Groupe de travail Réseau
Request for Comments : 5280
RFC rendues obsolètes : 3280, 4325, 4630
Catègorie : Sur la voie de la normalisation
mai 2008
Traduction Claude Brière de L'Isle

D. Cooper, NIST
S. Santesson, Microsoft
S. Farrell, Trinity College Dublin
S. Boeyen, Entrust
R. Housley, Vigil Security
W. Polk, NIST

Profil de certificat d'infrastructure de clé publique X.509 et de liste de révocation de certificat (CRL) pour l'Internet

Statut du présent mémoire

Le présent document spécifie un protocole Internet sur la voie de la normalisation pour la communauté de l'Internet. Il appelle à la discussion et à des suggestions pour son amélioration. Prière de se référer à l'édition actuelle des "Normes officielles des protocoles de l'Internet" (STD 1) pour connaître l'état de normalisation et le statut de ce protocole. La distribution du présent mémoire n'est soumise à aucune restriction.

Résumé

Le présent mémoire fait le profil du certificat X.509v3 et de la liste de révocation de certificat X.509 v2 (CRL) à utiliser dans l'Internet. Une vue générale de cette approche et de ce modèle est fournie en introduction. Le format du certificat X.509 v3 est décrit en détail, avec des informations complémentaires concernant le format et la sémantique des formes de nom Internet. Les extensions de certificat standard sont décrites et deux extensions spécifiques de l'Internet sont définies. Un ensemble d'extensions de certificat exigées est spécifié. Les formats de CRL X.509 v2 sont décrits en détail ainsi que les extensions standard et spécifiques de l'Internet. Un algorithme est décrit pour la validation de chemin de certification X.509. Un module ASN.1 et des exemples sont fournis dans les appendices.

Table des matières

1. Introduction

La présente spécification fait partie d'une famille de normes pour l'infrastructure de clé publique X.509 (PKI, *Public Key Infrastructure*) pour l'Internet.

La présente spécification fait le profil du format et de la sémantique des certificats et des listes de révocation de certificat (CRL, *Certificate Revocation List*) pour la PKI Internet. Les procédures sont décrites pour le traitement de la certification de chemins dans l'environnement de l'Internet. Finalement, les modules ASN.1 sont fournis dans les appendices pour toutes les structures de données définies ou référencées.

La Section 2 décrit les exigences de la PKI Internet et les hypothèses qui affectent la portée de ce document. La Section 3 présente un modèle d'architecture et décrit ses relations avec les normes précédentes de l'IETF et de l'ISO/CEI/UIT-T. En particulier, sont décrites les relations du présent document avec les spécifications PEM de l'IETF et les documents ISO/CEI/UIT-T X.509.

La Section 4 fait le profil du certificat X.509 version 3, et la Section 5 fait le profil du CRL X.509 version 2. Les profils incluent l'identification des extensions ISO/CEI/UIT-T et ANSI qui peuvent être utiles dans la PKI Internet. Les profils sont présentés dans la notation numéro un de syntaxe abstraite (ASN.1, *Abstract Syntax Notation One*) de 1988 plutôt que dans la syntaxe ASN.1 de 1997 utilisée dans les plus récentes normes ISO/CEI/UIT-T.

La Section 6 inclut les procédures de validation de chemin de certification. Ces procédures se fondent sur la définition de l'ISO/CEI/UIT-T. Il est EXIGÉ des mises en œuvre qu'elles déduisent les mêmes résultats mais il n'est pas exigé qu'elles utilisent les procédures spécifiées.

Les procédures pour l'identification et le codage des matériaux de clé publique et les signatures numériques sont définis dans les [RFC3279], [RFC4055], et [RFC4491]. Les mises en œuvre de la présente spécification ne sont pas obligées d'utiliser d'algorithmes de chiffrement particuliers. Cependant, les mises en œuvre conformes qui utilisent les algorithmes identifiés dans les [RFC3279], [RFC4055], et [RFC4491] DOIVENT identifier et coder les matériaux de clé publique et les signatures numériques comme décrit dans ces spécifications.

Finalement, trois appendices sont fournis pour aider à la mise en œuvre. L'Appendice A contient toutes les structures ASN.1 définies ou référencées au sein de la présente spécification. Comme ci-dessus, le matériel est présenté dans l'ASN.1 1988. L'Appendice B contient des notes sur les caractéristiques moins familières de la notation ASN.1 utilisées dans la présente spécification. L'Appendice C contient des exemples de certificats conformes et d'une CRL conforme.

La présente spécification rend obsolète la [RFC3280]. Les différences par rapport à la RFC3280 sont résumées ci-dessous :

* L'amélioration de la prise en charge des noms internationalisés est spécifiée à la Section 7, avec les règles de codage et la comparaison des noms de domaine internationalisés, des identifiants de ressource internationalisés (IRI), et des noms distinctifs. Ces règles sont alignées sur les règles de comparaison établies dans les RFC actuelles, incluant les [RFC3490], [RFC3987], et [RFC4518].
* Les paragraphes 4.1.2.4 et 4.1.2.6 incorporent les conditions pour la poursuite de l'utilisation des schémas traditionnels de codage de texte qui ont été spécifiés dans la [RFC4630]. Lorsque elles sont utilisées par une PKI établie, la transition à UTF8String pourrait causer un déni de service fondé sur l'échec du chaînage de noms ou un traitement incorrect des contraintes de dénomination.
* Le paragraphe 4.2.1.4 de la RFC3280, qui spécifiait l'extension de certificat privateKeyUsagePeriod mais déconseillait son utilisation, a été supprimé. L'utilisation de cette extension de la norme ISO n'est ni déconseillée ni recommandée dans la PKI Internet.

* Le paragraphe 4.2.1.5 recommande de marquer l'extension Transpositions de politique comme critique. La RFC3280 exigeait que l'extension Transpositions de politique soit marquée comme non critique.
* Le paragraphe 4.2.1.11 exige le marquage de l'extension Contraintes de politique comme critique. La RFC3280 permettait que l'extension Contraintes de politique soit marquée comme critique ou non critique.
* L'extension de CRL Accès aux informations d'autorité (AIA, *Authority Information Access*) comme spécifiée dans la [RFC4325], a été ajoutée au paragraphe 5.2.7.
* Les paragraphes 5.2 et 5.3 précisent respectivement les règles de traitement des extensions de CRL non reconnues et d'entrée de CRL.
* Le paragraphe 5.3.2 de la RFC3280, qui spécifiait l'extension d'entrée de CRL holdInstructionCode, a été supprimé.
* L'algorithme de validation de chemin spécifié à la Section 6 ne retrace plus le caractère critique des extensions de politiques de certificat dans une chaîne de certificats. Dans la RFC3280, ces informations étaient retournées à un consommateur d'assertions.
* La Section Considérations pour la sécurité vise le risque de dépendance circulaire provenant de l'utilisation de https ou de schémas similaires dans les points de distribution de CRL, les accès d'informations d'autorité, ou les extensions d'accès d'informations de sujet.
* La Section Considérations pour la sécurité traite des risques associés aux noms ambigus.
* La Section Considérations pour la sécurité fait référence à la RFC4210 pour les procédures de signalisation de changements dans le fonctionnement de CA.

Les modules ASN.1 dans l'Appendice A sont inchangés depuis la RFC3280, excepté que ub-emailaddress-length a été changé de 128 à 255 afin de l'aligner sur PKCS n° 9 [RFC2985].

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" dans le présent document sont à interpréter comme décrit dans la [RFC2119].

2. Exigences et hypothèses

Le but de la présente spécification est de développer un profil pour faciliter l'utilisation des certificats X.509 dans les applications Internet pour les communautés qui souhaitent utiliser la technologie X.509. De telles applications peuvent inclure la Toile mondiale, la messagerie électronique, l'authentification d'utilisateur, et IPsec. Afin de surmonter certains des obstacles à l'utilisation des certificats X.509, le présent document définit un profil pour promouvoir le développement des systèmes de gestion de certificat, le développement des outils d'application, et l'interopérabilité déterminée par la politique.

Certaines communautés auront besoin de compléter, ou éventuellement remplacer, ce profil afin de satisfaire aux exigences de domaines d'application ou environnements spécialisés avec des autorisations, des assurances, ou des exigences de fonctionnement supplémentaires. Cependant, pour les applications de base, les représentations courantes des attributs fréquemment utilisés sont définies afin que les développeurs d'applications puissent obtenir les informations nécessaires sans considération de la production d'un certificat particulier ou de liste de révocation de certificat (CRL).

Un utilisateur de certificat devrait revoir la politique de certificats générée par l'autorité de certification (CA, C*ertification Authority*) avant de faire confiance aux services d'authentification ou de non répudiation associés à la clé publique dans un certificat particulier. À cette fin, la présente norme ne prescrit pas de règle ou obligation légalement contraignante.

Au fur et à mesure qu'émergent des outils supplémentaires d'autorisation et de gestion d'attribut, comme les certificats d'attribut, il peut être approprié de limiter les attributs authentifiés qui sont inclus dans un certificat. Ces autres outils de gestion peuvent fournir des méthodes plus appropriées pour convoyer de nombreux attributs authentifiés.

2.1 Communication et topologie

Les utilisateurs de certificats vont fonctionner dans une large gamme d'environnements par rapport à leur topologie de communication, en particulier les utilisateurs de messagerie électronique sécurisée. Le présent profil prend en charge les utilisateurs qui n'ont pas le haut débit, la connexité IP en temps réel, ou une forte disponibilité de connexion. De plus, le profil permet la présence de pare-feu ou autres filtrages de communication.

Le présent profil ne suppose pas le déploiement d'un système de répertoire [X.500] ou d'un système de répertoire du protocole léger d'accès à un répertoire (LDAP, *Lightweight Directory Access Protocol*) [RFC4510]. Le profil n'interdit pas l'utilisation d'un répertoire X.500 ou d'un répertoire LDAP ; cependant, tout moyen de distribution de certificats et de listes de révocation de certificats (CRL) peut être utilisé.

2.2 Critères d'acceptabilité

Le but de l'infrastructure de clé publiques Internet (PKI, *Internet Public Key Infrastructure*) est de satisfaire les besoins des fonctions déterministes automatisées d'identification, d'authentification, de contrôle d'accès et d'autorisation. La prise en charge de ces services détermine les attributs contenus dans le certificat ainsi que les informations auxiliaires de contrôle dans le certificat, telles que les données de politique et les contraintes de chemin de certification.

2.3 Attentes de l'utilisateur

Les utilisateurs de la PKI Internet sont des gens et des processus qui utilisent un logiciel client et sont les sujets désignés dans les certificats. Ces utilisations incluent des lecteurs et des rédacteurs de messagerie électronique, les clients des navigateurs de la Toile mondiale, des serveurs de la Toile mondiale, et le gestionnaire de clé pour IPsec au sein d'un routeur. Le présent profil reconnaît les limitations des plate-formes qu'emploient ces utilisateurs et les limitations de la sophistication et de l'attention des utilisateurs eux-mêmes. Cela se manifeste par une responsabilité minimale de configuration par l'utilisateur (par exemple, clés de CA de confiance, règles) par des contraintes explicites d'utilisation de plate-forme au sein du certificat, des contraintes de chemin de certification qui protègent l'utilisateur contre de nombreuses actions malveillantes, et des applications qui automatisent sensiblement les fonctions de validation.

2.4 Attentes de l'administrateur

Comme avec les attentes de l'utilisateur, le profil de PKI Internet est structuré pour prendre en charge les individus qui font généralement fonctionner les CA. Fournir aux administrateurs des choix non limités augmente les chances qu'une faute subtile d'un administrateur de CA résulte en une large compromission. Aussi, les choix non limités compliquent beaucoup le logiciel qui traite et valide les certificats créés par la CA.

3. Généralités et approche

Ce qui suit est une vue simplifiée du modèle architectural supposé pour l'infrastructure de clé publique qui utilise les spécifications X.509 (PKIX).

Les composants de ce modèle sont :

entité d'extrémité : utilisateur des certificats PKI et/ou système utilisateur d'extrémité qui est le sujet d'un certificat ;

CA : autorité de certification ;

RA : autorité d'enregistrement, c'est-à-dire, un système facultatif auquel une CA délègue certaines fonctions de gestion ;

producteur de CRL : système qui génère et signe les CRL ;

dépôt : système ou collection de systèmes répartis qui mémorise les certificats et CRL et sert de moyen de distribution de ces certificats et CRL aux entités d'extrémité.

Les CA sont chargées d'indiquer le statut de révocation des certificats qu'elles délivrent. Les informations sur le statut de révocation peuvent être fournies en utilisant le protocole de statut de certificat en ligne (OCSP, *Online Certificate Status Protocol*) [RFC2560], les listes de révocation de certificat (CRL, *certificate revocation list*) ou d'autres mécanismes. En général, lorsque les informations de statut de révocation sont fournies en utilisant les CRL, la CA est aussi le producteur de la CRL. Cependant, une CA peut déléguer la responsabilité de produire les CRL à une entité différente.

Noter qu'une autorité d'attribut (AA, *Attribute Authority*) peut aussi choisir de déléguer la publication des CRL à un producteur de CRL.

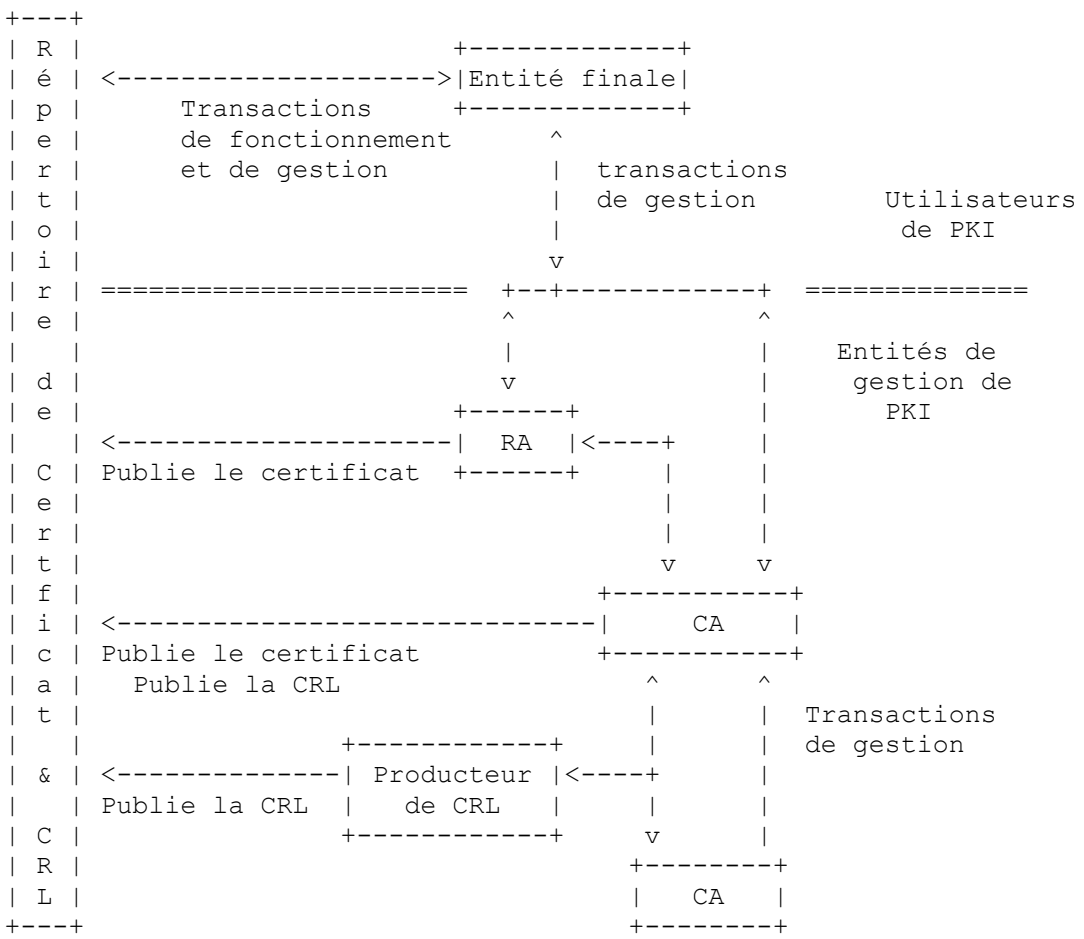


Figure 1 : Entités PKI

3.1 Certificat X.509 version 3

Les utilisateurs d'une clé publique exigent de pouvoir se fier au fait que la clé privée associée est possédée par le sujet (personne ou système) distant correct avec lequel un mécanisme de chiffrement ou de signature numérique sera utilisé. Cette confiance est obtenue grâce à l'utilisation de certificats de clé publique, qui sont des structures de données qui lient les valeurs de clé publique aux sujets. Le lien est affirmé par la signature numérique d'une CA de confiance sur chaque certificat. La CA peut fonder cette assertion sur des moyens techniques (autrement dit, une preuve de possession par un protocole de mise au défi - réponse) la présentation de la clé privée, ou sur une assertion par le sujet. Un certificat a une durée de validité limitée, qui est indiquée dans son contenu signé. Parce que la signature d'un certificat et sa pertinence temporelle peuvent être vérifiées de façon indépendante par un client utilisateur de certificat, les certificats peuvent être distribués via des communications et systèmes serveurs qui ne sont pas de confiance, et peuvent être mis en antémémoire dans une mémorisation non sûre dans les systèmes qui utilisent des certificats.

La Recommandation UIT-T X.509 (anciennement CCITT X.509) ou ISO/CEI 9594-8, qui a été publiée d'abord en 1988 au titre des recommandations de répertoire X.500, définit un format de certificat standard [X.509]. Le format de certificat dans la norme de 1988 est appelé le format de version 1 (v1). Lorsque X.500 a été révisée en 1993, deux champs ont été ajoutés, résultant en le format de version 2 (v2).

Les RFC sur la messagerie Internet à confidentialité améliorée (PEM, *Internet Privacy Enhanced Mail*) publiées en 1993, incluent des spécifications pour une infrastructure de clé publique fondée sur les certificats X.509 v1 [RFC1422]. L'expérience obtenue par les tentatives de déploiement de la RFC1422 montrait clairement que les formats de certificat v1 et v2 étaient déficients à plusieurs égards. Plus important, il fallait d'autres champs pour porter les informations que la conception de PEM et l'expérience de mise en œuvre avaient prouvées nécessaires. En réponse à ces nouvelles exigences, l'ISO/CEI, l'UIT-T, et l'ANSI X9 ont développé le format de certificat X.509 version 3 (v3). Le format v3 étend le format v2 en ajoutant des dispositions pour des champs d'extension supplémentaires. Les types de champs d'extension particuliers peuvent être spécifiés dans des normes ou peuvent être définis et enregistrés par toute organisation ou communauté. En juin 1996, la normalisation du format v3 de base a été achevée [X.509].

L'ISO/CEI, l'UIT-T, et l'ANSI X9 ont aussi développé des extensions standard à utiliser dans les champs d'extensions de la version 3 [X.509], [X9.55]. Ces extensions peuvent porter de telles données comme informations supplémentaires d'identification de sujet, informations d'attribut de clé, informations de politique, et contraintes de chemin de certification.

Cependant, les extensions standard de l'ISO/CEI, de l'UIT-T, et de l'ANSI X9 sont d'applicabilité trés large. Afin de développer des mises en œuvre interopérables des systèmes X.509 v3 pour l'usage de l'Internet, il est nécessaire de spécifier un profil pour l'usage des extensions X.509 v3 imaginées pour l'Internet. C'est un des buts du présent document de spécifier un profil pour la Toile de l'Internet, la messagerie électronique, et les applications IPsec. Les environnements qui ont des exigences supplémentaires peuvent s'appuyer sur le présent profil ou peuvent le remplacer.

3.2 Chemins de certification et confiance

Un utilisateur d'un service de sécurité qui exige la connaissance d'une clé publique a généralement besoin d'obtenir et valider un certificat contenant la clé publique requise. Si l'utilisateur de clé publique ne détient pas déjà une copie sûre de la clé publique de la CA qui a signé le certificat, le nom de la CA, et les informations qui s'y rapportent (comme la période de validité ou les contraintes de nom) il peut alors avoir besoin d'un certificat supplémentaire pour obtenir cette clé publique. En général, une chaîne de plusieurs certificats peut être nécessaire, comportant un certificat du propriétaire de la clé publique (l'entité d'extrémité) signé par une CA, et zéro, un ou plusieurs certificats supplémentaires de CA signés par d'autres CA. De telles chaînes, appelées chemins de certification, sont requises parce que un utilisateur de clé publique est seulement initialisé avec un nombre limité de clés publiques garanties par une CA.

Les CA peuvent être configurées de différentes façons afin que l'utilisateur de clé publiques soit capable de trouver les chemins de certification. Pour PEM, la RFC 1422 a défini une structure hiérarchique rigide des CA. Il y a trois types d'autorité de certification PEM :

(a) Autorité d'enregistrement de politique Internet (IPRA, *Internet Policy Registration Authority*) : cette autorité, qui fonctionne sous les auspices de la Internet Society, agit comme racine de la hiérarchie de certification PEM au niveau 1. Elle ne produit de certificats que pour le niveau d'autorité suivant, les CA. Tous les chemins de certification commencent par l'IPRA.

(b) Autorités de certification de politique (PCA, *Policy Certification Authorities*) : les PCA sont au niveau 2 de la hiérarchie, chaque PCA étant certifiée par l'IPRA. Une PCA doit établir et publier une déclaration de sa politique à l'égard des utilisateurs de certification ou des autorités de certification subordonnées. Des PCA distinctes visent à satisfaire des besoins d'utilisateur différents. Par exemple, une PCA (organisationnelle) peut prendre en charge les besoins généraux de messagerie électronique des organisations commerciales, et une autre PCA (de haute sécurité) peut avoir une politique plus stricte conçue pour satisfaire des exigences de signature numérique qui ont force légale.

(c) Autorités de certification (CA, *Certification Authorities*) : les CA sont au niveau 3 de la hiérarchie et peuvent aussi être à des niveaux inférieurs. Celles qui sont au niveau 3 sont certifiées par les PCA. Les CA représentent, par exemple, des organisations particulères, des unités d'organisation particulières (par exemple, des départements, des groupes, des sections) ou des zones géographiques particulières.

La RFC 1422 a de plus une règle de subordination de nom, qui exige qu'une CA puisse seulement produire des certificats pour des entités dont le nom est subordonné (dans l'arborescence de noms X.500) au nom de la CA elle-même. La confiance associée à un chemin de certifiation PEM est impliquée par le nom de PCA. La règle de subordination de nom assure que les CA en dessous de la PCA sont sensiblement contraintes à l'ensemble d'entités subordonnées qu'elles peuvent certifier (par exemple, une CA pour une organisation peut seulement certifier les entités dans l'arborescence de noms de cette organisation). Les systèmes d'utilisateur de certificat sont capables de vérifier mécaniquement que la règle de subordination de nom a été suivie.

La RFC 1422 utilise le format de certificat X.509 v1. Les limitations de X.509 v1 imposent plusieurs restrictions structurelles pour associer clairement les informations de politique ou restreindre l'utilité des certificats. Ces restrictions incluent :
(a) une pure hiérarchie descendante, avec tous les chemins de certifications commençant à la IPRA,
(b) une règle de subordination de désignation restreignant les noms des sujets d'une CA, et
(c) l'utilisation du concept de PCA, qui exige que la connaissance des PCA individuelles soit incorporée dans la logique de vérification de la chaîne de certificats. La connaissance des PCA individuelles était requise pour déterminer si une chaîne pouvait être acceptée.

Avec X.509 v3, la plupart des exigences traitées par la RFC 1422 peuvent l'être en utilisant des extensions de certificat, sans qu'il soit besoin de restreindre les structures de CA utilisées. En particulier, les extensions de certificat relatives aux

politiques de certificat suppriment le besoin des PCA et les extensions de contraintes suppriment le besoin de la règle de subordination de nom. Par suite, le présent document prend en charge une architecture plus souple, incluant :

(a) Les chemins de certification commencent par une clé publique d'une CA dans le domaine propre d'un utilisateur, ou avec la clé publique du sommet d'une hiérarchie. Commencer par la clé publique d'une CA dans le propre domaine d'un utilisateur présente certains avantages. Dans certains environnements, le domaine local est le plus digne de confiance.

(b) Les contraintes de nom peuvent être imposées par l'inclusion explicite d'une extension de contrainte de nom dans un certificat, mais ne sont pas exigées.

(c) Les extensions et transpositions de politique remplacent le concept de PCA, ce qui permet un degré supérieur d'automatisation. L'application peut déterminer si le chemin de certification est acceptable sur la base du contenu des certificats au lieu d'une connaissance a priori des PCA. Cela permet l'automatisation du traitement du chemin de certification.

X.509 v3 inclut aussi une extension qui identifie le sujet d'un certificat comme étant soit une CA, soit une entité d'extrémité, réduisant la dépendance aux informations hors bande demandées dans PEM.

La présente spécification couvre deux classes de certificats : les certificats de CA et les certificats d'entité d'extrémité. Les certificats de CA peuvent être redivisés en trois classes : les certificats croisés, les certificats auto produits, et les certificats auto signés. Les certificats croisés sont des certificats de CA dans lesquels le producteur et le sujet sont des entités différentes. Les certificats croisés décrivent une relation de confiance entre les deux CA. Les certificats auto produits sont des certificats de CA dans lesquels le producteur et le sujet sont la même entité. Les certificats auto produits sont générés pour prendre en charge des changements de politique ou de fonctionnement. Les certificats auto signés sont des certificats auto produits dans lesquels la signature numérique peut être vérifiée par la clé publique liée au certificat. Les certificats auto signés sont utilisés pour porter une clé publique à utiliser pour commencer le chemin de certification. Les certificats d'entité d'extrémité sont produits aux sujets qui ne sont pas autorisés à produire des certificats.

3.3 Révocation

Quand un certificat est produit, on s'attend à ce qu'il soit utilisé pendant toute sa période de validité. Cependant, diverses circonstances peuvent causer l'invalidité d'un certificat avant l'arrivée à expiration de sa période de validité. De telles circonstances incluent le changement de nom, le changement de l'association entre le sujet et la CA (par exemple, un employé quitte son emploi dans une organisation) et la compromission ou la présomption de compromission de la clé privée correspondante. Dans ces circonstances, la CA doit révoquer le certificat.

