec succès la validation de paramètres de domaine. La question de savoir s'il est nécessaire que quelqu'un d'autre effectue une validation de paramètres de domaine dépend de la relation de confiance entre le créateur et l'entité. Si un ensemble de paramètres de domaine non valables est utilisé, des résultats imprévisibles peuvent être obtenus, y compris la perte de la sécurité visée. Comme les paramètres de domaine sont typiquement publics, il vaut mieux que la validation puisse être effectuée dans un mode non connecté (c'est-à-dire, sans nécessiter que le créateur de paramètres de domaine réponde à des requêtes) et c'est classiquement le cas.

Généralement, une autorité de certification crée et valide un ensemble de paramètres de domaine qui peuvent alors faire l'objet d'une confiance implicite de la part de tous les membres de l'infrastructure PKI.

5.3.3 Validation de clé publique

La validation de clé publique est la validation d'une clé publique proclamée afin de garantir qu'elle est conforme aux prescriptions arithmétiques relatives à cette clé, c'est-à-dire que la clé publique proclamée est plausible. La validation de clé publique admet implicitement que tous les paramètres de domaine ont été validés précédemment.

EXEMPLES:

a) on peut exiger qu'un paramètre valide soit dans une gamme spécifique de valeurs: pour cette validation, le paramètre proclamé est soumis à l'essai afin de garantir qu'il se situe dans la gamme correcte;

b) on peut exiger qu'un paramètre valide présente un ordre spécifique (typiquement un ordre premier élevé): pour cette validation, le paramètre proclamé est soumis à l'essai afin de garantir qu'il présente l'ordre correct; ou

c) on peut exiger qu'un paramètre valide soit dans une relation arithmétique spécifique avec un certain autre paramètre ou certains autres paramètres: pour cette validation, la relation arithmétique est soumise à l'essai.

Si une clé publique non valide est utilisée, des résultats imprévisibles peuvent être obtenus, y compris la perte de la sécurité visée concernant le propriétaire de la clé privée associée, les destinataires de documents signés avec cette clé ou les deux. Un tiers de confiance, tel qu'une autorité de certification, peut effectuer une validation de clé publique afin d'assurer toutes les entités de son domaine.

5.3.4 Validation de certificat

Si une entité finale qui veut s'appuyer sur une signature numérique d'une autre entité finale n'est pas à même de vérifier le certificat correspond, elle peut demander à un tiers TTP de le faire.

La validation de certificat concerne la validité d'un certificat unique. Un certificat unique peut être fourni dans la demande ou il peut être fourni et suivi d'une séquence de certificats (sans nécessairement former un chemin de certification).

Deux conditions doivent être satisfaites pour soumettre à l'essai la validité d'un certificat:

a) un certificat unique ne peut être valide que par rapport à une politique de certification. Il est donc nécessaire de connaître la politique de certification qui s'applique. La politique de certification définit, entre autres, une règle: comment construire un chemin de certification valide. La tâche est donc de déterminer s'il est possible de construire un chemin de certification. Si les certificats fournis ne sont pas suffisants, le service peut essayer de rassembler lui-même les certificats manquants. Il peut y avoir plusieurs manières pour identifier la politique de certification mais il doit exister quelque pointeur

pointant sur la politique correcte. A cet effet, on peut utiliser un identificateur d'objet (OID, *object identifier*) ou une adresse Web (URL). Afin de s'assurer que le pointeur est correct, il convient d'ajouter une valeur de code de hachage de la politique. Une forme simple et dégénérée de politique est un certificat autosigné unique. Dans ce cas, il est nécessaire de pointer sur ce certificat autosigné en fournissant le nom de l'autorité de certification et le numéro de série du certificat.

b) il est également important de connaître la date à laquelle il convient d'effectuer l'essai de validité. Ceci est particulièrement important pour s'assurer que le certificat n'a pas été annulé avant cette date. En fonction de la politique de certification, la validité non seulement du certificat lui-même, mais aussi celle de tous les certificats dans le conduit de certificat doivent être vérifiés de manière ponctuelle à des moments pouvant différer pour chacun d'entre eux et dépendre du moment de la signature. La connaissance de la date de la signature est donc d'une importance particulière.

En guise de conclusion, les paramètres d'entrée pour ce service sont:

- un identificateur du certificat devant être soumis à l'essai de validité, suivi de zéro ou davantage de certificats;
- une ou plusieurs listes CRL (listes d'annulation de certificats);
- un identificateur de la politique de certification par rapport à laquelle le certificat doit être soumis à l'essai, c'est-à-dire un pointeur tel qu'un identificateur OID ou une adresse Web suivi d'un hachage de la politique de sécurité, ou bien l'identificateur d'un certificat autosigné (dans le cas dégénéré);
- le moment dans le temps auquel les essais doivent être effectués.

Il convient d'énumérer le paramètre de sortie.

5.3.5 Service d'archives

Sachant que des signatures numériques peuvent être utilisées pour signer des documents qui doivent être valides pendant longtemps, il doit être possible de déterminer la validité d'une signature même bien longtemps après l'expiration du certificat correspondant. A cet effet, des dispositions spéciales allant au-delà des procédures normales des autorités de certification doivent être prises car la période de validité du certificat représente l'intervalle de temps pendant lequel l'autorité de certification garantit qu'il conserve des informations sur le statut du certificat. La politique d'une infrastructure PKI conforme doit définir ces dispositions. L'utilisation de services d'horodatage ou celle des services d'archives constituent des dispositions possibles.

Les services d'archives peuvent être importants pour résoudre les conflits relatifs aux signatures numériques. Par exemple, des tiers TTP spécialisés peuvent archiver des certificats, des listes CRL et d'autres données car les services d'annuaires habituels ne peuvent contenir que des certificats et listes CRL qui n'ont pas expiré. Afin que cela puisse être réalisé, l'historique de la durée de vie des certificats et listes CRL peut également être enregistré de manière que l'on puisse reconstruire le cadre de leur durée de vie. La politique de certification peut exiger que l'autorité de certification elle-même archive tous les certificats et listes CRL qui ont été émis un jour.

6 Profil minimal de certificat et de liste CRL

6.1 Profil minimal de certificat

Les certificats doivent être conformes au format X.509 tel que spécifié dans la Rec. UIT-T X.509 | ISO/CEI 9594-8. Les entités qui vérifient les certificats doivent être à même de traiter ce format.

Aucun profil complet de certificat ne sera donné ici, mais les champs de certificat suivants doivent être implémentés pour des raisons de sécurité ou d'interopérabilité (les noms des champs de certificat se rapportent à la Rec. UIT-T X.509 | ISO/CEI 9594-8):

a) la version doit au moins être la version v3;

b) les extensions suivantes doivent être utilisées par les autorités de certification et les applications doivent être capables de les traiter:

- contraintes de base;
- utilisation de clés;
- identificateur de clé d'autorité (exception possible: certificats autosignés pour les clés de CA de confiance directe);
- politiques de certification;

c) les applications doivent pouvoir traiter l'extension en variante du nom du titulaire. Les autorités de certification doivent utiliser cette extension si le champ titulaire d'un certificat est vide;

d) les applications doivent pouvoir traiter les contraintes de nom, les contraintes de politique et les extensions d'utilisation de clé étendues. L'extension des contraintes de nom ne doit être utilisée que dans les certificats d'autorité de certification.

6.2 Profil minimal de liste CRL

Les listes CRL doivent être conformes au format X.509 tel que spécifié dans la Rec. UIT-T X.509 | ISO/CEI 9594-8. Les entités qui utilisent des listes CRL doivent être à même de traiter ce format.

Aucun profil complet de liste CRL ne sera donné ici mais les champs suivants doivent être implémentés pour des raisons de sécurité ou d'interopérabilité (les noms des champs de certificat se rapportent à la Rec. UIT-T X.509 | ISO/CEI 9594-8):

a) la version des listes CRL doit au moins être la version v2 telle que définie dans la Rec. UIT-T X.509 | ISO/CEI 9594-8;

b) les listes CRL doivent contenir le champ next update (prochaine mise à jour), l'extension du numéro de CRL et le champ keyIdentifier de l'extension de l'identificateur de clé de l'autorité. Les applications doivent pouvoir traiter ces champs.

7 Messages de gestion de certificats

Les messages sont définis dans le présent article pour tous les services pertinents de gestion de certificats. Les structures de données sont spécifiées conformément aux documents RFC, RFC 2510 et RFC 2511. Les messages sont spécifiés en détail dans l'article 8 en utilisant l'ASN.1. La syntaxe de l'ASN.1 est définie dans les Rec. UIT-T X.680 à X.683 | ISO/CEI 8824, parties 1 à 4, les règles de codage sont décrites dans la Rec. UIT-T X.690 | ISO/CEI 8825-1.

NOTE 1 – Il convient de noter que les protocoles en ligne ne constituent pas la seule façon d'implémenter ces services. Pour tous les services, il existe des méthodes en mode non connecté pour obtenir le même résultat et la présente Recommandation | Norme internationale oblige pas à utiliser des protocoles en ligne. Par exemple, lorsqu'on utilise des jetons matériels, on peut obtenir un grand nombre de services en tant que partie de la livraison par le jeton physique.

NOTE 2 – Le code ASN.1 est équivalent à celui des documents RFC susmentionnés bien que la syntaxe semble partiellement différente.

Fondamentalement, chaque type d'échange d'informations de tiers TTP consiste en un message de demande envoyé par l'initiateur et en un message de réponse renvoyé par le répondeur, comme le montre la Figure 2. En cas de problème, la réponse peut être remplacée par un message d'erreur.

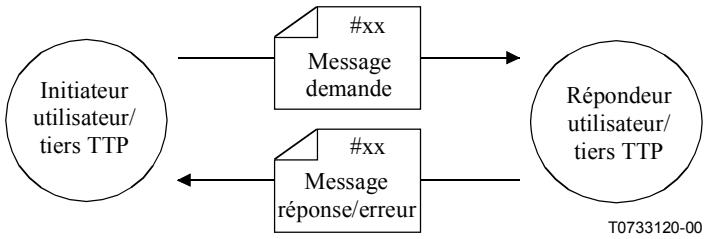

Figure 2 – Echange d'informations

La Figure 3 montre le flux d'informations de base dans un environnement de tiers TTP et toutes les entités impliquées.

On peut distinguer les classes suivantes de types d'échange d'informations de tiers TTP:

– types d'échange d'informations déclenchés par un tiers TTP en direction de tiers TTP: classe "TT";

– types d'échange d'informations déclenchés par un tiers TTP en direction des entités finales et vice versa: classe "TU";

– types d'échange d'informations déclenchés par une entité finale en direction d'autres entités finales: classe "UU".

La classe "UU" de types d'échange d'informations ne s'inscrit pas dans le domaine d'application de la présente Recommandation | Norme internationale.

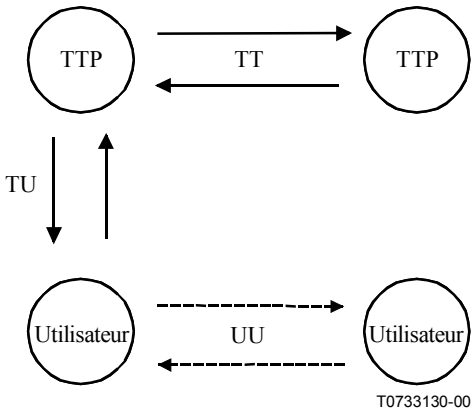

Figure 3 – Flux d'informations

7.1 Aperçu général des services et messages de gestion de certificats

Il convient de noter que ce ne sont pas tous les services de gestion d'infrastructure PKI qui aboutissent à la création d'un message de PKI.

Les services comprenant des entités et autorités de certification décrites ci-dessous peuvent en plus comprendre une autorité RA.

Les noms de message imprimés en caractères **gras** se rapportent aux définitions fournies dans le § 8.1.2.

7.1.1 Initialisation

7.1.1.1 Initialisation de CA

Avant d'émettre un certificat quelconque, une autorité CA nouvellement établie (qui prévoit d'émettre des listes CRL) doit produire des listes d'annulation initiales, c'est-à-dire des versions "vides" de chaque liste CRL qui doit être produite périodiquement.

Si une autorité CA nouvellement créée sert d'autorité CA de confiance directe pour certaines entités finales, elle doit produire au moins un "autocertificat" qui est un certificat pour la clé publique des autorités CA signé avec la propre clé de l'autorité CA. Il peut être nécessaire de disposer de différents certificats autosignés comprenant les contraintes d'attribution de nom afin de prendre en charge des applications différentes. Afin de rendre l'autocertificat de l'autorité CA utilisable par des entités finales qui n'acquièrent pas l'autocertificat par des moyens "hors bande", il convient que l'autorité de certification produise également un code de hachage pour chaque certificat autosigné. Les entités finales qui acquièrent, de manière sûre, ce code de hachage et un identificateur de l'autocertificat correspondant par le biais de moyens "hors bande" peuvent alors vérifier cet autocertificat et, donc, les autres attributs qu'il contient.

Du point de vue des protocoles de gestion d'infrastructure PKI, l'initialisation d'une autorité CA, qui n'est pas une autorité CA de confiance directe, est la même que l'initialisation d'une entité finale. La seule différence est que l'autorité CA doit aussi produire une liste d'annulation initiale.

7.1.1.2 Initialisation de CA à CA

Avant que des entités d'une autorité CA donnée puissent utiliser les clés publiques d'une entité certifiée par une autre autorité CA, la confiance entre ces autorités CA doit être établie. Ceci peut être réalisé en utilisant un certificat réciproque entre ces autorités CA ou au moyen d'un chemin de certification de confiance. Cette fonction peut être prise en charge par des mécanismes d'échange qui sont différents de ceux utilisés pour des interactions d'entités avec l'autorité de certification (par exemple l'échange physique entre des gestionnaires de CA) et, ainsi, il n'est pas nécessaire qu'elle soit prise en charge par des protocoles d'échange en ligne.

La certification réciproque implique l'utilisation de trois messages:

- message de demande de certification réciproque (**CrossCertReq**);
- message de réponse de certification réciproque (**CrossCertRep**);
- message de confirmation de l'infrastructure PKI (**PKIConfirm**).

L'autorité CA du demandeur est l'autorité de certification qui devient titulaire du certificat réciproque; l'autorité CA du répondeur devient l'émettrice du certificat réciproque.

7.1.1.3 Initialisation d'entité

Il s'agit du processus par lequel une entité finale se fait d'abord connaître d'une autorité CA ou RA, avant que l'autorité CA n'émette de certificat ou certificats pour cette entité finale. Le résultat final de ce processus (lorsqu'il réussit) est qu'une autorité CA émet un certificat pour une clé publique d'une entité finale et renvoie ce certificat à l'entité finale ou poste ce certificat dans un dépôt public. Ce processus peut impliquer, et en général implique, plusieurs "étapes", y compris une possible initialisation de l'équipement de l'entité finale. Par exemple, l'équipement de l'entité finale doit être initialisé en toute sécurité avec la clé publique d'une autorité CA afin d'être utilisé pour la validation de chemins de certification. De plus, une entité finale a généralement besoin d'être initialisée avec sa propre paire de clés ou ses propres paires de clés.

L'identité de l'entité doit être vérifiée par l'autorité RA. Cette procédure d'enregistrement peut comprendre des communications en mode non connecté et même des communications non électroniques (par exemple l'utilisation du service postal) et n'a pas besoin d'être prise en charge par de quelconques protocoles d'échange en ligne. Divers schémas d'authentification pour l'enregistrement initial d'une entité finale sont décrits au § 7.2.1.

Avant d'utiliser des services offerts par un tiers TTP, une entité peut avoir besoin d'obtenir des informations sur les fonctions prises en charge par le tiers TTP en même temps que les clés nécessaires pour communiquer en toute sécurité avec le tiers TTP et la clé publique du tiers TTP nécessaire à la vérification des certificats.

Le dialogue GenMsg de l'infrastructure PKI peut être utilisé pour demander et fournir cette information de PKI. Dans ce cas, la demande doit être le message **GenMsg**, la réponse doit être le message **GenRep** et le message d'erreur doit être le message **ErrorMsg**. Ces messages sont protégés à l'aide d'un code MAC fondé sur l'information secrète partagée (c'est-à-dire **PasswordBasedMAC**) ou tout autre moyen authentifié (si l'entité finale possède un certificat existant):

- demande d'information de PKI (message **GenMsg**);
- réponse d'information de PKI (message **GenRep**).

Cette paire de messages est conçue pour des demandes générales d'informations de PKI et peut aussi être utilisée après l'initialisation. Elle peut être également utilisée par d'autres tiers TTP pour obtenir des informations sur le statut actuel d'une autorité CA. L'autorité CA doit répondre à la demande en fournissant les informations demandées ou en envoyant une erreur si une partie des informations ne peut pas être fournie.

Lorsqu'elle effectue le processus d'enregistrement/certification, une entité peut indiquer que la demande est sa première demande de certification en utilisant le message suivant au lieu des messages **CertReq** et **CertRep** comme il est décrit plus loin:

- demande d'enregistrement/certification initial ou initiale (**InitReq**);
- réponse d'enregistrement/certification initial ou initiale (**InitRep**)

7.1.2 Création de clés

La paire de clés publique/privée utilisée dans un système asymétrique peut être créée soit par une entité soit par un tiers TTP. Chaque fois qu'une paire de clés est créée, elle doit l'être d'une manière sur laquelle on peut se fier pour une création correcte conformément à l'algorithme utilisé et pour la conservation de l'intégrité des deux clés et la confidentialité de la clé privée.

a) Création par une entité

Si l'entité crée la paire de clés publique/privée, la clé publique est transmise au tiers TTP en tant que partie du processus d'enregistrement/certification. Ceci est inclus dans un champ du modèle de certificat contenu dans le message **CertReqMessage**.

b) Création par un tiers TTP

Si le tiers TTP crée la paire de clés publique/privée, cette paire est transmise à l'entité en tant que partie du processus d'enregistrement/certification (se reporter au § 7.1.1.3). La clé privée est transmise par la livraison physique de l'environnement PSE ou à l'intérieur du champ clé privée de **CertKeyPair** dans **CertRep**. La clé publique est transmise à l'intérieur de son certificat; le champ certificat de **CertKeyPair** dans **CertRep** est contenu dans **CertRepContent**.

c) Mise à jour de clés

Lorsqu'une paire de clés doit expirer, l'entité finale pertinente peut demander une mise à jour de clés – c'est-à-dire, elle peut demander que le tiers TTP émette un nouveau certificat pour une nouvelle paire de clés. La demande est présentée en utilisant un message de demande de mise à jour de clés. Si l'entité finale possède déjà une paire de clés de signature (avec le certificat de vérification correspondant), ce message est généralement protégé par la signature numérique de l'entité. Le tiers TTP retourne le nouveau certificat (si la demande a réussi) dans un message de réponse à une mise à jour de clés:

- demande de certification pour une clé mise à jour (**KeyUpdReq**);
- réponse de certification pour une mise à jour de clé mise à jour (**KeyUpdRep**).

La nouvelle paire de clés peut être créée par l'entité ou par le tiers TTP. Dans le second cas, la clé est créée et transportée dans ces messages en tant que partie du processus d'enregistrement/ certification comme décrit ci-dessus.

NOTE – Dans le cas où l'entité souhaite allonger la vie d'une clé certifiée existante, il est alors nécessaire de mettre à jour le certificat.

d) Annonce de mise à jour de clés de CA

Les clés de CA (comme toutes les autres clés) ont une durée de vie finie et doivent être mises à jour à intervalles réguliers.

Des certificats spéciaux (voir le § 7.2.3) sont émis par l'autorité de certification pour aider les entités finales existantes qui détiennent l'ancien certificat de CA autosigné à effectuer une transition sécurisée vers un nouveau certificat de CA autosigné, et pour aider les nouvelles entités finales qui détiendront le nouveau certificat de CA autosigné, à acquérir l'ancien en toute sécurité aux fins de vérification de données existantes.

Lorsqu'une autorité de certification met à jour sa propre paire de clés publique/privée, le message suivant peut être utilisé pour annoncer l'événement aux entités:

- annonce de la mise à jour de clés de CA (**CAKeyUpdAnn**).

7.1.3 Certification de clé

Une entité finale initialisée peut demander un certificat à tout moment (en tant que partie d'une procédure de mise à jour ou pour tout autre besoin). Cette demande est faite à l'aide du message de demande de certification. Si l'entité finale possède déjà une paire de clés de signature (avec un certificat de vérification correspondant), ce message est généralement protégé par la signature numérique de l'entité. L'autorité de certification renvoie le nouveau certificat (si la demande a réussi) dans un message **CertRep**.

Une entité s'enregistre auprès d'un tiers TTP et demande un certificat en utilisant les messages suivants:

- demande d'enregistrement/certification (**CertReq**);
- réponse d'enregistrement/certification (**CertRep**).

Une demande/réponse de certification d'enregistrement initiale (**InitReq/InitRep**) peut être utilisée facultativement.