X.509 définit une méthode de révocation de certificat. Cette méthode implique que chaque CA produise périodiquement une structure de données signée appelée une liste de révocation de certificat (CRL, C*ertificate Revocation List*). Une CRL est une liste horodatée qui identifie les certificats révoqués et qui est signée par une CA ou par le producteur de CRL et est librement disponible dans un répertoire public. Chaque certificat révoqué est identifié dans une CRL par son numéro de série de certificat. Quand un système utilisateur de certificats utilise un certificat (par exemple, pour vérifier la signature numérique d'un utilisateur distant) ce système non seulement vérifie la signature et la validité du certificat mais aussi acquiert une CRL suffisamment récente et vérifie que le numéro de série du certificat n'est pas dans cette CRL. La signification de "suffisamment récente " peut varier selon la politique locale, mais signifie généralement la CRL la plus récemment publiée. Une nouvelle CRL est produite sur une base périodique régulière (par exemple, toutes les heures, jours ou semaines). Une entrée est ajoutée à la CRL au titre de la prochaine mise à jour qui suit la notification de révocation. Une entrée NE DOIT PAS être retirée de la CRL jusqu'à ce qu'elle apparaisse sur une CRL régulièrement programmée produite au delà de la période de validité du certificat révoqué.

Un avantage de cette méthode de révocation est que ces CRL peuvent être distribuées par exactement les mêmes moyens que les certificats eux-mêmes, à savoir via des serveurs et des communications qui ne sont pas de confiance.

Une limitation de la méthode de révocation par CRL, qui utilise des communications et serveurs qui ne sont pas de confiance, est que la granularité des révocations est limitée à la période de production des CRL. Par exemple, si une révocation est rapportée maintenant, elle ne sera fiablement notifiée aux systèmes qui utilisent des certificats que lorsque toutes les CRL actuellement produites seront mises à jour, ce qui peut prendre une heure, un jour ou une semaine selon la fréquence de production de ces CRL.

Comme avec le format de certificat X.509 v3, afin de faciliter l'interopérabilité des mises en œuvre de différents fabricants, le format de CRL X.509 v2 doit être profilé pour l'usage de l'Internet. C'est un des buts de ce document de spécifier ce profil. Cependant, le présent profil n'exige pas la production des CRL. Les formats de message et les protocoles qui prennent en charge la notification en ligne de la révocation sont définis dans d'autres spécifications PKIX. Les méthodes de

notification de révocation en ligne peuvent être applicables dans certains environnements comme solution de remplacement à la CRL X.509. La vérification de révocation en ligne peut significativement réduire la latence entre un rapport de révocation et la distribution des informations aux parties qui s'appuient sur elles. Une fois que la CA a accepté un rapport de révocation comme authentique et valide, toute interrogation au service en ligne va correctement refléter les impacts de la révocation sur la validation du certificat. Cependant, ces méthodes imposent de nouvelles exigences de sécurité : le valideur du certificat doit faire confiance au service de validation en ligne alors que le répertoire n'a pas besoin d'être de confiance.

3.4 Protocoles de fonctionnement

Des protocoles de fonctionnement sont requis pour livrer les certificats et les CRL (ou les informations d'état) aux systèmes clients utilisateurs de certificats. Des dispositions sont nécessaires pour divers moyens différents de livraison de certificat et CRL, incluant des procédures de distribution fondées sur LDAP, HTTP, FTP, et X.500. Les protocoles de fonctionnement qui prennent en charge ces fonctions sont définis dans d'autres spécifications PKIX. Ces spécifications peuvent inclure des définitions de formats et procédures de message pour la prise en charge de tous les environnements de fonctionnement ci-dessus, incluant la définition ou la référence aux types de contenu MIME appropriés.

3.5 Protocoles de gestion

Des protocoles de gestion sont requis pour prendre en charge les interactions en ligne entre l'utilisateur de PKI et les entités de gestion. Par exemple, un protocole de gestion peut être utilisé entre une CA et un système client auquel est associé une paire de clés, ou entre deux CA qui s'inter-certifient l'une l'autre. L'ensemble de fonctions qui ont potentiellement besoin d'être prises en charge par les protocoles de gestion inclut :

(a) enregistrement : c'est le processus par lequel un utilisateur se fait d'abord connaître d'une CA (directement, ou par une autorité d'enregistrement) avant que cette CA produise un ou des certificats pour cet utilisateur.

(b) initialisation : avant qu'un système client puisse fonctionner en toute sécurité, il est nécessaire d'installer des matériaux de clés qui ont les relations appropriées avec les clés mémorisées ailleurs dans l'infrastructure. Par exemple, le client a besoin d'être initialisé de façon sûre avec la clé publique et autres informations assurées de la ou des CA de confiance, pour être utilisées à valider les chemins de certification. De plus, un client a normalement besoin d'être initialisé avec sa ou ses propres paires de clés.

(c) certification : c'est le processus dans lequel une CA produit un certificat pour la clé publique d'un utilisateur, et retourne ce certificat au système client de l'utilisateur et/ou envoie ce certificat dans un répertoire.

(d) récupération de paire de clés : en option, les matériaux de clé de client d'utilisateur (par exemple, la clé privée d'un utilisateur utilisée pour le chiffrement) peuvent être sauvegardés par une CA ou un système de sauvegarde de clés. Si un usager a besoin de récupérer les matériaux de clé sauvegardés (par exemple, par suite d'un mot de passe oublié ou de la perte d'un fichier de chaîne de clés) un échange de protocole en ligne peut être nécessaire pour prendre en charge une telle récupération.

(e) mise à jour de paire de clés : toutes les paires de clé doivent être mises à jour régulièrement, c'est-à-dire, remplacées par une nouvelle paire de clés, et de nouveaux certificats produits.

(f) demande de révocation : une personne autorisée avise une CA d'une situation anormale exigeant la révocation du certificat.

(g) certification croisée : deux CA échangent les informations utilisées dans l'établissement d'un certificat croisé. Un certificat croisé est un certificat produit par une CA à une autre CA qui contient une clé de signature de CA utilisée pour produire des certificats.

Noter que les protocoles en ligne ne sont pas la seule façon de mettre en œuvre ces fonctions. Pour toutes les fonctions, il y a des méthodes hors ligne pour arriver au même résultat, et la présente spécification ne rend pas obligatoire l'utilisation de protocoles en ligne. Par exemple, quand des jetons matériels sont utilisés, beaucoup des fonctions peuvent être réalisées au titre de la livraison de jeton physique. De plus, certaines de ces fonctions peuvent être combinées en un échange de protocole. En particulier, deux fonctions, ou plus, d'enregistrement, initialisation, et certification peuvent être combinées en un échange de protocole.

La série de spécifications PKIX définit un ensemble de formats de message standard qui prennent en charge les fonctions ci-dessus. Les protocoles pour porter ces messages dans différents environnements (par exemple, messagerie électronique, transfert de fichier, et navigation sur la Toile) sont décrits dans ces spécifications.

4. Profil de certificat et d'extensions de certificat

Cette section présente un profil pour les certificats de clé publique pour permettre l'interopérabilité et une PKI réutilisable. Cette section se fonde sur le format de certificat X.509 v3 et les extensions standard de certificat définies dans [X.509]. Les documents de l'ISO/CEI et de l'UIT-T utilisent la version 1997 de l'ASN.1 tandis que le présent document utilise la syntaxe ASN.1 de 1988, mais les certificats codés et les extensions standard sont équivalents. Cette section définit aussi des extensions privées dont il est exigé qu'elles prennent en charge une PKI pour la communauté de l'Internet.

Les certificats peuvent être utilisés dans une large gamme d'applications et environnements couvrant un grand spectre d'objectifs d'interopérabilité et un plus large spectre d'exigences de fonctionnement et d'assurance. Le but de ce document est d'établir des fondements communs pour les applications génériques qui exigent une large interopérabilité et des exigences limitées de cas particuliers. L'accent sera mis particulièrement sur la prise en charge de l'utilisation des certificats X.509 v3 pour les applications de la messagerie électronique informelle de l'Internet, de IPsec, et de la Toile mondiale.

4.1 Champs de base de certificat

La syntaxe de base du certificat X.509 v3 est la suivante : pour le calcul de la signature, les données à signer sont codées en utilisant les règles de codage distinctives (DER) de l'ASN.1 [X.690]. Le codage ASN.1 en DER est un système de codage d'étiquette, longueur, valeur pour chaque élément.

```
Certificate ::= SEQUENCE {
      tbsCertificate          TBSCertificate,
      signatureAlgorithm   AlgorithmIdentifier,
      signatureValue         CHAINE BINAIRE }

TBSCertificate ::= SEQUENCE {
      version             [0]  EXPLICIT Version DEFAULT v1,
      serialNumber            CertificateSerialNumber,
      signature               AlgorithmIdentifier,
      issuer                  Name,
      validity                Validity,
      subject                 Name,
      subjectPublicKeyInfo SubjectPublicKeyInfo,
      issuerUniqueID   [1]   IMPLICITE UniqueIdentifier FACULTATIF,
-- Si il est présent, version DOIT être v2 ou v3
      subjectUniqueID [2]   IMPLICITE UniqueIdentifier FACULTATIF,
-- Si il est présent, version DOIT être v2 ou v3
      extensions          [3]   EXPLICITE Extensions FACULTATIF
-- Si il est présent, version DOIT être v3
   }

Version ::= ENTIER { v1(0), v2(1), v3(2) }

CertificateSerialNumber ::= ENTIER

Validity ::= SEQUENCE {
      notBefore     Time,
      notAfter      Time }

Time ::= CHOIX {
      utcTime         UTCTime,
      generalTime   GeneralizedTime }

UniqueIdentifier ::= CHAINE BINAIRE

SubjectPublicKeyInfo ::= SEQUENCE {
      algorithm             AlgorithmIdentifier,
      subjectPublicKey    CHAINE BINAIRE }
```

```
Extensions  ::=  TAILLE DE SEQUENCE (1..MAX) DE Extension

Extension  ::=  SEQUENCE  {
      extnID      IDENTIFIANT D'OBJET,
      critical    BOOLEEN DEFAUT FAUX,
      extnValue   CHAINE D'OCTETS
-- contient le codage en DER d'une valeur ASN.1 correspondant à l'identifiant de type d'extension identifié par extnID.
   }
```

Les éléments qui suivent décrivent le certificat X.509 v3 à utiliser dans l'Internet.

4.1.1 Champs de certificat

Le certificat est une SEQUENCE de trois champs exigés. Les champs sont décrits en détail dans les sous paragraphes suivants.

4.1.1.1 tbsCertificate

Le champ contient les noms du sujet et du producteur, une clé publique associée au sujet, une période de validité, et les autres informations associées. Les champs sont décrits en détails au paragraphe 4.1.2 ; le tbsCertificate inclut généralement des extensions, qui sont décrites au paragraphe 4.2.

4.1.1.2 signatureAlgorithm

Le champ signatureAlgorithm contient l'identifiant de l'algorithme de chiffrement utilisé par la CA pour signer ce certificat. Les [RFC3279], [RFC4055], et [RFC4491] font la liste des algorithmes de signature pris en charge, mais d'autres algorithmes de signature PEUVENT aussi être pris en charge.

Un identifiant d'algorithme est défini par la structure ASN.1 suivante :

```
AlgorithmIdentifier ::=  SEQUENCE  {
      algorithm            IDENTIFIANT D'OBJET,
      parameters           TOUS CEUX DEFINIS PAR algorithm FACULTATIF  }
```

L'identifiant d'algorithme est utilisé pour identifier un algorithme de chiffrement. Le composant IDENTIFIANT D'OBJET identifie l'algorithme (comme DSA avec SHA-1). Le contenu du champ Paramètres facultatif va varier selon l'algorithme identifié.

Ce champ DOIT contenir le même identifiant d'algorithme que le champ de signature dans la séquence tbsCertificate (paragraphe 4.1.2.3).

4.1.1.3 signatureValue

Le champ signatureValue contient une signature numérique calculée sur le tbsCertificate ASN.1 codé en DER. Le tbsCertificate ASN.1 codé en DER est utilisé comme entrée de la fonction de signature. Cette valeur de signature est codée comme CHAINE BINAIRE et incluse dans le champ signature. Les détails de ce processus sont spécifiés pour chacun des algorithmes cités dans les [RFC3279], [RFC4055], et [RFC4491].

En générant cette signature, une CA certifie la validité des informations dans le champ tbsCertificate. En particulier, la CA certifie le lien entre le matériel de clé publique et le sujet du certificat.

4.1.2 TBSCertificate

La séquence TBSCertificate contient les informations associées au sujet du certificat et à la CA qui l'a produit. Chaque TBSCertificate contient les noms du sujet et du producteur, une clé publique associée au sujet, une période de validité, un numéro de version, et un numéro de série ; certains PEUVENT contenir des champs d'identifiant univoque facultatifs. Le reste de cette section décrit la syntaxe et la sémantique de ces champs. Un TBSCertificate inclut généralement des extensions. Les extensions pour la PKI Internet sont décrites au paragraphe 4.2.

4.1.2.1 Version

Ce champ décrit la version du certificat codé. Quand des extensions sont utilisées, comme prévu par le présent profil, la version DOIT être 3 (la valeur est 2). Si aucune extension n'est présente, mais si un UniqueIdentifier est présent, la version

DEVRAIT être 2 (la valeur est 1) ; cependant, la version PEUT être 3. Si seulement les champs de base sont présents, la version DEVRAIT être 1 (la valeur est omise du certificat comme valeur par défaut) ; cependant, la version PEUT être 2 ou 3.

Les mises en œuvre DEVRAIENT être prêtes à accepter toute version de certificat. Au minimum, les mises en œuvre conformes DOIVENT reconnaître les certificats de version 3.

La génération de certificats de version 2 n'est pas attendue par les mises en œuvre qui se fondent sur le présent profil.

4.1.2.2 Numéro de série

Le numéro de série DOIT être un entier positif alloué par la CA à chaque certificat. Il DOIT être unique pour chaque certificat produit par une certaine CA (c'est-à-dire, le nom de producteur et le numéro de série identifient un unique certificat). Les CA DOIVENT forcer le serialNumber à être un entier non négatif.

Étant données les exigences d'unicité ci-dessus, les numéros de série peuvent être supposés contenir de longs entiers. Les utilisateurs de certificat DOIVENT être capables de traiter des valeurs de serialNumber jusqu'à 20 octets. Les CA conformes NE DOIVENT PAS utiliser des valeurs de serialNumber de plus de 20 octets.

Note : les CA non conformes peuvent produire des certificats avec des numéros de série qui sont négatifs ou zéro. Les utilisateurs de certificat DEVRAIENT être prêts à traiter en douceur de tels certificats.

4.1.2.3 Signature

Ce champ contient l'identifiant d'algorithme de l'algorithme utilisé par la CA pour signer le certificat.

Ce champ DOIT contenir le même identifiant d'algorithme que le champ signatureAlgorithm dans la séquence de certificats (paragraphe 4.1.1.2). Le contenu du champ Paramètres facultatifs va varier selon l'algorithme identifié. Les [RFC3279], [RFC4055], et [RFC4491] font la liste des algorithmes de signature acceptés, mais d'autres algorithmes de signature PEUVENT aussi être pris en charge.

4.1.2.4 Issuer

Le champ Issuer identifie l'entité qui a signé et produit le certificat. Le champ Issuer DOIT contenir un nom distinctif (DN, *Distinguished Name*) non vide. Le champ Issuer est défini comme le nom de type X.501 [X.501]. Le nom est défini par les structures ASN.1 suivantes :

```
Name ::= CHOIX {                              -- une seule possibilité actuellement --
   rdnSequence  RDNSequence }

RDNSequence ::= SEQUENCE DE RelativeDistinguishedName

RelativeDistinguishedName ::= ENSEMBLE TAILLE (1..MAX) DE AttributeTypeAndValue

AttributeTypeAndValue ::= SEQUENCE {
   type     AttributeType,
   value    AttributeValue }

AttributeType ::= IDENTIFIANT D'OBJET

AttributeValue ::= TOUTES -- DEFINI PAR AttributeType

DirectoryString ::= CHOIX {
      teletexString          TeletexString (TAILLE (1..MAX)),
      printableString        PrintableString (TAILLE (1..MAX)),
      universalString        UniversalString (TAILLE (1..MAX)),
      utf8String             UTF8String (TAILLE (1..MAX)),
      bmpString              BMPString (TAILLE (1..MAX)) }
```

Name décrit un nom hiérarchique composé d'attributs, comme un nom de pays, et les valeurs correspondantes, comme FR. Le type de la composante AttributeValue est déterminé par le AttributeType ; en général, il va être une DirectoryString.

Le type DirectoryString est défini comme un choix de PrintableString, TeletexString, BMPString, UTF8String, et UniversalString. Les CA conformes au présent profil DOIVENT utiliser le codage PrintableString ou UTF8String de DirectoryString, avec deux exceptions. Quand les CA ont préalablement produit des certificats avec un champ Issuer qui a des attributs codés en utilisant TeletexString, BMPString, ou UniversalString, la CA PEUT alors continuer d'utiliser ces codages de la DirectoryString pour préserver la rétro compatibilité. Aussi, les nouvelles CA qui sont ajoutées à un domaine où les CA existantes produisent des certificats dont les champs Issuer ont les attributs codés en utilisant TeletexString, BMPString, ou UniversalString, PEUVENT coder les attributs qu'elles partagent avec les CA existantes qui utilisent les mêmes codages que les CA existantes.

Comme noté ci-dessus, les noms distinctifs sont composés d'attributs. La présente spécification ne restreint pas l'ensemble de types d'attribut qui peuvent apparaître dans les noms. Cependant, les mises en œuvre conformes DOIVENT être prêtes à recevoir des certificats avec des noms de producteur contenant l'ensemble des types d'attribut définis ci-dessous. La présente spécification RECOMMANDE la prise en charge de types d'attribut supplémentaires.

Les ensembles d'attributs standard ont été définis dans la série de spécifications [X.520]. Les mises en œuvre de la présente spécification DOIVENT être prêtes à recevoir les types d'attribut standard suivants dans les noms de producteur et de sujet (paragraphe 4.1.2.6) :
* pays,
* organisation,
* unité organisationnelle,
* qualificatif de nom distinctif,
* nom d'état ou de province,
* nom propre (par exemple, "Susan Housley"), et
* numéro de série.

De plus, les mises en œuvre de la présente spécification DEVRAIENT être prêtes à recevoir les types d'attribut standard suivants dans les noms de producteur et de sujet :
* localité,
* titre,
* nom de famille,
* surnom,
* initiales,
* pseudonyme, et
* qualificatif de génération (par exemple, "Jr.", "3ème", ou "IV").

La syntaxe et les identifiants d'objet (OID, O*bject IDentifier*) associés à ces types d'attribut sont fournis dans les modules ASN.1 de l'Appendice A.

De plus, les mises en œuvre de la présente spécification DOIVENT être prêtes à recevoir l'attribut domainComponent, comme défini dans la [RFC4519]. Le système des noms de domaine (DNS, *Domain Name System*) fournit un système d'étiquetage de ressource hiérarchique. Cet attribut fournit un mécanisme convenable pour les organisations qui souhaitent utiliser des DN en parallèle avec leurs noms DNS. Ce n'est pas un remplacement du composant dNSName des extensions de nom de remplacement. Les mises en œuvre ne sont pas obligées de convertir de tels noms en noms DNS. La syntaxe et l'OID associé pour ce type d'attribut sont fournis dans les modules ASN.1 de l'Appendice A. Les règles de codage des noms de domaine internationalisés à utiliser avec le type d'attribut domainComponent sont spécifiées au paragraphe 7.3.

Les utilisateurs de certificat DOIVENT être prêts à traiter les champs de nom distinctif du producteur et de nom distinctif du sujet (paragraphe 4.1.2.6) pour effectuer le chaînage pour la validation du chemin de certification (Section 6). Le chaînage de noms est effectué en faisant correspondre le nom distinctif du producteur dans un certificat avec le nom de sujet dans un certificat de CA. Les règles de comparaison des noms distinctifs sont spécifiées au paragraphe 7.1. Si les noms dans le champ Producteur et Sujet dans un certificat correspondent selon les règles spécifiées au paragraphe 7.1, le certificat est alors auto-produit.

4.1.2.5 Validity

La période de validité d'un certificat est l'intervalle durant lequel la CA garantit qu'elle va maintenir les informations sur le statut du certificat. Le champ est représenté comme une SEQUENCE de deux dates : la date à laquelle commence la période de validité du certificat (notBefore) et la date à laquelle la période de validité du certificat se termine (notAfter). notBefore et notAfter peuvent tous deux être codés comme UTCTime ou GeneralizedTime.

Les CA conformes au présent profil DOIVENT toujours coder la date de validité de certificat jusqu'à l'année 2049 en UTCTime ; les dates de validité de certificat de 2050 ou plus tard DOIVENT être codées en GeneralizedTime. Les

applications conformes DOIVENT être capables de traiter des dates de validité qui sont codées en UTCTime ou en GeneralizedTime.

La période de validité pour un certificat est la période qui va de notBefore à notAfter, inclus.

Dans certaines situations, les appareils reçoivent des certificats pour lesquels une bonne date d'expiration ne peut être allouée. Par exemple, un appareil pourrait recevoir un certificat qui lie son modèle et son numéro de série à sa clé publique ; un tel certificat est destiné à être utilisé pour toute la durée de vie de l'appareil.

Pour indiquer qu'un certificat n'a pas de date d'expiration bien définie, le notAfter DEVRAIT recevoir la valeur de GeneralizedTime de 99991231235959Z.

Quand le producteur ne va pas être capable de conserver les informations d'état jusqu'à la date notAfter (incluant quand la date notAfter est 99991231235959Z) le producteur DOIT s'assurer qu'aucun chemin de certification valide n'existe pour le certificat après que le maintien des informations d'état est terminé. Ce peut être accompli par l'expiration ou la révocation de tous les certificats de CA contenant la clé publique utilisée pour vérifier la signature sur le certificat et en cessant d'utiliser la clé publique utilisée pour vérifier la signature sur le certificat comme une ancre de confiance.

4.1.2.5.1 UTCTime

Le type de temps universel, UTCTime, est un type standard ASN.1 destiné à la représentation des dates et des heures. UTCTime spécifie l'année par les deux chiffres de moindre poids et l'heure est spécifiée à la précision de une minute ou une seconde. UTCTime inclut soit Z (pour Zoulou, ou heure moyenne de Greenwich) soit le différentiel horaire.

Pour les besoins du présent profil, les valeurs de UTCTime DOIVENT être exprimées en heure moyenne de Greenwich (Zoulou) et DOIVENT inclure les secondes (c'est-à-dire, les heures sont AAMMJJHHMMSSZ) même lorsque le nombre de secondes est zéro. Les systèmes conformes DOIVENT interprêter le champ année (AA) comme suit : lorsque AA est supérieur ou égal à 50, l'année DEVRA être interprétée comme 19AA; et lorsque AA est inférieur à 50, l'année DEVRA être interprétée comme 20AA.

4.1.2.5.2 GeneralizedTime

Le type temps généralisé, GeneralizedTime, est un type standard ASN.1 pour une représentation de précision variable de l'heure. Facultativement, le champ GeneralizedTime peut inclure une représentation du différentiel horaire entre l'heure locale et l'heure moyenne de Greenwich.

Pour les besoins du présent profil, les valeurs de GeneralizedTime DOIVENT être exprimées en heure moyenne de Greenwich (Zoulou) et DOIVENT inclure les secondes (c'est-à-dire, les temps sont AAAAMMJJHHMMSSZ) même lorsque le nombre de secondes est zéro. Les valeurs de GeneralizedTime NE DOIVENT PAS inclure de fraction de seconde.

4.1.2.6 Subject

Le champ Subject identifie l'entité associée à la clé publique mémorisée dans le champ clé publique sujette. Le nom du sujet PEUT être porté dans le champ Subject et/ou dans l'extension subjectAltName. Si le sujet est une CA (par exemple, l'extension Contraintes de base, comme exposé au paragraphe 4.2.1.9, est présente et la valeur de CA est VRAI) le champ Subject DOIT être rempli avec un nom distinctif non vide correspondant au contenu du champ Producteur (paragraphe 4.1.2.4) dans tous les certificats produits par la CA sujette. Si le sujet est le producteur de CRL (par exemple, l'extension d'usage de clé, comme exposé au paragraphe 4.2.1.3, est présente et la valeur de cRLSign est VRAI) alors le champ Sujet DOIT être rempli avec un nom distinctif non vide correspondant au contenu du champ Producteur (paragraphe 5.1.2.3) dans toutes les CRL produites par le producteur de CRL sujet. Si les informations de dénomination du sujet ne sont présentes que dans l'extension subjectAltName (par exemple, une clé liée seulement à une adresse de messagerie ou un URI) alors le nom de sujet DOIT être une séquence vide et l'extension subjectAltName DOIT être critique.

Lorsque il n'est pas vide, le champ Subject DOIT contenir un nom distinctif X.500 (DN). Le DN DOIT être unique pour chaque entité sujette certifiée par la CA comme défini par le champ Producteur. Une CA PEUT produire plus d'un certificat avec le même DN à la même entité sujette.

Le champ sujet est défini comme nom de type X.501. Les exigences de mise en œuvre pour ce champ sont celles définies pour le champ Producteur (paragraphe 4.1.2.4). Les mises en œuvre de la présente spécification DOIVENT être prêtes à recevoir des noms de sujet contenant les types d'attribut requis pour le champ Producteur. Les mises en œuvre de la présente spécification DEVRAIENT être prêtes à recevoir des noms de sujet contenant les types d'attribut recommandés pour le champ Producteur. La syntaxe et les identifiants d'objet (OID) associés pour ces types d'attribut sont fournis dans

les modules ASN.1 de l'Appendice A. Les mises en œuvre de la présente spécification PEUVENT utiliser les règles de comparaison du paragraphe 7.1 pour traiter les types d'attribut non familiers (c'est-à-dire, pour le chaînage de noms) dont les valeurs d'attribut utilisent une des options de codage provenant de DirectoryString. La comparaison binaire devrait être utilisée quand des types d'attribut non familiers incluent des valeurs d'attribut avec des options de codage autres que celles qui se trouvent dans DirectoryString. Cela permet aux mises en œuvre de traiter les certificats qui ont des attributs non familiers dans le nom de sujet.