Lorsque l'entité identifie le besoin d'une nouvelle paire de clés et le certificat associé, des demandes de certification/enregistrement peuvent être acheminées facultativement en utilisant les messages suivants:

- demande d'enregistrement/certification dans le cas d'une mise à jour de clés (**KeyUpdReq**);
- réponse d'enregistrement/certification dans le cas d'une mise à jour de clés (**KeyUpdRep**).

L'entité inclut dans la demande toute information dont elle exige qu'elle soit placée dans le certificat, y compris son nom ou un pseudonyme. Certaines informations contenues dans la réponse, telles qu'une date d'expiration, peuvent différer de celles demandées par l'entité.

Afin d'achever le processus d'enregistrement/certification, d'autres échanges peuvent être nécessaires pour attester de la validité du nom de l'entité et des autres informations devant être incluses dans le certificat ou induites par le certificat.

Si l'entité crée la paire de clés, la clé publique peut être transmise au tiers TTP dans la demande. Si la paire de clés est créée par le tiers TTP, la clé privée cryptée peut être transmise à l'entité dans la réponse.

Si la paire de clés est créée par l'entité et si c'est pour la création et la vérification de signatures numériques, une signature peut être incluse dans la demande afin de prouver que la clé privée lui appartient.

Si exigé, l'entité peut envoyer le message suivant pour démontrer l'acceptation du certificat:

- confirmation (**PKIConfirm**).

7.1.4 Annonce de certificat

Après avoir achevé le processus d'enregistrement/certification, y compris si nécessaire la réception d'un message **PKIConfirm**, le tiers TTP met le certificat à la disposition des autres parties. Ceci peut être réalisé au moyen d'une gamme de mécanismes comprenant:

a) le placement du certificat dans un dépôt tel qu'un service d'annuaire ou un serveur Web;

b) la transmission du certificat à d'autres entités dont on sait qu'elles exigent le certificat (par exemple des membres d'une communauté notoire d'utilisateurs).

En variante, l'entité peut rendre son certificat disponible à d'autres parties en l'incluant avec d'éventuelles données protégées.

Un tiers TTP peut envoyer des certificats à des entités en utilisant le message suivant:

– annonce de certificat **CertAnn**.

Les échanges nécessaires pour placer des certificats dans un dépôt dépendent de la forme de dépôt utilisée.

Si une partie exigeant un certificat ne l'a pas déjà reçu par un message d'annonce de certificat ou en même temps que les données protégées, le certificat peut être obtenu d'un dépôt tel qu'un service d'annuaire ou un serveur Web.

Les échanges nécessaires pour obtenir un certificat d'un dépôt dépendent de la forme de dépôt utilisée.

7.1.5 Distribution de clés

Pour la distribution de clé, il est nécessaire de traiter la clé privée différemment de la clé publique.

Clé privée: pour la clé privée, il est nécessaire de faire une distinction en fonction de qui crée la paire de clés. Si une entité crée sa propre paire de clés, aucune distribution de la clé privée n'est nécessaire. Si la paire de clés n'est pas créée par l'entité (c'est-à-dire qu'elle est créée par le tiers TTP), il est nécessaire de distribuer la clé privée à l'entité de manière sécurisée.

NOTE – Une méthode de distribution de clés privées peut être trouvée au § 8.4.4. Pour des informations générales sur la distribution de clés privées, consulter l'ISO/CEI 11770 (dans toutes ses parties), qui présente différentes méthodes de distribution de clés.

Clé publique: les clés publiques des entités sont distribuées en même temps que les certificats qui leur sont attribués. Les messages associés sont ceux concernant l'annonce et la récupération de certificats.

7.1.6 Annulation de clé/certificat

Il convient qu'une annulation de certificat se produise chaque fois qu'il y a suspicion de dysfonctionnement concernant les clés (la clé privée est altérée) ou si des changements concernant le certificat associé à la clé publique se produisent (changement de nom, fin de la période d'embauche dans une organisation, etc.). Si une entité finale (ou toute autre entité autorisée) souhaite annuler la clé publique et le certificat qui lui est associé, une demande **RevReq** est envoyée au tiers TTP, qui répond par un message **RevRep**.

Si un tiers TTP a annulé ou est sur le point d'annuler le certificat demandé, il peut émettre une annonce de cet événement (**RevAnn**). Cela devrait être spécialement le cas si la demande d'annulation n'a pas été émise par l'entité.

En association avec les activités d'annulation, les saisies dans la liste CRL changent. Afin de signaler ces changements, le tiers TTP émet une annonce de la nouvelle liste CRL (**CRLAnn**). Il convient d'envoyer cette information à toutes les entités concernées.

7.2 Hypothèses et restrictions pour un certain nombre de services

7.2.1 Enregistrement/certification initial ou initiale

Il existe plusieurs schémas que l'on peut utiliser pour réaliser l'enregistrement initial et la certification initiale d'entités finales. Aucune méthode unique ne convient pour toutes les situations du fait de la gamme de politiques qu'une autorité de certification peut mettre en œuvre et du fait de la différence des types d'entités finales qui peuvent exister.

On peut cependant classifier les schémas d'enregistrement initial/certification initiale qui sont pris en charge par la présente Recommandation | Norme internationale. Cette classification est effectuée dans les situations où l'entité finale en question n'a pas eu de contact préalable avec l'infrastructure PKI. Lorsque l'entité finale possède déjà des clés certifiées, un certain nombre de simplifications/variantes sont possibles.

On peut utiliser les critères suivants afin de distinguer divers schémas:

a) Authentification auprès de l'autorité RA

Pendant l'enregistrement/certification initial ou initiale, l'entité finale traite avec l'autorité RA.

Elle peut être indépendante ou combinée avec l'autorité CA. On peut avoir besoin de l'autorité RA pour authentifier le demandeur à un certain moment avant l'émission d'un certificat. Cette authentification peut survenir avant l'échange d'un quelconque message avec l'autorité RA ou après l'échange d'un certain nombre de messages avec l'autorité RA. Cela peut être fait par contact direct (par exemple, la présentation d'une carte d'identité, d'un permis de conduire, d'un passeport) ou par des moyens hors bande (par exemple, procédure de rappel, reconnaissance vocale).

b) Informations initiales distribuées par des moyens hors bande

Il est nécessaire de distribuer des informations publiques et/ou secrètes par des moyens hors bande avant de pouvoir utiliser un quelconque protocole. Les informations publiques peuvent être constituées du nom de l'autorité RA (ou de l'autorité CA), de son emplacement, de la valeur et de l'algorithme de sa clé publique ou d'un pointeur vers un certificat autosigné et un hachage de celui-ci. Les informations secrètes peuvent être constituées d'un identificateur de l'entité finale et d'une clé d'authentification initiale. Cette clé d'authentification initiale peut alors être utilisée pour protéger les messages de PKI pertinents.

Les informations publiques peuvent être librement partagées par tous les utilisateurs d'une communauté et l'assurance de leur intégrité suffit. A titre d'exemple, ces informations peuvent être sur un CD-ROM.

L'entité finale ou l'organisation qu'elle représente reçoit les informations secrètes en tant que résultat de l'authentification initiale (voir ci-dessus). Il est nécessaire de protéger la partie secrète de ces informations contre la divulgation.

c) Authentification de l'origine du message émis par une entité finale

Les messages en ligne produits par l'entité finale qui exige un certificat et envoyés par cette entité finale à l'autorité RA ou CA peuvent être authentifiés ou non. Une procédure d'enregistrement initial/certification initiale peut être sécurisée lorsque les messages provenant de l'entité finale sont authentifiés par le biais de certains moyens hors bande (par exemple, une visite consécutive).

d) Emplacement de la création de clés

Il existe trois possibilités pour l'emplacement de la création de paires de clés: l'entité finale, une autorité RA et une autorité CA.

NOTE – Un service indépendant pour la création de clés n'en est pas pour autant interdit. – La paire de clés effective peut avoir été créée ailleurs et acheminée jusqu'à l'entité finale, l'autorité RA ou l'autorité CA à l'aide d'un protocole de demande/réponse de création de clés (propriétaire ou normalisé), protocole qui ne s'inscrit pas dans le domaine d'application de la présente Recommandation | Norme internationale.

e) Confirmation d'une certification réussie

A la suite de la création d'un certificat initial destiné à une entité finale, on peut obtenir une assurance supplémentaire lorsque l'entité finale confirme de façon explicite la réception réussie du message contenant le certificat (ou indiquant la création du certificat). Evidemment, ce message de confirmation doit être protégé (en s'appuyant sur une clé d'authentification initiale ou sur d'autres moyens). Il en découle deux possibilités supplémentaires: confirmation ou non-confirmation.

Les critères précédents assurent un grand nombre de schémas d'enregistrement/certification initial ou initiale. Quelques-uns sont décrits ici.

7.2.1.1 Schéma centralisé

En termes des critères précédents, dans ce schéma:

- une authentification initiale a lieu auprès de l'autorité RA avant un quelconque échange;
- des informations secrètes sont distribuées par des moyens hors bande par l'autorité RA pendant l'authentification initiale;
- aucun message en ligne provenant de l'entité finale n'est exigé;
- la création de paires de clés a lieu au niveau de l'autorité CA ou RA chargée de la spécification;
- aucun message de confirmation n'est exigé.

En termes de flux de messages, l'environnement PSE doit être retourné à l'entité finale puisque la création de paires de clés a lieu au niveau de l'autorité CA ou RA chargée de la certification. Le seul message exigé est envoyé de l'autorité CA à l'entité finale et contient la totalité de l'environnement PSE pour l'entité finale. Les informations secrètes distribuées au préalable par des moyens hors bande permettent à l'entité finale d'authentifier le message reçu et de déchiffrer les valeurs codées éventuelles. Une variante consiste en la livraison physique d'un jeton de PSE.

7.2.1.2 Schéma d'authentification préalable

En termes des critères précédents, dans ce schéma:

- l'authentification initiale a lieu auprès de l'autorité RA avant un quelconque échange;
- des informations secrètes sont distribuées par des moyens hors bande par l'autorité RA pendant l'authentification initiale;
- les messages provenant de l'entité finale sont authentifiés en utilisant ces informations secrètes;
- la création de paires de clés a lieu au niveau de l'entité finale;
- un message de confirmation est exigé.

En termes de flux de messages, ce schéma se présente comme suit:

Entité finale		RA/CA
	Authentification de l'entité finale auprès de l'autorité RA	
	Distribution hors bande de la clé d'authentification initiale (IAK) et de la valeur de référence (RA/CA → EE) par l'autorité RA	
Création de clés Création de la demande de certification Protection de la demande par la clé IAK		
	→→demande de certification→→	
		Vérification de la demande Traitement de la demande Création du certificat Création de la réponse
	←←réponse de certification←←	
Traitement de la réponse Création de la confirmation		
	→→message de confirmation→→	Vérification de la confirmation

(Lorsque la vérification de la confirmation échoue, l'autorité RA/CA doit annuler le certificat nouvellement émis, si nécessaire.)

7.2.1.3 Schéma de post-authentification

En termes des critères précédents, dans ce schéma:

- l'authentification initiale a lieu auprès de l'autorité RA après les échanges;
- les informations publiques sur l'autorité RA sont distribuées par des moyens hors bande (c'est-à-dire qu'aucune information secrète n'est distribuée);
- les messages provenant de l'entité finale ne sont pas authentifiés;
- la création de paires de clés a lieu au niveau de l'entité finale;
- un message de confirmation n'est pas exigé.

En termes de flux de messages, ce schéma se présente comme suit:

Entité finale		RA/CA
	Distribution hors bande d'informations publiques (RA/CA → EE)	
Création de clés Création de la demande de certification Aucune protection de la demande		
	→→demande de certification→→	
		Vérification de la demande Traitement de la demande Création de la réponse
	←←réponse de certification←←	
Traitement de la réponse Vérification de l'origine de la réponse Extraction du numéro d'enregistrement		
	Présentation du numéro d'enregistrement et authentification de l'entité finale auprès de l'autorité RA à l'aide de moyens hors bande	
		Création du certificat

L'entité finale peut à tout moment remplir une demande dès qu'elle détient les informations correctes sur l'autorité RA (sa clé publique, en particulier). L'autorité RA attribue un numéro d'enregistrement unique pour toute demande de certification non authentifiée reçue. Etant donné que la réponse de certification émise par l'autorité d'enregistrement est signée, l'entité finale peut en vérifier l'origine et en extraire le numéro d'enregistrement. L'entité finale peut alors présenter ce numéro à l'autorité RA et s'authentifier à l'aide de certains moyens hors bande.

Les avantages de ce schéma sont les suivants:

- aucun contact avec l'autorité d'enregistrement n'est exigé au préalable;
- on ne manipule jamais d'information secrète.

7.2.2 Preuve de la possession (POP, *proof of possession*) d'une clé privée

Afin de prévenir certaines attaques et de permettre à une autorité CA/RA de vérifier correctement la validité du lien entre une entité finale et une paire de clés de signature, les opérations de gestion d'infrastructure PKI qui sont spécifiées ici permettent à une entité finale de prouver qu'elle est en possession de (c'est-à-dire qu'elle est à même d'utiliser) la clé de signature privée correspondant à la clé de signature publique pour laquelle un certificat est exigé. Une autorité CA/RA donnée est libre de choisir la manière d'exiger la preuve POP [par exemple, par des moyens procéduraux hors bande en fonction des messages de protocole de gestion de certificats ("dans la bande")] dans ses échanges de certification (c'est-à-dire qu'il peut s'agir d'une question de politique). Toutefois les autorités CA/RA conformes doivent forcer la preuve POP par quelque moyen.

La présente Recommandation | Norme internationale assure de manière explicite les cas où une entité fournit la preuve pertinente à une autorité d'enregistrement et que celle-ci atteste en retour à l'autorité de certification que la preuve exigée a bien été reçue et validée. Par exemple, une entité finale qui souhaite obtenir un certificat pour une clé de signature pourrait envoyer la signature appropriée à l'autorité d'enregistrement qui indique alors simplement à l'autorité CA pertinente que l'entité finale a fourni la preuve exigée. Naturellement, une telle situation peut être interdite par un certain nombre de politiques (par exemple, les autorités de certification peuvent être les seules entités autorisées à vérifier la preuve POP pendant la certification).

Etant donné que la paire de clés de l'entité est créée pour des besoins de signatures numériques, l'entité finale peut signer une valeur convenable afin de prouver la possession de la clé privée. La valeur doit être choisie de manière à empêcher que des assaillants utilisent avec succès d'anciennes signatures de l'entité finale.

7.2.3 Mise à jour de clés de CA de confiance directe

La base de la procédure décrite ici est que l'autorité de certification protège sa nouvelle clé publique en utilisant sa précédente clé privée et vice versa. Ainsi, lorsqu'une autorité CA met à jour sa paire de clés, elle doit créer deux valeurs d'attribut de **cACertificate** supplémentaires si les certificats sont disponibles en utilisant l'annuaire X.500 (pour en obtenir quatre au total: **OldWithOld**; **OldWithNew**; **NewWithOld**; **NewWithNew**).

Lorsqu'une autorité CA change sa paire de clés, les entités qui ont acquis l'ancienne clé publique de l'autorité CA par des moyens "hors bande" sont les plus affectées. Ce sont ces entités finales qui ont besoin d'accéder à la nouvelle clé publique de l'autorité CA avec l'ancienne clé privée de l'autorité CA. Toutefois, elles ne l'exigent que pendant une période limitée (jusqu'à ce qu'elles acquièrent la nouvelle clé publique de l'autorité CA par un mécanisme "hors bande"). Ceci est généralement facilement réalisé lorsque les certificats de ces entités finales expirent.

La structure de données utilisée pour protéger les anciennes et les nouvelles clés publiques d'autorité CA est un certificat standard (qui peut également contenir des extensions). Aucune nouvelle structure de données n'est nécessaire.

NOTE 1 – Le présent schéma n'utilise pas d'extension de X.509 v3 ou v4 car il est conçu pour fonctionner même avec les certificats de la version 1. La présence de l'extension **KeyIdentifier** pourrait être utilisée pour des améliorations d'efficacité.

NOTE 2 – Bien que le schéma puisse être généralisé pour couvrir les cas où l'autorité de certification met plus d'une fois à jour sa paire de clés pendant la période de validité de l'un des certificats de ses entités finales, cette généralisation semble d'une valeur douteuse. L'absence de cette généralisation signifie simplement que la période de validité d'une paire de clés de CA doit être plus grande que celle de n'importe quel certificat émis par cette autorité CA en utilisant cette paire de clés.

NOTE 3 – Ce schéma force les entités finales à acquérir la nouvelle clé publique de l'autorité de certification au moment de l'expiration du dernier certificat qui leur appartenait et qui était signé avec l'ancienne clé privée de l'autorité de certification (par le biais de moyens "hors bande"). Ceci n'est pas nécessairement exigé de services de mise à jour de certificat ou de clés qui se produisent à d'autres moments (en fonction de l'équipement de l'entité finale).

Actions de CA

Pour changer la clé de l'autorité de certification, l'autorité CA effectue les actions suivantes:

1) création d'une nouvelle paire de clés;

2) création d'un certificat contenant l'ancienne clé publique de l'autorité CA signée avec la nouvelle clé privée (le certificat "l'ancienne avec la nouvelle").

   Les entités finales dont l'environnement PSE contient la nouvelle clé publique de l'autorité CA peuvent ainsi vérifier l'ancienne clé de CA et les certificats signés avec cette clé;

3) création d'un certificat contenant la nouvelle clé publique de l'autorité CA signée avec l'ancienne clé privée (le certificat "la nouvelle avec l'ancienne").

   Les entités finales dont l'environnement PSE contient l'ancienne clé publique de l'autorité CA peuvent ainsi vérifier la nouvelle clé de CA et les certificats signés avec cette clé;

4) création d'un certificat contenant la nouvelle clé publique de l'autorité CA signée avec la nouvelle clé privée (le certificat "la nouvelle avec la nouvelle").

   Les entités finales dont l'environnement PSE contient l'ancienne clé publique de l'autorité CA peuvent ainsi importer ce nouveau certificat autosigné;

5) publication de ces nouveaux certificats par le biais de l'annuaire ou tout autre moyen (en utilisant peut-être un message CAKeyUpdAnn);

6) exportation de la nouvelle clé publique de l'autorité de certification de manière à ce que les entités finales puissent l'acquérir en utilisant un mécanisme "hors bande" (si exigé).

L'ancienne clé privée de l'autorité CA n'est plus alors exigée. Toutefois, l'ancienne clé publique de l'autorité de certification sera utilisée pendant quelque temps encore. Le moment où cette clé publique ne sera plus exigée (pour d'autres raisons que la non-répudiation) est le moment où toutes les entités finales de cette autorité de certification auront acquis en toute sécurité la nouvelle clé publique de l'autorité de certification.

Le certificat "l'ancienne avec la nouvelle" doit avoir une période de validité commençant au moment de la création de l'ancienne paire de clés et se terminant à la date d'expiration de l'ancienne clé publique.

Le certificat "la nouvelle avec l'ancienne" doit avoir une période de validité commençant au moment de la création de la nouvelle paire de clés et se terminant au moment où toutes les entités finales de cette autorité de certification sont en possession en toute sécurité de la nouvelle clé publique de l'autorité de certification (au plus tard, la date d'expiration de l'ancienne clé publique).

Le certificat "la nouvelle avec la nouvelle" doit avoir une période de validité commençant au moment de la création de la nouvelle paire de clés et se terminant au moment où l'autorité de certification procédera à la mise à jour suivante de sa paire de clés.

La procédure de publication de ces certificats est utile pour des vérifications d'annulation également car l'autorité de certification peut avoir signé la liste CRL en utilisant une clé privée plus récente que celle qui est contenue dans l'environnement PSE de l'entité.

7.2.4 Certification réciproque

Le paragraphe suivant décrit un schéma possible de certification réciproque. D'autres schémas peuvent toutefois être utilisés en fonction des politiques des TTP rencontrées au cours de la certification réciproque. Voir l'Annexe A pour une discussion des scénarios possibles.

7.2.4.1 Schéma de demande-réponse à sens unique

Le schéma de certification réciproque est essentiellement un service à sens unique; c'est-à-dire que ce service, lorsqu'il est réussi, aboutit à la création d'un nouveau certificat réciproque unique. Si la prescription est que des certificats réciproques soient créés dans les "deux sens", chaque autorité de certification doit à son tour déclencher un service de certification réciproque (ou utiliser un autre schéma).

Ce schéma convient lorsque les deux autorités de certification en question peuvent déjà chacune vérifier les signatures de l'autre (elles ont certains points communs de confiance), ou lorsqu'il existe une vérification hors bande de l'origine de la demande de certification.

Description détaillée

La certification réciproque est déclenchée par l'autorité CA1. L'autorité CA1 identifie l'autorité CA (à savoir, CA2) qu'elle veut certifier de manière réciproque et l'équipement de l'autorité CA1 crée un code d'autorisation. L'autorité CA1 transmet ce code d'autorisation par des moyens hors bande à l'autorité CA2. L'autorité CA2 saisit le code d'autorisation afin de déclencher l'échange en ligne.