Quand elles codent des valeurs d'attribut du type DirectoryString, les CA conformes DOIVENT utiliser le codage PrintableString ou UTF8String, avec les exceptions suivantes :

(a) Quand le sujet du certificat est une CA, le champ Sujet DOIT être codé de la même façon que dans le champ Producteur (paragraphe 4.1.2.4) dans tous les certificats produits par la CA sujette. Donc, si la CA sujette code les attributs dans les champs Producteur des certificats qu'elle produit en utilisant les codages TeletexString, BMPString, ou UniversalString, le champ Sujet des certificats produits à cette CA DOIT alors utiliser le même codage.

(b) Quand le sujet du certificat est le producteur de CRL, le champ Sujet DOIT être codé de la même façon que dans le champ Producteur (paragraphe 5.1.2.3) dans toutes les CRL produites par le producteur de CRL sujet.

(c) TeletexString, BMPString, et UniversalString sont inclus pour la rétro compatibilité, et NE DEVRAIENT PAS être utilisés pour des certificats pour de nouveaux sujets. Cependant, ces types PEUVENT être utilisés dans des certificats où le nom été établi antérieurement, incluant les cas dans lesquels un nouveau certificat est produit à un sujet existant ou un certificat est produit à un nouveau sujet où les attributs à coder ont été établis précédemment dans des certificats produits à d'autres sujets. Les utilisateurs de certificat DEVRAIENT être prêts à recevoir des certificats de ces types.

Il existe des mises en œuvre traditionnelles où une adresse de messagerie électronique est incorporée dans le nom distinctif du sujet comme un attribut emailAddress [RFC2985]. La valeur d'attribut pour emailAddress est du type IA5String pour permettre l'inclusion du caractère '@', qui ne fait pas partie du jeu de caractères PrintableString. Les valeurs d'attribut emailAddress ne sont pas sensibles à la casse (par exemple, "subscriber@example.com" est le même que "SUBSCRIBER@EXAMPLE.COM").

Les mises en œuvre conformes qui génèrent de nouveaux certificats avec des adresses de messagerie électronique DOIVENT utiliser le rfc822Name dans l'extension de nom de remplacement sujet (paragraphe 4.2.1.6) pour décrire de telles identités. L'inclusion simultanée de l'attribut emailAddress dans le nom distinctif de sujet pour prendre en charge les mises en œuvre traditionnelles est déconseillée mais permise.

4.1.2.7 Informations de clé publique du sujet

Ce champ est utilisé pour porter la clé publique et identifier l'algorithme avec lequel la clé est utilisée (par exemple, RSA, DSA, ou Diffie-Hellman). L'algorithme est identifié en utilisant la structure AlgorithmIdentifier spécifiée au paragraphe 4.1.1.2. Les identifiants d'objet pour les algorithmes pris en charge et les méthodes pour le codage du matériel de clé publique (clé publique et paramètres) sont spécifiés dans les [RFC3279], [RFC4055], et [RFC4491].

4.1.2.8 Identifiants univoques

Ces champs DOIVENT apparaître seulement si la version est 2 ou 3 (paragraphe 4.1.2.1). Ces champs NE DOIVENT PAS apparaître si la version est 1. Les identifiants univoques du sujet et du producteur sont présents dans le certificat pour traiter la possibilité de réutilisation ultérieure du nom du sujet et/ou du producteur. Ce profil RECOMMANDE que les noms ne soient pas réutilisés pour des entités différentes et que les certificats Internet n'utilisent pas d'identifiants univoques. Les CA conformes au présent profil NE DOIVENT PAS générer de certificats avec des identifiants univoques. Les applications conformes au présent profil DEVRAIENT être capables d'analyser les certificats qui incluent des identifiant univoques, mais il n'y a pas d'exigence de traitement associée aux identifiants univoques.

4.1.2.9 Extensions

Ce champ DOIT apparaître seulement si la version est 3 (paragraphe 4.1.2.1). Si il est présent, ce champ est une SEQUENCE de une ou plusieurs extensions de certificat. Le format et le contenu des extensions de certificat dans la PKI Internet sont définis au paragraphe 4.2.

4.2 Extensions de certificat

Les extensions définies pour les certificats X.509 v3 fournissent des méthodes pour associer des attributs supplémentaires aux utilisateurs ou clés publiques et pour gérer les relations entre les CA. Le format de certificat X.509 v3 permet aussi à des communautés de définir des extensions privées pour porter des informations propres à ces communautés. Chaque

extension dans un certificat est désignée comme critique ou non critique. Un système utilisateur de certificats DOIT rejeter le certificat si il rencontre une extension critique qu'il ne reconnaît pas ou une extension critique qui contient des informations qu'il ne peut pas traiter. Une extension non critique PEUT être ignorée si elle n'est pas reconnue, mais DOIT être traitée si elle est reconnue. Les paragraphes qui suivent présentent les extensions recommandées utilisées dans les certificats Internet et les localisations standard pour information. Des communautés peuvent choisir d'utiliser des extensions supplémentaires ; cependant, il faut faire attention quand on adopte dans les certificats des extensions critiques qui pourraient empêcher leur utilisation dans un contexte général.

Chaque extension inclut un OID et une structure ASN.1. Quand une extension apparaît dans un certificat, l'OID apparaît dans le champ extnID et la structure ASN.1 codée en DER correspondante est la valeur de la chaîne d'octets extnValue. Un certificat NE DOIT PAS inclure plus d'une instance d'une extension particulière. Par exemple, un certificat peut contenir seulement une extension d'identifiant de clé d’autorité (paragraphe 4.2.1.1). Une extension inclut le booléen critique, avec une valeur par défaut de FAUX. Le texte pour chaque extension spécifie les valeurs acceptables de champ critique pour les CA conformes au présent profil.

Les CA conformes DOIVENT prendre en charge les identifiants de clé (paragraphes 4.2.1.1 et 4.2.1.2) les extensions de contraintes de base (paragraphe 4.2.1.9) d'usage de clé (paragraphe 4.2.1.3) et de politique de certificat (paragraphe 4.2.1.4). Si la CA produit des certificats avec une séquence vide pour le champ Sujet, la CA DOIT prendre en charge l'extension Nom de remplacement de sujet (paragraphe 4.2.1.6). La prise en charge des extensions restantes est FACULTATIVE. Les CA conformes PEUVENT prendre en charge des extensions qui ne sont pas identifiées dans la présente spécification ; les producteurs de certificat sont avertis que rendre de telles extensions critiques peut impacter l'interopérabilité.

Au minimum, les applications conformes au présent profil DOIVENT reconnaître les extensions suivantes : Usage de clé (paragraphe 4.2.1.3) Politiques de certificat (paragraphe 4.2.1.4) Nom de remplacement de sujet (paragraphe 4.2.1.6) Contraintes de base (paragraphe 4.2.1.9) Contraintes de nom (paragraphe 4.2.1.10) Contraintes de politique (paragraphe 4.2.1.11) Usage de clé étendu (paragraphe 4.2.1.12) et Inhiber toute politique (paragraphe 4.2.1.14).

De plus, les applications conformes au présent profil DEVRAIENT reconnaître les extensions Identifiant de clé d'autorité et de sujet (paragraphes 4.2.1.1 et 4.2.1.2) et Transpositions de politique (paragraphe 4.2.1.5).

4.2.1 Extensions standard

Ce paragraphe identifie les extensions de certificat standard définies dans [X.509] pour utilisation dans la PKI Internet. Chaque extension est associée à un OID défini dans [X.509]. Ces OID sont membres de l'arc id-ce, qui est défini comme suit :

IDENTIFIANT D’OBJET id-ce ::= { joint-iso-ccitt(2) ds(5) 29 }

4.2.1.1 Identifiant de clé d'autorité

L'extension Identifiant de clé d’autorité fournit un moyen pour identifier la clé publique correspondant à la clé privée utilisée pour signer un certificat. Cette extension est utilisée lorsque un producteur a plusieurs clés de signature (soit à cause de plusieurs paires de clé concurrentes, soit à cause d'un changement de clés). L'identification PEUT se fonder sur l'identifiant de clé (l'identifiant de clé de sujet dans le certificat du producteur) ou le nom et numéro de série du producteur.

Le champ keyIdentifier de l'extension authorityKeyIdentifier DOIT être inclus dans tous les certificats générés par les CA conformes pour faciliter la construction du chemin de certification. Il y a une exception, lorsque une CA distribue sa clé publique sous la forme d'un certificat "auto signé", l'identifiant de clé d’autorité PEUT être omis. La signature sur un certificat auto signé est générée avec la clé privée associée à la clé publique sujette du certificat. (Cela prouve que le producteur possède les deux clés publique et privée.) Dans ce cas, les identifiants de clé d’autorité et sujette vont être identiques, mais seul l'identifiant de clé de sujet est nécessaire pour la construction du chemin de certification.

La valeur du champ keyIdentifier DEVRAIT être déduite de la clé publique utilisée pour vérifier la signature du certificat ou d'une méthode qui génère des valeurs uniques. Deux méthodes courantes pour générer les identifiants de clé à partir de la clé publique sont décrites au paragraphe 4.2.1.2. Lorsque un identifiant de clé n'a pas été précédemment établi, la présente spécification RECOMMANDE l'utilisation d'une de ces méthodes pour générer les identifiants de clé ou l'utilisation d'une méthode similaire qui utilise un algorithme de hachage différent. Lorsque un identifiant de clé a été établi précédemment, la CA DEVRAIT l'utiliser.

Ce profil RECOMMANDE la prise en charge de la méthode d'identifiant de clé par tous les utilisateurs de certificat.

Les CA conformes DOIVENT marquer cette extension comme non critique.

IDENTIFIANT D'OBJET id-ce-authorityKeyIdentifier ::= { id-ce 35 }

```
AuthorityKeyIdentifier ::= SEQUENCE {
    keyIdentifier             [0] KeyIdentifier            FACULTATIF,
    authorityCertIssuer       [1] GeneralNames             FACULTATIF,
    authorityCertSerialNumber [2] CertificateSerialNumber FACULTATIF  }

KeyIdentifier ::= CHAINE D'OCTETS
```

4.2.1.2 Identifiant de clé de sujet

L'extension Identifiant de clé de sujet donne le moyen d'identifier les certificats qui contiennent une clé publique particulière.

Pour faciliter la construction du chemin de certification, cette extension DOIT apparaître dans tous les certificats de CA conformes, c'est-à-dire tous les certificats incluant l'extension Contraintes de base (paragraphe 4.2.1.9) lorsque la valeur de CA est VRAI. Dans les certificats de CA conformes, la valeur de l'identifiant de clé de sujet DOIT être la valeur placée dans le champ Identifiant de clé de l'extension Identifiant de clé d'autorité (paragraphe 4.2.1.1) des certificats produits par le sujet de ce certificat. Les applications ne sont pas obligées de vérifier que les identifiants de clé correspondent lors de la validation du chemin de certification.

Pour les certificats de CA, les identifiants de clé de sujet DEVRAIENT être déduits de la clé publique ou par une méthode qui génère des valeurs univoques. Deux méthodes courantes pour générer des identifiants de clé à partir de la clé publique sont :

(1) Le keyIdentifier est composé du hachage SHA-1 de 160 bits de la valeur de la CHAINE BINAIRE subjectPublicKey (excluant l'étiquette, longueur, et nombre de bits inutilisés).

(2) Le keyIdentifier est composé d'un champ de type de quatre bits avec la valeur 0100 suivie par les 60 bits de moindre poids du hachage SHA-1 de la valeur de la CHAINE BINAIRE subjectPublicKey (excluant l'étiquette, longueur, et nombre de bits inutilisés).

D'autres méthodes de génération de nombres univoques sont aussi acceptables.

Pour les certificats d'entité d'extrémité, l'exension Identifiant de clé de sujet donne un moyen pour identifier les certificats contenant la clé publique particulière utilisée dans une application. Lorsque une entité d'extrémité a obtenu plusieurs certificats, en particulier de plusieurs CA, l'identifiant de clé de sujet donne le moyen d'identifier rapidement l'ensemble de certificats contenant une clé publique particulière. Pour aider les applications à identifier le certificat d'entité d'extrémité approprié, cette extension DEVRAIT être incluse dans tous les certificats d'entité d'extrémité.

Pour les certificats d'entité d'extrémité, les identifiants de clé de sujet DEVRAIENT être déduits de la clé publique. Deux méthodes courantes pour générer les identifiants de clé à partir de la clé publique sont identifiées ci-dessus.

Lorsque un identifiant de clé n'a pas été établi précédemment, la présente spécification RECOMMANDE l'utilisation d'une de ces méthodes pour générer les identifiants de clé ou d'utiliser une méthode similaire qui utilise un algorithme de hachage différent. Lorsque un identifiant de clé a été établi précédemment, la CA DEVRAIT l'utiliser.

Les CA conformes DOIVENT marquer cette extension comme non critique.

IDENTIFIANT D'OBJET id-ce-subjectKeyIdentifier ::= { id-ce 14 }

SubjectKeyIdentifier ::= KeyIdentifier

4.2.1.3 Usage de clé

L'extension d'usage de clé définit l'objet (par exemple, chiffrement, signature, signature de certificat) de la clé contenue dans le certificat. Une restriction d'usage peut être employée quand une clé qui pourrait être utilisée pour plus d'une opération doit être restreinte. Par exemple, quand une clé RSA devrait être utilisée seulement pour vérifier les signatures sur des objets autres que des certificats de clé publique et des CRL, les bits digitalSignature et/ou nonRepudiation vont être établis. De même, quand une clé RSA devrait être utilisée seulement pour la gestion de clé, le bit keyEncipherment va être établi.

Les CA conformes DOIVENT inclure cette extension dans les certificats qui contiennent des clés publiques qui sont utilisées pour valider les signatures numériques sur d'autres certificats de clé publique ou CRL. Quand elle est présente, les CA conformes DEVRAIENT marquer cette extension comme critique.

IDENTIFIANT D'OBJET id-ce-keyUsage ::= { id-ce 15 }

```
KeyUsage ::= CHAINE BINAIRE {
        digitalSignature      (0),
        nonRepudiation        (1),          -- les éditions récentes de X.509 ont renommé ce bit contentCommitment
        keyEncipherment       (2),
        dataEncipherment      (3),
        keyAgreement          (4),
        keyCertSign           (5),
        cRLSign               (6),
        encipherOnly          (7),
        decipherOnly          (8) }
```

Les bits du type KeyUsage sont utilisés comme suit :

Le bit digitalSignature est établi quand la clé publique sujette est utilisée pour vérifier les signatures numériques, autres que les signatures sur les certificats (bit 5) et les CRL (bit 6), comme celles utilisées dans un service d'authentification d'entité, un service d'authentification d'origine des données, et/ou un service de protection de l'intégrité.

Le bit nonRepudiation est établi quand la clé publique sujette est utilisée pour vérifier les signatures numériques autres que les signatures sur les certificats (bit 5) et les CRL (bit 6), utilisées pour fournir un service de non répudiation qui protège contre le refus d'une certaine action par l'entité signataire. Dans le cas d'un conflit ultérieur, un tiers de confiance peut déterminer l'authenticité des données signées. (Noter que les éditions récentes de X.509 ont renommé le bit nonRepudiation en contentCommitment *(engagement de contenu)*.)

Le bit keyEncipherment est établi quand la clé publique sujette est utilisée pour chiffrer les clés privées ou secrètes, c'est-à-dire, pour le transport de clé. Par exemple, ce bit devra être établi quand une clé publique RSA doit être utilisée pour chiffrer une clé symétrique de déchiffrement de contenu ou une clé privée asymétrique.

Le bit dataEncipherment est établi quand la clé publique sujette est utilisée pour chiffrer directement des données d'utilisateur brutes sans l'utilisation d'un chiffrement symétrique intermédiaire. Noter que l'utilisation de ce bit est extrêmement rare ; presque toutes les applications utilisent le transport de clé ou l'accord de clé pour établir une clé symétrique.

Le bit keyAgreement est établi quand la clé publique sujette est utilisée pour l'accord de clé. Par exemple, quand une clé Diffie-Hellman va être utilisée pour la gestion de clé, ce bit est alors établi.

Le bit keyCertSign est établi quand la clé publique sujette est utilisée pour vérifier les signatures sur des certificats de clé publique. Si le bit keyCertSign est établi, alors le bit cA dans l'extension Contraintes de base (paragraphe 4.2.1.9) DOIT aussi être établi.

Le bit cRLSign est établi quand la clé publique sujette est utilisée pour vérifier les signatures sur des listes de révocation de certificat (par exemple, des CRL, des CRL delta, ou des ARL).

La signification du bit encipherOnly est indéfinie en l'absence du bit keyAgreement. Quand le bit encipherOnly est établi et que le bit keyAgreement est aussi établi, la clé publique sujette ne peut être utilisée que pour le chiffrement de données en effectuant un accord de clé.

La signification du bit decipherOnly est indéfinie en l'absence du bit keyAgreement. Quand le bit decipherOnly est établi et que le bit keyAgreement est aussi établi, la clé publique sujette ne peut être utilisé que pour déchiffrer des données tout en effectuant l'accord de clé.

Si l'extension keyUsage est présente, alors la clé publique sujette NE DOIT PAS être utilisée pour vérifier les signatures sur les certificats ou les CRL sauf si le bit keyCertSign ou cRLSign correspondant est établi. Si la clé publique sujette est à utiliser seulement pour vérifier les signatures sur les certificats et/ou les CRL, alors les bits digitalSignature et nonRepudiation NE DEVRAIENT PAS être établis. Cependant, les bits digitalSignature et/ou nonRepudiation PEUVENT être établis en plus des bits keyCertSign et/ou cRLSign si la clé publique sujette est à utiliser pour vérifier les signatures sur les certificats et/ou les CRL ainsi que d'autres objets.

Combiner le bit nonRepudiation dans l'extension de certificat keyUsage avec d'autres bits keyUsage peut avoir des implications pour la sécurité selon le contexte dans lequel le certificat va être utilisé. D'autres distinctions entre les bits digitalSignature et nonRepudiation peuvent être fournies dans des politiques de certificat spécifiques.

Ce profil n'interdit pas les combinaisons de bits qui peuvent être établies dans une instance d'extension keyUsage. Cependant, les valeurs appropriées pour les extensions keyUsage pour des algorithmes particuliers sont spécifiées dans les [RFC3279], [RFC4055], et [RFC4491]. Quand l'extension keyUsage apparaît dans un certificat, au moins un des bits DOIT être réglé à 1 *(établi)*.

4.2.1.4 Politiques de certificat

L'extension Politiques de certificat contient une séquence d'un ou plusieurs termes d'informations de politique, dont chacun consiste en un identifiant d'objet (OID) et des qualificatifs facultatifs. Les qualificatifs facultatifs, qui PEUVENT être présents, ne sont pas supposés changer la définition de la politique. Un OID de politique de certificat NE DOIT PAS apparaître plus d'une fois dans une extension Politiques de certificat.

Dans un certificat d'entité d'extrémité, les termes d'informations de politique indiquent la politique sous laquelle le certificat a été produit et les objets pour lesquels le certificat peut être utilisé. Dans un certificat de CA, ces termes d'informations de politique limitent l'ensemble de politiques pour les chemins de certification qui incluent ce certificat. Quand une CA ne souhaite pas limiter l'ensemble de politiques pour les chemins de certification qui incluent ce certificat, elle PEUT établir la politique spéciale anyPolicy, avec une valeur de { 2 5 29 32 0 }.

Les applications qui ont des exigences de politique spécifiques sont supposées avoir une liste de ces politiques qu'elles vont accepter et comparer les OID de politique dans le certificat à cette liste. Si cette extension est critique, le logiciel de validation de chemin DOIT être capable d'interpréter cette extension (incluant le qualificatif facultatif) ou DOIT rejeter le certificat.

Pour promouvoir l'interopérabilité, le présent profil RECOMMANDE que les termes d'informations de politique consistent seulement en un OID. Lorsque un OID seul est insuffisant, le présent profil recommande fortement que l'utilisation de qualificatifs soit limitée à ceux identifiés dans cette section. Quand des qualificatifs sont utilisés avec la politique spéciale anyPolicy, ils DOIVENT être limités aux qualificatifs identifiés dans cette section. Seuls les qualificatifs retournés par suite d'une validation de chemin sont considérés.

La présente spécification définit deux types de qualificatifs de politique à utiliser par les auteurs de politique de certificat et les producteurs de certificat. Les types de qualificatifs sont Pointeur de CPS et Remarque d'utilisateur.

Le qualificatif Pointeur de CPS contient un pointeur sur une déclaration de pratique de certification (CPS, *Certification Practice Statement*) publiée par la CA. Le pointeur est sous la forme d'un URI. Les exigences de traitement pour ce qualificatif sont une affaire locale. Aucune action n'est obligatoire selon la présente spécification sans considération de la valeur de criticité établie par l'extension.

La remarque d'utilisateur est destinée à l'affichage à un utilisateur quand un certificat est utilisé. Seules les remarques d'utilisateur retournées par suite d'une validation de chemin sont destinées à être affichées à l'utilisateur. Si une remarque est dupliquée, une seule copie doit être affichée. Pour empêcher une telle duplication, ce qualificatif DEVRAIT n'être présent que dans des certificats d'entité d'extrémité et des certificats de CA produits à d'autres organisations.

La remarque d'utilisateur a deux champs facultatifs : le champ noticeRef et le champ explicitText. Les CA conformes NE DEVRAIENT PAS utiliser l'option noticeRef.

Le champ noticeRef, si il est utilisé, nomme une organisation et identifie, par un nombre, une déclaration textuelle particulière préparée par cette organisation. Par exemple, il peut identifier l'organisation "CertsRUs" et le numéro de remarque 1. Dans une mise en œuvre normale, le logiciel d'application va avoir un fichier de remarques contenant l'ensemble courant de remarques pour CertsRUs ; l'application va extraire le texte de la remarque du fichier et l'afficher. Les messages PEUVENT être multilingues, permettant au logiciel de choisir le langage particulier du message pour son propre environnement.

Un champ explicitText inclut la déclaration textuelle directement dans le certificat. Le champ explicitText est une chaîne avec une taille maximum de 200 caractères. Les CA conformes DEVRAIENT utiliser le codage UTF8String pour explicitText, mais PEUVENT utiliser IA5String. Les CA conformes NE DOIVENT PAS coder explicitText comme VisibleString ou BMPString. La chaîne explicitText NE DEVRAIT PAS inclure de caractère de contrôle (par exemple, de U+0000 à U+001F et de U+007F à U+009F). Quand le codage UTF8String est utilisé, toutes les séquences de caractères DEVRAIENT être normalisées conformément au format de normalisation Unicode C [NFC].

Si les deux options noticeRef et explicitText sont incluses dans le même qualificatif et si le logiciel d'application peut localiser le texte de la remarque indiquée par l'option noticeRef, ce texte DEVRAIT alors être affiché ; autrement, la chaîne explicitText DEVRAIT être affichée.

Note : bien que explicitText ait une taille maximum de 200 caractères, certaines CA non conformes excèdent cette limite. Donc, l'utilisateur de certificat DEVRAIT traiter en douceur un explicitText de plus de 200 caractères.

```
IDENTIFIANT D'OBJET id-ce-certificatePolicies ::=  { id-ce 32 }

IDENTIFIANT D'OBJET anyPolicy ::= { id-ce-certificatePolicies 0 }

certificatePolicies ::= TAILLE DE SEQUENCE (1..MAX) DE PolicyInformation

PolicyInformation ::= SEQUENCE {
      policyIdentifier    CertPolicyId,
      policyQualifiers   TAILLE DE SEQUENCE (1..MAX) DE PolicyQualifierInfo FACULTATIF }

IDENTIFIANT D'OBJET CertPolicyId ::= IDENTIFIANT D'OBJET

PolicyQualifierInfo ::= SEQUENCE {
    policyQualifierId    PolicyQualifierId,
    qualifier               TOUT DEFINI PAR policyQualifierId }

-- policyQualifierId pour qualificatifs de politique Internet

IDENTIFIANT D'OBJET id-qt           ::=  { id-pkix 2 }
IDENTIFIANT D'OBJET id-qt-cps       ::=  { id-qt 1 }
IDENTIFIANT D'OBJET id-qt-unotice ::=  { id-qt 2 }

PolicyQualifierId ::= IDENTIFIANT D'OBJET ( id-qt-cps | id-qt-unotice )

Qualifier ::= CHOIX {
      cPSuri          CPSuri,
      userNotice     UserNotice }

CPSuri ::= IA5String

UserNotice ::= SEQUENCE {
      noticeRef       NoticeReference FACULTATIF,
      explicitText     DisplayText FACULTATIF }

NoticeReference ::= SEQUENCE {
      organization     DisplayText,
      noticeNumbers    SEQUENCE DE ENTIER }

DisplayText ::= CHOIX {
      ia5String        IA5String      (TAILLE (1..200)),
      visibleString   VisibleString  (TAILLE (1..200)),
      bmpString      BMPString      (TAILLE (1..200)),
      utf8String       UTF8String     (TAILLE (1..200)) }
```

4.2.1.5 Transpositions de politique

Cette extension est utilisée dans les certificats de CA. Elle fait la liste des paires d'OID ; chaque paire inclut un issuerDomainPolicy et un subjectDomainPolicy. La paire indique que la CA productrice considère son issuerDomainPolicy comme équivalent au subjectDomainPolicy de la CA sujette.

Les utilisateurs de la CA productrice peuvent accepter un issuerDomainPolicy pour certaines applications. La transposition de politique définit la liste des politiques associées à la CA sujette qui peuvent être acceptées comme comparables à la issuerDomainPolicy.

Chaque issuerDomainPolicy nommée dans l'extension Transpositions de politique DEVRAIT aussi être affirmée dans l'extension Politiques de certificat dans le même certificat. Les politiques NE DOIVENT PAS être transposées de ou en la valeur spéciale anyPolicy (paragraphe 4.2.1.4).