Le code d'autorisation est utilisé pour des besoins d'authentification et d'intégrité. Ceci est fait par la création d'une clé symétrique fondée sur le code d'autorisation et par l'utilisation de la clé symétrique pour créer des codes d'authentification de messages (MAC) sur tous les messages échangés.

L'autorité CA2 déclenche l'échange en créant un numéro aléatoire (numéro aléatoire de demandeur). Elle envoie ensuite à l'autorité CA1 le message de demande de certification réciproque (**CrossCertReq**). Un code MAC fondé sur le code d'autorisation protége les champs contenus dans ce message contre les modifications.

Lorsqu'elle reçoit le message de demande de certification réciproque, l'autorité CA1 vérifie la version du protocole, sauvegarde le numéro aléatoire du demandeur, crée son propre numéro aléatoire (numéro aléatoire du répondeur) et valide le code MAC. Elle crée ensuite (et archive, si elle le souhaite) un nouveau certificat de demandeur qui contient la clé publique de l'autorité CA2 et qui est signé avec la clé privée de signature de l'autorité CA1. L'autorité CA1 répond avec le message de réponse de certification réciproque (**CrossCertRes**). Un code MAC fondé sur le code d'autorisation protège les champs contenus dans ce message contre les modifications.

Lorsqu'elle reçoit le message de réponse de certification réciproque, l'autorité CA2 vérifie que son propre temps système est proche du temps système de l'autorité CA1, vérifie les numéros aléatoires reçus et valide le code MAC. L'autorité CA2 répond avec le message **PKIConfirm**. Un code MAC fondé sur le code d'autorisation protège les champs contenus dans ce message contre les modifications. L'autorité CA2 rédige le certificat de demandeur et l'envoie au Dépôt.

Lorsqu'elle reçoit le message **PKIConfirm**, l'autorité CA1 vérifie les nombres aléatoires et valide le code MAC.

NOTE 1 – Le message de demande de certification réciproque doit contenir une demande de certification "complète", c'est-à-dire que tous les champs (y compris, par exemple, une extension **BasicConstraints**) doivent être spécifiés par l'autorité CA2.

NOTE 2 – Il convient que le message de réponse de certification réciproque contienne le certificat de vérification de l'autorité CA1 – l'autorité CA2 doit alors vérifier ce certificat s'il est présent (par exemple, par le biais d'un mécanisme "hors bande").

8 Structures de données pour les messages de gestion de certificat

Le présent article contient des descriptions de structures de données qui sont exigées pour les messages de gestion de certificat. Les structures de données sont spécifiées conformément aux documents RFC, RFC 2510 et RFC 2511. La présente Recommandation | Norme internationale utilise la syntaxe ASN.1 et importe les définitions ASN.1 contenues dans la Rec. UIT-T X.509 | ISO/CEI 9594-8.

8.1 Message global

Tous les messages utilisés dans la présente Recommandation | Norme internationale pour des besoins de gestion de certificat utilisent la structure suivante:

```
PKIMessage ::= SEQUENCE {
    header          PKIHeader,
    body            PKIBody,
    protection      [0] PKIProtection   OPTIONAL,
    extraCerts      [1] SEQUENCE SIZE (1..MAX) OF Certificate OPTIONAL
}
```

L'en-tête de message de PKI ainsi que son corps et sa protection sont exposés en détail dans les paragraphes ci-dessous.

Le champ certificats supplémentaires **extraCerts** peut contenir des certificats qui peuvent être utiles au destinataire. Par exemple, un tiers TTP peut l'utiliser pour présenter une entité finale avec des certificats dont elle a besoin afin de vérifier son propre nouveau certificat (si le tiers TTP ayant émis le certificat de l'entité finale n'est pas une autorité CA de confiance directe pour cette entité finale). Il convient également de noter que ce champ ne contient pas nécessairement un chemin de certification – le destinataire peut devoir trier les certificats supplémentaires, en choisir ou sinon les traiter, afin de les utiliser.

8.1.1 En-tête de message de PKI

Tous les messages de tiers TTP exigent certaines informations d'en-tête pour l'adressage et l'identification de transactions. Certaines de ces informations peuvent être également présentes dans une enveloppe spécifique à chaque transport; toutefois, si le message est protégé, ces informations sont également protégées (c'est-à-dire qu'aucune hypothèse n'est faite sur le transport sécurisé).

La structure de données suivante est utilisée pour contenir ces informations:

```
PKIHeader ::= SEQUENCE {
                    pvno            INTEGER     { version2 (1) },
                    sender          GeneralName,
                    recipient       GeneralName,
                    messageTime     [0] GeneralizedTime         OPTIONAL,
                    protectionAlg   [1] AlgorithmIdentifier     OPTIONAL,
                    senderKID       [2] KeyIdentifier           OPTIONAL,
                    recipKID        [3] KeyIdentifier           OPTIONAL,
                    transactionID   [4] OCTET STRING            OPTIONAL,
                    senderNonce     [5] OCTET STRING            OPTIONAL,
                    recipNonce      [6] OCTET STRING            OPTIONAL,
                    freeText        [7] PKIFreeText             OPTIONAL,
                    generalInfo     [8] SEQUENCE SIZE (1..MAX) OF InfoTypeAndValue OPTIONAL
                                -- on peut l'utiliser pour acheminer des informations spécifiques à chaque contexte
}
```

Le champ **pvno** est fixe pour la présente version de la norme TTP. Le champ **sender** contient le nom de l'expéditeur du message **PKIMessage**. Il convient que l'on puisse utiliser ce nom (en association avec le champ **senderKID**, s'il est fourni) pour vérifier la protection figurant sur le message.

Si l'entité émettrice ne sait rien sur l'expéditeur (par exemple, DN, nom de mél., adresse IP, etc.), le champ "sender" doit contenir une valeur "NULL"; c'est-à-dire que la SEQUENCE OF des noms distinctifs correspondants a une longueur égale à zéro. Dans un tel cas, le champ **senderKID** doit contenir un identificateur (c'est-à-dire un numéro de référence) qui indique au destinataire les informations secrètes partagées qui sont appropriées à être utilisées pour vérifier le message.

Le champ **recipient** (destinataire) contient le nom du destinataire du message **PKIMessage**. Il convient que l'on puisse utiliser ce nom (en association avec le champ **recipKID**, s'il est fourni) pour vérifier la protection figurant sur le message. Le champ **messageTime** contient l'heure à laquelle l'expéditeur a créé le message (il est utilisé lorsque l'expéditeur pense que le transport est "convenable"; c'est-à-dire que le temps aura encore un sens au moment de la réception). Ceci peut être utile pour permettre aux entités finales de corriger leur heure locale pour la rendre cohérente avec l'heure d'un système central. Le champ **protectionAlg** spécifie l'algorithme utilisé pour protéger le message. Si aucun bit de protection n'est fourni (le champ **PKIProtection** est facultatif), ce champ doit être omis; si des bits de protection sont fournis, ce champ doit être fourni.

Les champs **senderKID** et **recipKID** sont utilisés pour indiquer quelles clés ont été utilisées pour protéger le message. Le champ **transactionID** à l'intérieur de l'en-tête de message est utile au destinataire d'un message de réponse qui peut le corréler avec une demande émise précédemment. Par exemple, dans le cas d'une autorité RA, il peut y avoir plusieurs demandes en attente à un moment donné.

Les champs **senderNonce** et **recipNonce** protègent le message **PKIMessage** contre les attaques par redéfilement. Les mots créés pour la circonstance (nonces) sont utilisés pour fournir une protection contre le redéfilement, **senderNonce** est inséré par le créateur de ce message; **recipNonce** est un nonce précédemment inséré dans un message apparenté par le destinataire prévu de ce message. Le champ **freeText** peut être utilisé pour envoyer au destinataire un message compréhensible par l'homme. La structure de ce champ est:

```
PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String
        -- texte codé comme une chaîne UTF-8 String (NOTE – Il convient que chaque UTF8String
        -- comprenne une étiquette de langue RFC 1766 afin d'indiquer la langue
        -- du texte contenu)
```

8.1.2 Corps de message de PKI

```
PKIBody ::= CHOICE {      -- message-specific body elements
    ir          [0]  CertReqMessages,         -- Demande d'initialisation (InitReq)
    ip          [1]  CertRepMessage,          -- Réponse d'initialisation (InitRep)
    cr          [2]  CertReqMessages,         -- Demande de certification (CertReq)
    cp          [3]  CertRepMessage,          -- Réponse de certification (CertRep)
    p10cr       [4]  CertificationRequest,    -- Demande de certif. PKCS #10 (variante de formulaire pour CertRep)
                     -- demande de certification comme défini dans PKCS #10 pour compatibilité
    kur         [7]  CertReqMessages,         -- Demande de mise à jour de clés (KeyUpdReq)
    kup         [8]  CertRepMessage,          -- Réponse de mise à jour de clés (KeyUpdRep)
    rr          [11] RevReqContent,           -- Demande d'annulation (RevReq)
    rp          [12] RevRepContent,           -- Réponse d'annulation (RevRep)
    ccr         [13] CertReqMessages,         -- Demande de certif. réciproque (CrossCertReq)
    ccp         [14] CertRepMessage,          -- Réponse de certif. réciproque (CrossCertRep)
    ckuann      [15] CAKeyUpdAnnContent,      -- Annonce de mise à jour de clés de CA (CAKeyUpdAnn)
    cann        [16] CertAnnContent,          -- Annonce de certification (CertAnn)
    rann        [17] RevAnnContent,           -- Annonce d'annulation (RevAnn)
    crlann      [18] CRLAnnContent,           -- Annonce de liste CRL (CRLAnn)
    conf        [19] PKIConfirmContent,       -- Confirmation de PKI (PKIConfirm)
    nested      [20] NestedMessageContent,    -- Message imbriqué
    genm        [21] GenMsgContent,           -- Message général (GenMsg)
    genp        [22] GenRepContent,           -- Réponse générale (GenRep)
    error       [23] ErrorMsgContent,         -- Message d'erreur (ErrorMsg)
}
```

Les types spécifiques sont décrits aux § 8.3 et 8.4 ci-dessous.

8.1.3 Protection de message de PKI

Certains messages PKI sont protégés pour leur intégrité. (Il convient de noter que, si un algorithme asymétrique a été utilisé pour protéger un message et que le composant public correspondant a déjà été certifié, l'origine du message peut également être authentifiée. D'autre part, si le composant public n'est pas certifié, l'origine du message ne peut pas être authentifiée automatiquement mais peut l'être par le biais de moyens hors bande.)

Si une protection est appliquée, on utilise la structure suivante:

```
PKIProtection ::= BIT STRING
```

La donnée d'entrée du calcul de PKIProtection est le codage selon DER de la structure de données suivante:

```
ProtectedPart ::= SEQUENCE {
    header      PKIHeader,
    body        PKIBody
}
```

Il peut exister des cas où la chaîne BIT STRING de PKIProtection n'est pas utilisée volontairement pour protéger un message (c'est-à-dire que ce champ FACULTATIF est omis) car une autre protection, externe à la présente Recommandation | Norme internationale, sera appliquée à la place. Un tel choix est autorisé de façon explicite dans la présente Recommandation | Norme internationale.

Toutefois, il convient de remarquer que plusieurs de ces mécanismes externes exigent que l'entité finale possède déjà un certificat de clé publique ou un nom distinctif unique, ou d'autres de ces informations liées à l'infrastructure. Ainsi, ils peuvent ne pas être appropriés pour un enregistrement initial ou pour tout autre processus ayant des caractéristiques de "bootstrap" ou amorçage. Dans ces cas, il peut être nécessaire d'utiliser le paramètre PKIProtection.

En fonction des circonstances, les bits de PKIProtection peuvent contenir un code d'authentification de message (MAC) ou une signature. Les cas suivants peuvent se produire:

Cas 1: Informations secrètes partagées

Dans le cas où l'expéditeur et le destinataire partagent des informations secrètes (établies par des moyens hors bande ou à partir d'un précédent service de gestion de tiers TTP). Le champ PKIProtection contient une valeur de code MAC.

Le champ protectionAlg est le suivant:

```
PasswordBasedMac ::= OBJECT IDENTIFIER     --{1 2 840 113533 7 66 13}
PBMParameter ::= SEQUENCE {
    salt            OCTET STRING,
    owf             AlgorithmIdentifier,
    -- AlgorithmIdentifier  pour une fonction à une voie (SHA-1 recommandée)
    iterationCount          INTEGER,
    -- nombre de fois que la fonction OWF est appliquée
    mac             AlgorithmIdentifier
    -- AlgorithmIdentifier de MAC (par exemple DES-MAC, Triple-DES-MAC comme dans PKCS #11,
}   -- ou HMAC comme dans RFC2104, RFC2202)
```

Dans le champ **protectionAlg** ci-dessus, la valeur salt est ajoutée à la donnée d'entrée secrète partagée. La fonction OWF est ensuite appliquée un nombre **iterationCount** de fois, la valeur secrète munie de salt étant la donnée d'entrée de la première itération et, pour chaque itération successive, la donnée d'entrée est réglée pour être la donnée en sortie de l'itération précédente. La donnée en sortie de l'itération finale (appelée "BASEKEY" pour commodité de référence, avec une taille de "H") est ce qui est utilisé pour former la clé symétrique. Si l'algorithme MAC exige une clé de K bits et que K <= H, les K bits de poids fort de BASEKEY sont utilisés. Si K > H, la totalité de BASEKEY est utilisée pour les H bits de poids fort de la clé, la fonction OWF("1" || BASEKEY) est utilisée pour les H bits de poids fort suivants de la clé, la fonction OWF("2" || BASEKEY) est utilisée pour les H bits de poids fort suivants de la clé, et ainsi de suite, jusqu'à ce que tous les K bits soient obtenus. [Ici "N" est l'octet ASCII qui code le nombre N et "||" représente la concaténation.]

Cas 2: Paires de clés de Diffie-Hellman

Lorsque l'expéditeur et le destinataire possèdent des certificats de Diffie-Hellman avec des paramètres de DH compatibles, l'entité finale doit créer une clé symétrique fondée sur la valeur de sa clé privée de DH et sur la clé publique de DH du destinataire du message afin de protéger le message. Le champ PKIProtection contient une valeur de code MAC codée avec la clé symétrique obtenue.

```
DHBasedMac ::= OBJECT IDENTIFIER --{1 2 840 113533 7 66 30}
  DHBMParameter ::= SEQUENCE {
    owf             AlgorithmIdentifier,
-- AlgorithmIdentifier pour une fonction à une voie (SHA-1 recommandée)
    mac             AlgorithmIdentifier
    -- AlgorithmIdentifier de MAC (par exemple DES-MAC, Triple-DES-MAC comme dans PKCS#11,
}   -- ou HMAC comme dans RFC2104, RFC2202)
```

Dans le champ protectionAlg ci-dessus, une fonction OWF est appliquée au résultat du calcul de Diffie-Hellman. La donnée en sortie de la fonction OWF (appelée "BASEKEY" pour commodité de référence, avec une taille de "H") est ce qui est utilisé pour former la clé symétrique. Si l'algorithme MAC exige une clé de K bits et que K <= H, les K bits de poids fort de BASEKEY sont utilisés. Si K > H, la totalité de BASEKEY est utilisée pour les H bits de poids fort de la clé, la fonction OWF("1" || BASEKEY) est utilisée pour les H bits de poids fort suivants de la clé, la fonction OWF("2" || BASEKEY) est utilisée pour les H bits de poids fort suivants de la clé, et ainsi de suite, jusqu'à ce que tous les K bits soient obtenus. [Ici "N" est l'octet ASCII qui code le nombre N et "||" représente la concaténation.]

Cas 3: Signature numérique

Lorsque l'expéditeur possède une paire de clés de signature, il peut simplement signer le message. Le champ **PKIProtection** contient la valeur de signature et le champ **protectionAlg** est un identificateur **AlgorithmIdentifier** pour une signature numérique (par exemple **md5WithRSAEncryption** ou **dsaWithSha-1**).

Cas 4: Protection multiple

Dans les cas où une entité finale envoie un message protégé à une autorité d'enregistrement, l'autorité RA peut transmettre ce message à une autorité de certification, en joignant sa propre protection (qui peut être un code MAC ou une signature, en fonction des informations et des certificats partagés entre l'autorité RA et l'autorité CA). Ceci est réalisé en imbriquant le message entier envoyé par l'entité finale dans un nouveau message de tiers TTP. La structure utilisée est la suivante.

```
NestedMessageContent ::= PKIMessage
```

8.2 Structures de données communes

Dans le présent paragraphe, on définit un certain nombre de structures de données qui sont utilisées dans plus d'un message de PKI.

8.2.1.1 Contenus du certificat demandés

Divers messages de gestion de tiers TTP exigent que l'expéditeur du message indique un certain nombre de champs dont la présence est exigée dans un certificat. La structure **CertTemplate** permet à une entité finale ou à une autorité d'enregistrement de spécifier autant qu'elle le souhaite à propos d'un certificat qu'elle exige. Le modèle **CertTemplate** est identique à un certificat mais tous ses champs sont facultatifs.

Il convient de noter que même si l'expéditeur spécifie complètement le contenu d'un certificat qu'il exige, une autorité de certification est libre de modifier des champs dans le certificat réellement émis.

Se reporter au § 8.3 pour la syntaxe de **CertTemplate**.

8.2.2 Valeurs codées

Lorsque des valeurs codées (restreintes dans la présente spécification à être soit des clés privées soit des certificats) sont envoyées dans les messages de PKI, on utilise la structure de données **EncryptedValue**.

Se reporter au § 8.6.1 pour la syntaxe de **EncryptedValue**.

L'utilisation de cette structure de données exige que le créateur et le destinataire prévu soient à même de respectivement chiffrer et déchiffrer. Généralement, cela signifie que l'expéditeur et le destinataire ont une clé secrète partagée ou sont capables d'en créer une.

Si le destinataire du message PKIMessage possède déjà une clé privée utilisable pour un déchiffrement, le champ **encSymmKey** peut contenir une clé de session codée en utilisant la clé publique du destinataire.

8.2.3 Code de statut et informations d'échec pour messages de PKI

Tous les messages de réponse comprennent certaines informations de statut. On définit les valeurs suivantes:

```
PKIStatus ::= INTEGER {
    granted                 (0),
    -- vous avez obtenu exactement ce que vous avez demandé
    grantedWithMods         (1),
    -- vous avez obtenu quelque chose qui ressemble à ce que vous avez demandé; le
    -- demandeur est responsable d'identifier les différences
    rejection               (2),
    -- vous ne l'obtenez pas, de plus amples informations se trouvent ailleurs dans le message
    waiting                 (3),
    -- le corps de la demande n'a pas encore été traité,
    -- attendez-vous à en savoir davantage plus tard
    revocationWarning       (4),
    -- ce message contient un avertissement indiquant qu'une annulation est
    -- imminente
    revocationNotification  (5),
    -- notification qu'une annulation s'est produite
    keyUpdateWarning        (6)
    -- mise à jour déjà effectuée pour le oldCertId spécifié dans
    -- le message de demande de mise à jour de clés
}
```

Les répondeurs peuvent utiliser la syntaxe suivante pour fournir davantage d'informations sur les cas d'échec.

```
PKIFailureInfo ::= BIT STRING {
-- car une demande peut échouer de plus d'une façon!
-- davantage de codes peuvent être ajoutés dans l'avenir si exigés/lorsque exigés.
    badAlg                  (0),
    -- Identificateur d'algorithme non reconnu ou non pris en charge
    badMessageCheck         (1),
    -- échec de la vérification d'intégrité (par exemple, la signature n'a pas été vérifiée)
    badRequest              (2),
    -- transaction non autorisée ou non prise en charge
```

```
    badTime                 (3),
    -- l'heure du message n'est pas suffisamment proche de l'heure système,
    -- tel que le définit la politique locale
    badCertId               (4),
    -- aucun certificat satisfaisant aux critères fournis n'a pu être trouvé
    badDataFormat           (5),
    -- les données soumises ont le mauvais format
    wrongAuthority          (6),
    -- l'autorité indiquée dans la demande est différente de
    -- celle qui crée le jeton de réponse
    incorrectData           (7),
    -- les données du demandeur sont incorrectes (utilisées pour des services de notarisation)
    missingTimeStamp        (8)
    -- lorsque l'horodatage manque alors qu'il devrait y être (par politique)
}


PKIStatusInfo ::= SEQUENCE {
    status          PKIStatus,
    statusString    PKIFreeText     OPTIONAL,
    failInfo        PKIFailureInfo  OPTIONAL
}
```

8.2.4 Identification de certificat

Afin d'identifier des certificats particuliers, on utilise la structure de données de **CertId**. Se reporter au § 8.3 pour la syntaxe de **CertId**.

8.2.5 Clé publique de CA de confiance directe "hors bande"

Chaque autorité CA de confiance directe doit pouvoir publier sa clé publique en vigueur par le biais de moyens "hors bande". Bien que de tels mécanismes ne s'inscrivent pas dans le domaine d'application de la présente Recommandation | Norme internationale, des structures de données pouvant prendre en charge ces mécanismes y sont définies.