En général, les politiques de certificat qui apparaissent dans le champ issuerDomainPolicy de l'extension Transpositions de politique ne sont pas considérées comme des politiques acceptables pour inclusion dans les certificats suivants sur le chemin de certification. Dans certaines circonstances, une CA peut souhaiter transposer d'une politique (p1) dans une autre (p2), mais veut quand même que le issuerDomainPolicy (p1) soit considéré comme acceptable pour inclusion dans les certificats suivants. Cela peut se produire, par exemple, si la CA est en cours de transition de l'utilisation de la politique p1 à celle de la politique p2 et a des certificats valides qui ont été produits sous chacune des politiques. Une CA peut indiquer cela en incluant deux transpositions de politique dans les certificats de CA qu'elle produit. Chaque transposition de politique aurait un issuerDomainPolicy de p1 ; une transposition de politique aurait un subjectDomainPolicy de p1 et l'autre aurait un subjectDomainPolicy de p2.

Cette extension PEUT être prise en charge par les CA et/ou applications. Les CA conformes DEVRAIENT marquer cette extension comme critique.

IDENTIFIANT D'OBJET id-ce-policyMappings ::= { id-ce 33 }

PolicyMappings ::= TAILLE DE SEQUENCE (1..MAX) DE SEQUENCE {
issuerDomainPolicy CertPolicyId,
subjectDomainPolicy CertPolicyId }

4.2.1.6 Nom de remplacement de sujet

L'extension Nom de remplacement de sujet permet de lier les identités au sujet du certificat. Ces identités peuvent être incluses en plus ou à la place de l'identité dans le champ Sujet du certificat. Les options définies incluent une adresse Internet de messagerie électronique, un nom DNS, une adresse IP, et un identifiant de ressource universel (URI, *Uniform Resource Identifier*). D'autres options existent, incluant des définitions complètement locales. Plusieurs formes de nom, et plusieurs instances de chaque forme de nom, PEUVENT être incluses. Chaque fois que de telles identités sont à lier dans un certificat, l'extension Nom de remplacement de sujet (ou Nom de remplacement de producteur) DOIT être utilisée ; cependant, un nom DNS PEUT aussi être représenté dans le champ Sujet en utilisant l'attribut domainComponent décrit au paragraphe 4.1.2.4. Noter que lorsque de tels noms sont représentés dans le champ Sujet, les mises en œuvre ne sont pas obligées de les convertir en noms DNS.

Comme le nom de remplacement de sujet est considéré comme définitivement lié à la clé publique, toutes les parties du nom de remplacement de sujet DOIVENT être vérifiées par la CA.

De plus, si la seule identité de sujet incluse dans le certificat est une forme de nom de remplacement (par exemple, une adresse de messagerie électronique) alors le nom distinctif du sujet DOIT être vide (une séquence vide) et l'extension subjectAltName DOIT être présente. Si le champ Sujet contient une séquence vide, alors la CA productrice DOIT inclure une extension subjectAltName qui est marquée comme critique. Quand elles incluent l'extension subjectAltName dans un certificat qui a un nom distinctif de sujet non vide, les CA conformes DEVRAIENT marquer l'extension subjectAltName comme non critique.

Quand l'extension subjectAltName contient une adresse de messagerie Internet, l'adresse DOIT être mémorisée dans le rfc822Name. Le format d'un rfc822Name est une "Mailbox" comme défini au paragraphe 4.1.2 de la [RFC2821]. Une Mailbox a la forme "Partie-locale@Domaine". Noter qu'une Mailbox n'a pas de phrase (comme un nom propre) avant elle, n'a pas de commentaire (du texte entre parenthèses) après elle, et n'est pas entouré de "<" et ">". Les règles de codage des adresses de messagerie Internet qui incluent des noms de domaine internationalisés sont spécifiées au paragraphe 7.5.

Quand l'extension subjectAltName contient une iPAddress, l'adresse DOIT être mémorisée dans la chaîne d'octets dans "l'ordre des octets du réseau", comme spécifié dans la [RFC0791]. Le bit de moindre poids (LSB) de chaque octet est le LSB de l'octet correspondant dans l'adresse réseau. Pour IP version 4, comme spécifié dans la [RFC0791], la chaîne d'octets DOIT contenir exactement quatre octets. Pour IP version 6, comme spécifié dans la [RFC2460], la chaîne d'octets DOIT contenir exactement seize octets.

Quand l'extension subjectAltName contient une étiquette du système des noms de domaine, le nom de domaine DOIT être mémorisé dans le dNSName (une IA5String). Le nom DOIT être dans la "syntaxe de nom préférée", comme spécifié au paragraphe 3.5 de la [RFC1034] et comme modifié par le paragraphe 2.1 de la [RFC1123]. Noter qu'alors que les lettres majuscules et minuscules sont autorisées dans les noms de domaines, aucune signification ne s'attache à la casse. De plus, alors que la chaîne " " est un nom de domaine légal, les extensions subjectAltName avec un dNSName de " " NE DOIVENT PAS être utilisées. Finalement, l'utilisation de la représentation DNS pour les adresses de messagerie Internet

(abonné.exemple.com au lieu de abonné@exemple.com) NE DOIT PAS être présente ; de telles identités sont à coder comme des rfc822Name. Les règles pour le codage des noms de domaine internationalisés sont spécifiées au paragraphe 7.2.

Quand l'extension subjectAltName contient un URI, le nom DOIT être mémorisé dans le uniformResourceIdentifier (une IA5String). Le nom NE DOIT PAS être un URI relatif, et il DOIT suivre la syntaxe d'URI et les règles de codage spécifiées dans la [RFC3986]. Le nom DOIT inclure un schéma (par exemple, "http" ou "ftp") et des parties d'URI spécifiques de schéma. Les URI qui incluent une autorité ([RFC3986], paragraphe 3.2) DOIVENT inclure un nom de domaine pleinement qualifié ou une adresse IP comme hôte. Les règles pour le codage des identifiants de ressource internationalisés (IRI) sont spécifiées au paragraphe 7.4.

Comme spécifié dans la [RFC3986], le nom de schéma n'est pas sensible à la casse (par exemple, "http" est équivalent à "HTTP"). La partie hôte, si elle est présente, est aussi non sensible à la casse, mais d'autres composants de la partie spécifique du schéma peuvent être sensibles à la casse. Les règles pour comparer les URI sont spécifiées au paragraphe 7.4.

Quand l'extension subjectAltName contient un DN dans le directoryName, les règles de codage sont les mêmes que celles spécifiées pour le champ Producteur au paragraphe 4.1.2.4. Le DN DOIT être univoque pour chaque entité sujette certifiée par la CA comme défini par le champ Producteur. Une CA PEUT produire plus d'un certificat avec le même DN à la même entité sujette.

Le subjectAltName PEUT porter des types de noms supplémentaires par l'utilisation du champ otherName. Le format et la sémantique du nom sont indiqués par l'IDENTIFIANT D'OBJET dans le champ type-id. Le nom lui-même est porté comme champ de valeur dans otherName. Par exemple, les noms de format Kerberos [RFC4120] peuvent être codés dans le otherName, en utilisant un OID de nom de principal Kerberos 5 et une SEQUENCE de Realm et PrincipalName.

Les noms de remplacement de sujet PEUVENT être contraints de la même manière que les noms distinctifs de sujet en utilisant les extensions Contraintes de nom comme décrit au paragraphe 4.2.1.10.

Si l'extension subjectAltName est présente, la séquence DOIT contenir au moins une entrée. À la différence du champ Sujet, les CA conformes NE DOIVENT PAS produire de certificats avec des subjectAltName contenant des champs GeneralName vides. Par exemple, un rfc822Name est représenté comme une IA5String. Alors qu'une chaîne vide est une IA5String valide, un tel rfc822Name n'est pas permis par le présent profil. Le comportement des clients qui rencontrent un tel certificat lors du traitement d'un chemin de certification n'est pas défini par le présent profil.

Finalement, la sémantique des noms de remplacement de sujet qui incluent des caractères générique (par exemple, comme bouche trou pour un ensemble de noms) n'est pas traitée par la présente spécification. Les applications qui ont des exigences spécifiques PEUVENT utiliser de tels noms, mais elles doivent en définir la sémantique.

```
IDENTIFIANT D'OBJET id-ce-subjectAltName ::=  { id-ce 17 }

SubjectAltName ::= GeneralNames

GeneralNames ::= TAILLE DE SEQUENCE (1..MAX) DE GeneralName

GeneralName ::= CHOIX {
    otherName                   [0]   OtherName,
    rfc822Name                  [1]   IA5String,
    dNSName                     [2]   IA5String,
    x400Address                 [3]   ORAddress,
    directoryName               [4]   Name,
    ediPartyName                [5]   EDIPartyName,
    uniformResourceIdentifier   [6]   IA5String,
    iPAddress                   [7]   CHAINE D'OCTETS,
    registeredID                [8]   IDENTIFIANT D'OBJET }

OtherName ::= SEQUENCE {
       type-id    IDENTIFIANT D'OBJET,
       value      [0] EXPLICIT ANY DEFINED BY type-id }

EDIPartyName ::= SEQUENCE {
    nameAssigner            [0]   DirectoryString FACULTATIF,
    partyName               [1]   DirectoryString }
```

4.2.1.7 Nom de remplacement du producteur

Comme au paragraphe 4.2.1.6, cette extension est utilisée pour associer les identités de style Internet au producteur de certificat. Le nom de remplacement de producteur DOIT être codé comme au paragraphe 4.2.1.6. Les noms de remplacement de producteur ne sont pas traités au titre de l'algorithme de validation de chemin de certification de la Section 6. (C'est-à-dire que les noms de remplacement de producteur ne sont pas utilisés dans le chaînage de nom et les contraintes de nom ne sont pas appliquées.)

Lorsque elle est présente, les CA conformes DEVRAIENT marquer cette extension comme non critique.

IDENTIFIANT D'OBJET id-ce-issuerAltName ::= { id-ce 18 }

IssuerAltName ::= GeneralNames

4.2.1.8 Attributs de répertoire sujet

L'extension Attributs de répertoire sujet est utilisée pour porter les attributs d'identification (par exemple, nationalité) du sujet. L'extension est définie comme une séquence d'un ou plusieurs attributs. Les CA conformes DOIVENT marquer cette extension comme non critique.

IDENTIFIANT D'OBJET id-ce-subjectDirectoryAttributes ::= { id-ce 9 }

SubjectDirectoryAttributes ::= TAILLE DE SEQUENCE (1..MAX) DE Attribute

4.2.1.9 Contraintes de base

L'extension Contraintes de base identifie si le sujet du certificat est une CA et la profondeur maximum de la certification de chemin valide qui inclut ce certificat.

Le booléen cA indique si la clé publique certifiée peut être utilisée pour vérifier les signatures de certificat. Si le booléen cA n'est pas affirmé, alors le bit keyCertSign dans l'extension d'usage de clé NE DOIT PAS être affirmé. Si l'extension Contraintes de base n'est pas présente dans un certificat de version 3, ou si l'extension est présente mais que le booléen cA n'est pas affirmé, alors la clé publique certifiée NE DOIT PAS être utilisée pour vérifier les signatures de certificat.

Le champ pathLenConstraint n'est significatif que si le booléen cA est affirmé et si l'extension d'usage de clé, si présente, affirme le bit keyCertSign (paragraphe 4.2.1.3). Dans ce cas, il donne le nombre maximum de certificats intermédiaires non auto produits qui peuvent suivre ce certificat dans un chemin de certification valide. (Note : le dernier certificat dans le chemin de certification n'est pas un certificat intermédiaire, et n'est pas inclus dans cette limite. Généralement, le dernier certificat est un certificat d'entité d'extrémité, mais il peut être un certificat de CA.) Une pathLenConstraint de zéro indique qu'aucun certificat de CA non auto produit intermédiaire ne peut suivre dans un chemin de certification valide. Lorsque il apparaît, le champ pathLenConstraint DOIT être supérieur ou égal à zéro. Lorsque pathLenConstraint n'apparaît pas, aucune limite n'est imposée.

Les CA conformes DOIVENT inclure cette extension dans tous les certificats de CA qui contiennent des clés publiques utilisées pour valider les signatures numériques sur les certificats et DOIVENT marquer l'extension comme critique dans de tels certificats. Cette extension PEUT apparaître comme extension critique ou non critique dans les certificats de CA qui contiennent des clés publiques utilisées exclusivement pour des besoins autres que de validation de signatures numériques sur des certificats. De tels certificats de CA incluent ceux qui contiennent des clés publiques utilisées exclusivement pour valider les signatures numériques sur les CRL et ceux qui contiennent des clés publiques de gestion de clé utilisées avec des protocoles d'engagement de certificat. Cette extension PEUT apparaître comme extension critique ou non critique dans les certificats d'entité d'extrémité.

Les CA NE DOIVENT PAS inclure de champ pathLenConstraint sauf si le booléen cA est affirmé et si l'extension d'usage de clé affirme le bit keyCertSign.

IDENTIFIANT D'OBJET id-ce-basicConstraints ::= { id-ce 19 }

```
BasicConstraints ::= SEQUENCE {
     cA                    BOOLEEN DEFAUT FAUX,
     pathLenConstraint     ENTIER (0..MAX) FACULTATIF }
```

4.2.1.10 Contraintes de nom

L'extension Contraintes de nom, qui DOIT n'être utilisée que dans un certificat de CA, indique un espace de noms au sein duquel tous les noms sujets dans les certificats suivants dans le chemin de certification DOIVENT être localisés. Les restrictions s'appliquent au nom distinctif sujet et s'appliquent aux noms de remplacement de sujet. Les restrictions ne s'appliquent que quand la forme de nom spécifiée est présente. Si aucun nom de ce type n'est dans le certificat, le certificat est acceptable.

Les contraintes de noms ne sont pas appliquées aux certificats auto produits (sauf si le certificat est le certificat final dans le chemin). (Cela pourrait empêcher les CA qui utilisent des contraintes de nom d'employer des certificats auto produits pour mettre en œuvre le retournement de clé.)

Les restrictions sont définies en termes de sous arborescences de noms permises ou exclues. Tout nom qui correspond à une restriction dans le champ excludedSubtrees *(sous arborescence exclue)* est invalide sans considération des informations qui apparaissent dans les sous arborescence permises *(permittedSubtrees)*. Les CA conformes DOIVENT marquer cette extension comme critique et NE DEVRAIENT PAS imposer de contrainte de nom sur les formes de noms adresses X.400 *(x400Address)*, nom de partie EDI *(ediPartyName)*, ou identifiant enregistré *(registeredID)*. Les CA conformes NE DOIVENT PAS produire de certificat où la contrainte de nom est une séquence vide. C'est-à-dire que soit le champ permittedSubtrees, soit le champ excludedSubtrees DOIT être présent.

Les applications qui se conforment au présent profil DOIVENT être capables de traiter les contraintes de nom qui sont imposées à la forme de nom directoryName *(nom de répertoire)* et DEVRAIENT être capables de traiter les contraintes de nom qui sont imposées aux formes de nom rfc822Name *(nom de la RFC0822)*, uniformResourceIdentifier *(identifiant de ressource universel)*, dNSName *(nom DNS)*, et iPAddress *(adresse IP)*. Si une extension de contraintes de nom qui est marquée comme critique impose des contraintes sur une forme de nom particulière, et si une instance de cette forme de nom apparaît dans le champ sujet ou l'extension subjectAltName d'un certificat suivant, alors l'application DOIT soit traiter la contrainte, soit rejeter le certificat.

Dans le présent profil, les champs de minimum et maximum ne sont pas utilisés avec une forme de nom, et donc, le minimum DOIT être zéro, et le maximum DOIT être absent. Cependant, si une application rencontre une extension critique de contraintes de nom qui spécifie d'autres valeurs pour minimum ou maximum pour une forme de nom qui apparaît dans un certificat suivant, l'application DOIT soit traiter ces champs, soit rejeter le certificat.

Pour les URI, la contrainte s'applique à la partie hôte du nom. La contrainte DOIT être spécifiée comme un nom de domaine pleinement qualifié et PEUT spécifier un hôte ou un domaine. Des exemples pourraient être "hôte.exemple.com" et ".exemple.com". Quand la contrainte commence par un point, elle PEUT être étendue par une ou plusieurs étiquettes. C'est-à-dire que la contrainte ".exemple.com" est satisfaite par hôte.exemple.com et par mon.hôte.exemple.com. Cependant, la contrainte ".exemple.com" n'est pas satisfaite par "exemple.com". Quand la contrainte ne commence pas par un point, elle spécifie un hôte. Si une contrainte est appliquée à la forme de nom uniformResourceIdentifier *(identifiant de ressource universel)* et si un certificat suivant inclut une extension subjectAltName avec un uniformResourceIdentifier qui n'inclut pas un composant d'autorité avec un nom d'hôte spécifié comme nom de domaine pleinement qualifié (par exemple, si l'URI n'inclut pas un composant d'autorité ou inclut un composant d'autorité dans lequel le nom d'hôte est spécifié comme une adresse IP) alors l'application DOIT rejeter le certificat.

Une contrainte de nom pour des adresses de messagerie Internet PEUT spécifier une boîte aux lettres particulière chez un hôte particulier, ou toutes les boîtes aux lettres dans un domaine. Pour indiquer une boîte aux lettres particulière, la contrainte est l'adresse de messagerie complète. Par exemple, "racine@exemple.com" indique la boîte aux lettres racine sur l'hôte "exemple.com". Pour indiquer toutes les adresses de messagerie Internet sur un hôte particulier, la contrainte est spécifiée par le nom d'hôte. Par exemple, la contrainte "exemple.com" est satisfaite par toute adresse de messagerie chez l'hôte "exemple.com". Pour spécifier une adresse au sein d'un domaine, la contrainte est spécifiée avec un point en tête (comme avec les URI). Par exemple, ".exemple.com" indique toutes les adresses de messagerie Internet dans le domaine "exemple.com", mais pas les adresses de messagerie Internet chez l'hôte "exemple.com".

Les restrictions de nom DNS sont exprimées comme hôte.exemple.com. Tout nom DNS qui peut être construit en ajoutant simplement zéro, une ou plusieurs étiquettes à gauche du nom satisfait à la contrainte de nom. Par exemple, www.hôte.exemple.com satisferait à la contrainte mais pas hôte1.exemple.com.

Il existe des mises en œuvre traditionnelles d'adresse de messagerie électronique où l'adresse est incorporée dans le nom distinctif sujet dans un attribut de type emailAddress (paragraphe 4.1.2.6). Quand des contraintes sont imposées à la forme de nom rfc822Name, mais où le certificat n'inclut pas de nom de remplacement de sujet, la contrainte rfc822Name DOIT être appliquée à l'attribut de type emailAddress dans le nom distinctif sujet. La syntaxe ASN.1 pour emailAddress et l'IOD correspondant est fournie à l'Appendice A.

Les restrictions de la forme directoryName DOIVENT être appliquées au champ Sujet dans le certificat (quand le certificat inclut un champ Sujet non vide) et à tous les noms de type directoryName dans l'extension subjectAltName. Les restrictions de la forme x400Address DOIVENT être appliquées à tout nom de type x400Address dans l'extension subjectAltName.

Quand elle applique des restrictions de la forme directoryName, une mise en œuvre DOIT comparer les attributs du DN. Au minimum, les mises en œuvre DOIVENT effectuer les règles de comparaison de DN spécifiées au paragraphe 7.1. Les CA qui produisent des certificats avec une restriction de la forme directoryName NE DEVRAIENT PAS compter sur une mise en œuvre complète de l'algorithme de comparaison de nom DN ISO. Cela implique que les restrictions de nom DOIVENT être déclarées de façon identique au codage utilisé dans le champ Sujet ou l'extension subjectAltName.

La syntaxe de iPAddress DOIT être celle décrite au paragraphe 4.2.1.6 avec les ajouts suivants spécifiquement pour les contraintes de nom. Pour les adresses IPv4, le champ iPAddress de GeneralName DOIT contenir huit (8) octets, codés dans le style de la RFC 4632 (CIDR) pour représenter une gamme d'adresses [RFC4632]. Pour les adresses IPv6, le champ iPAddress DOIT contenir 32 octets codés de façon similaire. Par exemple, une contrainte de nom pour le sous réseau de "classe C" 192.0.2.0 est représentée par les octets C0 00 02 00 FF FF FF 00, représentant la notation CIDR 192.0.2.0/24 (gabarit 255.255.255.0).

Des règles supplémentaires pour les contraintes de codage et de traitement de nom sont spécifiées à la Section 7.

La syntaxe et la sémantique pour les contraintes de nom pour otherName, ediPartyName, et registeredID ne sont pas définies par la présente spécification, cependant, la syntaxe et la sémantique pour les contraintes de nom pour d'autres formes de nom peuvent être spécifiées dans d'autres documents.

IDENTIFIANT D'OBJET id-ce-nameConstraints ::= { id-ce 30 }

```
NameConstraints ::= SEQUENCE {
      permittedSubtrees       [0]    GeneralSubtrees FACULTATIF,
      excludedSubtrees        [1]    GeneralSubtrees FACULTATIF }

GeneralSubtrees ::= TAILLE DE SEQUENCE (1..MAX) DE GeneralSubtree

GeneralSubtree ::= SEQUENCE {
      base                    GeneralName,
      minimum         [0]     BaseDistance DEFAUT 0,
      maximum         [1]     BaseDistance FACULTATIF }

BaseDistance ::= ENTIER (0..MAX)
```

4.2.1.11 Contraintes de politique

L'extension Contraintes de politique peut être utilisée dans des certificats produits par les CA. L'extension Contraintes de politique contraint la validation de chemin de deux façons. Elle peut être utilisée pour interdire une transposition de politique ou pour exiger que chaque certificat dans un chemin contienne un identifiant de politique acceptable.

Si le champ inhibitPolicyMapping est présent, la valeur indique le nombre de certificats supplémentaires qui peuvent apparaître dans le chemin avant que la transposition de politique ne soit plus permise. Par exemple, une valeur de un indique que la transposition de politique peut être traitée dans les certificats produits par le sujet de ce certificat, mais pas dans des certificats supplémentaires sur le chemin.

Si le champ requireExplicitPolicy est présent, la valeur de requireExplicitPolicy indique le nombre de certificats supplémentaires qui peuvent apparaître dans le chemin avant qu'une politique explicite soit exigée pour le chemin entier. Quand une politique explicite est exigée, il est nécessaire que tous les certificats dans le chemin contiennent un identifiant de politique acceptable dans l'extension Politiques de certificat. Un identifiant de politique acceptable est l'identifiant d'une politique exigée par l'utilisateur du chemin de certification ou l'identifiant d'une politique qui a été déclarée équivalente par une transposition de politique.

Les applications conformes DOIVENT être capables de traiter le champ requireExplicitPolicy et DEVRAIENT être capables de traiter le champ inhibitPolicyMapping. Les applications qui prennent en charge le champ inhibitPolicyMapping DOIVENT aussi prendre en charge l'extension policyMappings. Si l'extension policyConstraints est marquée comme critique et si le champ inhibitPolicyMapping est présent, les applications qui ne mettent pas en œuvre la prise en charge du champ inhibitPolicyMapping DOIVENT rejeter le certificat.

Les CA conformes NE DOIVENT PAS produire des certificats où les contraintes de politique sont une séquence vide. C'est-à-dire que soit le champ inhibitPolicyMapping, soit le champ requireExplicitPolicy DOIT être présent. Le comportement des clients qui rencontrent un champ Contraintes de politique vide n'est pas traité dans le présent profil.

Les CA conformes DOIVENT marquer cette extension comme critique.

IDENTIFIANT D'OBJET id-ce-policyConstraints ::= { id-ce 36 }

PolicyConstraints ::= SEQUENCE {
requireExplicitPolicy [0] SkipCerts FACULTATIF,
inhibitPolicyMapping [1] SkipCerts FACULTATIF }

SkipCerts ::= ENTIER (0..MAX)

4.2.1.12 Usage de clé étendu

Cette extension indique un ou plusieurs objets pour lesquels la clé publique certifiée peut être utilisée, en plus ou à la place des objets de base indiqués dans l'extension d'usage de clé. En général, cette extension ne va apparaître que dans un certificat d'entité d'extrémité. Cette extension est définie comme suit :

IDENTIFIANT D'OBJET : id-ce-extKeyUsage := { id-ce 37 }

ExtKeyUsageSyntax ::= TAILLE DE SEQUENCE (1..MAX) DE KeyPurposeId

KeyPurposeId ::= IDENTIFIANT D'OBJET

L'objet de la clé peut être défini par toute organisation qui a un besoin. Les identifiants d'objets utilisés pour identifier les objets de clé DOIVENT être alloués en accord avec l'IANA ou la Recommandation UIT-T X.660 [X.660].

Cette extension PEUT, au choix du producteur du certificat, être critique ou non critique.

Si l'extension est présente, alors le certificat DOIT être seulement utilisé pour un des objets indiqués. Si plusieurs objets sont indiqués, l'application n'a pas besoin de reconnaître tous les objets indiqués, tant que l'objet prévu est présent. Les applications qui utilisent des certificats PEUVENT exiger que l'extension d'usage de clé étendu soit présente et qu'un objet particulier soit indiqué afin que le certificat soit acceptable pour cette application.

Si une CA inclut des usages de clé étendus pour satisfaire de telles applications, mais ne souhaite pas restreindre les usages de la clé, la CA peut inclure le KeyPurposeId spécial anyExtendedKeyUsage en plus des objets de clé particuliers exigés par les applications. Les CA conformes NE DEVRAIENT PAS marquer cette extension comme critique si le KeyPurposeId anyExtendedKeyUsage est présent. Les applications qui exigent la présence d'un objet particulier PEUVENT rejeter les certificats qui incluent l'OID anyExtendedKeyUsage mais pas l'OID particulier attendu par l'application.

Si un certificat contient à la fois une extension d'usage de clé et une extension d'usage de clé étendue, les deux extensions DOIVENT alors être traitées indépendamment et le certificat DOIT être seulement utilisé pour un objet cohérent pour les deux extensions. Si il n'y a pas d'objet cohérent pour les deux extensions, le certificat NE DOIT alors être utilisé pour aucun objet.