On dispose généralement de deux méthodes: soit l'autorité CA publie directement son certificat autosigné; soit ces informations sont disponibles par le biais de l'annuaire (ou son équivalent) et l'autorité CA publie un code de hachage de cette valeur afin de permettre la vérification de son intégrité avant utilisation.

```
OOBCert ::= Certificate
```

Les champs dans ce certificat sont limités de la manière suivante:

- le certificat doit être autosigné (c'est-à-dire que la signature doit être vérifiable en utilisant le champ SubjectPublicKeyInfo);
- les champs subject (titulaire) et issuer (émetteur) doivent être identiques;
- si le champ subject est **NULL,** les deux extensions **subjectAltNames** et **issuerAltNames** doivent être présentes et avoir exactement la même valeur;
- les valeurs de toutes les autres extensions doivent convenir à un certificat autosigné (par exemple les identificateurs de clé pour le titulaire et l'émetteur doivent être les mêmes).

```
OOBCertHash ::= SEQUENCE {
    hashAlg     [0] AlgorithmIdentifier     OPTIONAL,
    certId      [1] CertId                  OPTIONAL,
    hashVal     BIT STRING
    -- hashVal est calculée sur le certificat
    -- autosigné avec l'identificateur certID.
}
```

L'objectif du code de hachage est que toute personne ayant reçu en toute sécurité le code de hachage (par des moyens hors bande) puisse vérifier un certificat autosigné pour cette autorité de certification.

8.2.6 Informations sur la publication

Les demandeurs peuvent indiquer leur souhait que l'infrastructure PKI publie un certificat en utilisant la structure **PKIPublicationInfo**. Se reporter au § 8.3 pour la syntaxe de **PKIPublicationInfo**.

8.2.7 Structures de preuve de possession

La preuve de possession d'une clé de signature privée est démontrée par l'utilisation de la structure **POPOSigningKey**. Se reporter au § 8.3 pour la syntaxe **POPOSigningKey**, mais il convient de noter que, dans la présente spécification, l'entrée POPOSigningKeyInput présente les stipulations sémantiques suivantes:

```
POPOSigningKeyInput ::= SEQUENCE {
        authInfo        CHOICE {
            sender              [0] GeneralName,
            -- provenant de l'en-tête PKIHeader [uniquement utilisé si une identité authentifiée
            -- a été établie pour l'émetteur (par exemple, un DN provenant
            -- d'un certificat précédemment émis et en cours de validité)]
            publicKeyMAC    [1] PKMACValue
            -- utilisé si aucun nom GeneralName authentifié n'existe actuellement
            -- pour l'émetteur; publicKeyMAC contient un code MAC fondé sur un mot de passe
            -- (utilisant l'identificateur AlgorithmIdentifier de protectionAlg provenant de l'en-tête PKIHeader)
            -- sur la valeur codée selon DER de publicKey
            },
        publicKey       SubjectPublicKeyInfo-- provenant de CertTemplate
        }
```

8.3 Structures de données spécifiques pour les messages de demande de certificat du type CertReq

Le présent paragraphe décrit le format de message de demande de certificat (CRMF, *certificate request message format*). Cette syntaxe est utilisée pour acheminer une demande de certificat à une autorité de certification (CA) [probablement par l'intermédiaire d'une autorité d'enregistrement (RA)] pour les besoins de la production de certificat X.509. La demande contient classiquement une clé publique et des informations d'enregistrement associées.

8.3.1 Aperçu général

La construction d'une demande de certification comprend les étapes suivantes:

a) une valeur CertRequest est construite. Cette valeur peut comprendre la clé publique, tout ou partie du nom de l'entité finale (EE), d'autres champs de certificat exigés et des informations de contrôle supplémentaires liées au processus d'enregistrement;

b) une valeur de preuve de possession (de la clé privée correspondant à la clé publique pour laquelle le certificat est demandé) peut être calculée dans la valeur de CertRequest;

c) des informations d'enregistrement supplémentaires peuvent être combinées avec la valeur de la preuve de possession et avec la structure CertRequest pour former un message CertReqMessage;

d) le message CertReqMessage est communiqué en toute sécurité à une autorité CA (comme il est spécifié au § 8.1.3).

8.3.2 Syntaxe de CertReqMessage

Un corps de message de demande de certificat est composé d'une demande de certificat, d'un champ facultatif de preuve de possession et d'un champ facultatif d'informations d'enregistrement.

```
CertReqMessages ::= SEQUENCE SIZE (1..MAX) OF CertReqMsg

CertReqMsg ::= SEQUENCE {
   certReq  CertRequest,
   pop      ProofOfPossession  OPTIONAL,
   regInfo  SEQUENCE SIZE(1..MAX) OF AttributeTypeAndValue OPTIONAL }
```

Le champ de preuve de possession est utilisé pour démontrer que l'entité devant être associée au certificat est effectivement en possession de la clé privée correspondante. Le champ peut être calculé avec le contenu du champ **certReq**, sa structure et son contenu variant avec le type d'algorithme de clé publique et avec le mode de service.

Il convient que le champ **regInfo** ne contienne que des informations supplémentaires liées au contexte de la demande de certification lorsque ces informations sont exigées pour satisfaire une demande de certification. Ces informations peuvent comprendre des informations de contact avec les abonnés, des informations de facturation ou d'autres informations auxiliaires utiles à la satisfaction de la demande de certification.

Il convient d'inclure dans le contenu de **certReq** les informations directement liées au contenu du certificat. Cependant, l'inclusion d'un contenu supplémentaire de **certReq** par des autorités d'enregistrement peut invalider le champ pop. Des données destinées au contenu du certificat peuvent donc être fournies dans **regInfo**.

8.3.3 Preuve de possession (POP)

Afin de prévenir certaines attaques et de permettre à une autorité CA/RA de vérifier correctement la validité du lien entre une entité finale et une paire de clés, les services de gestion d'infrastructure PKI qui sont spécifiés ici permettent à une entité finale de prouver qu'elle est en possession de (c'est-à-dire qu'elle est à même d'utiliser) la clé privée correspondant à la clé publique pour laquelle un certificat est exigé. Une autorité CA/RA donnée est libre de choisir la manière d'exiger la preuve POP (par exemple par des moyens procéduraux hors bande en fonction du message CRMF "dans la bande") dans ses échanges de certification (c'est-à-dire qu'il peut s'agir d'une question de politique). Toutefois, les autorités CA/RA conformes doivent exiger la preuve POP par quelque moyen.

La présente Recommandation | Norme internationale tient compte des cas où la preuve POP est validée par l'autorité de certification, par l'autorité d'enregistrement ou par les deux. Certaines politiques peuvent exiger que l'autorité CA vérifie la preuve POP pendant la certification, auquel cas l'autorité d'enregistrement doit transmettre les champs non altérés CertRequest et ProofOfPossession de l'entité finale à l'autorité de certification et, facultativement, peuvent également vérifier la preuve POP. Si l'autorité de certification n'est pas tenue par politique de vérifier la preuve POP, il convient que l'autorité d'enregistrement transmette sans altération la demande de l'entité finale et la preuve à l'autorité de certification comme ci-dessus. En cas d'impossibilité (par exemple parce que l'autorité d'enregistrement vérifie la preuve POP par une méthode hors bande), l'autorité d'enregistrement peut attester à l'autorité de certification que la preuve exigée a été validée. Si l'autorité de certification utilise une méthode hors bande pour vérifier la preuve POP (comme une livraison physique de clés privées créées par une autorité de certification), le champ ProofOfPossession n'est pas utilisé.

8.3.3.1 Syntaxe de la preuve de possession

```
ProofOfPossession ::= CHOICE {
   raVerified      [0] NULL,
   -- utilisé si l'autorité RA a déjà vérifié que le demandeur est en
   -- possession de la clé privée
   signature       [1] POPOSigningKey,
}

POPOSigningKey ::= SEQUENCE {
   poposkInput             [0] POPOSigningKeyInput OPTIONAL,
   algorithmIdentifier     AlgorithmIdentifier,
   signature               BIT STRING }
   -- La signature (utilisant "algorithmIdentifier") est sur la
   -- valeur de poposkInput codée selon DER.  NOTE – Si le modèle CertTemplate de certReq de
   -- CertReqMsg contient les valeurs de subject (titulaire) et de publicKey,
   -- le champ poposkInput doit être omis et la signature doit être
   -- calculée sur la valeur codée selon DER de certReq de CertReqMsg. Si
   -- le modèle CertTemplate de certReq de CertReqMsg ne contient pas les valeurs
   -- de clé publique et de titulaire, le champ poposkInput doit être présent et
   -- doit être signé. Cette stratégie garantit que la clé publique n'est
   -- pas présente à la fois dans les deux champs poposkInput et CertTemplate de certReq de CertReqMsg.

POPOSigningKeyInput ::= SEQUENCE {
   authInfo        CHOICE {
      sender       [0] GeneralName,
      -- utilisé uniquement si une identité authentifiée a été
      -- établie pour l'émetteur (par exemple, un DN provenant
      -- d'un certificat précédemment émis et en cours de validité)
      publicKeyMAC     PKMACValue },
      -- utilisé si aucun nom GeneralName authentifié n'existe actuellement
      -- pour l'émetteur; publicKeyMAC contient un code MAC fondé sur un mot de passe
      -- sur la valeur codée selon DER de publicKey
   publicKey       SubjectPublicKeyInfo }  -- from CertTemplate

PKMACValue ::= SEQUENCE {
   algId  AlgorithmIdentifier,
   -- la valeur d'algorithme doit être PasswordBasedMac
   -- {1 2 840 113533 7 66 13}
   -- la valeur de paramètre est PBMParameter
   value  BIT STRING }
```

8.3.3.2 Utilisation de code MAC fondé sur un mot de passe

L'algorithme suivant doit être utilisé lorsque **publicKeyMAC** est employé dans **POPOSigningKeyInput** afin de prouver l'authenticité de la demande.

```
PBMParameter ::= SEQUENCE {
    salt            OCTET STRING,
    owf             AlgorithmIdentifier,
    -- AlgorithmIdentifier pour une fonction à une voie (SHA-1 recommandée)
    iterationCount  INTEGER,
    -- nombre de fois que la fonction OWF est appliquée
    mac             AlgorithmIdentifier
    -- AlgorithmIdentifier de MAC (par exemple DES-MAC, Triple-DES-MAC [PKCS 11],
}   -- ou HMAC [RFC2104, RFC2202])
```

Le processus d'utilisation de **PBMParameter** pour calculer **publicKeyMAC** et ainsi authentifier l'origine de la demande de certification de clé publique comprend deux étapes. La première étape utilise des informations secrètes partagées pour créer une clé MAC. La seconde étape code selon MAC la clé publique en question en utilisant cette clé MAC pour produire une valeur authentifiée.

L'initialisation de la première étape d'algorithme admet implicitement l'existence d'un secret partagé distribué de manière confiante entre l'autorité CA/RA et l'entité finale. La valeur salt est ajoutée au secret partagé et la fonction à une voie (owf) est appliquée un nombre iterationCount de fois, le secret muni de salt étant la donnée d'entrée de la première itération et, pour chaque itération suivante, la donnée d'entrée est réglée pour être la donnée en sortie de l'itération précédente, ce qui donne une clé K.

Dans la seconde étape, la clé K et la clé publique sont les données d'entrée du HMAC comme il est exposé dans RFC 2104 pour fournir une valeur pour **publicKeyMAC** de la manière suivante:

publicKeyMAC = Hash(K XOR opad, Hash(K XOR ipad, public key))

où ipad et opad sont définis dans RFC 2104.

L'identificateur AlgorithmIdentifier pour owl doit être SHA-1{1 3 14 3 2 26} et il doit être HMAC-SHA-1 {1 3 6 1 5 5 8 1 2} pour mac.

8.3.4 Syntaxe de CertRequest

La syntaxe de CertRequest comprend un identificateur de demande, un modèle de contenu de certificat et une séquence facultative d'informations de contrôle.

```
CertRequest ::= SEQUENCE {
        certReqId     INTEGER,          -- ID pour demande et réponse en correspondance
        certTemplate  CertTemplate,     -- Champs sélectionnés du certificat à émettre
        controls      Controls OPTIONAL } -- Attributs affectant l'émission

CertTemplate ::= SEQUENCE {
    version         [0] Version                 OPTIONAL,
    serialNumber    [1] INTEGER                 OPTIONAL,
    signingAlg      [2] AlgorithmIdentifier     OPTIONAL,
    issuer          [3] Name                    OPTIONAL,
    validity        [4] OptionalValidity        OPTIONAL,
    subject         [5] Name                    OPTIONAL,
    publicKey       [6] SubjectPublicKeyInfo    OPTIONAL,
    issuerUID       [7] UniqueIdentifier        OPTIONAL,
    subjectUID      [8] UniqueIdentifier        OPTIONAL,
    xtensions       [9] Extensions              OPTIONAL }

OptionalValidity ::= SEQUENCE {
    notBefore   [0] Time OPTIONAL,
    notAfter    [1] Time OPTIONAL } --l'un au moins doit être présent

Time ::= CHOICE {
    utcTime     TCTime,
    generalTime eneralizedTime }
```

8.3.5 Syntaxe de Controls (des contrôles)

Le créateur d'une demande CertRequest peut inclure une ou plusieurs valeurs de contrôle relatives au traitement de la demande.

```
Controls ::= SEQUENCE SIZE(1..MAX) OF AttributeTypeAndValue
```

Les contrôles ci-après sont définis (il est admis que cette liste peut s'allonger avec le temps): **regToken; authenticator; pkiPublicationInfo; oldCertID; protocolEncrKey**.

8.3.5.1 Contrôle du jeton d'enregistrement (*registration token control*)

Un contrôle **regToken** contient des informations à usage unique (fondées soit sur une valeur secrète soit sur la connaissance) destinées à être utilisées par l'autorité de certification afin de vérifier l'identité du titulaire avant l'émission d'un certificat. Lorsqu'elle reçoit une demande de certification contenant une valeur de **regToken**, l'autorité CA réceptrice vérifie l'information afin de confirmer l'identité proclamée dans la demande de certification.

La valeur de **regToken** peut être créée par l'autorité de certification et être fournie hors bande à l'abonné, ou sinon elle peut être mise à la disposition à la fois de l'autorité de certification et de l'abonné. Il convient de mesurer la sécurité d'un échange hors bande à la hauteur du risque que l'autorité de certification accepte une valeur interceptée provenant de quelqu'un d'autre que l'abonné prévu.

Le contrôle **regToken** n'est généralement utilisé que pour l'initialisation d'une entité finale dans l'infrastructure PKI, alors que le contrôle d'authentifiant (se reporter au prochain paragraphe) est généralement utilisé pour des demandes de certification aussi bien initiales qu'ultérieures.

Dans certains cas d'utilisation, la valeur de **regToken** pourrait être une chaîne de texte ou une quantité numérique telle qu'un nombre aléatoire. Dans ce dernier cas, la valeur pourrait en principe être représentée soit comme une quantité binaire soit comme une chaîne de texte. Afin d'assurer un codage uniforme des valeurs indépendamment de la nature de la quantité, il est exigé que **regToken** soit codé comme une valeur de type UTF8String.

8.3.5.2 Contrôle d'authentifiant (*authenticator*)

Un contrôle d'authentifiant contient des informations utilisées sur une base d'actualité afin d'établir une vérification non cryptographique d'identité dans la communication avec l'autorité de certification. Des exemples en sont: le nom de jeune fille de la mère, les quatre derniers chiffres du numéro de sécurité sociale ou d'autres informations fondées sur la connaissance qui sont partagées avec l'autorité de certification de l'abonné; un code de hachage de ces informations; ou d'autres informations produites pour le besoin. La valeur du contrôle authenticator peut être créée par l'abonné ou par l'autorité de certification.

Dans certains cas d'utilisation, la valeur pour **authenticator** pourrait être une chaîne de texte ou une quantité numérique telle qu'un nombre aléatoire. Dans ce dernier cas, la valeur pourrait en principe être représentée soit comme une quantité binaire soit comme une chaîne de texte. Afin d'assurer un codage uniforme des valeurs indépendamment de la nature de la quantité, il est exigé qu'**authenticator** soit codé comme une valeur de type UTF8String.

8.3.5.3 Contrôle d'information de publication

Le contrôle **pkiPublicationInfo** permet aux abonnés de contrôler la publication du certificat par l'autorité de certification. Il est défini par la syntaxe suivante:

```
PKIPublicationInfo ::= SEQUENCE {
   action     INTEGER {
              dontPublish (0),
              pleasePublish (1) },
   pubInfos  SEQUENCE SIZE (1..MAX) OF SinglePubInfo OPTIONAL }
    -- pubInfos NE DOIT PAS être présent si l'action est "dontPublish"
    -- (si l'action est "pleasePublish" et que pubInfos est omis,
    -- "dontCare" est implicite)

SinglePubInfo ::= SEQUENCE {
   pubMethod    INTEGER {
      dontCare       (0),
      x500           (1),
      web            (2),
      ldap           (3) },
   pubLocation  GeneralName OPTIONAL }
```

Si l'option **dontPublish** est choisie, le demandeur indique qu'il convient que l'infrastructure PKI ne publie pas le certificat (cela peut indiquer que le demandeur a l'intention de publier lui-même ou elle-même le certificat).

Si la méthode **dontCare** est choisie ou si le contrôle **PKIPublicationInfo** est omis de la demande, le demandeur indique que l'infrastructure PKI peut publier le certificat par tout moyen de son choix.

Si le demandeur souhaite que le certificat apparaisse dans au moins certains endroits mais souhaite permettre à l'autorité de certification de rendre le certificat disponible dans d'autres dépôts, on règle deux valeurs de **SinglePubInfo** pour **pubInfos**: une avec la valeur **x500**, **web** ou **ldap** et l'autre avec **dontCare**.

Le champ **pubLocation**, s'il est fourni, indique que le demandeur souhaiterait que l'on trouve le certificat (il convient de noter que le CHOICE dans **GeneralName** comprend une adresse Web et une adresse IP, par exemple).

8.3.5.4 Contrôle OldCert ID

S'il est présent, le contrôle **OldCertID** spécifie le certificat devant être mis à jour par la demande actuelle de certification. La syntaxe de sa valeur est:

```
CertId ::= SEQUENCE {
    issuer          GeneralName,
    serialNumber    INTEGER
}
```

8.3.5.5 Contrôle de clé de chiffrement de protocole

S'il est présent, le contrôle **protocolEncrKey** spécifie une clé que l'autorité de certification doit utiliser pour chiffrer une réponse à des messages **CertReqMessages**. Ce contrôle peut être utilisé lorsqu'une autorité de certification détient des informations à envoyer à l'abonné qu'il est nécessaire de coder. Ces informations comprennent une clé privée créée par l'autorité de certification pour être utilisée par l'abonné.

Le codage de **protocolEncrKey** doit être **SubjectPublicKeyInfo**.

8.3.6 Identificateurs d'objet (*object identifiers*)

id-pkix de l'identificateur OID a la valeur

```
id-pkix  OBJECT IDENTIFIER  ::= { iso(1) identified-organization(3)
dod(6) internet(1) security(5) mechanisms(5) pkix(7) }

-- arc pour les protocoles de PKI X.509 pour l'Internet et leurs composants
id-pkip  OBJECT IDENTIFIER :: { id-pkix pkip(5) }

-- Contrôles d'enregistrement dans le format CRMF
id-regCtrl  OBJECT IDENTIFIER ::= { id-pkip regCtrl(1) }
id-regCtrl-regToken                 OBJECT IDENTIFIER ::= { id-regCtrl 1 }
id-regCtrl-authenticator            OBJECT IDENTIFIER ::= { id-regCtrl 2 }
id-regCtrl-pkiPublicationInfo       OBJECT IDENTIFIER ::= { id-regCtrl 3 }
id-regCtrl-oldCertID                OBJECT IDENTIFIER ::= { id-regCtrl 5 }
id-regCtrl-protocolEncrKey          OBJECT IDENTIFIER ::= { id-regCtrl 6 }

-- Info d'enregistrement dans le format CRMF
id-regInfo      OBJECT IDENTIFIER ::= { id-pkip id-regInfo(2) }
id-regInfo-asciiPairs   OBJECT IDENTIFIER ::= { id-regInfo 1 }
-- avec la syntaxe OCTET STRING
id-regInfo-certReq      OBJECT IDENTIFIER ::= { id-regInfo 2 }
-- avec la syntaxe CertRequest
```

8.4 Structures de données spécifiques pour d'autres messages

8.4.1 Demande d'initialisation

Un message de demande d'initialisation contient comme corps **PKIBody** une structure de données **CertReqMessages** qui spécifie le ou les certificats demandés. Classiquement, **SubjectPublicKeyInfo**, **KeyId** et **Validity** sont des champs modèles qui peuvent être fournis pour chaque certificat demandé. Ce message est destiné à être utilisé pour les entités qui s'initialisent pour la première fois dans l'infrastructure PKI.