Les objets d'usage de clé suivants sont définis :

IDENTIFIANT D'OBJET anyExtendedKeyUsage ::= { id-ce-extKeyUsage 0 }

IDENTIFIANT D'OBJET id-kp ::= { id-pkix 3 }

IDENTIFIANT D'OBJET id-kp-serverAuth ::= { id-kp 1 }
-- authentification de serveur TLS WWW
-- bits d'usage de clé qui peuvent être cohérents : digitalSignature, keyEncipherment ou keyAgreement

IDENTIFIANT D'OBJET id-kp-clientAuth ::= { id-kp 2 }
-- bits d'usage de clé d'authentification de client TLS WWW qui peuvent être cohérents : digitalSignature et/ou keyAgreement

IDENTIFIANT D'OBJET id-kp-codeSigning ::= { id-kp 3 }
-- Signature des bits d'usage de clé de code exécutable téléchargeable qui peuvent être cohérents : digitalSignature

IDENTIFIANT D'OBJET id-kp-emailProtection ::= { id-kp 4 }
-- bits d'usage de clé de protection de messagerie électronique qui peuvent être cohérents : digitalSignature, nonRepudiation, et/ou (keyEncipherment ou keyAgreement)

IDENTIFIANT D'OBJET id-kp-timeStamping ::= { id-kp 8 }
-- lien du hachage d'un objet à l'heure des bits d'usage de clé qui peuvent être cohérents : digitalSignature et/ou nonRepudiation

IDENTIFIANT D'OBJET id-kp-OCSPSigning ::= { id-kp 9 }
-- Signature de réponses OCSP
-- bits d'usage de clé qui peuvent être cohérents : digitalSignature et/ou nonRepudiation

4.2.1.13 Points de distribution de CRL

L'extension Points de distribution de CRL identifie comment sont obtenues les informations de CRL. L'extension DEVRAIT être non critique, mais le présent profil RECOMMANDE la prise en charge de cette extension par les CA et les applications. Une discussion plus poussée de la gestion des CRL figure à la Section 5.

L'extension cRLDistributionPoints est une SEQUENCE de DistributionPoint. Un DistributionPoint consiste en trois champs, dont chacun est facultatif : distributionPoint, raisons, et cRLIssuer. Bien que chacun de ces champs soit facultatif, un DistributionPoint NE DOIT PAS consister seulement en le champ raisons ; distributionPoint ou cRLIssuer DOIT être présent. Si le producteur de certificat n'est pas le producteur de CRL, le champ cRLIssuer DOIT alors être présent et contenir le nom du producteur de CRL. Si le producteur de certificat est aussi le producteur de CRL, alors les CA conformes DOIVENT omettre le champ cRLIssuer et DOIVENT inclure le champ distributionPoint.

Quand le champ distributionPoint est présent, il contient soit une SEQUENCE de noms généraux, soit une seule valeur, nameRelativeToCRLIssuer. Si le DistributionPointName contient plusieurs valeurs, chaque nom décrit un mécanisme différent pour obtenir la même CRL. Par exemple, la même CRL pourrait être disponible pour restitution par LDAP et HTTP.

Si le champ distributionPoint contient un directoryName, l'entrée pour ce directoryName contient la CRL actuelle pour les raisons associées et la CRL est produite par le cRLIssuer associé. La CRL peut être mémorisée dans l'attribut certificatRevocationList ou authorityRevocationList. La CRL est obtenue par l'application à partir de tout serveur de répertoire configuré localement. Le protocole qu'utilise l'application pour accéder au répertoire (par exemple, DAP ou LDAP) est une affaire locale.

Si le DistributionPointName contient un nom général de type URI, la sémantique suivante DOIT être supposée : l'URI est un pointeur sur la CRL courante pour les raisons associées et va être produit par le cRLIssuer associé. Quand le schéma d'URI HTTP ou FTP est utilisé, l'URI DOIT pointer sur une seule CRL codée en DER comme spécifié dans la [RFC2585]. Les mises en œuvre de serveur HTTP auxquelles accède l'URI DEVRAIENT spécifier le type de support application/pkix-crl dans le champ d'en-tête Type de contenu de la réponse. Quand le schéma d'URI LDAP [RFC4516] est utilisé, l'URI DOIT inclure un champ <dn> contenant le nom distinctif de l'entrée contenant la CRL, DOIT inclure une seule <attrdesc> qui contient une description d'attribut appropriée pour l'attribut qui contient la CRL [RFC4523], et DEVRAIT inclure un <host> (par exemple, <ldap://ldap.exemple.com/cn=exemple%20CA,dc=exemple,dc=com? certificateRevocationList ;binary>). Omettre le <host> (par exemple, <ldap:///cn=CA,dc=exemple,dc=com?authorityRevocationList;binary>) a pour effet de s'appuyer sur la connaissance a priori que le client pourrait avoir pour contacter un serveur approprié. Quand il est présent, DistributionPointName DEVRAIT inclure au moins un URI LDAP ou HTTP.

Si le DistributionPointName contient la seule valeur nameRelativeToCRLIssuer, elle donne un fragment de nom distinctif. Le fragment est ajouté au nom distinctif X.500 du producteur de CRL pour obtenir le nom du point de distribution. Si le champ cRLIssuer dans le DistributionPoint est présent, alors le fragment de nom est ajouté au nom distinctif qu'il contient ; autrement, le fragment de nom est ajouté au nom distinctif du producteur de certificat. Les CA conformes NE DEVRAIENT PAS utiliser nameRelativeToCRLIssuer pour spécifier des noms de point de distribution. Le DistributionPointName NE DOIT PAS utiliser la solution de remplacement nameRelativeToCRLIssuer quand le cRLIssuer contient plus d'un nom distinctif.

Si le DistributionPoint omet le champ Raisons, la CRL DOIT inclure des informations de révocation pour toutes les raisons. Ce profil RECOMMANDE de ne pas segmenter les CRL par code de cause. Quand une CA conforme inclut une extension cRLDistributionPoints dans un certificat, elle DOIT inclure au moins un DistributionPoint qui pointe sur une CRL qui couvre le certificat pour toutes les raisons.

Le cRLIssuer identifie l'entité qui signe et produit la CRL. Si il est présent, le cRLIssuer DOIT contenir seulement le nom distinctif (DN) provenant du champ Producteur de la CRL sur laquelle le DistributionPoint est pointé. Le codage du nom

dans le champ cRLIssuer DOIT être exactement le même que le codage dans le champ Producteur de la CRL. Si le champ cRLIssuer est inclus et si le DN dans ce champ ne correspond pas à une entrée de répertoire X.500 ou LDAP où la CRL est située, alors les CA conformes DOIVENT inclure le champ distributionPoint.

IDENTIFIANT D'OBJET id-ce-cRLDistributionPoints ::= { id-ce 31 }

CRLDistributionPoints ::= TAILLE DE SEQUENCE (1..MAX) DE DistributionPoint

```
DistributionPoint ::= SEQUENCE {
    distributionPoint   [0]    DistributionPointName FACULTATIF,
    reasons             [1]    ReasonFlags FACULTATIF,
    cRLIssuer           [2]    GeneralNames FACULTATIF }

DistributionPointName ::= CHOIX {
    fullName                    [0]    GeneralNames,
    nameRelativeToCRLIssuer [1]    RelativeDistinguishedName }

ReasonFlags ::= CHAINE BINAIRE {
    unused                  (0),
    keyCompromise           (1),
    cACompromise            (2),
    affiliationChanged      (3),
    superseded              (4),
    cessationOfOperation    (5),
    certificateHold         (6),
    privilegeWithdrawn      (7),
    aACompromise            (8) }
```

4.2.1.14 Inhiber toute politique

L'extension Inhiber toute politique peut être utilisée dans les certificats produits aux CA. L'extension Inhiber toute politique indique que l'OID spécial anyPolicy, avec la valeur { 2 5 29 32 0 }, n'est pas considéré comme une correspondance explicite pour les autres politiques de certificat sauf quand il apparaît dans un certificat de CA intermédiaire auto-produit. La valeur indique le nombre de certificats non auto-produits supplémentaires qui peuvent apparaître dans le chemin avant que anyPolicy ne soit plus permis. Par exemple, une valeur de un indique que anyPolicy peut être traité dans les certificats produits par le sujet de ce certificat, mais pas dans les certificats supplémentaires dans le chemin.

Les CA conformes DOIVENT marquer cette extension comme critique.

IDENTIFIANT D'OBJET id-ce-inhibitAnyPolicy ::= { id-ce 54 }

InhibitAnyPolicy ::= SkipCerts

SkipCerts ::= ENTIER (0..MAX)

4.2.1.15 CRL la plus fraîche (delta de point de distribution de CRL)

L'extension de CRL la plus fraîche identifie comment les informations de delta de CRL sont obtenues. L'extension DOIT être marquée comme non critique par les CA conformes. Les détails de la gestion de CRL sont à la Section 5.

La même syntaxe est utilisée pour cette extension et l'extension cRLDistributionPoints, et est décrite au paragraphe 4.2.1.13. Les mêmes conventions s'appliquent aux deux extensions.

IDENTIFIANT D'OBJET id-ce-freshestCRL ::= { id-ce 46 }

FreshestCRL ::= CRLDistributionPoints

4.2.2 Extensions Internet privées

Ce paragraphe définit deux extensions à utiliser dans l'Infrastructure de clé publique de l'Internet. Ces extensions peuvent être utilisées pour diriger les applications sur des informations en ligne sur le producteur ou le sujet. Chaque extension contient une séquence de méthodes et localisations d'accès. La méthode d'accès est un identifiant d'objet qui indique le type

d'informations disponible. La localisation d'accès est un GeneralName qui spécifie implicitement la situation et le format des informations et la méthode pour les obtenir.

Les identifiants d'objets sont définis pour les extensions privées. Les identifiants d'objet associés aux extensions privées sont définis sous l'arc id-pe au sein de l'arc id-pkix. Toute future extension définie pour la PKI Internet est aussi supposée être définie sous l'arc id-pe.

IDENTIFIANT D'OBJET id-pkix ::= { iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) }

IDENTIFIANT D'OBJET id-pe ::= { id-pkix 1 }

4.2.2.1 Accès aux information d'autorité

L'extension Accès aux informations d'autorité indique comment accéder aux informations et services sur le producteur du certificat dans lequel l'extension apparaît. Les informations et services peuvent inclure des services de validation en ligne et des données de politique de CA. (La localisation des CRL n'est pas spécifiée dans cette extension ; cette information est fournie par l'extension cRLDistributionPoints.) Cette extension peut être incluse dans une entité d'extrémité ou des certificats de CA. Les CA conformes DOIVENT marquer cette extension comme non critique.

IDENTIFIANT D'OBJET id-pe-authorityInfoAccess ::= { id-pe 1 }

AuthorityInfoAccessSyntax ::= TAILLE DE SEQUENCE (1..MAX) DE AccessDescription

```
AccessDescription  ::=  SEQUENCE {
      accessMethod          IDENTIFIANT D'OBJET,
      accessLocation        GeneralName  }
```

IDENTIFIANT D'OBJET id-ad ::= { id-pkix 48 }

IDENTIFIANT D'OBJET id-ad-caIssuers ::= { id-ad 2 }

IDENTIFIANT D'OBJET id-ad-ocsp ::= { id-ad 1 }

Chaque entrée dans la séquence AuthorityInfoAccessSyntax décrit le format et la localisation des informations supplémentaires fournies par le producteur du certificat dans lequel cette extension apparaît. Le type et le format des informations sont spécifiés par le champ accessMethod ; le champ accessLocation spécifie la localisation de l'information. Le mécanisme de restitution peut être impliqué par la méthode d'accès ou spécifié par la localisation de l'accès.

Ce profil définit deux OID de accessMethod : id-ad-caIssuers et id-ad-ocsp.

Dans un certificat de clé publique, l'OID id-ad-caIssuers est utilisé quand des informations supplémentaires font la liste des certificats qui ont été produits à la CA qui a produit le certificat contenant cette extension. La description des producteurs de la CA référencée est destinée à aider les utilisateurs de certificat dans le choix d'un chemin de certification qui se termine à un point de confiance pour l'utilisateur du certificat.

Quand id-ad-caIssuers apparaît comme méthode d'accès, le champ accessLocation décrit le serveur de description référencé et le protocole d'accès pour obtenir la description référencée. Le champ accessLocation est défini comme un nom général qui peut prendre plusieurs formes.

Quand le accessLocation est un nom de répertoire, les informations sont à obtenir par l'application de tout serveur de répertoire configuré en local. L'entrée du nom de répertoire contient des certificats de CA dans les attributs crossCertificatePair et/ou cACertificate comme spécifié dans la [RFC4523]. Le protocole qu'utilise l'application pour accéder au répertoire (par exemple, DAP ou LDAP) est une affaire locale.

Lorsque les informations sont disponibles via LDAP, le accessLocation DEVRAIT être un uniformResourceIdentifier. L'URI LDAP [RFC4516] DOIT inclure un champ <dn> contenant le nom distinctif de l'entrée qui détient les certificats, DOIT inclure un champ <attributes> qui fait la liste des descriptions appropriées d'attributs pour les attributs qui détiennent les certificats ou paires de certificats croisés codés en DER [RFC4523], et DEVRAIT inclure un <host> (par exemple, <ldap://ldap.exemple.com/cn=CA, dc=exemple,dc=com?cACertificate;binary,crossCertificatePair;binary>). Omettre le <host> (par exemple, <ldap:///cn=exempleCA,dc=exemple,dc=com? cACertificate;binary>) a pour effet de reposer sur une connaissance à priori que pourrait avoir le client pour contacter un serveur approprié.

Lorsque les informations sont disponibles via HTTP ou FTP, accessLocation DOIT être un uniformResourceIdentifier et l'URI DOIT pointer sur un seul certificat codé en DER comme spécifié dans la [RFC2585] ou une collection de certificats dans un message de CMS codé en BER ou en DER "certs-only" comme spécifié dans la [RFC2797].

Les applications conformes qui prennent en charge HTTP ou FTP pour accéder aux certificats DOIVENT être capables d'accepter des certificats individuels codés en DER et DEVRAIENT être capables d'accepter les messages "certs-only" de CMS.

Les mises en œuvre de serveur HTTP accédées via l'URI DEVRAIENT spécifier le type de support application/pkix-cert [RFC2585] dans le champ Type de contenu de la réponse pour un seul certificat codé en DER et DEVRAIENT spécifier le type de support application/pkcs7-mime [RFC2797] dans le champ d'en-tête Type de contenu de la réponse pour les messages "certs-only" de CMS. Pour FTP, le nom d'un fichier qui contient un seul certificat codé en DER DEVRAIT avoir un suffixe de ".cer" [RFC2585] et le nom d'un fichier qui contient un message de CMS "certs-only" DEVRAIT avoir un suffixe de ".p7c" [RFC2797]. Les clients consommateurs peuvent utiliser le type de support ou l'extension de fichier comme une indication du contenu, mais ne devraient pas dépendre seulement de la présence du type de support ou extension de fichier correct dans la réponse du serveur.

La sémantique des autres formes de nom id-ad-caIssuers accessLocation n'est pas définie.

Une extension authorityInfoAccess peut inclure plusieurs instances de la méthode d'accès id-ad-caIssuers. Les différentes instances peuvent spécifier différentes méthodes pour accéder aux mêmes informations ou peuvent pointer sur différentes informations. Quand la méthode d'accès id-ad-caIssuers est utilisée, au moins une instance DEVRAIT spécifier une accessLocation qui soit un URI HTTP [RFC2616] ou LDAP [RFC4516].

L'OID id-ad-ocsp est utilisé quand des informations de révocation pour le certificat contenant cette extension sont disponibles en utilisant le protocole d'état de certificat en ligne (OSCP, *Online Certificate Status Protocol*) [RFC2560].

Quand id-ad-ocsp apparaît comme méthode d'accès, le champ accessLocation est la localisation du répondant OCSP, en utilisant les conventions définies dans la [RFC2560].

Des descripteurs d'accès supplémentaires peuvent être définis dans d'autres spécifications PKIX.

4.2.2.2 Accès aux informations de sujet

L'extension Accès aux information de sujet indique comment accéder aux informations et services pour le sujet du certificat dans lequel l'extension apparaît. Quand le sujet est une CA, les informations et services peuvent inclure des services de validation de certificat et des données de politique de CA. Quand le sujet est une entité d'extrémité, les informations décrivent le type des services offerts et comment y accéder. Dans ce cas, le contenu de cette extension est défini dans les spécifications de protocole des services pris en charge. Cette extension peut être incluse dans les certificats d'entité d'extrémité ou de CA. Les CA conformes DOIVENT marquer cette extension comme non critique.

IDENTIFIANT D'OBJET id-pe-subjectInfoAccess ::= { id-pe 11 }

SubjectInfoAccessSyntax ::= TAILLE DE SEQUENCE (1..MAX) DE AccessDescription

```
AccessDescription  ::=  SEQUENCE {
      accessMethod          IDENTIFIANT D'OBJET,
       accessLocation       GeneralName  }
```

Chaque entrée dans la séquence de SubjectInfoAccessSyntax décrit le format et la localisation des informations supplémentaires fournies par le sujet du certificat dans lequel cette extension apparaît. Le type et le format des informations sont spécifiés par le champ accessMethod ; le champ accessLocation spécifie la localisation des informations. Le mécanisme de restitution peut être impliqué par la méthode d'accès ou spécifié par la localisation d'accès.

Ce profil définit une méthode d'accès à utiliser quand le sujet est une CA et une méthode d'accès à utiliser quand le sujet est une entité d'extrémité. Des méthodes d'accès supplémentaires pourront être définies à l'avenir dans les spécifications de protocole d'autres services.

L'OID id-ad-caRepository est utilisé quand le sujet est une CA qui publie des certificats qu'elle produit dans un répertoire. Le champ accessLocation est défini comme un GeneralName, qui peut prendre plusieurs formes.

Quand accessLocation est un directoryName, les informations à obtenir par l'application proviennent de tout serveur de répertoire configuré en local. Quand l'extension est utilisée pour pointer sur des certificats de CA, l'entrée pour le

directoryName contient des certificats de CA dans les attributs crossCertificatePair et/ou cACertificate comme spécifié dans la [RFC4523]. Le protocole qu'utilise l'application pour accéder au répertoire (par exemple, DAP ou LDAP) est une affaire locale.

Lorsque les informations sont disponibles via LDAP, la accessLocation DEVRAIT être un uniformResourceIdentifier (URI). L'URI LDAP [RFC4516] DOIT inclure un champ <dn> contenant le nom distinctif de l'entrée qui détient les certificats, DOIT inclure un champ <attributes> qui fait la liste des descriptions d'attribut appropriées pour les attributs qui détiennent les certificats codés en DER ou les paires de certificats croisés [RFC4523], et DEVRAIT inclure un <host> (par exemple, <ldap://ldap.example.com/cn=CA, dc=example,dc=com?cACertificate;binary,crossCertificatePair;binary>).

Omettre le <host> (par exemple, <ldap:///cn=exampleCA,dc=example,dc=com? cACertificate;binary>) a pour effet de s'appuyer sur la connaissance a priori que le client pourrait avoir pour contacter un serveur approprié.

Lorsque les informations sont disponibles via HTTP ou FTP, accessLocation DOIT être un uniformResourceIdentifier et l'URI DOIT pointer sur un seul certificat codé en DER comme spécifié dans la [RFC2585] ou une collection de certificats dans un message "certs-only" de CMS codé en BER ou DER comme spécifié dans la [RFC2797].

Les applications conformes qui prennent en charge HTTP ou FTP pour accéder aux certificats DOIVENT être capables d'accepter des certificats individuels codés en DER et DEVRAIENT être capables d'accepter des messages de CMS "certs-only".

Les mises en œuvre de serveur HTTP accédées via l'URI DEVRAIENT spécifier le type de support application/pkix-cert [RFC2585] dans le champ Type de contenu de la réponse pour un seul certificat codé en DER et DEVRAIENT spécifier le type de support application/pkcs7-mime [RFC2797] dans le champ d'en-tête Type de contenu de la réponse pour les messages de CMS "certs-only". Pour FTP, le nom d'un fichier qui contient un seul certificat codé en DER DEVRAIT avoir un suffixe de ".cer" [RFC2585] et le nom d'un fichier qui contient un message de CMS "certs-only" DEVRAIT avoir un suffixe de ".p7c" [RFC2797]. Les clients consommateurs peuvent utiliser le type de support ou l'extension de fichier comme une indication du contenu, mais ne devraient pas dépendre seulement de la présence du type correct de support ou d'extension de fichier dans la réponse du serveur.

La sémantique des autres formes de nom id-ad-caRepository accessLocation n'est pas définie.

Une extension subjectInfoAccess peut inclure plusieurs instances de la méthode d'accès id-ad-caRepository. Les différentes instances peuvent spécifier des méthodes différentes pour accéder aux mêmes informations ou peuvent pointer sur des informations différentes. Quand la méthode d'accès id-ad-caRepository est utilisée, au moins une instance DEVRAIT spécifier une accessLocation qui soit un URI HTTP [RFC2616] ou LDAP [RFC4516].

L'OID id-ad-timeStamping est utilisé quand le sujet offre des services d'horodatages utilisant le protocole d'horodatage défini dans la [RFC3161]. Lorsque les services d'horodatage sont disponibles via HTTP ou FTP, accessLocation DOIT être un uniformResourceIdentifier. Lorsque les services d'horodatage sont disponibles via la messagerie électronique, accessLocation DOIT être un rfc822Name. Lorsque les services d'horodatage sont disponibles en utilisant TCP/IP, les formes de nom dNSName ou iPAddress peuvent être utilisées. La sémantique des autres formes de noms de accessLocation (quand la méthode d'accès est id-ad-timeStamping) ne sont pas définies par la présente spécification.

Des descripteurs d'accès supplémentaires peuvent être définis dans d'autres spécifications PKIX.
IDENTIFIANT D'OBJET id-ad ::= { id-pkix 48 }
IDENTIFIANT D'OBJET id-ad-caRepository ::= { id-ad 5 }
IDENTIFIANT D'OBJET id-ad-timeStamping ::= { id-ad 3 }

5. Profil de CRL et d'extensions de CRL

Comme discuté ci-dessus, l'un des objectifs de ce profil CRL X.509 v2 est de favoriser la création d'une PKI Internet interopérable et réutilisable. Pour atteindre cet objectif, des lignes directrices sur l'utilisation des extensions ont été spécifiées, et certaines hypothèses ont été formulées quant à la nature des informations incluses dans la CRL.

Les CRL peuvent être utilisées dans un large éventail d'applications et d'environnements, couvrant un large spectre d'objectifs d'interopérabilité et un spectre encore plus large d'exigences opérationnelles et d'assurance. Ce profil établit une base de référence commune pour les applications génériques nécessitant une large interopérabilité. Le profil définit un ensemble d'informations qui peut être attendu dans chaque CRL. En outre, le profil définit des emplacements communs au sein de la CRL pour les attributs fréquemment utilisés, ainsi que des représentations communes pour ces attributs.

Les producteurs de CRL produisent des CRL. Le producteur de CRL est soit la CA, soit une entité qui a été autorisée par la CA à produire des CRL. Les CA publient les CRL pour fournir des informations d'état sur les certificats qu'elles produisent. Cependant, une CA peut déléguer cette responsibilité à une autre autorité de confiance.

Chaque CRL a une portée particulière. La portée de la liste de révocation de certificats est l'ensemble des certificats qui peuvent apparaître sur une liste de révocation de certificats donnée. Par exemple, la portée pourrait être "tous les certificats produits par la CA X ", "tous les certificats de CA produits par la CA X", "tous les certificats produits par la CA X qui ont été révoqués pour des raisons de clé compromise et de CA compromise", ou un ensemble de certificats fondé sur des informations locales arbitraires, telles que "tous les certificats produits aux employés du NIST situés à Boulder".

Une CRL complète fait la liste de tous les certificats non expirés, dans son champ d'application, qui ont été révoqués pour l'une des raisons de révocation couvertes par le champ d'application de la CRL. Une CRL pleine et complète fait la liste de tous les certificats non expirés produits par une autorité de certification qui ont été révoqués pour une raison quelconque. (Noter que puisque les CA et les producteurs de CRL sont identifiés par le nom, la portée de la CRL n'est pas affectée par la clé utilisée pour signer la CRL ni la ou les clés utilisées pour signer les certificats.)

Si la portée de la CRL inclut un ou plusieurs certificats produits par une entité autre que le producteur de CRL, alors elle est une CRL indirecte. La portée d'une CRL indirecte peut être limitée aux certificats produits par une seule CA ou peut inclure des certificats produit par plusieurs CA. Si le producteur de la CRL indirecte est une CA, alors la portée de la CRL indirecte PEUT aussi inclure des certificats produits par le producteur de la CRL.

Le producteur de CRL PEUT aussi générer des CRL delta. Une CRL delta fait seulement la liste des certificats, dans sa portée, dont le statut de révocation a changé depuis la production d'une CRL complète référencée. La CRL complète référencée est appelée une CRL de base. La portée d'une CRL delta DOIT être la même que celle de la CRL de base qu'elle référence.

Ce profil définit une extension de CRL Internet privée mais ne définit aucune extension d'entrée de CRL privée.

Les environnements avec des exigences supplémentaires ou particulières peuvent s'appuyer sur le présent profil ou peuvent le remplacer.

Les CA conformes ne sont pas obligées de produire des CRL si d'autres mécanismes de révocation ou d'état de certificat sont fournis. Quand des CRL sont produites, les CRL DOIVENT être des CRL de version 2, inclure la date à laquelle la prochaine CRL sera produite dans le champ nextUpdate (paragraphe 5.1.2.5) inclure l'extension Numéro de CRL (paragraphe 5.2.3) et inclure l'extension Identifiant de clé d'autorité (paragraphe 5.2.1). Il est EXIGÉ des applications conformes qui prennent en charge les CRL qu'elles traitent les CRL complètes de version 1 et version 2 qui fournissent les informations de révocation pour tous les certificats produits par une CA. Les applications conformes ne sont pas obligées de prendre en charge le traitement des CRL delta, des CRL indirectes, ou des CRL avec une portée autre que tous les certificats produits par une CA.