Se reporter au § 8.3 pour la syntaxe de **CertReqMessages**.

8.4.2 Réponse d'initialisation

Un message de réponse d'initialisation contient comme corps **PKIBody** une structure de données **CertRepMessage** qui présente pour chaque certificat demandé un champ **PKIStatusInfo**, un certificat de titulaire et une possible clé privée (normalement codée avec une clé de session qui est elle-même codée avec la clé **protocolEncKey**).

Se reporter au § 8.4.4 pour la syntaxe de **CertRepMessage**.

8.4.3 Demande d'enregistrement/certification

Un message de demande d'enregistrement/certification contient comme corps **PKIBody** une structure de données **CertReqMessages** qui spécifie les certificats demandés. Ce message est destiné à être utilisé pour les entités de PKI existantes qui souhaitent obtenir des certificats supplémentaires.

Se reporter au § 8.3 pour la syntaxe de **CertReqMessages**.

En variante, le corps **PKIBody** peut être une demande **CertificationRequest** (cette structure est entièrement spécifiée par la structure **CertificationRequest** de l'ASN.1 fournie en PKCS#10). Cette structure peut être exigée pour les demandes de certificats destinés aux paires de clés de signature lorsqu'un interservice avec des systèmes patrimoniaux est souhaité mais son utilisation est fortement déconseillée chaque fois qu'elle n'est pas absolument nécessaire.

8.4.4 Réponse d'enregistrement/certification

Un message de réponse d'enregistrement contient comme corps **PKIBody** une structure de donnnées **CertRepMessage** qui présente une valeur de statut pour chaque certificat demandé et a facultativement une clé publique de CA, des informations d'échec, un certificat de titulaire et une clé privée codée.

```
CertRepMessage ::= SEQUENCE {
   caPubs        [1] SEQUENCE SIZE (1..MAX) OF Certificate OPTIONAL,
   response          SEQUENCE OF CertResponse
}

CertResponse ::= SEQUENCE {
   certReqId         INTEGER,
   -- pour faire correspondre cette réponse avec la demande correspondante (une valeur
   -- de –1 doit être utilisée si certReqId n'est pas spécifié dans la
   -- demande correspondante)
   status            PKIStatusInfo,
   certifiedKeyPair  CertifiedKeyPair  OPTIONAL,
   rspInfo           OCTET STRING      OPTIONAL
   -- analogues à id-regInfo-asciiPairs OCTET STRING définie
   -- pour regInfo dans CertReqMsg au § 8.3.6.
}

CertifiedKeyPair ::= SEQUENCE {
   certOrEncCert     CertOrEncCert,
   privateKey      [0] EncryptedValue    OPTIONAL,
publicationInfo [1] PKIPublicationInfo    OPTIONAL
}

CertOrEncCert ::= CHOICE {
   certificate       [0] Certificate,
   encryptedCert     [1] EncryptedValue
}
```

Un seul des champs **failInfo** (dans **PKIStatusInfo**) et **certificate** (dans **CertifiedKeyPair**) peut être présent dans chaque réponse **CertResponse** (en fonction du statut). Pour certaines valeurs de statut [par exemple, waiting (en attente)] aucun des champs facultatifs n'est présent.

En recevant une valeur **EncryptedCert** et la clé de déchiffrement correspondante, le certificat peut être obtenu. L'objectif en est de permettre à une autorité de certification de renvoyer la valeur d'un certificat mais avec la contrainte que seul le destinataire prévu peut obtenir le certificat réel. L'avantage de cette approche est que l'autorité de certification peut répondre avec un certificat même en l'absence d'une preuve que le demandeur est l'entité finale qui peut utiliser la clé privée correspondante (il convient de noter que la preuve n'est pas obtenue jusqu'à ce que l'autorité de certification reçoive le message **PKIConfirm**). Ainsi, l'autorité de certification n'aura pas à annuler ce certificat lorsqu'il se produit un dysfonctionnement concernant la preuve de possession.

8.4.5 Contenu de la demande de mise à jour de clé

Pour les demandes de mise à jour de clé, on utilise la syntaxe **CertReqMessages**. Classiquement, **SubjectPublicKeyInfo**, **KeyId** et **Validity** sont les champs modèles qui peuvent être fournis pour chaque clé devant être mise à jour. Ce message est destiné à être utilisé pour demander des mises à jour de certificats existants (non annulés et non expirés).

Se reporter au § 8.3 pour la syntaxe de **CertReqMessages**.

8.4.6 Contenu de la réponse de mise à jour de clés

Pour les réponses de mise à jour de clés, on utilise la syntaxe **CertRepMessage**. La réponse est identique à la réponse d'initialisation.

Se reporter au § 8.4.4 pour la syntaxe de **CertRepMessage**.

8.4.7 Contenu de la demande d'annulation

Lorsque l'on demande l'annulation d'un certificat (ou de plusieurs certificats), on utilise la structure de donnée ci-après. Le nom du demandeur est présent dans la structure **PKIHeader**.

```
RevReqContent ::= SEQUENCE OF RevDetails

RevDetails ::= SEQUENCE {
    certDetails       CertTemplate,
    -- permet au demandeur de spécifier autant qu'il peut sur le
    -- certificat pour lequel une annulation est demandée
    -- (par exemple, pour les cas où un serialNumber n'est pas disponible)
    revocationReason  ReasonFlags      OPTIONAL,
    -- raison pour laquelle l'annulation est demandée
    badSinceDate      GeneralizedTime  OPTIONAL,
    -- indique la meilleure connaissance de l'émetteur
    crlEntryDetails   Extensions       OPTIONAL
    -- crlEntryExtensions demandées
}
```

8.4.8 Contenu de la réponse d'annulation

Il s'agit de la réponse au message ci-dessus. Si elle est créée, elle est envoyée au demandeur de l'annulation. (Un message d'annonce d'annulation distinct peut être envoyé au titulaire du certificat dont l'annulation a été demandée.)

```
RevRepContent ::= SEQUENCE {
    status      SEQUENCE SIZE (1..MAX) OF PKIStatusInfo,
    -- dans le même ordre d'envoi que dans RevReqContent
    revCerts  [0] SEQUENCE SIZE (1..MAX) OF CertId OPTIONAL,
    -- Identificateurs dont l'annulation a été demandée (dans le même ordre que statut)
    crls    [1] SEQUENCE SIZE (1..MAX) OF CertificateList  OPTIONAL
    -- les listes CRL qui en découlent (il peut y en avoir plus d'une)
}
```

8.4.9 Contenu de la demande de certification réciproque

Les demandes de certification réciproque utilisent la même syntaxe (**CertReqMessages**) que les demandes de certification normale avec la restriction que la paire de clés doit être créée par l'autorité CA qui fait la demande et que la clé privée ne doit pas être envoyée à l'autorité CA qui répond.

Se reporter au § 8.3 pour la syntaxe de **CertReqMessages**.

8.4.10 Contenu de la réponse de certification réciproque

Les réponses de certification réciproque utilisent la même syntaxe (**CertRepMessage**) que les réponses de certification normale avec la restriction qu'aucune clé privée codée ne peut être envoyée.

Se reporter au § 8.4.4 pour la syntaxe de **CertRepMessage**.

8.4.11 Contenu de l'annonce de mise à jour de clés

Lorsqu'une autorité de certification met à jour sa propre paire de clés, la structure de données suivante peut être utilisée pour annoncer cet événement.

```
CAKeyUpdAnnContent ::= SEQUENCE {
    oldWithNew     Certificate, -- ancienne clé publique signée avec nouvelle clé privée
    newWithOld     Certificate, -- nouvelle clé publique signée avec ancienne clé privée
    newWithNew     Certificate  -- nouvelle clé publique signée avec nouvelle clé privée
}
```

8.4.12 Annonce de certificat

Cette structure peut être utilisée pour annoncer l'existence de certificats.

Il convient de noter que ce message est destiné à être utilisé pour les cas (s'il en existe) où il n'y a pas de méthode préexistante de publication de certificats; il n'est pas destiné à être utilisé lorsque, par exemple, X.500 est la méthode de publication de certificats.

```
CertAnnContent ::= Certificate
```

8.4.13 Annonce d'annulation

Lorsqu'une autorité de certification a annulé ou est sur le point d'annuler un certificat particulier, elle peut émettre une annonce de cet événement (possible prochainement).

```
RevAnnContent ::= SEQUENCE {
    status          PKIStatus,
    certId          CertId,
    willBeRevokedAt    GeneralizedTime,
    badSinceDate       GeneralizedTime,
    crlDetails         Extensions  OPTIONAL
    -- détails supplémentaires de liste CRL (par exemple, numéro de liste CRL, raison, emplacement, etc.)
}
```

Une autorité de certification peut utiliser une telle annonce pour avertir (ou aviser) un titulaire que son certificat est sur le point d'être (ou a été) annulé. Ceci est classiquement utilisé lorsque la demande d'annulation ne vient pas du titulaire concerné.

Le champ **willBeRevokedAt** contient l'heure à laquelle une nouvelle saisie est ajoutée aux listes CRL concernées.

8.4.14 Annonce de listes d'annulation de certificat (CRL)

Lorsqu'une autorité de certification émet une nouvelle liste CRL (ou un ensemble de listes CRL), on peut utiliser la structure de données ci-après pour annoncer l'événement.

```
CRLAnnContent ::= SEQUENCE OF CertificateList
```

8.4.15 Contenu de la confirmation de PKI

Cette structure de données est utilisée dans des protocoles à trois voies en tant que le message **PKIMessage** final. Son contenu est le même dans tous les cas – en fait, il n'y a pas de contenu car l'en-tête **PKIHeader** contient toutes les informations nécessaires.

```
PKIConfirmContent ::= NULL
```

8.4.16 Contenu du message général de PKI

```
InfoTypeAndValue ::= SEQUENCE {
    infoType          TYPE-IDENTIFIER.&id({InfoTable}),
    infoValue         TYPE-IDENTIFIER.&Type ({InfoTable}{@infoType}) OPTIONAL
}
    -- Des exemples de contenus de InfoTypeAndValue comprennent, sans s'y limiter:
    -- { CAProtEncCert    = {id-it 1}, Certificate                }
    -- { SignKeyPairTypes = {id-it 2}, SEQUENCE OF AlgorithmIdentifier }
    -- { EncKeyPairTypes  = {id-it 3}, SEQUENCE OF AlgorithmIdentifier }
    -- { PreferredSymmAlg = {id-it 4}, AlgorithmIdentifier        }
    -- { CAKeyUpdateInfo  = {id-it 5}, CAKeyUpdAnnContent         }
    -- { CurrentCRL       = {id-it 6}, CertificateList            }
    -- où {id-it} = {id-pkix 4} = {1 3 6 1 5 5 7 4}
    -- Cette construction peut également être utilisée pour définir de nouveaux messages
    -- de demande et de réponse de protocole de gestion de certificat PKIX ou des
    -- messages d'usage général (par exemple, annonce) pour de futurs besoins ou pour
    -- des environnements spécifiques.
```

```
GenMsgContent ::= SEQUENCE OF InfoTypeAndValue
   -- peut être envoyé par l'entité EE, l'autorité RA ou CA (selon le contenu du message).
   -- Le paramètre FACULTATIF infoValue de InfoTypeAndValue est généralement
   -- omis dans certains des exemples fournis ci-dessus. Le destinataire est
   -- libre d'ignorer tout identificateur d'objet (OBJECT ID) contenu qu'il ne reconnaît pas.
   -- Lorsqu'il est envoyé d'une entité finale à une autorité de certification, l'ensemble vide indique
   -- que l'autorité de certification peut envoyer toute information ou toutes les informations qu'elle souhaite.
```

8.4.17 Contenu de la réponse générale de PKI

```
GenRepContent ::= SEQUENCE OF InfoTypeAndValue
-- Le destinataire est libre d'ignorer tout identificateur d'objet (OBJECT ID) contenu qu'il
-- ne reconnaît pas.
```

8.4.18 Contenu du message d'erreur

```
ErrorMsgContent ::= SEQUENCE {
   pKIStatusInfo        PKIStatusInfo,
   errorCode            INTEGER       OPTIONAL,
   -- codes d'erreur spécifiques à chaque implémentation
   errorDetails         PKIFreeText   OPTIONAL
   -- détails d'erreur spécifiques à chaque implémentation
}
```

8.5 Protocoles de transport

Aucun protocole de transport spécifique n'est obligatoire pour les entités finales, les autorités d'enregistrement et les autorités de certification pour se passer des messages entre elles. Il n'y a aucune prescription relative à des mécanismes de sécurité spécifiques devant être appliquée à ce niveau si les messages de PKI sont convenablement protégés (c'est-à-dire, si le paramètre de PKIProtection facultatif est utilisé comme il est spécifié pour chaque message).

Les messages de PKI peuvent être acheminés dans des fichiers ne contenant que le codage selon DER d'un seul message de PKI, c'est-à-dire sans en-tête étranger ou information de fin dans le fichier. Ces fichiers peuvent être utilisés pour acheminer des messages de PKI en utilisant, par exemple, le protocole de transfert de fichiers FTP. Ces fichiers peuvent également être joints à des méls ou transférés par HTTP (on pourrait définir des objets MIME spéciaux à cet effet).

8.6 Module complet de l'ASN.1

8.6.1 Module spécifique pour le format de message de demande de certification (CRMF)

```
CRMF DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- EXPORTER TOUT; --
IMPORTS
-- Cadre d'information de l'annuaire (X.501)
      Name
        FROM InformationFramework {
          joint-iso-itu-t(2) ds(5) module(1) informationFramework(1) 3 }

   -- Cadre d'authentification de l'annuaire (X.509)
      AlgorithmIdentifier, Extensions, SubjectPublicKeyInfo, Time,
        Version
        FROM AuthenticationFramework {
          joint-iso-itu-t(2) ds(5) module(1) authenticationFramework(7) 3 }

   -- Attributs sélectionnés de l'annuaire (X.520)
      UniqueIdentifier
        FROM SelectedAttributeTypes {
          joint-iso-itu-t(2) ds(5) module(1) selectedAttributeTypes(5) 3 }

    -- Extensions de certificat (X.509)
      GeneralName
        FROM CertificateExtensions {joint-iso-itu-t(2) ds(5)
          module(1) certificateExtensions(26) 0}
```

```
    -- Syntaxe de message cryptographique
      EnvelopedData
        FROM CryptographicMessageSyntax { iso(1) member-body(2)
          us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16)
           modules(0) cms(1) };

CertReqMessages ::= SEQUENCE SIZE (1..MAX) OF CertReqMsg

CertReqMsg ::= SEQUENCE {
  certReq   CertRequest,
  pop       ProofOfPossession  OPTIONAL,
   -- le contenu dépend du type de clé
  regInfo   SEQUENCE SIZE(1..MAX) OF AttributeTypeAndValue OPTIONAL }

CertRequest ::= SEQUENCE {
  certReqId    INTEGER,                 -- ID pour demande et réponse en correspondance
certTemplate  CertTemplate,             -- Champs sélectionnés du certificat à émettre
  controls     Controls OPTIONAL }      -- Attributs affectant l'émission

CertTemplate ::= SEQUENCE {
  version          [0] Version                 OPTIONAL,
  serialNumber     [1] INTEGER                 OPTIONAL,
  signingAlg       [2] AlgorithmIdentifier     OPTIONAL,
  issuer           [3] EXPLICIT Name           OPTIONAL,
  validity         [4] OptionalValidity        OPTIONAL,
  subject          [5] EXPLICIT Name           OPTIONAL,
  publicKey        [6] SubjectPublicKeyInfo    OPTIONAL,
  issuerUID        [7] UniqueIdentifier        OPTIONAL,
  subjectUID       [8] UniqueIdentifier        OPTIONAL,
  extensions       [9] Extensions              OPTIONAL }

OptionalValidity ::= SEQUENCE {
  notBefore        [0] EXPLICIT Time OPTIONAL,
  notAfter         [1] EXPLICIT Time OPTIONAL } -- l'un au moins DOIT être présent

Controls  ::= SEQUENCE SIZE(1..MAX) OF AttributeTypeAndValue

AttributeTypeAndValue ::= SEQUENCE {
  type             TYPE-IDENTIFIER.&id ({CRMF-Table}),
  value            TYPE-IDENTIFIER.&Type ({CRMF-Table}{@type})
}

CRMF-Table TYPE-IDENTIFIER::={ ... }

ProofOfPossession ::= CHOICE {
  raVerified       [0] NULL,
  -- utilisé si l'autorité RA a déjà vérifié que le demandeur est en
  -- possession de la clé privée
  signature        [1] POPOSigningKey }

POPOSigningKey ::= SEQUENCE {
  poposkInput          [0] POPOSigningKeyInput OPTIONAL,
  algorithmIdentifier   AlgorithmIdentifier,
  signature             BIT STRING }
  -- La signature (utilisant "algorithmIdentifier") est sur la
  -- DER-encoded value of poposkInput. NOTE –  Si le modèle CertTemplate
  -- de certReq de CertReqMsg contient les valeurs de subject et publicKey,
  -- le champ poposkInput DOIT être omis et la signature DOIT être
  -- calculée sur la valeur codée selon DER de certReq de CertReqMsg.  Si
  -- le modèle de certReq de CertReqMsg ne contient pas les valeurs de clé
  -- publique et de titulaire (subject), le champ poposkInput DOIT être présent et
  -- DOIT être signé. Cette stratégie garantit que la clé publique n'est
  -- pas présente à la fois dans les deux champs poposkInput et CertTemplate de certReq
  -- CertReqMsg à la fois.

POPOSigningKeyInput ::= SEQUENCE {
  authInfo         CHOICE {
     sender        [0] EXPLICIT GeneralName,
     -- utilisé uniquement si une identité authentifiée a été
     -- établie pour l'émetteur (par exemple, un DN provenant
     -- d'un certificat précédemment émis et en cours de validité)
```

```
      publicKeyMAC     PKMACValue },
      -- utilisé si aucun nom GeneralName authentifié n'existe actuellement
      -- pour l'émetteur; publicKeyMAC contient un code MAC fondé sur un mot de passe
      -- sur la valeur codée selon DER de publicKey
   publicKey        SubjectPublicKeyInfo }  -- provenant de CertTemplate

PKMACValue ::= SEQUENCE {
   algId  AlgorithmIdentifier,
   -- la valeur de l'algorithme doit être PasswordBasedMac {1 2 840 113533 7 66 13}
   -- la valeur de paramètre est PBMParameter
   value  BIT STRING }

PBMParameter ::= SEQUENCE {
     salt            OCTET STRING,
     owf             AlgorithmIdentifier,
     -- AlgorithmIdentifier pour une fonction à une voie (SHA-1 recommandée)
     iterationCount     INTEGER,
     -- nombre de fois que la fonction OWF est appliquée
     mac             AlgorithmIdentifier
     -- AlgorithmIdentifierde MAC (par exemple  DES-MAC, Triple-DES-MAC comme dans PKCS #11,
}   -- ou HMAC comme dans RFC2104, RFC2202)

-- Affectations d'identificateurs d'objet --

id-pkix  OBJECT IDENTIFIER  ::= { iso(1) identified-organization(3)
dod(6) internet(1) security(5) mechanisms(5) 7 }

-- arc pour les protocoles de PKI  X.509 pour l'Internet et leurs composants
id-pkip  OBJECT IDENTIFIER ::= { id-pkix 5 }

-- Contrôles d'enregistrement dans le format CRMF
id-regCtrl OBJECT IDENTIFIER ::= { id-pkip 1 }

id-regCtrl-regToken OBJECT IDENTIFIER ::= { id-regCtrl 1 }
-- avec la syntaxe:
RegToken ::= UTF8String

id-regCtrl-authenticator OBJECT IDENTIFIER ::= { id-regCtrl 2 }
-- avec la syntaxe:
Authenticator ::= UTF8String

id-regCtrl-pkiPublicationInfo OBJECT IDENTIFIER ::= { id-regCtrl 3 }
-- avec la syntaxe:
PKIPublicationInfo ::= SEQUENCE {
   action     INTEGER {
              dontPublish (0),
              pleasePublish (1) },
   pubInfos  SEQUENCE SIZE (1..MAX) OF SinglePubInfo OPTIONAL }
   -- pubInfos NE DOIT PAS être présent si l'action est "dontPublish"
   -- (si l'action est "pleasePublish" et que pubInfos est omis,
   -- "dontCare" est implicite)

SinglePubInfo ::= SEQUENCE {
   pubMethod    INTEGER {
      dontCare    (0),
      x500        (1),
      web         (2),
      ldap        (3) },
   pubLocation  GeneralName OPTIONAL }

EncryptedKey ::= CHOICE {
   encryptedValue       EncryptedValue,
   envelopedData      [0] EnvelopedData }
   -- La clé privée codée DOIT être placée dans envelopedData
   -- encryptedContentInfo encryptedContent OCTET STRING.

EncryptedValue ::= SEQUENCE {
     intendedAlg    [0] AlgorithmIdentifier    OPTIONAL,
     symmAlg        [1] AlgorithmIdentifier    OPTIONAL,
     encSymmKey     [2] BIT STRING             OPTIONAL,
```

```
    keyAlg          [3] AlgorithmIdentifier    OPTIONAL,
    valueHint       [4] OCTET STRING           OPTIONAL,
    encValue            BIT STRING
  }

KeyGenParameters ::= OCTET STRING

id-regCtrl-oldCertID        OBJECT IDENTIFIER ::= { id-regCtrl 5 }
-- avec la syntaxe:
OldCertId ::= CertId

CertId ::= SEQUENCE {
  issuer          GeneralName,
  serialNumber    INTEGER }

id-regCtrl-protocolEncrKey    OBJECT IDENTIFIER ::= { id-regCtrl 6 }
-- avec la syntaxe:
ProtocolEncrKey ::= SubjectPublicKeyInfo

-- Info d'enregistrement dans le format CRMF
id-regInfo OBJECT IDENTIFIER ::= { id-pkip 2 }

id-regInfo-utf8Pairs    OBJECT IDENTIFIER ::= { id-regInfo 1 }
-- avec la syntaxe
UTF8Pairs ::= UTF8String

id-regInfo-certReq      OBJECT IDENTIFIER ::= { id-regInfo 2 }
-- avec la syntaxe
CertReq ::= CertRequest

END
```