5.1 Champs de CRL

La syntaxe de CRL X.509 v2 est la suivante. Pour le calcul de signature, les données à signer sont codées en DER ASN.1. Le codage en DER ASN.1 est un système de codage d'étiquette, longueur, valeur pour chaque élément.

```
CertificateList ::= SEQUENCE  {
    tbsCertList             TBSCertList,
    signatureAlgorithm   AlgorithmIdentifier,
    signatureValue         CHAINE BINAIRE  }

TBSCertList ::= SEQUENCE  {
    version                 Version FACULTATIF,      -- si présent, DOIT être v2
    signature               AlgorithmIdentifier,
    issuer                  Name,
    thisUpdate              Time,
    nextUpdate              Time FACULTATIF,
    revokedCertificates   SEQUENCE DE SEQUENCE  {
        userCertificate        CertificateSerialNumber,
        revocationDate         Time,
      crlEntryExtensions       Extensions FACULTATIF                    -- si présent, version DOIT être v2
                    }  FACULTATIF,
    crlExtensions           [0]  EXPLICIT Extensions FACULTATIF       -- si présent, version DOIT être v2
```

```
}

-- Version, Time, CertificateSerialNumber, et Extensions sont tous définis dans l'ASN.1 du paragraphe 4.1
-- AlgorithmIdentifier est défini au paragraphe 4.1.1.2
```

Les éléments qui suivent décrivent l'utilisation de la CRL X.509 v2 dans la PKI Internet.

5.1.1 Champs de CertificateList

CertificateList est une SEQUENCE de trois champs exigés. Les champs sont décrits en détail dans les paragraphes suivants.

5.1.1.1 tbsCertList

Le premier champ dans la séquence est tbsCertList. Ce champ est lui-même une séquence contenant le nom du producteur, la date de production, la date de production de la prochaine liste, la liste facultative des certificats révoqués, et les extensions de CRL facultatives. Quand il n'y a pas de certificat révoqué, la liste des certificats révoqués est absente. Quand un ou plusieurs certificats sont révoqués, chaque entrée de la liste de certificats révoqués est définie par une séquence de numéro de série de certificat d'utilisateur, date de révocation, et extensions d'entrée de CRL facultative.

5.1.1.2 signatureAlgorithm

Le champ signatureAlgorithm contient l'identifiant d'algorithme pour l'algorithme utilisé par le producteur de CRL pour signer la CertificateList. Le champ est du type AlgorithmIdentifier, qui est défini au paragraphe 4.1.1.2. Les [RFC3279], [RFC4055], et [RFC4491] font la liste des algorithmes pris en charge pour la présente spécification, mais d'autres algorithmes de signature PEUVENT aussi être pris en charge.

Ce champ DOIT contenir le même identifiant d'algorithme que le champ Signature dans la séquence tbsCertList (paragraphe 5.1.2.2).

5.1.1.3 signatureValue

Le champ signatureValue contient une signature numérique calculée sur la tbsCertList ASN.1 codée en DER. La tbsCertList ASN.1 codée en DER est utilisée comme entrée à la fonction de signature. Cette valeur de signature est codée comme une CHAINE BINAIRE et incluse dans le champ signatureValue de CRL. Les détails de ce processus sont spécifiés pour chacun des algorithmes pris en charge dans les [RFC3279], [RFC4055], et [RFC4491].

Les CA qui sont aussi productrices de CRL PEUVENT utiliser une clé privée pour signer numériquement des certificats et des CRL, ou PEUVENT utiliser des clés privées séparées pour signer numériquement les certificats et CRL. Quand des clés privées séparées sont employées, chacune des clés publiques associées à ces clés privées est placée dans un certificat séparé, un avec le bit keyCertSign établi dans l'extension d'usage de clé, et un avec le bit cRLSign établi dans l'extension d'usage de clé (paragraphe 4.2.1.3). Quand des clés privées séparées sont employées, les certificats produits par la CA contiennent un identifiant de clé d'autorité, et les CRL correspondantes contiennent un identifiant de clé d'autorité différent. L'utilisation de certificats de CA séparés pour la validation des signatures de certificat et des signatures de CRL peut offrir des caractéristiques de sécurité améliorées ; cependant, elle impose une charge aux applications, et elle pourrait limiter l'interopérabilité. De nombreuses applications construisent un chemin de certification, et ensuite valident le chemin de certification (Section 6). La vérification de CRL exige à son tour qu'un chemin de certification séparé soit construit et validé pour le certificat de validation de signature de CRL de la CA. Les applications qui effectuent la vérification de CRL DOIVENT prendre en charge la validation de chemin de certification quand les certificats et les CRL sont signés numériquement avec la même clé privée de CA. Ces applications DEVRAIENT prendre en charge la validation de chemin de certification quand les certificats et les CRL sont signées numériquement avec des clés privées de CA différentes.

5.1.2 Liste de certificats "à signer"

La liste des certificats à signer, ou TBSCertList, est une séquence de champs exigés et facultatifs. Les champs exigés identifient le producteur de CRL, l'algorithme utilisé pour signer la CRL, et la date et heure de production de la CRL.

Les champs facultatifs incluent la date et heure à laquelle le producteur de CRL va produire la prochaine CRL, la liste des certificats révoqués, et les extensions de CRL. La liste de certificats révoqués est de prise en charge facultative dans le cas où une CA n'a pas révoqué de certificat non expiré qu'elle a produit. Ce profil exige que les producteurs de CRL conformes incluent le champ nextUpdate et le numéro de CRL et l'identifiant de clé d'autorité d'extensions de CRL dans toutes les CRL produites.

5.1.2.1 Version

Ce champ facultatif décrit la version de la CRL codée. Quand des extensions sont utilisées, comme exigé par le présent profil, cet champ DOIT être présent et DOIT spécifier la version 2 (la valeur d'entier est 1).

5.1.2.2 Signature

Ce champ contient l'identifiant d'algorithme pour l'algorithme utilisé pour signer la CRL. Les [RFC3279], [RFC4055], et [RFC4491] font la liste des OID pour les algorithmes de signature les plus courants utilisés dans la PKI Internet.

Ce champ DOIT contenir le même identifiant d'algorithme que le champ signatureAlgorithm dans la séquence CertificateList (paragraphe 5.1.1.2).

5.1.2.3 Nom du producteur

Le nom du producteur identifie l'entité qui a signé et produit la CRL. L'identité du producteur est portée dans le champ Producteur. Des formes de nom de remplacement peuvent aussi apparaître dans l'extension issuerAltName (paragraphe 5.2.2). Le champ Producteur DOIT contenir un nom distinctif (DN) X.500 non vide. Le champ Producteur est défini comme le nom de type X.501, et DOIT suivre les règles de codage pour le champ Nom de producteur dans le certificat (paragraphe 4.1.2.4).

5.1.2.4 thisUpdate

Ce champ indique la date de production de cette CRL. thisUpdate peut être codé comme UTCTime ou GeneralizedTime.

Les producteurs de CRL conformes au présent profil DOIVENT coder thisUpdate comme UTCTime pour les dates jusqu'à l'année 2049. Les producteurs de CRL conformes au présent profil DOIVENT coder thisUpdate comme GeneralizedTime pour les dates à partir de l'an 2050. Les applications conformes DOIVENT être capables de traiter les dates codées en UTCTime ou GeneralizedTime.

Lorsque codé en UTCTime, thisUpdate DOIT être spécifié et interprété comme défini au paragraphe 4.1.2.5.1. Lorsque codé en GeneralizedTime, thisUpdate DOIT être spécifié et interprété comme défini au paragraphe 4.1.2.5.2.

5.1.2.5 nextUpdate

Ce champ indique la date à laquelle la prochaine CRL va être produite. La prochaine CRL pourrait être produite avant la date indiquée, mais elle ne va pas être produite plus tard que la date indiquée. Les producteurs de CRL DEVRAIENT produire des CRL avec une date nextUpdate égale ou plus tard que toutes les précédentes CRL. nextUpdate peut être codé comme UTCTime ou comme GeneralizedTime.

Les producteurs de CRL conformes DOIVENT inclure le champ nextUpdate dans toutes les CRL. Noter que la syntaxe ASN.1 de TBSCertList décrit ce champ comme FACULTATIF, ce qui est cohérent avec la structure ASN.1 définie dans [X.509]. Le comportement des clients traitant des CRL qui omettent nextUpdate n'est pas spécifié dans le présent profil.

Les producteurs de CRL conformes au présent profil DOIVENT coder nextUpdate comme UTCTime pour les dates jusqu'à l'an 2049. Les producteurs de CRL conformes au présent profil DOIVENT coder nextUpdate comme GeneralizedTime pour les dates à partir de l'an 2050 et après. Les applications conformes DOIVENT être capables de traiter les dates qui sont codées en UTCTime ou en GeneralizedTime.

Lorsque codé comme UTCTime, nextUpdate DOIT être spécifié et interprété comme défini au paragraphe 4.1.2.5.1. Lorsque codé comme GeneralizedTime, nextUpdate DOIT être spécifié et interprété comme défini au paragraphe 4.1.2.5.2.

5.1.2.6 Certificats révoqués

Quand il n'y a pas de certificat révoqué, la liste des certificats révoqués DOIT être absente. Autrement, les certificats révoqués sont mentionnés par leurs numéro de séries. Les certificats révoqués par la CA sont identifiés de façon univoque par le numéro de série du certificat. La date à laquelle la révocation s'est produite est spécifiée. L'heure de revocationDate DOIT être exprimée comme décrit au paragraphe 5.1.2.4. Des informations supplémentaires peuvent être fournies dans les extensions d'entrée de CRL; les extensions d'entrée de CRL sont discutées au paragraphe 5.3.

5.1.2.7 Extensions

Ce champ peut seulement apparaître si la version est 2 (paragraphe 5.1.2.1). Si il est présent, ce champ est une séquence de une ou plusieurs extensions de CRL. Les extensions de CRL sont discutées au paragraphe 5.2.

5.2 Extensions de CRL

Les extensions définies par ANSI X9, ISO/CEI, et l'UIT-T pour les CRL X.509 v2 [X.509] [X9.55] fournissent des méthodes pour associer des attributs supplémentaires aux CRL. Le format de CRL X.509 v2 permet aussi à des communautés de définir des extensions privées pour porter des informations particulières pour ces communautés. Chaque extension dans une CRL peut être désignée comme critique ou non critique. Si une CRL contient une extension critique que l'application ne peut pas traiter, alors l'application NE DOIT PAS utiliser cette CRL pour déterminer le statut des certificats. Cependant, les applications peuvent ignorer les extensions non critiques non reconnues. Les sous paragraphes qui suivent présentent les extensions utilisées dans les CRL Internet. Les communautés peuvent choisir d'inclure des extensions dans des CRL qui ne sont pas définies dans la présente spécification. Cependant, on devrait être prudent pour adopter des extensions critiques dans des CRL qui pourraient être utilisées dans un contexte général.

Il est EXIGÉ des producteurs de CRL conformes d'inclure les extensions Identifiant de clé d'autorité (paragraphe 5.2.1) et Numéro de CRL (paragraphe 5.2.3) dans toute CRL produite.

5.2.1 Identifiant de clé d'autorité

L'extension d'identifiant de clé d'autorité donne un moyen pour identifier la clé publique correspondant à la clé privée utilisée pour signer une CRL. L'identification peut être fondée sur l'identifiant de clé (l'identifiant de clé sujet dans le certificat du signataire de la CRL) ou le nom du producteur et le numéro de série. Cette extension est particulièrement utile lorsque un producteur a plus d'une clé signante, soit à cause de multiples paires de clé concurrentes, soit du fait de leur changement.

Les producteurs de CRL conformes DOIVENT utiliser la méthode d'identifiant de clé, et DOIVENT inclure cette extension dans toute CRL produite.

La syntaxe pour cette extension de CRL est définie au paragraphe 4.2.1.1.

5.2.2 Autre nom de producteur

L'extension Autre nom de producteur permet que des identités supplémentaires soient associées au producteur de la CRL. Les options définies incluent une adresse de messagerie électronique (rfc822Name) un nom DNS, une adresse IP, et un URI. Plusieurs instances d'une forme de nom et plusieurs formes de nom peuvent être incluses. Chaque fois que de telles identités sont utilisées, l'extension Autre nom de producteur DOIT être utilisée ; cependant, un nom DNS PEUT être représenté dans le champ Producteur en utilisant l'attribut domainComponent comme décrit au paragraphe 4.1.2.4.

Les producteurs de CRL conformes DEVRAIENT marquer l'extension issuerAltName comme non critique.

L'OID et la syntaxe pour cette extension de CRL sont définis au paragraphe 4.2.1.7.

5.2.3 Numéro de CRL

Le numéro de CRL est a extension de CRL non critique qui porte un numéro de séquence à croissance monotone pour une certaine portée de CRL et le producteur de CRL. Cette extension permet aux utilisateurs de déterminer facilement quand une CRL particulière supplante une autre CRL. Les numéros de CRL prennent aussi en charge l'identification de CRL complètes complémentaires et de CRL delta. Les producteurs de CRL conformes au présent profil DOIVENT inclure cette extension dans toutes les CRL et DOIVENT marquer cette extension comme non critique.

Si un producteur de CRL génère des CRL delta en plus de CRL complètes pour une certaine portée, les CRL complètes et les CRL delta DOIVENT partager une séquence de numérotation. Si une CRL delta et une CRL complète qui couvrent la même portée sont produites au même moment, elles DOIVENT avoir le même numéro de CRL et fournir les mêmes informations de révocation. C'est-à-dire que la combinaison de la CRL delta et d'une CRL complète acceptable DOIT fournir les mêmes informations de révocation que la CRL complète produite simultanément.

Si un producteur de CRL génère deux CRL (deux CRL complètes, deux CRL delta, ou une CRL complète et une CRL delta) pour la même portée à des moments différents, les deux CRL NE DOIVENT PAS avoir le même numéro de CRL. C'est-à-dire, si le champ thisUpdate (paragraphe 5.1.2.4) dans les deux CRL n'est pas identique, les numéros de CRL DOIVENT être différents.

Étant données les exigences ci-dessus, on peut s'attendre à ce que les numéros de CRL contiennent de grands entiers. Les vérificateurs de CRL DOIVENT être capables de traiter des valeurs de CRLNumber jusqu'à 20 octets. Les producteurs de CRL conformes NE DOIVENNT PAS utiliser des valeurs de CRLNumber de plus de 20 octets.

IDENTIFIANT D'OBJET id-ce-cRLNumber ::= { id-ce 20 }

CRLNumber ::= ENTIER (0..MAX)

5.2.4 Indicateur de CRL delta

L'indicateur de CRL delta est une extension de CRL critique qui identifie une CRL comme étant une CRL delta. Les CRL delta contiennent des mises à jour aux informations de révocation précédemment distribuées, plutôt que toutes les informations qui apparaîtraient dans une CRL complète. L'utilisation des CRL delta peut significativement réduire la charge du réseau et le temps de traitement dans certains environnements. Les CRL delta sont généralement plus petites que les CRL qu'elles mettent à jour, de sorte que les applications qui obtiennent des CRL delta consomment moins de bande passante du réseau que les applications qui obtiennent les CRL complètes correspondantes. Les applications qui mémorisent les informations de révocation dans un format autre que la structure de CRL peuvent ajouter de nouvelles informations de révocation à la base de données locale sans retraiter les informations.

L'extension Indicateur de CRL delta contient la seule valeur de type BaseCRLNumber. Le numéro de CRL identifie la CRL, complète pour une certaine portée, qui a été utilisée comme point de départ de la génération de cette CRL delta. Un producteur de CRL conforme DOIT publier la CRL de base référencée comme une CRL complète. La CRL delta contient toutes les mises à jour du statut de révocation pour cette même portée. La combinaison d'une CRL delta plus la CRL de base référencée est équivalente à une CRL complète, pour la portée applicable, au moment de la publication de la CRL delta.

Quand un producteur de CRL conforme génère une CRL delta, la CRL delta DOIT include une extension critique d'indicateur de CRL delta.

Quand une CRL delta est produite, elle DOIT couvrir le même ensemble de raisons et le même ensemble de certificats que couverts dans la CRL de base qu'elle référence. C'est-à-dire, la portée de la CRL delta DOIT être la même que la portée de la CRL complète référencée comme base. La CRL de base référencée et la CRL delta DOIVENT omettre l'extension de point de distribution producteur ou contenir des extensions de point de distribution producteur identiques. De plus, le producteur de CRL DOIT utiliser la même clé privée pour signer la CRL delta et toute CRL complète qu'elle met à jour.

Une application qui prend en charge les CRL delta peut construire une CRL qui est complète pour une certaine portée en combinant une CRL pour cette portée avec une CRL produite qui est complète pour cette portée ou une CRL construite localement qui est complète pour cette portée.

Quand une CRL delta est combinée avec une CRL complète ou une CRL construite localement, la CRL construite localement résultante a le numéro de CRL spécifié dans l'extension Numéro de CRL trouvé dans la CRL delta utilisée dans sa construction. De plus, la CRL construite localement résultante a les temps thisUpdate et nextUpdate spécifiés dans les champs correspondants de la CRL delta utilisée dans sa construction. De plus, la CRL construite localement hérite du point de distribution producteur provenant de la CRL delta.

Une CRL complète et une CRL delta PEUVENT être combinées si les quatre conditions suivantes sont satisfaites :

(a) La CRL complète et la CRL delta ont le même producteur.

(b) La CRL complète et la CRL delta ont la même portée. Les deux CRL ont la même portée si d'une des conditions suivante est satisfaite :
   (1) L'extension issuingDistributionPoint est omise pour la CRL complète et pour la CRL delta.
   (2) L'extension issuingDistributionPoint est présent dans la CRL complète et dans la CRL delta, et les valeurs pour chacun des champs dans les extensions sont les mêmes dans les deux CRL.

(c) Le numéro de CRL de la CRL complète est égal ou supérieur au BaseCRLNumber spécifié dans la CRL delta. C'est-à-dire, la CRL complète contient (au minimum) toutes les informations de révocation détenues par la CRL de base référencée.

(d) Le numéro de CRL de la CRL complète est inférieur au numéro de CRL de la CRL delta. C'est-à-dire, la CRL delta suit la CRL complète dans la séquence de numérotation.

Les producteurs de CRL DOIVENT s'assurer que la combinaison d'une CRL delta et de toute CRL complète appropriée reflète précisément le statut de révocation actuel. Le producteur de CRL DOIT inclure une entrée dans la CRL delta pour chaque certificat dans la portée de la CRL delta dont le statut a changé depuis la génération de la CRL de base référencée :

(a) Si le certificat est révoqué pour une raison incluse dans la portée de la CRL, marquer le certificat comme révoqué.

(b) Si le certificat est valide et a été marqué sur la CRL de base référencée ou toute CRL suivante avec le code de cause certificateHold, et si le code de cause certificateHold est inclus dans la portée de la CRL, marquer le certificat avec le code de cause removeFromCRL.

(c) Si le certificat est révoqué pour une raison en dehors de la portée de la CRL, mais si le certificat était marqué sur la CRL de base référencée ou toute CRL suivante avec un code de cause inclus dans la portée de cette CRL, marquer le certificat comme révoqué mais omettre le code de cause.

(d) Si le certificat est révoqué pour une raison en dehors de la portée de la CRL et si le certificat n'était ni marqué sur la CRL de base référencée ni sur une autre CRL suivante avec un code de cause inclus dans la portée de cette CRL, ne pas marquer le certificat sur cette CRL.

Le statut d'un certificat est considéré avoir changé si il est révoqué (pour toute raison de révocation, incluant certificateHold) si il est libéré de garde, ou si sa rasion de révocation change.

Il est approprié de marquer un certificat avec le code de cause removeFromCRL sur une CRL delta même si le certificat n'était pas en garde dans la CRL de base référencée. Si le certificat était placé en garde dans une CRL produite après la CRL de base mais avant cette CRL delta et ensuite libére de garde, il DOIT être marqué sur la CRL delta avec la raison de révocation removeFromCRL.

Un producteur de CRL PEUT facultativement marquer un certificat sur une CRL delta avec le code de cause removeFromCRL si le temps notAfter spécifié dans le certificat précède l'heure thisUpdate spécifiée dans la CRL delta et si le certificat était marqué sur la CRL de base référencée ou sur toute CRL produite après la CRL de base mais avant cette CRL delta.

Si une notification de révocation de certificat apparaît en premier dans une CRL delta, il est alors possible que la période de validité du certificat expire avant que la prochaine CRL complète pour la même portée soit produite. Dans ce cas, la notification de révocation DOIT être incluse dans toutes les CRL delta suivantes jusqu'à ce que la notification de révocation soit incluse dans au moins une CRL complète explicitement produite pour cette portée.

Une application qui prend en charge les CRL delta DOIT être capable de construire une CRL complète courante en combinant une CRL complète précédemment produite et la plus récente CRL delta. Une application qui prend en charge les CRL delta PEUT aussi être capable de construire une CRL complète courante en combinant une CRL complète construite précédement en local et la CRL delta courante. Une CRL delta est considérée être celle en cours si l'heure actuelle est entre les heures contenues dans les champs thisUpdate et nextUpdate. Dans certaines circonstances, le producteur de CRL peut publier une ou plusieurs CRL delta avant l'heure indiquée par le champ nextUpdate. Si la CRL delta la plus actuelle pour une portée donnée est rencontrée, l'application DEVRAIT considérer celle avec la dernière valeur dans thisUpdate comme étant la plus récente.

IDENTIFIANT D'OBJET id-ce-deltaCRLIndicator ::= { id-ce 27 }

BaseCRLNumber ::= CRLNumber

5.2.5 Point de distribution producteur

Le point de distribution producteur est une extension de CRL critique qui identifie le point de distribution et la portée de CRL pour une CRL particulière, et il indique si la CRL couvre la révocation pour des certificats d'entité d'extrémité seulement, des certificats de CA seulement, des certificats d'attribut seulement, ou un ensemble limité de codes de cause. Bien que l'extension soit critique, les mises en œuvre conformes ne sont pas obligées de prendre en charge cette extension. Cependant, les mises en œuvre qui ne prennent pas en charge cette extension DOIVENT soit traiter le statut de tout certificat ne figurant pas sur cette CRL comme inconnu, soit localiser une autre CRL qui ne contient pas d'extensions critiques non reconnues.

La CRL est signée en utilisant la clé privée du producteur de CRL. Les points de distribution de CRL n'ont pas leurs propres paires de clés. Si la CRL est mémorisée dans le répertoire X.500, elle est mémorisée dans l'entrée de répertoire correspondant au point de distribution de CRL, qui peut être différent de l'entrée de répertoire du producteur de CRL.

Les codes de cause associés à un point de distribution DOIVENT être spécifiés dans onlySomeReasons. Si onlySomeReasons n'apparaît pas, le point de distribution DOIT contenir des révocations pour tous les codes de cause. Les

CA peuvent utiliser les points de distribution de CRL pour partager la CRL sur la base des révocations pour compromission et de routine. Dans ce cas, les révocations avec le code de cause keyCompromise (1), cACompromise (2), et aACompromise (8) apparaissent sur un point de distribution, et les révocations avec d'autres codes de cause apparaissent dans un autre point de distribution.

Si une CRL inclut une extension issuingDistributionPoint avec onlySomeReasons présent, alors chaque certificat dans la portée de la CRL qui est révoquée DOIT avoir une raison de révocation allouée autre que non spécifiée. La raison de révocation allouée est utilisée pour déterminer sur quelles CRL marquer le certificat révoqué, cependant, il n'est pas exigé d'inclure l'extension d'entrée de CRL reasonCode dans l'entrée de CRL correspondante.

La syntaxe et la sémantiquc du champ distributionPoint sont les mêmes que pour le champ distributionPoint dans l'extension cRLDistributionPoints (paragraphe 4.2.1.13). Si le champ distributionPoint est présent, il DOIT alors inclure au moins un des noms provenant du champ distributionPoint correspondant de l'extension cRLDistributionPoints de chaque certificat qui est dans la portée de cette CRL. Le codage identique DOIT être utilisé dans les champs distributionPoint du certificat et de la CRL.

Si le champ distributionPoint est absent, la CRL DOIT contenir des entrées pour tous les certificats révoqués non expirés produits par le producteur de CRL, si il en est, dans la portée de la CRL.

Si la portée de la CRL inclut seulement des certificats produits par le producteur de CRL, alors le booléen indirectCRL DOIT être réglé à FAUX. Autrement, si la portée de la CRL inclut des certificats produits par une ou plusieurs autorités autres que le producteur de CRL, le booléen indirectCRL DOIT être réglé à VRAI. L'autorité responsable de chaque entrée est indiquée par l'extension d'entrée de producteur de certificat de CRL (paragraphe 5.3.3).

Si la portée de la CRL inclut seulement des certificats de clé publique d'entité d'extrémité, alors onlyContainsUserCerts DOIT être réglé à VRAI. Si la portée de la CRL inclut seulement des certificats de CA, alors onlyContainsCACerts DOIT être réglé à VRAI. Si onlyContainsUserCerts ou onlyContainsCACerts est réglé à VRAI, alors la portée de la CRL NE DOIT PAS inclure de certificat de version 1 ou de version 2. Les producteurs de CRL conformes DOIVENT régler le booléen onlyContainsAttributeCerts à FAUX.

Les producteurs de CRL conformes NE DOIVENT PAS produire des CRL où le codage en DER de l'extension de point de distribution producteur est une séquence vide. C'est-à-dire que si onlyContainsUserCerts, onlyContainsCACerts, indirectCRL, et onlyContainsAttributeCerts sont tous FAUX, alors le champ distributionPoint field ou le champ onlySomeReasons DOIT être présent.

IDENTIFIANT D'OBJET id-ce-issuingDistributionPoint ::= { id-ce 28 }

```
IssuingDistributionPoint ::= SEQUENCE {
    distributionPoint          [0] DistributionPointName FACULTATIF,
    onlyContainsUserCerts      [1] BOOLÉEN FAUX PAR DÉFAUT,
    onlyContainsCACerts        [2] BOOLEAN DEFAULT FAUX,
    onlySomeReasons            [3] ReasonFlags FACULTATIF,
    indirectCRL                [4] BOOLÉEN FAUX PAR DÉFAUT,
    onlyContainsAttributeCerts [5] BOOLÉEN FAUX PAR DÉFAUT }

-- au plus un de onlyContainsUserCerts, onlyContainsCACerts, et onlyContainsAttributeCerts peut être réglé à VRAI.
```

5.2.6 CRL la plus fraîche (point de distribution de delta de CRL)

La plus fraîche extension de CRL identifie comment les informations de CRL delta pour cette CRL complète sont obtenues. Les producteurs de CRL conformes DOIVENT marquer cette extension comme non critique. Cette extension NE DOIT PAS apparaître dans les CRL delta.