8.6.2 Module général

```
-- Il convient de noter que cette syntaxe supplémentaire apparaît dans le module CRMF ci-dessus.
GeneralModule DEFINITIONS EXPLICIT TAGS ::=

  BEGIN

  -- EXPORTER TOUT --

  IMPORTS

--  Cadre d'information (X.501) --

  Attribute, Name
    FROM InformationFramework {
      joint-iso-itu-t ds(5) module(1) informationFramework(1) 3 }

-- Cadre d'authentification de l'annuaire (X.509) --

  AlgorithmIdentifier, Certificate, CertificateList, Extensions,
  SubjectPublicKeyInfo
    FROM AuthenticationFramework {
      joint-iso-itu-t ds(5) module(1) authenticationFramework(7) 3 }

-- Extensions de certificat (X.509)

  GeneralName, KeyIdentifier, ReasonFlags
    FROM CertificateExtensions {
      joint-iso-itu-t(2) ds(5) module(1) certificateExtensions(26) 0 }

-- X.843 ISO 15945 (CRMF) --

  CertTemplate, PKIPublicationInfo, EncryptedValue, CertId,
    CertReqMessages
        FROM CRMF;
```

```
-- CertificationRequest compatible avec PKCS#10

CertificationRequest ::= SEQUENCE {
  certificationRequestInfo  CertificationRequestInfo,
  signatureAlgorithm        AlgorithmIdentifier,
  signature                 BIT STRING
}

CertificationRequestInfo ::= SEQUENCE {
  version       INTEGER,
  subject       Name,
  subjectPKInfo  SubjectPublicKeyInfo,
  attributes    [0] IMPLICIT Attributes
}

Attributes ::= SET SIZE(0..MAX) OF Attribute

        -- Identificateurs OID définis localement ---- Il est à noter que l'étiquetage est EXPLICIT dans ce module.

PKIMessage ::= SEQUENCE {
  header           PKIHeader,
  body             PKIBody,
  protection       [0] PKIProtection OPTIONAL,
  extraCerts       [1] SEQUENCE SIZE (1..MAX) OF Certificate OPTIONAL
}

PKIHeader ::= SEQUENCE {
  pvno             INTEGER   { version1 (0) },
  sender           GeneralName,
  -- identifie l'émetteur
  recipient        GeneralName,
  -- identifie le destinataire prévu
  messageTime   [0] GeneralizedTime       OPTIONAL,
  -- heure de création de ce message (utilisé lorsque l'expéditeur
  -- pense que le transport est "convenable"; c'est-à-dire
  -- que le temps aura encore un sens au moment de la réception)
  protectionAlg  [1] AlgorithmIdentifier   OPTIONAL,
  -- algorithme utilisé pour le calcul de bits de protection
  senderKID      [2] KeyIdentifier         OPTIONAL,
  recipKID       [3] KeyIdentifier         OPTIONAL,
  -- pour identifier des clés spécifiques utilisées pour la protection
  transactionID  [4] OCTET STRING      OPTIONAL,
  -- identifie la transaction; c'est-à-dire qu'il s'agit du même identificateur dans
  -- les messages de demande, de réponse et de confirmation correspondants
  senderNonce    [5] OCTET STRING      OPTIONAL,
  recipNonce     [6] OCTET STRING      OPTIONAL,
  -- des mots créés pour la circonstance (nonces) utilisés pour fournir une protection contre le redéfilement
  -- senderNonce est inséré par le créateur de ce message; recipNonce
  -- est un mot créé pour la circonstance précédemment inséré dans un message apparenté par
  -- le destinataire prévu de ce message
  freeText       [7] PKIFreeText           OPTIONAL,
  -- ceci peut être utilisé pour indiquer des instructions spécifiques à chaque contexte
  -- (ce champ est destiné à un usage humain)
  generalInfo    [8] SEQUENCE SIZE (1..MAX) OF
                     InfoTypeAndValue      OPTIONAL
  -- on peut l'utiliser pour acheminer des informations spécifiques à chaque contexte
  -- (ce champ n'est pas destiné à l'origine pour un usage humain)
}

PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String
  -- texte codé comme une chaîne UTF-8 String (NOTE –  Il convient que chaque UTF8String
  -- comprenne une étiquette de langue RFC 1766 afin d'indiquer la langue
  -- du texte contenu)

PKIBody ::= CHOICE {                    -- éléments de corps propres au message
    ir       [0]  CertReqMessages,      -- Demande d'initialisation
    ip       [1]  CertRepMessage,       -- Réponse d'initialisation
    cr       [2]  CertReqMessages,      -- Demande de certification
    cp       [3]  CertRepMessage,       -- Réponse de certification
    p10cr    [4]  CertificationRequest, -- pour compatibilité avec [PKCS#10]
```

```
  kur       [7]  CertReqMessages,          -- Demande de mise à jour de clés
  kup       [8]  CertRepMessage,           -- Réponse de mise à jour de clés
  rr        [11] RevReqContent,            -- Demande d'annulation
  rp        [12] RevRepContent,            -- Réponse d'annulation
  ccr       [13] CertReqMessages,          -- Demande de certification reciproque
  ccp       [14] CertRepMessage,           -- Réponse de certification reciproque
  ckuann    [15] CAKeyUpdAnnContent,       -- Annonce de mise à jour de clés de CA
  cann      [16] CertAnnContent,           -- Annonce de certification
  rann      [17] RevAnnContent,            -- Annonce d'annulation
  crlann    [18] CRLAnnContent,            -- Annonce de liste CRL
  conf      [19] PKIConfirmContent,        -- Confirmation imbriquée
  nested    [20] NestedMessageContent,     -- Message imbriqué
  genm      [21] GenMsgContent,            -- Message général
  genp      [22] GenRepContent,            -- Réponse générale
  error     [23] ErrorMsgContent           -- Message d'erreur
}

PKIProtection ::= BIT STRING

ProtectedPart ::= SEQUENCE {
  header    PKIHeader,
  body      PKIBody
}

PasswordBasedMac ::= OBJECT IDENTIFIER --{1 2 840 113533 7 66 13}

PBMParameter ::= SEQUENCE {
  salt              OCTET STRING,
  owf               AlgorithmIdentifier,
  -- AlgorithmIdentifier pour une fonction à une voie (SHA-1 recommandée)
  iterationCount    INTEGER,
  -- nombre de fois que la fonction OWF est appliquée
  mac               AlgorithmIdentifier
  -- AlgorithmIdentifier de MAC (par exemple  DES-MAC, Triple-DES-MAC comme dans PKCS #11,
} -- ou HMAC comme dans RFC2104, RFC2202)

DHBasedMac ::= OBJECT IDENTIFIER --{1 2 840 113533 7 66 30}

DHBMParameter ::= SEQUENCE {
  owf               AlgorithmIdentifier,
  -- AlgorithmIdentifier pour une fonction à une voie  (SHA-1 recommandée)
  mac               AlgorithmIdentifier
   -- AlgorithmIdentifier de MAC (par exemple  DES-MAC, Triple-DES-MAC comme dans PKCS #11,
} -- ou HMAC comme dans RFC2104, RFC2202)

NestedMessageContent ::= PKIMessage

PKIStatus ::= INTEGER {
  granted                   (0),
  -- vous avez obtenu exactement ce que vous avez demandé
  grantedWithMods           (1),
  -- vous avez obtenu quelque chose qui ressemble à ce que vous avez demandé; le
  -- demandeur est responsable d'identifier les différences
  rejection                 (2),
  -- vous ne l'obtenez pas, de plus amples informations se trouvent ailleurs dans le message
  waiting                   (3),
  -- le corps de la demande n'a pas encore été traité,
  -- attendez-vous à en savoir davantage plus tard
  revocationWarning         (4),
  --  ce message contient un avertissement indiquant qu'une annulation est
  -- imminente
  revocationNotification    (5),
  -- notification qu'une annulation s'est produite
  keyUpdateWarning          (6)
  -- mise à jour déjà effectuée pour le oldCertId spécifié dans
  -- CertReqMsg
}
```

```
PKIFailureInfo ::= BIT STRING {
  -- car une demande peut échouer de plus d'une façon!
  -- Plus de codes peuvent être ajoutés à l'avenir si exigés/lorsque exigés.
  badAlg                (0),
  -- Identificateur d'algorithme non reconnu ou non pris en charge
  badMessageCheck       (1),
  -- la vérification d'intégrité a échoué (par exemple, la signature n'a pas été vérifiée)
  badRequest            (2),
  -- transaction non autorisée ou non prise en charge
  badTime               (3),
  -- l'heure du message n'est pas suffisamment proche de l'heure système,
  -- tel que le définit la politique locale
  badCertId             (4),
  -- aucun certificat satisfaisant aux critères fournis n'a pu être trouvé
  badDataFormat         (5),
  -- les données soumises ont le mauvais format
  wrongAuthority        (6),
  -- l'autorité indiquée dans la demande est différente de
  -- celle qui crée le jeton de réponse
  incorrectData         (7),
  -- les données du demandeur sont incorrectes (pour des services de notarisation)
  missingTimeStamp      (8)
  -- lorsque l'horodatage manque alors qu'il devrait y être (par politique)
}

PKIStatusInfo ::= SEQUENCE {
  status            PKIStatus,
  statusString      PKIFreeText     OPTIONAL,
  failInfo          PKIFailureInfo  OPTIONAL
}

OOBCert ::= Certificate

OOBCertHash ::= SEQUENCE {
  hashAlg       [0] AlgorithmIdentifier    OPTIONAL,
  certId        [1] CertId                 OPTIONAL,
  hashVal           BIT STRING
-- hashVal est calculée sur le codage selon DER
-- du champ subjectPublicKey du certificat correspondant
}

CertRepMessage ::= SEQUENCE {
  caPubs        [1] SEQUENCE SIZE (1..MAX) OF Certificate OPTIONAL,
  response          SEQUENCE OF CertResponse
}

CertResponse ::= SEQUENCE {
  certReqId         INTEGER,
  -- pour faire correspondre cette réponse avec la demande correspondante (une valeur
  -- de –1 doit être utilisée si certReqId n'est pas spécifié dans la
  -- demande correspondante)
  status                PKIStatusInfo,
  certifiedKeyPair      CertifiedKeyPair   OPTIONAL,
  rspInfo               OCTET STRING       OPTIONAL
  -- analogues à id-regInfo-asciiPairs OCTET STRING défini
  -- pour regInfo dans CertReqMsg
}

CertifiedKeyPair ::= SEQUENCE {
  certOrEncCert       CertOrEncCert,
  privateKey        [0] EncryptedValue       OPTIONAL,
  publicationInfo   [1] PKIPublicationInfo   OPTIONAL
}

CertOrEncCert ::= CHOICE {
  certificate       [0] Certificate,
  encryptedCert     [1] EncryptedValue
}
```

```
KeyRecRepContent ::= SEQUENCE {
  status              PKIStatusInfo,
  newSigCert    [0]   Certificate             OPTIONAL,
  caCerts       [1]   SEQUENCE SIZE (1..MAX) OF
                      Certificate             OPTIONAL,
  keyPairHist   [2]   SEQUENCE SIZE (1..MAX) OF
                      CertifiedKeyPair        OPTIONAL
}

RevReqContent ::= SEQUENCE OF RevDetails

RevDetails ::= SEQUENCE {
  certDetails         CertTemplate,
  -- permet au demandeur de spécifier autant qu'il peut sur le
  -- certificat pour lequel une annulation est demandée
  -- (par exemple, pour les cas où un serialNumber n'est pas disponible)
  revocationReason    ReasonFlags         OPTIONAL,
  -- la raison pour laquelle l'annulation est demandée
  badSinceDate        GeneralizedTime     OPTIONAL,
  -- indique la meilleure connaissance de l'émetteur
  crlEntryDetails     Extensions          OPTIONAL
  -- crlEntryExtensions demandées
}

RevRepContent ::= SEQUENCE {
  status       SEQUENCE SIZE (1..MAX) OF PKIStatusInfo,
  -- dans le même ordre d'envoi que dans RevReqContent
  revCerts     [0] SEQUENCE SIZE (1..MAX) OF CertId OPTIONAL,
  -- Identificateurs dont l'annulation a été demandée (dans le même ordre que statut)
  crls         [1] SEQUENCE SIZE (1..MAX) OF CertificateList  OPTIONAL
  -- les listes CRL qui en découlent  (il peut y en avoir plus d'une)
}

CAKeyUpdAnnContent ::= SEQUENCE {
  oldWithNew        Certificate, -- ancienne clé publique signée avec nouvelle clé privée
  newWithOld        Certificate, -- nouvelle clé publique signée avec ancienne clé privée
  newWithNew        Certificate  -- nouvelle clé publique signée avec nouvelle clé privée
}

CertAnnContent ::= Certificate

RevAnnContent ::= SEQUENCE {
  status          PKIStatus,
  certId          CertId,
  willBeRevokedAt    GeneralizedTime,
  badSinceDate       GeneralizedTime,
  crlDetails      Extensions  OPTIONAL
  -- détails supplémentaires de liste CRL (par exemple, numéro de liste CRL, raison, emplacement, etc.)
}

CRLAnnContent ::= SEQUENCE OF CertificateList

PKIConfirmContent ::= NULL

InfoTypeAndValue ::= SEQUENCE {
  infoType            TYPE-IDENTIFIER.&id ({InfoTable}),
  infoValue           TYPE-IDENTIFIER.&Type ({InfoTable}@infoType}) OPTIONAL
}

InfoTable TYPE-IDENTIFIER ::= { ... }

-- Des exemples de contenus de InfoTypeAndValue comprennent, sans s'y limiter:
--  { CAProtEncCert    = {id-it 1}, Certificate                }
--  { SignKeyPairTypes = {id-it 2}, SEQUENCE OF AlgorithmIdentifier }
--  { EncKeyPairTypes  = {id-it 3}, SEQUENCE OF AlgorithmIdentifier }
--  { PreferredSymmAlg = {id-it 4}, AlgorithmIdentifier          }
--  { CAKeyUpdateInfo  = {id-it 5}, CAKeyUpdAnnContent           }
--  { CurrentCRL       = {id-it 6}, CertificateList              }
--où {id-it} = {id-pkix 4} = {1 3 6 1 5 5 7 4}
```

```
-- Cette construction peut également être utilisée pour définir de nouveaux messages
-- de demande et de réponse de protocole de gestion de certificat PKIX ou des
-- messages d'usage général (par exemple: annonce) pour de futurs besoins ou pour
-- des environnements spécifiques.

GenMsgContent ::= SEQUENCE OF InfoTypeAndValue
-- peut être envoyé par l'entité EE, l'autorité RA ou CA (selon le contenu du message).
-- Le paramètre FACULTATIF infoValue de InfoTypeAndValue est généralement
-- omis dans certains des exemples fournis ci-dessus. Le destinataire est
-- libre d'ignorer tout identificateur d'objet (OBJECT ID) contenu qu'il ne reconnaît pas.
-- Lorsqu'il est envoyé d'une entité finale à une autorité de certification, l'ensemble vide indique que l'autorité de
-- certification peut envoyer toute information ou toutes les informations qu'elle souhaite.

GenRepContent ::= SEQUENCE OF InfoTypeAndValue
-- Le destinataire est libre d'ignorer tout identificateur d'objet (OBJECT ID) contenu qu'il
-- ne reconnaît pas.

ErrorMsgContent ::= SEQUENCE {
  pKIStatusInfo          PKIStatusInfo,
  errorCode              INTEGER           OPTIONAL,
  -- codes d'erreur spécifiques à chaque implémentation
  errorDetails           PKIFreeText       OPTIONAL
  -- détails d'erreur spécifiques à chaque implémentation
}

END
```

9 Protocole de statut de certificat en ligne

Le présent article spécifie un protocole utile pour déterminer le statut actuel d'un certificat sans exiger de liste d'annulation de certificats (CRL). La structure de données est spécifiée conformément au document RFC 2560. Un aperçu général du protocole est fourni au § 9.1. Des prescriptions fonctionnelles sont spécifiées au § 9.2. Des détails du protocole figurent au § 9.3. Le § 9.4 contient le module ASN.1 pour les structures de données nécessaires au protocole.

NOTE – Le code ASN.1 est équivalent à celui du RFC susmentionné bien que la syntaxe semble partiellement être différente.

9.1 Aperçu général du protocole

Le protocole de statut de certificat en ligne (OCSP, *on-line certificate status protocol*) permet aux applications de déterminer l'état d'un certificat identifié. Le protocole OCSP peut être utilisé pour satisfaire à certaines des prescriptions d'exploitation relatives à la fourniture de davantage d'informations d'annulation opportunes qu'il n'est possible avec les listes CRL et il peut être également utilisé pour obtenir des informations de statut supplémentaires. Une entité finale utilisant des services OCSP émet une demande de statut auprès d'un tiers TTP du protocole OCSP et suspend l'acceptation du certificat en question jusqu'à ce que le tiers fournisse une réponse.

Ce protocole spécifie les données qu'il est nécessaire d'échanger entre une entité finale qui vérifie le statut du certificat et le tiers TTP qui fournit ce statut.

9.1.1 Demande

Une demande de protocole OCSP contient les données suivantes:

- la version du protocole;
- la demande de services;
- l'identificateur du certificat visé;
- des extensions facultatives qui peuvent être traitées par le tiers TTP du protocole OCSP.

A la réception d'une demande, un tiers TTP du protocole OCSP détermine:

1) si le message est bien formé;
2) si le tiers TTP est configuré pour fournir le service demandé; et
3) si la demande contient les informations dont le tiers TTP a besoin.

Si l'une quelconque des conditions précédentes n'est pas satisfaite, le tiers TTP du protocole OCSP produit un message d'erreur; sinon, il renvoie une réponse définitive.

9.1.2 Réponse

Une réponse OCSP peut être de divers types. Une réponse OCSP est constituée d'un type de réponse et des octets de la réponse effective. Il y a un type fondamental de réponse OCSP qui doit être pris en charge par tous les tiers TTP fournissant des services OCSP et par toutes les entités utilisant ces services. Le reste du présent paragraphe se rapporte uniquement à ce type de réponse fondamental.

Tous les messages de réponse définitive doivent être signés numériquement. La clé utilisée pour signer la réponse doit appartenir à l'un des éléments suivants:

- l'autorité de certification qui a émis le certificat en question;
- un tiers TTP dont la clé publique a la confiance du demandeur;
- un tiers TTP désigné par l'autorité de certification (un tiers TTP autorisé) qui détient un certificat spécial émis par l'autorité de certification qui indique qu'il peut émettre des réponses OCSP pour cette autorité de certification.

Un message de réponse définitive est composé:

- de la version de la syntaxe de réponse;
- du nom du tiers TTP;
- des réponses pour chacun des certificats figurant dans une demande;
- d'extensions facultatives;
- de l'identificateur OID de l'algorithme de signature;
- d'une signature calculée par hachage de la réponse.

La réponse pour chacun des certificats figurant dans une demande est composée:

- de l'identificateur de certificat visé;
- de la valeur du statut de certificat;
- de la période de validité de réponse;
- d'extensions facultatives.

La présente Recommandation | Norme internationale définit les indicateurs de réponse définitive suivants pour une utilisation dans la valeur du statut de certificat:

- **good** (bon);
- **revoked** (annulé);
- **unknown** (inconnu).

L'état "**bon**" indique une réponse positive à la demande de renseignements sur le statut. Au minimum, cette réponse positive indique que le certificat n'est pas annulé mais elle ne signifie pas nécessairement que le certificat a été émis un jour ou que le moment de la production de la réponse est dans la limite de la période de validité du certificat. Des extensions à la réponse peuvent être utilisées pour acheminer des informations supplémentaires à la suite d'assertions faites par le tiers TTP concernant le statut du certificat telles qu'une déclaration positive sur l'émission, la validité, etc.

L'état "**annulé** " indique que le certificat a été annulé (soit de façon permanente soit de façon temporaire [en suspens)].

L'état "**inconnu** " indique que le tiers TTP ne sait rien du certificat demandé.

9.1.3 Cas d'exception

En cas d'erreurs, le tiers TTP du protocole OCSP peut renvoyer un message d'erreur. Ces messages ne sont pas signés. Les erreurs peuvent être des types suivants:

- **malformedRequest;**
- **internalError;**
- **tryLater;**
- **sigRequired;**
- **unauthorized.**

Un tiers TTP produit la réponse "**malformedRequest**" si la demande reçue n'est pas conforme à la syntaxe du protocole OCSP.

La réponse "**internalError**" indique que le tiers TTP du protocole a atteint un état interne incohérent. Il convient que la demande soit répétée, potentiellement avec un autre tiers TTP.

Dans le cas où le tiers TTP du protocole OCSP est opérationnel mais incapable de renvoyer un statut pour le certificat demandé, la réponse "**tryLater**" peut être utilisée pour indiquer que le service existe mais n'est pas temporairement à même de répondre.

La réponse "**sigRequired**" est renvoyée dans les cas où le tiers TTP exige que l'entité finale signe la demande afin de construire une réponse.

La réponse "**unauthorized**" est renvoyée dans les cas où l'entité finale n'est pas autorisée à poser cette requête auprès de ce tiers TTP.

9.1.4 Sémantique de thisUpdate, nextUpdate et producedAt

Les réponses peuvent contenir trois temps – **thisUpdate**, **nextUpdate** et **producedAt**. La sémantique de ces champs est:

- **thisUpdate**: moment où l'on sait que le statut en cours d'indication est correct;
- **nextUpdate**: moment auquel ou avant lequel des informations plus récentes sont disponibles sur le statut du certificat;
- **producedAt:** moment auquel le tiers TTP du protocole OCSP a signé cette réponse.

Si le champ **nextUpdate** n'est pas réglé, le tiers TTP indique que des informations d'annulation plus récentes sont disponibles tout le temps.

9.1.5 Production préalable de réponse

Des tiers TTP du protocole OCSP peuvent produire au préalable des réponses signées spécifiant le statut des certificats à un moment spécifié. Le moment auquel on savait le statut correct doit être reflété dans le champ **thisUpdate** de la réponse. Le moment auquel ou avant lequel des informations plus récentes sont disponibles est reflété dans le champ **nextUpdate** tandis que le moment auquel la réponse a été produite apparaît dans le champ **producedAt** de la réponse.

9.1.6 Délégation d'autorité pour la signature OCSP

Il n'est pas nécessaire que la clé qui signe des informations de statut d'un certificat soit la même clé qui a signé le certificat. Un émetteur de certificat délègue de manière explicite l'autorité de signature OCSP en émettant un certificat qui contient une valeur unique pour **extendedKeyUsage** dans le certificat du signataire OCSP.

9.1.7 Altération de clé de CA

Si un tiers TTP du protocole OCSP sait qu'une clé privée d'une autorité de certification particulière a été altérée, il peut renvoyer l'état "annulé" pour tous les certificats émis par cette autorité de certification.

9.2 Prescriptions fonctionnelles

9.2.1 Contenu de certificat

Afin d'acheminer à des entités finales un point d'accès notoire aux informations, des autorités de certification doivent fournir la capacité d'inclure l'extension **AuthorityInfoAccess** (définie dans RFC 2459, § 4.2.2.1) dans des certifcats que l'on peut vérifier à l'aide du protocole OCSP. En variante, l'emplacement **accessLocation** du fournisseur de protocole OCSP peut être configuré localement au niveau de l'équipement des entités finales.

Les autorités de certification qui prennent en charge un service de protocole OCSP, soit hébergé localement soit fourni par un tiers TTP autorisé, peuvent fournir une valeur pour un emplacement **accessLocation** d'un indicateur **uniformResourceIndicator** (URI) et la valeur **id-ad-ocsp** de l'identificateur d'objet pour une méthode **accessMethod** dans la séquence **AccessDescription SEQUENCE**.

La valeur du champ **accessLocation** dans le certificat du titulaire définit le mécanisme de transport (par exemple HTTP) utilisé pour accéder au tiers TTP de protocole OCSP et peut contenir d'autres informations dépendant du transport (par exemple une adresse Web).

9.2.2 Prescriptions d'acceptation de réponse signée

Avant d'accepter une réponse signée comme étant valide, les entités finales doivent confirmer:

1) que le certificat identifié dans la réponse reçue correspond à celui qui a été identifié dans la demande correspondante;
2) que la signature figurant sur la réponse est valide;

3) que l'identité du signataire correspond à celle du destinataire prévu de la demande;

4) que le signataire est actuellement autorisé à signer la réponse;

5) que le moment auquel on sait que le statut indiqué est correct (thisUpdate) est suffisamment récent;

6) le cas échéant, que le temps auquel ou avant lequel des informations sont disponibles sur le statut du certificat (nextUpdate) est supérieur au temps actuel.

9.3 Protocole détaillé

La syntaxe ASN.1 importe des termes définis dans RFC 2459. Pour le calcul de signature, les données devant être signées sont codées à l'aide des règles de codage distinctives (DER, *distinguished encoding rules*) de l'ASN.1.

L'étiquetage **EXPLICIT** de l'ASN.1 est utilisé par défaut, sauf spécification contraire.

Les termes importés d'ailleurs sont: **Extensions**, **CertificateSerialNumber**, **SubjectPublicKeyInfo**, **Name**, **AlgorithmIdentifier**, **CRLReason**

9.3.1 Demandes

Le présent paragraphe spécifie la spécification ASN.1 pour une demande de confirmation.

9.3.1.1 Syntaxe de demande

```
OCSPRequest     ::=     SEQUENCE {
        tbsRequest              TBSRequest,
        optionalSignature   [0]  Signature OPTIONAL }

TBSRequest      ::=     SEQUENCE {
        version             [0]   Version DEFAULT v1,
        requestorName       [1]   GeneralName OPTIONAL,
        requestList               SEQUENCE OF Request,
        requestExtensions   [2]  Extensions OPTIONAL }

Signature       ::=SEQUENCE {
        signatureAlgorithm      AlgorithmIdentifier,
        signature               BIT STRING,
        certs               [0]  SEQUENCE OF Certificate
        OPTIONAL}

Version         ::=     INTEGER  {  v1(0) }

Request         ::=SEQUENCE {
        reqCert             CertID,
        singleRequestExtensions     [0]  Extensions OPTIONAL }

CertID          ::=SEQUENCE {
        hashAlgorithm           AlgorithmIdentifier,
        issuerNameHash          OCTET STRING, -- Hachage du DN de l'émetteur
        issuerKeyHash           OCTET STRING, -- Hachage de la clé publique des émetteurs
        serialNumber            CertificateSerialNumber }
```

issuerNameHash est le hachage du nom distinctif de l'émetteur. Le hachage doit être calculé sur le codage selon DER du champ nom de l'émetteur figurant dans le certificat en cours de vérification. **issuerKeyHash** est le hachage de la clé publique de l'Emetteur. Le hachage doit être calculé sur la valeur (étiquette et longueur non comprises) du champ clé publique du titulaire figurant dans le certificat de l'émetteur. L'algorithme de hachage utilisé pour ces deux hachages est identifié dans **hashAlgorithm**.

9.3.1.2 Remarques sur la syntaxe de demande

La raison principale de l'utilisation à la fois du nom et de la clé publique afin d'identifier l'émetteur est qu'il est possible que deux autorités de certification choisissent d'utiliser le même nom (l'unicité dans le nom est une recommandation que l'on ne peut pas faire respecter). Cependant, deux autorités de certification n'ont jamais la même clé publique sauf si elles décident de manière explicite de partager leur clé privée ou si la clé de l'une de ces autorités a été altérée.

La prise en charge de l'une quelconque des extensions spécifiques est facultative. Il ne convient pas que le fanion critique soit mis pour l'une quelconque d'entre elles. Le § 9.3.4 suggère plusieurs extensions utiles. Les extensions non reconnues doivent être ignorées (sauf si leur fanion critique est mis et qu'elles ne sont pas comprises).

Le demandeur peut choisir de signer la demande OCSP. Dans ce cas, la signature est calculée sur la structure **tbsRequest**. Si la demande est signée, le demandeur doit spécifier son nom dans le champ **requestorName**. Pour les demandes signées, le demandeur peut également inclure des certificats qui aident le tiers TTP du protocole OCSP à vérifier la signature du demandeur dans le champ certs de signature.

9.3.2 Syntaxe de réponse

Le présent paragraphe spécifie la spécification ASN.1 pour une réponse de confirmation.

9.3.2.1 Spécification de l'ASN.1 pour une réponse OCSP

Une réponse OCSP comprend au minimum un champ **responseStatus** qui indique le statut de traitement de la demande précédente. Si la valeur de **responseStatus** est l'une des valeurs de conditions d'erreur, les octets **responseBytes** ne sont pas mis.

```
OCSPResponse ::= SEQUENCE {
        responseStatus OCSPResponseStatus,
        responseBytes[0]  ResponseBytes OPTIONAL }

OCSPResponseStatus ::= ENUMERATED {
        successful          (0),  -- la réponse a des confirmations valides
        malformedRequest    (1),  -- demande de confirmation illégale
        internalError       (2),  -- erreur interne dans l'émetteur
        tryLater            (3),  -- réessayer plus tard
        --(4) n'est pas utilisé
        sigRequired         (5),  -- doit signer la demande
        unauthorized        (6)  -- demande non autorisée
}
```

La valeur des octets responseBytes est constituée d'un identificateur d'objet **OBJECT IDENTIFIER** et d'une syntaxe de réponse identifiée par cet identificateur d'objet OID codé comme une chaîne d'octets **OCTET STRING**:

```
ResponseBytes ::=SEQUENCE {
        responseType   OBJECT IDENTIFIER,
        response       OCTET STRING }
```

Pour un tiers TTP d'OCSP de base, **responseType** est **id-pkix-ocsp-basic**:

```
id-pkix-ocsp          OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-basic    OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }
```

Les tiers TTP de protocole OCSP doivent être capables de répondre en fournissant des réponses du type de réponse **id-pkix-ocsp-basic.** En conséquence, les entités finales doivent être capables de recevoir et traiter des réponses du type de réponse **id-pkix-ocsp-basic**.

La valeur pour la réponse doit être le codage selon DER de **BasicOCSPResponse**:

```
BasicOCSPResponse ::= SEQUENCE {
        tbsResponseData       ResponseData,
        signatureAlgorithm    AlgorithmIdentifier,
        signature             BIT STRING,
        certs                 [0]  SEQUENCE OF Certificate OPTIONAL
}
```

La valeur pour la signature doit être calculée sur le hachage du codage selon DER de **ResponseData**.

```
ResponseData ::= SEQUENCE {
        version             [0]  Version DEFAULT v1,
        responderID              ResponderID,
        producedAt               GeneralizedTime,
        responses                SEQUENCE OF SingleResponse,
        responseExtensions  [1]  Extensions OPTIONAL }

ResponderID ::= CHOICE {
        byName   [1] Name,
        byKey    [2] KeyHash }

KeyHash ::= OCTET STRING -- hachage SHA-1de la clé publique du tiers TTP (les champs tag et length non compris)
```

```
SingleResponse ::= SEQUENCE {
        certID CertID,
        certStatus CertStatus,
        thisUpdate GeneralizedTime,
        nextUpdate          [0] GeneralizedTime OPTIONAL,
        singleExtensions    [1] Extensions OPTIONAL }

CertStatus ::= CHOICE {
        good            [0]IMPLICIT NULL,
        revoked         [1]IMPLICIT RevokedInfo,
        unknown         [2]IMPLICIT UnknownInfo }

RevokedInfo ::= SEQUENCE {
        revocationTime GeneralizedTime,
        revocationReason[0] CRLReason OPTIONAL }

UnknownInfo ::= NULL
```

9.3.2.2 Remarques sur les réponses OCSP

9.3.2.2.1 **Time** (*heure*)

Les champs **thisUpdate** et **nextUpdate** définissent une période de validité recommandée. Cette période correspond à la période {**thisUpdate**, **nextUpdate**} figurant dans les listes CRL. Il convient de considérer non fiables les réponses dont la valeur de **nextUpdate** est plus ancienne que la valeur de l'heure locale du système.

Il convient de considérer non fiables les réponses dont l'heure **thisUpdate** est plus tardive que l'heure locale du système. Les réponses dont la valeur **nextUpdate** n'est pas réglée sont équivalentes à une liste CRL qui ne possède pas d'heure pour **nextUpdate** (se reporter au § 9.1.4).

L'heure **producedAt** est le moment où cette réponse a été signée.

9.3.2.2.2 Tiers de confiance autorisés (*authorized TTP*)

Il n'est pas nécessaire que la clé qui signe des informations de statut d'un certificat soit la même clé qui a signé le certificat. Un émetteur de certificat peut de manière explicite déléguer l'autorité de signature OCSP en émettant un certificat qui inclut une extension **extendedKeyUsage** dans le certificat du signataire OCSP qui contient la valeur **id-kp-OCSPSigning**.

```
id-kp-OCSPSigning OBJECT IDENTIFIER ::= {id-kp 9}
```

Etant donné qu'un tiers TTP d'OCSP autorisé fournit des informations de statut destinées à une autorité de certification, les entités finales ont besoin de savoir comment vérifier qu'un certificat d'un tiers TTP autorisé n'a pas été annulé. Les autorités de certification peuvent choisir de traiter ce problème de l'une des trois manières suivantes:

- une autorité de certification peut spécifier qu'une entité finale peut faire confiance à un tiers TTP pendant la durée de vie du certificat du tiers TTP. L'autorité de certification le fait en incorporant l'extension **id-pkix-ocsp-nocheck**. Il convient qu'il s'agisse d'une extension non critique. Il convient que la valeur de l'extension soit **NULL**. Il convient que les autorités de certification qui émettent un tel certificat se rendent compte qu'une altération de la clé de tiers TTP est aussi grave que l'altération de la clé de l'autorité de certification, au moins pendant la période de validité de ce certificat. Des autorités de certification peuvent choisir d'émettre ce type de certificat pour une durée de vie très courte et le renouveler fréquemment;

  ```
  id-pkix-ocsp-nocheck OBJECT IDENTIFIER ::= { id-pkix-ocsp 5 }
  ```

- une autorité de certification peut spécifier la manière de vérifier la révocation d'un certificat du tiers TTP. Cela peut être réalisé à l'aide des points de distribution de liste CRL s'il convient que la vérification soit effectuée à l'aide des listes CRL ou des points de distribution de liste CRL ou bien à l'aide d'accès à l'information d'autorité s'il convient que la vérification soit effectuée d'une certaine autre manière;

- une autorité de certification peut choisir de ne spécifier aucune méthode de vérification de l'annulation du certificat du tiers TTP, auquel cas il appartient à la politique de sécurité locale de l'entité finale de décider s'il convient de vérifier ou non l'annulation de ce certificat.

9.3.3 Algorithmes cryptographiques obligatoires et facultatifs

Des entités finales qui demandent des services OCSP doivent être capables de traiter des réponses signées en utilisant des clés DSA identifiées par le **sig-alg-oid** spécifié au § 7.2.2 de RFC 2459. Il convient que les entités finales soient également capables de traiter des signatures RSA telles que spécifiées au § 7.2.1 de RFC 2459. Les tiers TTP de protocole OCSP doivent prendre en charge l'algorithme de hachage SHA-1.

9.3.4 Extensions

Le présent paragraphe définit un certain nombre d'extensions standard. La prise en charge de toutes les extensions est facultative. Pour chaque extension, la définition indique sa syntaxe, le traitement réalisé par le tiers TTP du protocole OCSP et les éventuelles extensions qui sont incluses dans la réponse correspondante.

9.3.4.1 **Nonce** (mot créé pour la circonstance)

Le mot créé pour la circonstance (nonce) lie cryptographiquement une demande et une réponse afin de prévenir des attaques par redéfilement. Le mot créé est incorporé en tant qu'une des extensions **requestExtensions** dans les demandes, alors que dans les réponses il serait inclus en tant qu'une des extensions **responseExtensions**. Tant dans la demande que dans la réponse, le mot créé pour la circonstance est identifié par l'identificateur d'objet **id-pkix-ocsp-nonce**, **extnValue** étant la valeur du mot créé pour la circonstance.

```
id-pkix-ocsp-nonce     OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
```

9.3.4.2 Références de liste CRL

II peut être souhaitable que le tiers TTP du protocole OCSP indique la liste de révocation de certificat dans laquelle se trouve un certificat annulé ou **onHold (en suspens).** Ceci peut être utile lorsque le protocole OCSP est utilisé entre des dépôts et peut être également utilisé en tant que mécanisme d'audit. La liste CRL peut être spécifiée par une adresse Web (l'URL à laquelle la liste est disponible), par un numéro (numéro de CRL) ou par une heure (heure de la création de la liste CRL pertinente). Ces extensions sont spécifiées comme **singleExtensions**. L'identificateur de cette extension est **id-pkix-ocsp-crl** alors que la valeur est **CrlID**.

```
id-pkix-ocsp-crl        OBJECT IDENTIFIER ::= { id-pkix-ocsp 3 }

CrlID ::= SEQUENCE {
        crlUrl      [0]   IA5String OPTIONAL,
        crlNum      [1]   INTEGER OPTIONAL,
        crlTime     [2]   GeneralizedTime OPTIONAL }
```

Pour le choix **crlUrl**, la chaîne **IA5String** spécifie l'adresse Web à laquelle la liste CRL est disponible. Pour le choix **crlNum**, le nombre entier **INTEGER** spécifie la valeur de l'extension du numéro de liste CRL pour la liste CRL pertinente. Pour le choix **crlTime**, l'heure **GeneralizedTime** indique l'heure d'émission de la liste CRL pertinente.

9.3.4.3 Types de réponse acceptables

Une entité finale peut souhaiter spécifier les sortes de types de réponse qu'elle comprend. A cet effet, il convient qu'elle utilise une extension avec l'identificateur d'objet **id-pkix-ocsp-response** et la valeur **AcceptableResponses**. Les identificateurs d'objet inclus dans **AcceptableResponses** sont ceux des divers types de réponses que cette entité peut accepter (par exemple **id-pkix-ocsp-basic**).

```
id-pkix-ocsp-response  OBJECT IDENTIFIER ::= { id-pkix-ocsp 4 }

AcceptableResponses ::= SEQUENCE OF OBJECT IDENTIFIER
```

Comme il a été remarqué au § 9.3.2.1, les tiers TTP de protocole OCSP doivent être capables de répondre en fournissant des réponses du type de réponse **id-pkix-ocsp-basic**. En conséquence, les entités finales doivent être capables de recevoir et traiter des réponses du type de réponse **id-pkix-ocsp-basic**.

9.3.4.4 Limite d'archivage

Un tiers TTP de protocole OCSP peut choisir de conserver des informations d'annulation au-delà de l'expiration d'un certificat. La date obtenue en soustrayant cette valeur de période de rétention de l'heure **producedAt** figurant dans une réponse est définie comme la date de "limite d'archivage" du certificat.

Les applications permises par le protocole OCSP utilisent une date de limite d'archivage OCSP pour contribuer à la preuve qu'une signature numérique était (ou n'était pas) fiable à la date de sa création même si le certificat nécessaire à la validation de la signature a expiré longtemps après.

Il convient que les tiers TTP de protocole OCSP qui fournissent une prise en charge de telles références historiques incorporent une extension de date de limite d'archivage dans les réponses. Si elle est incluse, cette valeur doit être fournie en tant qu'extension **singleResponse** du protocole OCSP identifiée par la limite **id-pkix-ocsp-archive-cutoff** et ayant une syntaxe **GeneralizedTime**:

```
id-pkix-ocsp-archive-cutoff  OBJECT IDENTIFIER ::= { id-pkix-ocsp 6 }

ArchiveCutoff ::= GeneralizedTime
```

A titre d'illustration, si un tiers TTP est exploité avec une politique de période de rétention de sept ans et que le statut a été créé au temps t1, la valeur pour la limite **ArchiveCutoff** dans la réponse est de (t1 – 7 années).

9.3.4.5 Extensions d'entrée de liste CRL

Les extensions d'entrée de liste CRL sont également prises en charge en tant que **singleExtensions**.

9.3.4.6 Localisateur de service

Un tiers TTP de protocole OCSP peut être exploité dans un mode où le tiers TTP reçoit une demande et l'achemine à un tiers TTP de protocole OCSP dont on sait qu'il a autorité pour ce qui concerne le certificat identifié. L'extension de demande **serviceLocator** est définie dans ce but.

```
id-pkix-ocsp-service-locator OBJECT IDENTIFIER ::= { id-pkix-ocsp 7 }

ServiceLocator ::= SEQUENCE {
        issuer    Name,
        locator   AuthorityInfoAccessSyntax OPTIONAL }
```

Les valeurs pour ces champs sont tirées des champs correspondants dans le certificat du titulaire.