La même syntaxe est utilisée pour cette extension que dans l'extension de certificat cRLDistributionPoints, et est décrite au paragraphe 4.2.1.13. Cependant, seul le champ point de distribution est significatif dans ce contexte. Les champs Raisons et cRLIssuer DOIVENT être omis de cette extension de CRL.

Chaque nom de point de distribution fournit la localisation à laquelle une CRL delta pour cette CRL complète peut être trouvée. La portée de ces CRL delta DOIT être la même que la portée de cette CRL complète. Le contenu de cette extension de CRL est seulement utilisé pour localiser les CRL delta ; les contenus ne sont pas utilisés pour valider la CRL ou les CRL delta référencées. Les conventions de codage définies pour les points de distribution au paragraphe 4.2.1.13 s'appliquent à cette extension.

IDENTIFIANT D'OBJET id-ce-freshestCRL ::= { id-ce 46 }

FreshestCRL ::= CRLDistributionPoints

5.2.7 Accès aux informations d'autorité

Ce paragraphe définit l'utilisation de l'extension Accès aux informations d'autorité dans une CRL. La syntaxe et la sémantique définies au paragraphe 4.2.2.1 pour l'extension de certificat sont aussi utilisées pour l'extension de CRL.

Cette extension de CRL DOIT être marquée comme non critique.

Quand elle est présente dans une CRL, cette extension DOIT inclure au moins une AccessDescription spécifiant id-ad-caIssuers comme la méthode d'accès. L'OID id-ad-caIssuers est utilisé quand les informations disponibles font la liste des certificats qui peuvent être utilisés pour vérifier la signature sur la CRL (c'est-à-dire, des certificats qui ont un nom de sujet qui correspond au nom du producteur sur la CRL et ont une clé publique sujette qui correspond à la clé privée utilisée pour signer la CRL). Les types de méthode d'accès autres que id-ad-caIssuers NE DOIVENT PAS être inclus. Au moins une instance de AccessDescription DEVRAIT spécifier une accessLocation qui est un URI HTTP [RFC2616] ou LDAP [RFC4516].

Lorsque les informations sont disponibles via HTTP ou FTP, accessLocation DOIT être un identifiant de ressource universel et cet URI DOIT pointer sur un seul certificat codé en DER comme spécifié dans la [RFC2585] ou sur une collection de certificats dans un message de CMS "certs-only" codé en BER ou DER comme spécifié dans la [RFC2797].

Les applications conformes qui prennent en charge HTTP ou FTP pour accéder aux certificats DOIVENT être capables d'accepter des certificats individuels codés en DER et DEVRAIENT être capables d'accepter les messages "certs-only" de CMS.

Les mises en œuvre de serveur HTTP accédées via l'URI DEVRAIENT spécifier le type de support application/pkix-cert [RFC2585] dans le champ d'en-tête content-type de la réponse pour un seul certificat codé en DER et DEVRAIENT spécifier le type de support application/pkcs7-mime [RFC2797] dans le champ d'en-tête content-type de la réponse pour les messages de CMS "certs-only". Pour FTP, le nom d'un fichier qui contient un seul certificat codé en DER DEVRAIT avoir un suffixe de ".cer" [RFC2585] et le nom d'un fichier qui contient un message de CMS "certs-only" DEVRAIT avoir un suffixe de ".p7c" [RFC2797]. Les clients consommateurs peuvent utiliser le type de support ou l'extension de fichier comme indication du contenu, mais ne devraient pas dépendre seulement de la présence du type de support ou extension de fichier correct dans la réponse du serveur.

Quand la localisation d'accès est un nom de répertoire, les informations sont obtenues par l'application à partir de laquelle le serveur de répertoire est configuré en local. Quand une clé publique de CA est utilisée pour valider les signatures sur les certificats et les CRL, le certificat de CA désiré est mémorisé dans les attributs crossCertificatePair et/ou cACertificate comme spécifié dans la [RFC4523]. Quand différentes clés publiques sont utilisées pour valider les signatures sur les certificats et les CRL, le certificat désiré est mémorisé dans l'attribut userCertificate comme spécifié dans la [RFC4523]. Donc, les mises en œuvre qui prennent en charge la forme directoryName de accessLocation DOIVENT être prêtes à trouver le certificat nécessaire dans un de ces trois attributs. Le protocole qu'utilise une application pour accéder au répertoire (par exemple, DAP ou LDAP) est une affaire locale.

Lorsque les informations sont disponibles via LDAP, la localisation d'accès DEVRAIT être un URI. L'URI LDAP [RFC4516] DOIT inclure un champ <dn> contenant le nom distinctif de l'entrée qui contient les certificats, DOIT inclure un champ <attributes> qui fait la liste des descriptions d'attribut appropriées pour les attributs qui contiennent les certificats codés en DER ou les paires de certificats croisés [RFC4523], et DEVRAIT inclure un <host> (par exemple, <ldap://ldap.example.com/cn=CA, dc=example,dc=com?cACertificate;binary,crossCertificatePair;binary>). Omettre le <host> (par exemple, <ldap:///cn=exampleCA,dc=example,dc=com? cACertificate;binary>) a pour effet de s'appuyer sur une connaissance à priori que le client pourrait avoir pour contacter un serveur approprié.

5.3 Extensions d'entrée de CRL

Les extensions d'entrée de CRL définies par l'ISO/CEI, l'UIT-T, et ANSI X9 pour les CRL X.509 v2 fournissent des méthodes pour associer des attributs supplémentaires aux entrées de CRL [X.509] [X9.55]. Le format de CRL X.509 v2 permet aussi aux communautés de définir des extensions privées d'entrée de CRL pour porter des informations uniques pour ces communautés. Chaque extension dans une entrée de CRL peut être désignée comme critique ou non critique. Si une CRL contient une extension d'entrée critique de CRL que l'application ne peut pas traiter, alors l'application NE DOIT

PAS utiliser cette CRL pour déterminer le statut des certificats. Cependant, les applications peuvent ignorer les extensions d'entrée de CRL non critiques non reconnues.

Les sous pararaphes qui suivent présentent les extensions recommandées utilisées dans les entrées de CRL Internet et les localisations standard pour information. Des communautés peuvent choisir d'utiliser des extensions d'entrée de CRL suplémentaires ; cependant, la prudence devrait être de mise pour adopter toute extension d'entrée de CRL critique dans des CRL qui pourraient être utilisées dans un contexte général.

La prise en charge des extensions d'entrée de CRL définies dans la présente spécification est facultative pour les producteurs et applications de CRL conformes. Cependant, les producteurs de CRL DEVRAIENT inclure des codes de cause (paragraphe 5.3.1) et des dates d'invalidité (paragraphe 5.3.2) chaque fois que cette information est disponible.

5.3.1 Code de cause

Le code de cause (reasonCode) est une extension d'entrée de CRL non critique qui identifie la raison de la révocation du certificat. Les producteurs de CRL sont fortrmrnt encouragés à inclure des codes de cause significatifs dans les entrées de CRL ; cependant, l'extension d'entrée de code de cause de CRL DEVRAIT être absent plutôt que d'utiliser la valeur non spécifiée (0) de reasonCode.

La valeur de code de cause removeFromCRL (8) ne peut apparaître que dans les CRL delta et indique qu'un certificat est à supprimer d'une CRL parce que soit le certificat a expiré, soit qu'il a été retiré de la garde. Tous les autres codes de cause peuvent apparaître dans toute CRL et indiquer que le certificat spécifié devrait être considéré comme révoqué.

IDENTIFIANT D'OBJET id-ce-cRLReasons ::= { id-ce 21 }

-- reasonCode ::= { CRLReason }

```
CRLReason ::= ENUMERATED {
    unspecified             (0),        ; (non spécifée)
    keyCompromise           (1),        ; (clé compromise)
    cACompromise            (2),        ; (CA compromise)
    affiliationChanged      (3),        ; (changement d'affiliation)
    superseded              (4),        ; (remplacée)
    cessationOfOperation    (5),        ; (cessation de fonctio
    certificateHold         (6),        ; (certificat en garde)
-- la valeur 7 n'est pas utilisée
    removeFromCRL           (8),        ; (retiré de la CRL)
    privilegeWithdrawn      (9),        ; (privilège supprimé)
    aACompromise           (10) }       ; (AA compromise)
```

5.3.2 Date d'invalidité

La date d'invalidité est une extension d'entrée de CRL non critique qui donne la date à laquelle il est connu ou suspecté que la clé privée a été compromise ou que le certificat est autrement devenu invalide. Cette date peut être autérieure à la date de révocation dans l'entrée de CRL, qui est la date à laquelle la CA a traité la révocation. Quand une révocation est envoyée pour la première fois par le producteur de CRL dans une CRL, la date d'invalidité peut précéder la date de production de CRL antérieures, mais la date de révocation NE DEVRAIT PAS précéder la date de production des CRL antérieures. Chaque fois que cette information est disponible, les producteurs de CRL sont fortement encouragés à la partager avec les utilisateurs de CRL.

Les valeurs de GeneralizedTime incluses dans ce champ DOIVENT être exprimées en heure moyenne de Greenwich (Zoulou) et DOIVENT être spécifiées et interprétées comme défini au paragraphe 4.1.2.5.2.

IDENTIFIANT D'OBJET id-ce-invalidityDate ::= { id-ce 24 }

```
InvalidityDate ::=  GeneralizedTime
```

5.3.3 Producteur du certificat

Cette extension d'entrée de CRL identifie le producteur de certificat associé à une entrée dans une CRL indirecte, c'est-à-dire, une CRL qui a l'indicateur indirectCRL établi dans son extension de point de distribution producteur. Quand elle est présente, l'extension d'entrée de CRL de producteur de certificat inclut un ou plusieurs noms provenant du champ

producteur et/ou de l'extension de nom de remplacement de producteur du certificat qui correspond à l'entrée de CRL. Si cette extension n'est pas présente sur la première entrée dans une CRL indirecte, le producteur de certificat est par défaut le producteur de CRL. Sur les entrées suivantes dans une CRL indirecte, si cette extension n'est pas présente, le producteur de certificat pour l'entrée est le même que pour l'entrée précédante. Ce champ est défini comme suit :

IDENTIFIANT D'OBJET id-ce-certificateIssuer ::= { id-ce 29 }

CertificateIssuer ::= GeneralNames

Les producteurs de CRL conformes DOIVENT inclure dans cette extension le nom distinctif (DN) provenant du champ Producteur du certificat qui correspond à cette entrée de CRL. Le codage du DN DOIT être identique au codage utilisé dans le certificat.

Les producteurs de CRL DOIVENT marquer cette extension comme critique car une mise en œuvre qui ignore cette extension ne pourrait pas correctement attribuer les entrées de CRL aux certificats. La présente spécification RECOMMANDE que les mises en œuvre reconnaissent cette extension.

6. Validation du chemin de certification

Les procédures de validation du chemin de certification pour la PKI Internet sont fondées sur l'algorithme fourni dans [X.509]. Le traitement de chemin de certification vérifie le lien entre le nom distinctif de sujet et/ou le nom de remplacement de sujet et de clé publique sujette. Le lien est limité par les contraintes qui sont spécifiées dans les certificats qui composent le chemin et les entrées qui sont spécifiées par le consommateur d'assertions. Les contraintes de base et les extensions de contraintes de politiques permettent que la logique de traitement de chemin de certification automatise le processus de prise de décision.

Cette section décrit un algorithme pour valider la certification de chemins. Les mises en œuvre conformes de la présente spécification ne sont pas obligées de mettre en œuvre cet algorithme, mais DOIVENT fournir une fonctionnalité équivalente au comportement externe résultant de cette procédure. Tout algorithme peut être utilisé par une mise en œuvre particulière pour autant qu'elle déduit le résultat correct.

Au paragraphe 6.1, le texte décrit la validation de base de chemin. Les chemins valides commencent avec les certificats produits par une ancre de confiance. L'algorithme exige la clé publique de la CA, le nom de la CA, et toutes les contraintes sur l'ensemble de chemins qui peut être validé en utilisant cette clé.

Le choix d'une ancre de confiance est une affaire de politique : elle pourrait être la CA sommitale d'une PKI hiérarchique, la CA qui a produit le propre certificat du vérificateur ou toute autre CA dans une PKI de réseau. La procédure de validation de chemin est la même sans considération du choix de l'ancre de confiance. De plus, différentes applications peuvent s'appuyer sur des ancres de confiance différentes, ou peuvent accepter des chemins qui commencent par tout ensemble d'ancres de confiance.

Le paragraphe 6.2 décrit les méthodes pour utiliser l'algorithme de validation de chemin dans des mises en œuvre spécifiques.

Le paragraphe 6.3 décrit les étapes nécessaires pour déterminer si un certificat est révoqué quand des CRL sont le mécanisme de révocation utilisé par le producteur de certificat.

6.1 Validation du chemin de base

Ce texte décrit un algorithme pour le traitement de chemin X.509. Une mise en œuvre conforme DOIT inclure une procédure de traitement de chemin X.509 fonctionnellement équivalente au comportement externe de cet algorithme. Cependant, la prise en charge de certaines des extensions de certificat traitées dans cet algorithme est FACULTATIVE pour les mises en œuvre conformes. Les clients qui ne prennent pas en charge ces extensions PEUVENT omettre les étapes correspondantes dans l'algorithme de validation de chemin.

Par exemple, les clients ne sont pas obligés de prendre en charge l'extension de transpositions de politique. Les clients qui ne prennent pas en charge cette extension PEUVENT omettre les étapes de validation de chemin où des transpositions de politique sont traitées. Noter que les clients DOIVENT rejeter le certificat si il contient une extension critique non prise en charge.

Alors que les profils de certificat et de CRL spécifiés aux Sections 4 et 5 du présent document spécifient les valeurs pour les champs et extensions de certificat et de CRL qui sont considérés être appropriés pour la PKI Internet, l'algorithme présenté dans cette section ne se limite pas à accepter les certificats et CRL qui se conforment à ces profils. Donc, l'algorithme inclut seulement la vérification que le chemin de certification est valide selon X.509 et n'inclut pas de vérification que les certificats et CRL se conforment au présent profil. Bien que l'algorithme pourrait être étendu pour inclure des vérification de conformité aux profils des sections 4 et 5, le présent profil RECOMMANDE de ne pas inclure de telles vérifications.

L'algorithme présenté dans cette section valide le certificat à l'égard de la date et heure courantes. Une mise en œuvre conforme PEUT aussi prendre en charge la validation par rapport à un instant dans le passé. Noter qu'il n'y a pas de mécanisme disponible pour valider un certificat par rapport à un instant en dehors de la période de validité du certificat.

L'ancre de confiance est une entrée de l'algorithme. Il n'est pas exigé que la même ancre de confiance soit utilisée pour valider toutes les certifications de chemin. Différentes ancres de confiance PEUVENT être utilisées pour valider différents chemins, comme expliqué plus en détails au paragraphe 6.2.

Le principal but de la validation de chemin est de vérifier le lien entre un nom distinctif de sujet ou un nom de remplacement de sujet et la clé publique sujette, comme représentée dans le certificat cible, sur la base de la clé publique de l'ancre de confiance. Dans la plupart des cas, le certificat cible va être un certificat d'entité d'extrémité, mais le certificat cible peut être un certificat de CA pour autant que la clé publique sujette est à utiliser pour un objet autre que de vérifier la signature sur un certificat de clé publique. Vérifier le lien entre le nom et la clé publique sujette exige d'obtenir une séquence de certificats qui prennent en charge ce lien. La procédure suivie pour obtenir cette séquence de certificats sort du domaine d'application de la présente spécification.

Pour atteindre cet objectif, le processus de validation de chemin vérifie, entre autres choses, qu'un chemin de certification prospectif (une séquence de n certificats) satisfait les conditions suivantes :
(a) pour tout x dans {1, ..., n-1}, le sujet du certificat x est le producteur du certificat x+1 ;
(b) le certificat 1 est produit par l'ancre de confiance ;
(c) le certificat n est le certificat à valider (c'est-à-dire, le certificat cible) ; et
(d) pour tout x dans {1, ..., n}, le certificat était valide à l'instant en question.

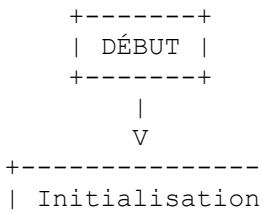
Un certificat NE DOIT PAS apparaître plus d'une fois dans un chemin de certification prospectif.

Quand l'ancre de confiance est fournie sous la forme d'un certificat auto-signé, ce certificat auto-signé n'est pas inclus au titre du chemin de certification prospectif. Les informations sur les ancres de confiance sont fournies comme entrées à l'algorithme de validation de chemin de certification (paragraphe 6.1.1).

Un chemin de certification particulier peut cependant ne pas être approprié pour toutes les applications. Donc, une application PEUT augmenter cet algorithme pour limiter l'ensemble de chemins valides. Le processus de validation de chemin détermine aussi l'ensemble de politiques de certificat qui sont valides pour ce chemin, sur la base de l'extension de politiques de certificat, de l'extension des transpositions de politique, de l'extension des contraintes de politique, et inhibe l'extension anyPolicy. Pour réaliser cela, l'algorithme de validation de chemin construit une arborescence des politiques valides. Si l'ensemble des politiques de certificat qui sont valides pour ce chemin n'est pas vide, alors le résultat va être une arborescence de politiques valides de profondeur n, autrement le résultat va être une arborescence de politique valide nulle.

Un certificat est auto-produit si le même DN apparaît dans les champs Sujet et Producteur (les deux DN sont le même si ils correspondent selon les règles spécifiées au paragraphe 7.1). En général, le producteur et le sujet des certificats qui constituent un chemin sont différents pour chaque certificat. Cependant, une CA peut produire un certificat à elle-même pour prendre en charge le changement de clé ou les changements de politique de certificats. Ces certificats auto-produits ne sont pas comptés lors de l'évaluation de la longueur de chemin ou les contraintes de nom.

Ce paragraphe présente l'algorithme en quatre étapes de base : (1) initialisation, (2) traitement de base du certificat, (3) préparation du prochain certificat, et (4) enveloppement. Les étapes (1) et (4) sont effectuées exactement une fois. L'étape (2) est effectuée pour tous les certificats dans le chemin. L'étape (3) est effectuée pour tous les certificats dans le chemin sauf le certificat final. La Figure 2 donne un diagramme général de cet algorithme.



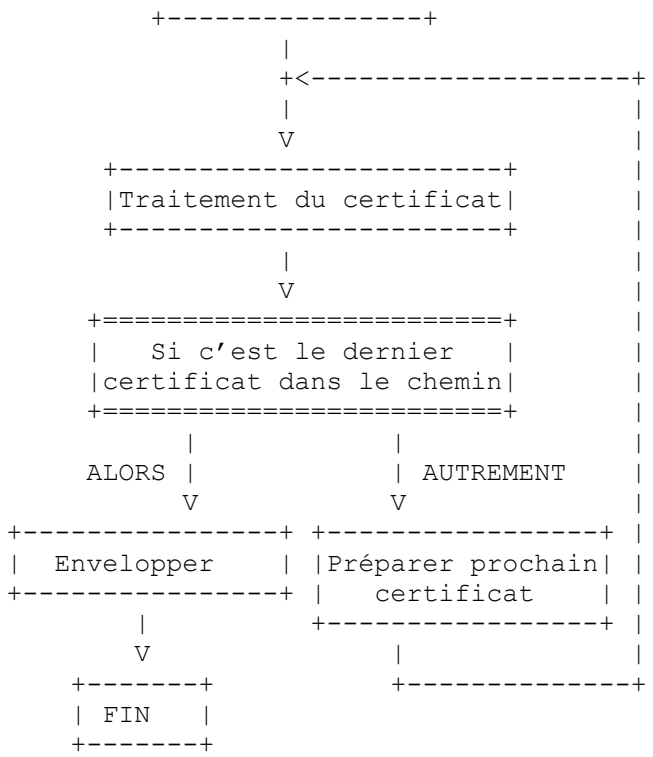


Figure 2 : Diagramme de traitement du chemin de certification

6.1.1 Entrées

Cet algorithme suppose que les neuf entrées suivantes sont fournies à la logique de traitement de chemin :

(a) un chemin de certification prospectif de longueur n.

(b) la date/heure actuelle.

(c) ensemble-initial-de-politique-d'utilisateur : un ensemble d'identifiants de politique de certificat désignant les politiques acceptables pour l'utilisateur du certificat. Le ensemble-initial-de-politique-d'utilisateur contient la valeur spéciale any-policy si l'utilisateur n'est pas concerné par la politique de certificat.

(d) informations d'ancre de confiance, décrivant une CA qui sert d'ancre de confiance pour le chemin de certification. Les informations d'ancre de confiance incluent :
(1) le nom du producteur de confiance,
(2) l'algorithme de clé publique de confiance,
(3) la clé publique de confiance, et
(4) facultaticement, les paramètres de clé publique de confiance associés à la clé publique.
Les informations d'ancre de confiance peuvent être fournies à la procédure de traitement de chemin sous la forme d'un certificat auto-signé. Quand les informations d'ancre de confiance sont fournies sous la forme d'un certificat, le nom dans le champ sujet est utilisé comme nom du producteur de confiance et le contenu du champ subjectPublicKeyInfo est utilisé comme source de l'algorithme de clé publique de confiance et de clé publique de confiance. Les informations d'ancre de confiance sont de confiance parce qu'elles ont été livrées à la procédure de traitement de chemin par une procédure digne de confiance hors-bande. Si l'algorithme de clé publique de confiance exige des paramètres, ceux-ci sont alors fournis avec la clé publique de confiance.

(e) inhiber-la-transposition-de-politique-initiale, indique si la transposition de politique est permise dans le chemin de certification.

(f) politique-initiale-explicite, indique si le chemin doit être valide pour au moins une des politiques de certificat dans le "ensemble-initial-de-politique-d'utilisateur".

(g) inhiber-toute-politique-initiale, indique si l'OID anyPolicy devrait être traité si il est inclus dans un certificat.

(h) sous-arborescences-initiales-permises, qui indique pour chaque type de nom (par exemple, noms distinctifs X.500, adresses de messagerie, ou adresses IP) un ensemble de sous arborescences au sein desquelles tous les noms de sujet dans chaque certificat sur le chemin de certification DOIVENT entrer. L'entrée sous-arborescences-initiales-permises inclut un ensemble pour chaque type de nom. Pour chaque type de nom, l'ensemble peut consister en une seule sous arborescence qui inclut tous les noms de ce type de nom ou une ou plusieurs sous arborescences qui chacune spécifie un sous ensemble des noms de ce type de nom, ou l'ensemble peut être vide. Si l'ensemble pour un type de nom est vide, alors le chemin de certification va être considéré invalide si un certificat dans le chemin de certification inclut un nom de ce type de nom.

(i) sous-arborescences-initiales-exclues, qui indique pour chaque type de nom (par exemple, noms distinctifs X.500, adresses de messagerie, ou adresses IP) un ensemble de sous arborescences au sein desquelles aucun nom de sujet dans aucun certificat dans le chemin de certification ne peut tomber. L'entrée de sous-arborescences-initiales-exclues inclut un ensemble pour chaque type de nom. Pour chaque type de nom, l'ensemble peut être vide ou peut consister en une ou plusieurs sous arborescences qui chacune spécifie un sous ensemble des noms de ce type de nom. Si l'ensemble pour un type de nom est vide, alors aucun nom de ce type de nom n'est exclu.

Les mises en œuvre conformes ne sont pas obligées de prendre en charge le réglage de toutes ces entrées. Par exemple, une mise en œuvre conforme peut être conçue pour valider toutes les certifications de chemin qui utilisent une valeur de FAUX pour inhiber-toute-politique-initiale.

6.1.2 Initialisation

Cette phase d'initialisation établit onze variables d'état fondées sur neuf entrées :

(a) arborescence_des_politiques_valides : arborescence des politiques de certificat avec leurs qualificatifs facultatifs ; chacune des feuilles de l'arborescence représente une politique valide à cette étape de la valication de chemin de certification. Si des politiques valides existent à cette étape dans la validation de chemin de certification, la profondeur de l'arborescence est égale au nombre de certificats dans la chaîne qui a été traitée. Si des politiques valides n'existent pas à cette étape de la validation de chemin de certification, l'arborescence est réglée à NUL. Une fois que l'arborescence est réglée à NUL, le traitement de politique cesse.

Chaque nœud dans arborescence_des_politiques_valides inclut trois objets de données : la politique valide, un ensemble de qualificatifs de politique associés, et un ensemble de une ou plusieurs valeurs de politique attendues. Si le nœud est à la profondeur x, les composants du nœud ont la sémantique suivante :

(1) politique_valide est un seul OID de politique représentant une politique valide pour le chemin de longueur x.
(2) ensemble_de_qualificatifs est un ensemble de qualificatifs de politique associés à la politique valide dans le certificat x.
(3) ensemble_de_politiques_attendues contient un ou plusieurs OID de politique qui vont satisfaire cette politique dans le certificat x+1.

La valeur initiale de arborescence_des_politiques_valides est un seul nœud avec politique_valide anyPolicy, un ensemble_de_qualificatifs vide, et un ensemble_de_politiques_attendues avec la seule valeur de anyPolicy. Ce nœud est considéré être à la profondeur zéro.

La Figure 3 est une représentation graphique de l'état initial de l'arborescence_des_politiques_valides. Des figures supplémentaires vont utiliser ce format pour décrire les changements dans arborescence_des_politiques_valides durant le traitement de chemin.

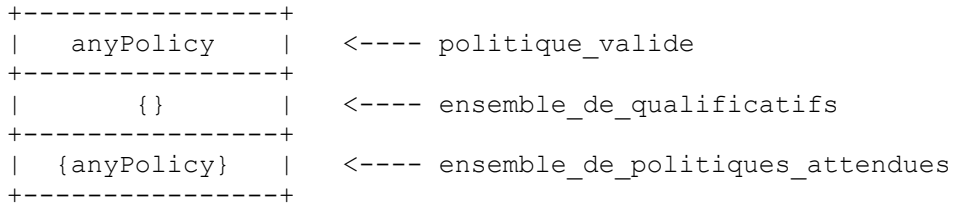


Figure 3 : Valeur initiale de la variable d'état arborescence_des_politiques_valides

(b) sous-arborescences_permises : ensemble de noms racine pour chaque type de nom (par exemple, noms distinctifs X.500, adresses de messagerie, ou adresses IP) définissant un ensemble de sous arborescences au sein desquelles tous les noms de sujet dans les certificats suivants dans le chemin de certification DOIVENT entrer. Cette variable inclut un ensemble pour chaque type de nom, et la valeur initiale est sous-arborescences-initiales-permises.