9.4 Module ASN.1 pour le protocole OCSP

```
OCSP DEFINITIONS EXPLICIT TAGS ::=

BEGIN

IMPORTS
-- Cadre d'information de l'annuaire (X.501) --
  Name
    FROM InformationFramework {
      joint-iso-itu-t ds(5) module(1) informationFramework(1) 3 }
  -- Cadre d'authentification de l'annuaire (X.509) --
  AlgorithmIdentifier, Certificate, CertificateSerialNumber,
  Extensions
    FROM AuthenticationFramework {
      joint-iso-itu-t ds(5) module(1) authenticationFramework(7) 3 }
  -- Extensions de certificat de l'annuaire (X.509) --
  CRLReason, GeneralName
    FROM CertificateExtensions {joint-iso-itu-t ds(5)
      module(1) certificateExtensions(26) 0 }
  -- PKIX (RFC 2459) --
  AuthorityInfoAccessSyntax
    FROM PKIX1Implicit93 {
      iso(1) identified-organization(3) dod(6) internet(1)
        security(5) mechanisms(5) pkix(7) id-mod(0)
          id-pkix1-implicit-93(4) }
  id-kp, id-ad-ocsp
    FROM PKIX1Explicit93 {
      iso(1) identified-organization(3) dod(6) internet(1)
        security(5) mechanisms(5) pkix(7) id-mod(0)
          id-pkix1-explicit-93(3) };

OCSPRequest    ::=SEQUENCE {
            tbsRequest      TBSRequest,
            optionalSignature  [0] Signature OPTIONAL }

TBSRequest   ::=SEQUENCE {
            version                 [0] Version DEFAULT v1,
            requestorName           [1] GeneralName OPTIONAL,
            requestList                 SEQUENCE OF Request,
            requestExtensions       [2] Extensions OPTIONAL }
```

```
Signature           ::=SEQUENCE {
        signatureAlgorithm   AlgorithmIdentifier,
        signature BIT STRING,
        certs   [0] SEQUENCE OF Certificate OPTIONAL }

Version  ::=  INTEGER  {  v1(0) }

Request ::=    SEQUENCE {
        reqCert     CertID,
        singleRequestExtensions   [0] Extensions OPTIONAL }

CertID ::= SEQUENCE {
        hashAlgorithm      AlgorithmIdentifier,
        issuerNameHash OCTET STRING, -- Hachage du DN de l'émetteur
        issuerKeyHash OCTET STRING, -- Hachage de la clé publique des émetteurs
        serialNumber CertificateSerialNumber }

OCSPResponse ::= SEQUENCE {
        responseStatus      OCSPResponseStatus,
        responseBytes       [0] ResponseBytes OPTIONAL }

OCSPResponseStatus ::= ENUMERATED {
        successful(0), -- la réponse a des confirmations valides
        malformedRequest(1), -- demande de confirmation illégale
        internalError(2), -- erreur interne dans l'émetteur
        tryLater(3), -- réessayer plus tard
        --(4) n'est pas utilisé
        sigRequired(5), -- doit signer la demande
        unauthorized(6) -- demande non autorisée
}

ResponseBytes ::=SEQUENCE {
        responseType   OBJECT IDENTIFIER,
        response       OCTET STRING }

BasicOCSPResponse ::= SEQUENCE {
        tbsResponseData ResponseData,
        signatureAlgorithm AlgorithmIdentifier,
        signature BIT STRING,
        certs   [0] SEQUENCE OF Certificate OPTIONAL
}

ResponseData ::= SEQUENCE {
        version                 [0] Version DEFAULT v1,
        responderID                 ResponderID,
        producedAt                  GeneralizedTime,
        responses                   SEQUENCE OF SingleResponse,
        responseExtensions  [1] Extensions OPTIONAL }

ResponderID ::= CHOICE {
        byName   [1] Name,
        byKey    [2] KeyHash }

KeyHash ::= OCTET STRING -- hachage SHA-1 de la clé publique du tiers TTP
        -- (les champs tag et length non compris)

SingleResponse ::= SEQUENCE {
        certID CertID,
        certStatus CertStatus,
        thisUpdate GeneralizedTime,
        nextUpdate[0] GeneralizedTime OPTIONAL,
        singleExtensions[1] Extensions OPTIONAL }

CertStatus ::= CHOICE {
        good        [0]     IMPLICIT NULL,
        revoked     [1]     IMPLICIT RevokedInfo,
        unknown     [2]     IMPLICIT UnknownInfo }

RevokedInfo ::= SEQUENCE {
        revocationTime   GeneralizedTime,
        revocationReason[0]         CRLReason OPTIONAL }

UnknownInfo ::= NULL
```

```
ArchiveCutoff ::= GeneralizedTime

AcceptableResponses ::= SEQUENCE OF OBJECT IDENTIFIER

ServiceLocator ::= SEQUENCE {
        issuer    Name,
        locator   AuthorityInfoAccessSyntax }

-- Identificateurs d'objets
id-kp-OCSPSigning           OBJECT IDENTIFIER ::= { id-kp 9 }
id-pkix-ocsp                OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-basic          OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }
id-pkix-ocsp-nonce          OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
id-pkix-ocsp-crl            OBJECT IDENTIFIER ::= { id-pkix-ocsp 3 }
id-pkix-ocsp-response       OBJECT IDENTIFIER ::= { id-pkix-ocsp 4 }
id-pkix-ocsp-nocheck        OBJECT IDENTIFIER ::= { id-pkix-ocsp 5 }
id-pkix-ocsp-archive-cutoff OBJECT IDENTIFIER ::= { id-pkix-ocsp 6 }
id-pkix-ocsp-service-locator OBJECT IDENTIFIER ::= { id-pkix-ocsp 7 }
END
```

Annexe A

Interfonctionnement

(Cette annexe ne fait pas partie intégrante de la présente Recommandation | Norme internationale

La fourniture de services de gestion de certificats peut exiger l'interfonctionnement de différents fournisseurs de services. Si l'on considère un ensemble d'autorités de certification, chacune gérant son service dans son propre domaine, les entités ont besoin d'avoir des clés publiques garanties d'autorités de certification appartenant à d'autres domaines. Il existe deux modèles de base possibles: le premier modèle est un modèle hiérarchique fondé sur des chaînes de certificats, et dans le second modèle, des autorités de certification peuvent se certifier réciproquement les unes les autres. Des modèles hybrides de ces deux modèles sont possibles.

Dans le premier modèle, les autorités sont placées sous une autorité CA "racine" qui émet des certificats aux autorités CA appelées subordonnées. Ces autorités CA peuvent émettre des certificats aux autres autorités CA situées hiérarchiquement sous elles ou aux entités.

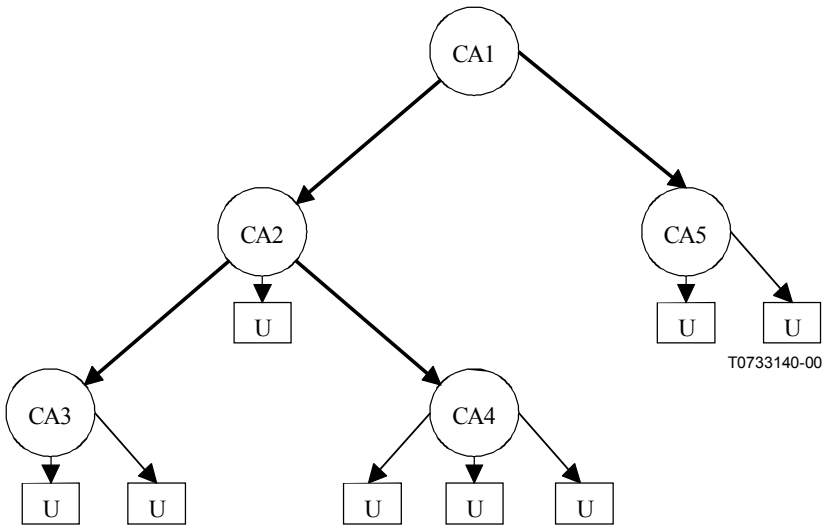

Figure A.1 – Hiérarchie d'autorités de certification (CA)

L'autorité CA1 est appelée CA "racine", se reporter à la Figure A.1. La tâche de l'autorité CA racine est d'enregistrer ses autorités CA subordonnées, dans la Figure A.1 ce sont les autorités CA2 et CA5. Les autorités CA subordonnées peuvent enregistrer d'autres CA ou des entités. Il convient que chaque autorité de certification CA fonctionne conformément à une politique commune, de manière à pouvoir obtenir un niveau commun de confiance ou de qualité de service.

Dans le second modèle, des autorités CA indépendantes se certifient réciproquement les unes les autres, ce qui aboutit à un réseau d'autorités CA interconnectées. La certification réciproque est un accord bilatéral entre deux autorités de certification, que l'une d'elles émette un certificat pour l'autre ou que toutes les deux émettent un certificat l'une pour l'autre.

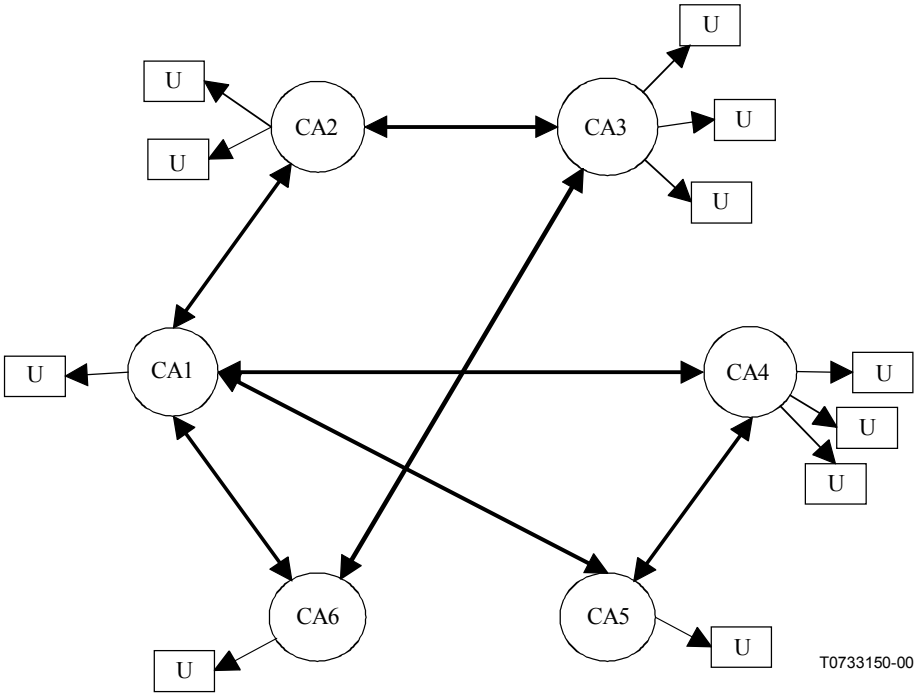

Figure A.2 – Exemple de réseau d'autorités de certification certifiées réciproquement

Annexe B

Algorithmes

(Cette annexe ne fait pas partie intégrante de la présente Recommandation | Norme internationale)

B.1 Algorithmes de hachage

Les algorithmes de hachage ci-après conviennent pour une utilisation dans le contexte de signatures numériques:

a) RIPEMD-160:

DOBBERTIN, BOSSELAERS (A.), PRENEEL (B.): *RIPEMD-160: A strengthened version of RIPEMD, Fast Software Encryption*, Cambridge Workshop 1996, LNCS, Band 1039, S. 71-82, Springer-Verlag, 1996.

b) SHA-1:

NIST: FIPS Publication 180-1: *Secure Hash Standard (SHS-1),* mai 1995.

Tous deux sont décrits dans:

ISO/CEI 10118-3:1998, *Technologies de l'information – Techniques de sécurité – Fonctions de brouillage – Partie 3: Fonctions de hachage dédiées*.

B.2 Algorithmes de signature digitale

Les algorithmes de signature digitale ci-après conviennent pour une utilisation dans le contexte de la présente Recommandation:

a) DSA:

NIST: FIPS Publication 186: *Digital Signature Standard (DSS)*, mai 1994.

b) RSA:

RIVEST, SHAMIR (A.), ADLEMAN (B.): A method for obtaining digital signatures and public key cryptosystems, *Communications of the ACM*, vol. 21, Nº 2, 1978.

c) Des schémas apparentés à DSA fondés sur des courbes elliptiques:

- ISO/CEI 14883-3, Annexe A.2.1 ("Elliptic Curve DSA");
- ISO/ CEI WD 15946-2;
- IEEE: Norme P1363 (Projet), 6 février 1997, § 5.3.3 ("Nyberg-Rueppel version");
- IEEE: Norme P1363 (Projet), 6 février 1997, § 5.3.4 ("DSA version").

Plusieurs algorithmes sont décrits dans:

ISO/CEI 14888-3:1998, *Technologies de l'information – Techniques de sécurité – Signatures digitales avec appendice – Partie 3: Mécanismes fondés sur certificat*.

Annexe C

Bibliographie

(Cette annexe ne fait pas partie intégrante de la présente Recommandation | Norme internationale)

COM (1997) 503, *"Ensuring Security and Trust in Electronic Communication", Communication from the Commission to the European Parliament, the Council, the Economic and Social Committee and the Committee of the Regions, octobre 1997*.

Directive 1999/93/EC du Parlement européen et du Conseil relative à un cadre communautaire pour les signatures électroniques du 13 décembre 1999.

ECBS TR 402, European Committee for Banking Standards, Technical Report 402: Certification Authorities, Vol. 1, 1997.

ETSI EG/SEC-003000 Requirements for Trusted Third Party Services (Edition 1), Version 7.0, juillet 1997.

FIPS PUB 140-1, *Federal Information Processing Standard Publication 140-1, "Security Requirements for Cryptographic Modules", U.S. Department of commerce, National Institute of Standards and Technology,* janvier 1994.

Recommandation UIT-T X.511 (1997) | ISO/CEI 9594-3:1998, *Technologies de l'information – Interconnexion des systèmes ouverts (OSI) – L'annuaire: définition de service abstrait.*

Recommandation UIT-T X.519 (1997) | ISO/CEI 9594-5:1998, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: spécification du protocole.*

Recommandation UIT-T X.810 (1995) | ISO/CEI 10181-1:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadre de sécurité pour les systèmes ouverts: aperçu général.*

Recommandation UIT-T X.811 (1995) | ISO/CEI 10181-2:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadre de sécurité pour les systèmes ouverts: cadre d'authentification.*

Recommandation UIT-T X.812 (1995) | ISO/CEI 10181-3:1996, *Technologies de l'information – Interconnexion des systèmes ouvert s– Cadre de sécurité pour les systèmes ouverts: cadre de contrôle d'accès.*

Recommandation UIT-T X.813 (1996) | ISO/CEI 10181-4:1997, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadre de sécurité pour les systèmes ouverts: non-répudiation.*

Recommandation UIT-T X.814 (1995) | ISO/CEI 10181-5:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadre de sécurité pour les systèmes ouverts: cadre de confidentialité.*

Recommandation UIT-T X.815 (1995) | ISO/CEI 10181-6:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadre de sécurité pour les systèmes ouverts: cadre d'intégrité.*

Recommandation UIT-T X.816 (1995) | ISO/CEI 10181-7:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadre de sécurité pour les systèmes ouverts: cadre d'audit et d'alarmes de sécurité.*

ISO/DIS 15782-1, *Gestion de certificats pour les services financiers – Partie 1: Certificats de clé publique.*

ISO/CEI 10118-1:1994*, Technologies de l'information – Techniques de sécurité – Fonctions de brouillage – Partie 1: Généralités.*

ISO/CEI 10118-2:1994*, Technologies de l'information – Techniques de sécurité – Fonctions de brouillage – Partie 2: Fonctions de brouillage utilisant un algorithme de chiffrement par blocs de n bits.*

ISO/CEI 10118-3:1998*, Technologies de l'information – Techniques de sécurité – Fonctions de brouillage – Partie 3: Fonctions de hachage dédiées.*

ISO 9735:1998, *Echange de données informatisées pour l'administration, le commerce et le transport (EDIFACT) – Règles de syntaxe au niveau de l'application.*

ISO/CEI 15408-1:1999, *Technologies de l'information – Techniques de sécurité – Critères d'évaluation pour la sécurité TI – Partie 1: Introduction et modèle général.*

ISO/CEI 15408-2:1999, *Technologies de l'information – Techniques de sécurité – Critères d'évaluation pour la sécurité TI – Partie 2: Exigences fonctionnelles de sécurité.*

ISO/CEI 15408-3:1999, *Technologies de l'information – Techniques de sécurité – Critères d'évaluation de la sécurité TI – Partie 3: Prescriptions d'assurance de sécurité.*

ITSEC, *Information Technology Security Evaluation Criteria (ITSEC), Harmonized Criteria of France, Germany, the Netherlands, the United Kingdom, Version 1.2,* juin 1992.

Minimum Interoperability Specification for PKI Components (MISPC), NIST Special Publication 800-15, septembre 1997.

Documents PKCS

PKCS #1, RSA Laboratories, PKCS #1: *RSA Encryption Standard,* Version 1.5, novembre 1993.

PKCS #10, RSA Laboratories, PKCS #10: *Certification Request Standard*, Version 1.5, novembre 1993.

PKCS #11, RSA Laboratories, PKCS #11: *Cryptographic Token Interface Standard*, Version 1.0, avril 1995.

PKCS #3, RSA Laboratories, PKCS #3: *Diffie-Hellman Key Agreement Standard,* Version 1.4, novembre 1993.

PKCS #5, RSA Laboratories, PKCS #5: *Password-Based Encryption Standard*, Version 1.5, novembre 1993.

PKCS #6, RSA Laboratories, PKCS #6: *Extended-Certificate Syntax Standard*, Version 1.5, novembre 1993.

PKCS #7, RSA Laboratories, PKCS #7: *Cryptographic Message Standard*, Version 1.5, novembre 1993, Extensions et revisions, 1997.

PKCS #8, RSA Laboratories, PKCS #8: *Private Key Information Syntax Standard*, Version 1.5, novembre 1993.

PKCS #9, RSA Laboratories, PKCS #9: *Selected Attribute Types*, Version 1.5, novembre 1993.

Documents PKIX

RFC 1421, *Privacy Enhancement for Electronic Mail: Part 1: Message Encryption and Authentication Procedures,* février 1993.

RFC 1422, *Privacy Enhancement for Electronic Mail: Part 2: Certificate-Based Key Management,* février 1993.

RFC 1423, *Privacy Enhancement for Electronic Mail: Part 3: Algorithms, Modes, and Identifiers,* février 1993.

RFC 1424, *Privacy Enhancement for Electronic Mail: Part 4: Key Certification and Related Services,* février 1993.

RFC 1510, *The Kerberos Network Authentication Services,* septembre 1993.

RFC 1750, *Randomness Recommendations for Security,* décembre 1994.

RFC 1766, *Tags for the Identification of Languages,* mars 1995.

RFC 2104, *Keyed-Hashing for Message Authentication,* février 1997.

RFC 2459, *Internet X.509 Public Key Infrastructure Certificate and CRL Profile,* janvier 1999.

RFC 2510, *Internet X.509 Public Key Infrastructure Certificate Management Protocols,* mars 1999.

RFC 2511, *Internet X.509 Public Key Infrastructure Certificate Request Message Format,* mars 1999.

RFC 2527, *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework,* mars 1999.

RFC 2559, *Internet X.509 Public Key Infrastrtucture, Operational Protocols – LDAPv2,* avril 1999.

RFC 2560, *Internet X.509 Public Key Infrastructure, Online Certificate Status Protocol – OCSP,* juin 1999.

Time Stamp Protocols, Internet Draft, septembre 1998 (en cours); Adams C., Cain P., Pinkas D., Zuccherato R.

SET Secure Electronic Transaction Specification, Book 1: Business Description, Version 1.0, 31 mai 1997.

SET Secure Electronic Transaction Specification, Book 2: Programmer's Guide, Version 1.0, 31 mai 1997.

SET Secure Electronic Transaction Specification, Book 3: Formal Protocol Definition, Version 1.0, 31 mai 1997.

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série B	Moyens d'expression: définitions, symboles, classification
Série C	Statistiques générales des télécommunications
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	Systèmes audiovisuels et multimédias
Série I	Réseau numérique à intégration de services
Série J	Réseaux câblés et transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	RGT et maintenance des réseaux: systèmes de transmission, circuits téléphoniques, télégraphie, télécopie et circuits loués internationaux
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux
Série Q	Commutation et signalisation
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	Terminaux des services télématiques
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	**Réseaux de données et communication entre systèmes ouverts**
Série Y	Infrastructure mondiale de l'information et protocole Internet
Série Z	Langages et aspects généraux logiciels des systèmes de télécommunication

Genève, 2002