(c) sous-arborescences_exclues : ensemble de noms racine pour chaque type de nom (par exemple, noms distinctifs X.500, adresses de messagerie, ou adresses IP) définissant un ensemble de sous arborescences au sein desquelles aucun nom de sujet dans les certificats suivants sur le chemin de certification ne peut tomber. Cette variable inclut un ensemble pour chaque type de nom, et la valeur initiale est sous-arborescences-initiales-exclues.

(d) politique_explicite : entier qui indique si une arborescence_des_politiques_valides non NULLE est exigée. L'entier indique le nombre de certificats non auto-produits à traiter avant que cette exigence soit imposée. Une fois établie, cette variable peut être diminuée, mais ne peut pas être augmentée. C'est-à-dire, si un certificat dans le chemin exige une arborescence_des_politiques_valides non NULLE, un certificat ultérieur ne peut pas supprimer cette exigence. Si politique-initiale-explicite est établi, alors la valeur initiale est 0, autrement la valeur initiale est n+1.

(e) inhiber_anyPolicy : entier qui indique si l'identifiant de politique anyPolicy est considéré correspondre. L'entier indique que le nombre de certificats non auto-produits à traiter avant l'OID anyPolicy, si il est affirmé dans un certificat autre que un certificat auto-produit intermédiaire, est ignoré. Une fois établie, cette variable peut être diminuée, mais ne peut pas être augmentée. C'est-à-dire, si un certificat dans le chemin inhibe le traitement de anyPolicy, un certificat ultérieur ne peut pas le permettre. Si inhiber-toute-politique-initiale est établi, alors la valeur initiale est 0, autrement la valeur initiale est n+1.

(f) transposition_de_politique : entier qui indique si la transposition de politique est permise. L'entier indique le nombre de certificats non auto produits à traiter avant que la transposition de politique soit inhibée. Une fois établie, cette variable peut être diminuée, mais ne peut pas être augmentée. C'est-à-dire, si un certificat dans le chemin spécifie que la transposition de politique n'est pas permise, elle ne peut pas être outrepassée par un certificat ultérieur. Si inhiber-la-transposition-de-politique-initiale est établi, la valeur initiale est alors 0, autrement la valeur initiale est n+1.

(g) algorithme_de_clé_publique_activé : algorithme de signature numérique utilisé pour vérifier la signature d'un certificat. algorithme_de_clé_publique_activé est initialisé à partir de l'algorithme de clé publique de confiance fourni dans les informations d'ancre de confiance.

(h) clé_publique_activée : la clé publique utilisée pour vérifier la signature d'un certificat. La clé_publique_activée est initialisée à partir de la clé publique de confiance fournie dans les informations d'ancre de confiance.

(i) paramètres_de_clé_publique_activés : paramètres associés à la clé publique actuelle qui peuvent être exigés pour vérifier une signature (selon l'algorithme). La variable paramètres_de_clé_publique_activés est initialisée à partir de des paramètres de clé publique de confiance fournis dans les informations d'ancre de confiance.

(j) nom_de_producteur_actif : le nom distinctif de producteur attendu dans le prochain certificat de la chaîne. Le nom_de_producteur_actif est initialisé au nom du producteur de confiance fourni dans les informations d'ancre de confiance.

(k) longueur_maximale_de_chemin : cet entier est initialisé à n, est décrémenté pour chaque certificat non auto-produit dans le chemin, et peut être réduit à la valeur dans le champ contrainte de longueur de chemin dans l'extension Contraintes de base d'un certificat de CA.

À l'achèvement des étapes d'initialisation, effectuer les étapes de base de traitement de certificat spécifiées en 6.1.3.

6.1.3 Traitement de base de certificat

Les actions de base du traitement de chemin à effectuer pour le certificat i (pour tout i dans [1..n]) sont énumérées ci-dessous.

(a) Vérifier les informations de base du certificat. Le certificat DOIT satisfaire chacune des conditions suivantes :
  (1) La signature sur le certificat peut être vérifiée en utilisant algorithme_de_clé_publique_activé, clé_publique_activée, et paramètres_de_clé_publique_activés.
  (2) La période de validité du certificat inclut l'heure actuelle.
  (3) À l'heure actuelle, le certificat n'est pas révoqué. Cela peut être déterminé en obtenant la CRL appropriée (paragraphe 6.3), par des informations d'état, ou par des mécanismes hors bande.
  (4) Le nom du producteur du certificat est le nom_de_producteur_actif.

(b) Si le certificat i est auto-produit et n'est pas le certificat final dans le chemin, sauter cette étape pour le certificat i. Autrement, vérifier que le nom de sujet est dans une des sous-arborescences_permises pour les noms distinctifs X.500, et vérifier que chacun des noms de remplacement dans l'extension subjectAltName (critique ou non critique) est dans une des sous-arborescences_permises pour ce type de nom.

(c) Si le certificat i est auto-produit et n'est pas le certificat final dans le chemin, sauter cette étape pour le certificat i. Autrement, vérifier que le nom de sujet n'est pas dans une des sous-arborescences_exclues pour les noms distinctifs X.500, et vérifier que chacun des noms de remplacement dans l'extension subjectAltName (critique ou non critique) n'est dans aucune des sous-arborescences_exclues pour ce type de nom.

(d) Si l'extension de politiques de certificat est présente dans le certificat et si arborescence_des_politiques_valides n'est pas NULLE, traiter les informations de politique en effectuant les étapes suivantes dans l'ordre :

(1) Pour chaque politique P non égale à anyPolicy dans l'extension de politiques de certificat, soit P-OID qui note l'OID pour la politique P et P-Q qui note l'ensemble de qualificatifs pour la politique P. Effectuer les étapes suivantes dans l'ordre :

(i) Pour chaque nœud de profondeur i-1 dans l'arborescence_des_politiques_valides où P-OID est dans l'ensemble_de_politiques_attendues, créer un nœud fils comme suit : régler la politique_valide à P-OID, régler l'ensemble_de_qualificatifs à P-Q, et régler l'ensemble_de_politiques_attendues à {P-OID}.
Par exemple, considérons une arborescence_des_politiques_valides avec un nœud de profondeur i-1 où l'ensemble_de_politiques_attendues est {doré, blanc}. Supposons que les politiques de certificat Doré et Argent apparaissent dans l'extension de politiques de certificat du certificat i. La politique Doré correspond, mais la politique Argent ne correspond pas. Cette règle va générer un nœud fils de profondeur i pour la politique Doré. Le résultat est montré à la Figure 4.

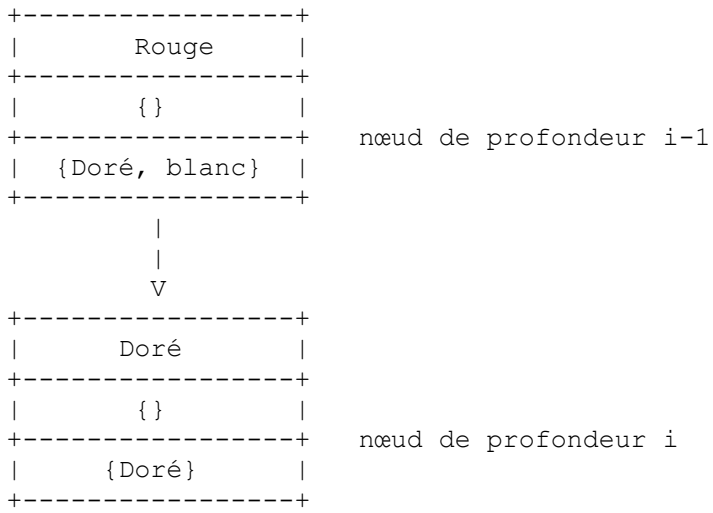
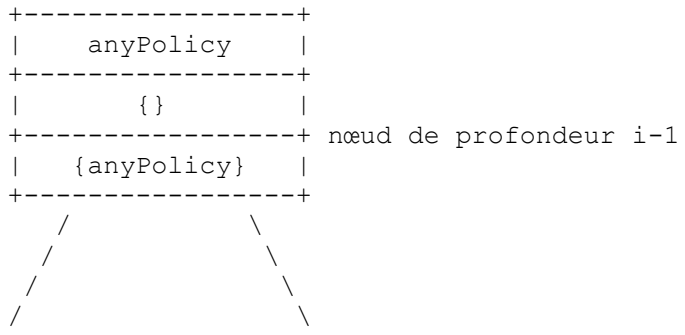


Figure 4 : Traitement d'une correspondance exacte

(ii) Si il n'y a pas de correspondance dans l'étape (i) et si l'arborescence_des_politiques_valides inclut un nœud de profondeur i-1 avec la politique_valide anyPolicy, générer un nœud fils avec les valeurs suivantes : régler la politique_valide à P-OID, régler l'ensemble_de_qualificatifs à P-Q, et régler l'ensemble_de_politiques _attendues à {P-OID}.
Par exemple, considérons une arborescence_des_politiques_valides avec un nœud de profondeur i-1 où la politique_valide est anyPolicy. Supposons que les politiques de certificat Doré et Argent apparaissent dans l'extension de politiques de certificat du certificat i. La politique Doré n'a pas de qualificatif, mais la politique Argent a le qualificatif Q-Silver. Si Doré et Argent ne correspondent pas dans le (i) ci-dessus, cette règle va générer deux nœuds fils de profondeur i, un pour chaque politique. Le résultat est montré à la Figure 5.



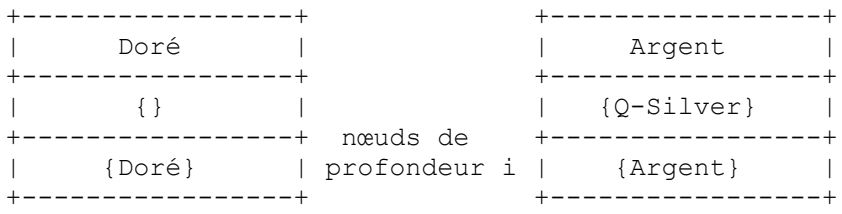


Figure 5 : Traitement de politiques non satisfaites quand un nœud feuille spécifie anyPolicy

(2) Si l'extension de politiques de certificat inclut la politique anyPolicy avec l'ensemble de qualificatifs AP-Q et soit (a) inhiber_anyPolicy est supérieur à 0, soit (b) I < n et le certificat est auto-produit, alors :
pour chaque nœud dans l'arborescence_des_politiques_valides de profondeur i-1, pour chaque valeur dans l'ensemble_de_politiques_attendues (incluant anyPolicy) qui n'apparaît pas dans un nœud fils, créer un nœud fils avec les valeurs suivantes : régler la politique_valide à la valeur provenant de ensemble_de_politiques_attendues dans le nœud parent, régler ensemble_de_qualificatifs à AP-Q, et régler ensemble_de_politiques_attendues à la valeur dans la politique_valide provenant de ce nœud.

Par exemple, considérons une arborescence_des_politiques_valides avec un nœud de profondeur i-1 où l'ensemble_de_politiques_attendues est {Doré, Argent}. Supposons que anyPolicy apparaisse dans l'extension de politiques de certificat du certificat i sans qualificatif de poltique, mais Doré et Argent n'apparaisent pas. Cette règle va générer deux nœuds fils de profondeur i, un pour chaque politique. Le résultat est montré à la Figure 6.

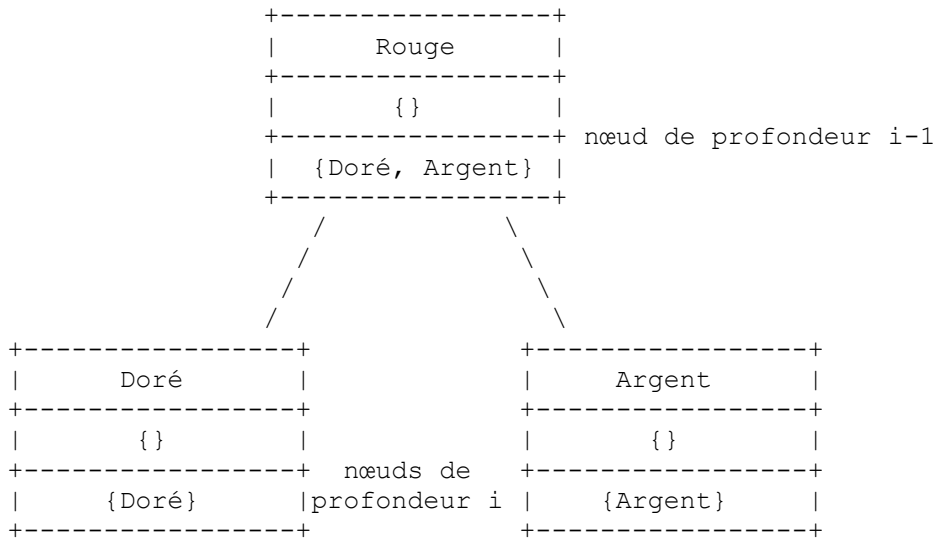


Figure 6 : Traitement de politiques non satisfaites quand l'extension Politiques de certificat spécifie anyPolicy

(3) Si il y a un nœud dans l'arborescence_des_politiques_valides de profondeur i-1 ou moins sans aucun nœud fils, supprimer ce nœud. Répéter cette étape jusqu'à ce qu'il n'y ait pas de nœud de profondeur i-1 ou moins sans enfant.

Par exemple, considérons l'arborescence_des_politiques_valides montrée à la Figure 7. Les deux nœuds de profondeur i-1 qui sont marqués avec un 'X' n'ont pas d'enfant, et ils sont supprmés. Appliquer cette règle à l'arborescence résultante va causer la suppression du nœud de profondeur i-2 qui est marqué avec un 'Y'. Dans l'arborescence résultante, il n'y a pas de nœud de profondeur i-1 ou moins sans enfant, et cette étape est achevée.

(e) Si l'extension de politiques de certificat n'est pas présente, régler arborescence_des_politiques_valides à NUL.

(f) Vérifier que soit politique_explicite est supérieur à 0, soit que arborescence_des_politiques_valides n'est pas égal à NUL.

Si une des étapes (a), (b), (c), ou (f) échoue, la procédure se termine, retournant une indication d'échec et une raison appropriée.

Si i n'est pas égal à n, continuer en effectuant les étapes préparatoires mentionnées au paragraphe 6.1.4. Si i est égal à n, effectuer les étapes d'envelopppe mentionnées au paragraphe 6.1.5.

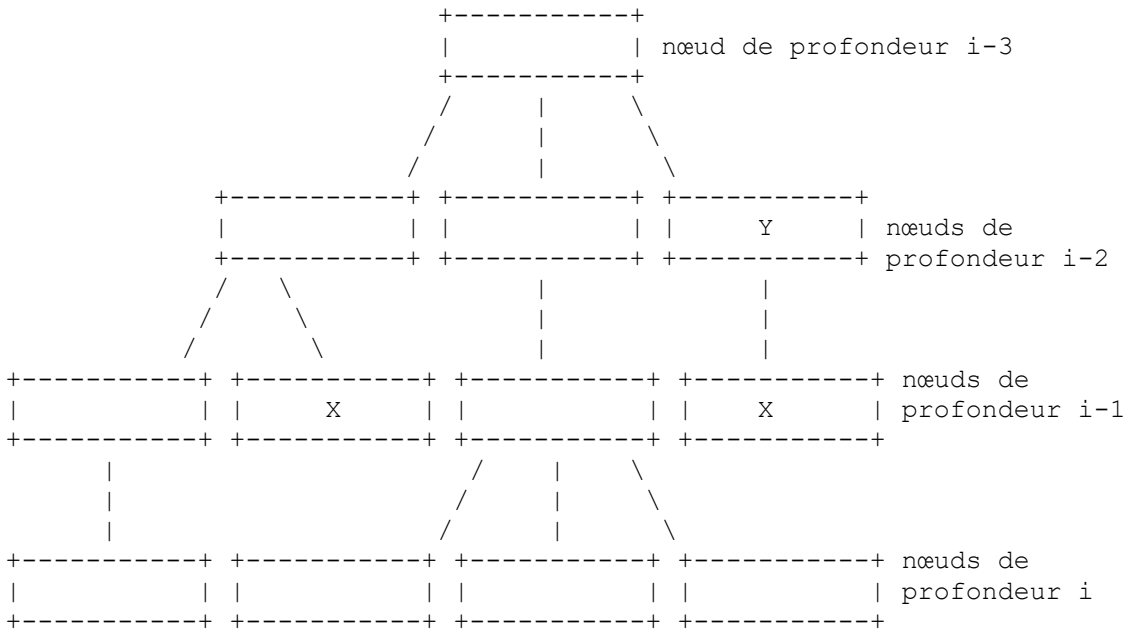


Figure 7 : Élagage de arborescence_des_politiques_valides

6.1.4 Préparation pour le certificat i+1

Pour préparer le traitement du certificat i+1, effectuer les étapes suivantes pour le certificat i :

(a) Si une extension Transpositions de politique est présente, vérifier que la valeur spéciale anyPolicy n'apparaît pas comme issuerDomainPolicy ou subjectDomainPolicy.

(b) Si une extension Transpositions de politique est présente, alors pour chaque ID-P issuerDomainPolicy dans l'extension Transpositions de politique :

(1) Si la variable transposition_de_politique variable est supérieure à 0, pour chaque nœud dans arborescence_des_politiques_valides de profondeur i où ID-P est la politique_valide, régler ensemble_de_politiques_attendues à l'ensetmble de valeurs de subjectDomainPolicy qui sont spécifiées comme équivalentes à ID-P par l'extension Transpositions de politique. Si aucun nœud de profondeur i dans l'arborescence_des_politiques_valides n'a de politique_valide de ID-P mais si il y a un nœud de profondeur i avec une politique_valide de anyPolicy, générer alors un nœud fils du nœud de profondeur i-1 qui a une politique_valide de anyPolicy comme suit :
   (i) régler la politique_valide à ID-P ;
   (ii) régler l'ensemble_de_qualificatifs à l'ensemble de qualificatifs de la politique anyPolicy dans l'extension de politiques de certificat du certificat I ; et
   (iii) régler l'ensemble_de_politiques_attendues à l'ensemble de valeurs de subjectDomainPolicy qui sont spécifiées comme équivalentes à ID-P par l'extension Transpositions de politique.

(2) Si la variable transposition_de_politique est égale à 0 :
   (i) supprimer chaque nœud de profondeur i dans l'arborescence_des_politiques_valides où ID-P est la politique_valide.
   (ii) Si il y a un nœud dans l'arborescence_des_politiques_valides de profondeur i-1 ou moins sans aucun nœud fils, supprimer ce nœud. Répéter cette étape jusqu'à ce qu'il n'y ait plus de nœud de profondeur i-1 ou moins sans enfant.

(c) Allouer le nom de sujet de certificat au nom_de_producteur_actif.

(d) Allouer le certificat subjectPublicKey à la clé_publique_activée.

(e) Si le champ subjectPublicKeyInfo du certificat contient un champ Algorithme avec des paramètres non nuls, allouer les paramètres à la variable paramètres_de_clé_publique_activés. Si le champ subjectPublicKeyInfo du certificat contient un champ Algorithme avec des paramètres nuls ou si les paramètres sont omis, comparer l'algorithme subjectPublicKey

de certificat à l'algorithme_de_clé_publique_activé. Si ils sont différents, régler à nul les paramètres_de_clé_publique_activés.

(f) Allouer l'algorithme subjectPublicKey de certificat à la variable algorithme_de_clé_publique_activé.

(g) Si une extension contraintes de nom est incluse dans le certificat, modifier les variables d'état sous-arborescences_permises et sous-arborescences_exclues comme suit :
   (1) Si permittedSubtrees est présent dans le certificat, régler la variable d'état sous-arborescences_permises à l'intersection de sa valeur précédente et de la valeur indiquée dans le champ d'extension. Si permittedSubtrees n'inclut pas de type de nom particulier, la variable d'état sous-arborescences_permises est inchangée pour ce type de nom. Par exemple, l'intersection de exemple.com et foo.exemple.com est foo.exemple.com. Et l'intersection de exemple.com et exemple.net est l'ensemble vide.
   (2) Si excludedSubtrees est présent dans le certificat, régler la variable d'état sous-arborescences_exclues à l'union de sa valeur précédente et de la valeur indiquée dans le champ d'extension. Si excludedSubtrees n'inclut pas de type de nom particulier, la variable d'état sous-arborescences_exclues est inchangée pour ce type de nom. Par exemple, l'union des espaces de noms exemple.com et foo.exemple.com est exemple.com. Et l'union de exemple.com et exemple.net est les deux espaces de noms.

(h) Si le certificat i n'est pas auto-produit :
   (1) Si politique_explicite n'est pas 0, décrémenter politique_explicite de 1.
   (2) Si transposition_de_politique n'est pas 0, décrémenter transposition_de_politique de 1.
   (3) Si inhiber_anyPolicy n'est pas 0, décrémenter inhiber_anyPolicy de 1.

(i) Si une extension contraintes de politique est incluse dans le certificat, modifier les variables d'état politique_explicite et transposition_de_politique comme suit :
   (1) Si requireExplicitPolicy est présent et est moins que politique_explicite, régler politique_explicite à la valeur de requireExplicitPolicy.
   (2) Si inhibitPolicyMapping est présent et est moins que transposition_de_politique, régler transposition_de_politique à la valeur de inhibitPolicyMapping.

(j) Si l'extension inhibitAnyPolicy est incluse dans le certificat et est moins que inhiber_anyPolicy, régler inhiber_anyPolicy à la valeur de inhibitAnyPolicy.

(k) Si le certificat i est un certificat de version 3, vérifier que l'extension basicConstraints est présente et que cA est réglé à VRAI. (Si le certificat i est un certificat de version 1 ou version 2, l'application DOIT alors vérifier que le certificat i est un certificat de CA par des moyens hors bande ou rejeter le certificat. Les mises en œuvre conformes peuvent choisir de rejeter tous les certificats intermédiaires de version 1 et version 2.)

(l) Si le certificat n'était pas auto-produit, vérifier que longueur_maximale_de_chemin est supérieur à zéro et décrémenter longueur_maximale_de_chemin de 1.

(m) Si pathLenConstraint est présent dans le certificat et est moins que longueur_maximale_de_chemin, régler longueur_maximale_de_chemin à la valeur de pathLenConstraint.

(n) Si une extension d'usage de clé est présente, vérifier que le bit keyCertSign est établi.

(o) Reconnaître et traiter toute autre extension critique présente dans le certificat. Traiter toute autre extension non critique reconnue présente dans le certificat qui est pertinente pour le traitement de chemin.

Si les vérifications (a), (k), (l), (n), ou (o) échouent, la procédure se termine, reteurnant une indication d'échec et une raison appropriée.

Si (a), (k), (l), (n), et (o) se sont achevées avec succès, incrémenter i et effectuer le traitement de base de certificat spécifié au paragraphe 6.1.3.

6.1.5 Procédure d'enveloppement

Pour achever le traitement du certificat cible, effectuer les étapes suivantes pour le certificat n :

(a) Si politique_explicite n'est pas 0, décrémenter politique_explicite de 1.

(b) Si une extension contraintes de politique est incluse dans le certificat et si requireExplicitPolicy est présent et a une valeur de 0, régler la variable d'état politique_explicite à 0.

(c) Allouer le certificat subjectPublicKey à clé_publique_activée.

(d) Si le champ subjectPublicKeyInfo du certificat contient un champ Algorithme avec des paramètres non nuls, allouer les paramètres à la variable paramètres_de_clé_publique_activés. Si le champ subjectPublicKeyInfo du certificat contient un champ Algorithme avec des paramètres nuls ou si les paramètres sont omis, comparer l'algorithme subjectPublicKey de certificat à l'algorithme_de_clé_publique_activé. Si ils sont différents, régler à nul les paramètres_de_clé_publique_activés.

(e) Allouer l'algorithme subjectPublicKey de certificat à la variable algorithme_de_clé_publique_activé.

(f) Reconnaître et traiter toute autre extension critique présente dans le certificat n. Traiter toute autre extension non critique reconnue présente dans le certificat n qui est pertinente pour le traitement de chemin.

(g) Calculer l'intersection de arborescence_des_politiques_valides et de ensemble-initial-de-politique-d'utilisateur, comme suit :

(i) Si arborescence_des_politiques_valides est NULLE, l'intersection est NULLE.

(ii) Si arborescence_des_politiques_valides est non NULLE et si ensemble-initial-de-politique-d'utilisateur est any-policy, l'intersection est arborescence_des_politiques_valides entière.

(iii) Si arborescence_des_politiques_valides est non NULLE et si ensemble-initial-de-politique-d'utilisateur n'est pas any-policy, calculer l'intersection de arborescence_des_politiques_valides et de ensemble-initial-de-politique-d'utilisateur comme suit :

1. Déterminer l'ensemble de nœuds de politique dont les nœuds parents ont une politique_valide de anyPolicy. C'est l'ensemble_de_nœuds_de_politique_valide.
2. Si la politique_valide d'un nœud dans l'ensemble_de_nœuds_de_politique_valide n'est pas dans l'ensemble-initial-de-politique-d'utilisateur et n'est pas anyPolicy, supprimer ce nœud et tous ses enfants.
3. Si arborescence_des_politiques_valides inclut un nœud de profondeur n avec la politique_valide de anyPolicy et si l'ensemble-initial-de-politique-d'utilisateur n'est pas any-policy, effectuer les étapes suivantes :
   a. Régler P-Q à ensemble_de_qualificatifs dans le nœud de profondeur n à la politique_valide de anyPolicy.
   b. Pour chaque P-OID dans l'ensemble-initial-de-politique-d'utilisateur qui n'est pas la politique_valide d'un nœud dans l'ensemble_de_nœuds_de_politique_valide, créer un nœud fils dont le parent est le nœud de profondeur n-1 avec la politique_valide de anyPolicy. Régler les valeurs dans le nœud fils comme suit : régler la politique_valide à P-OID, régler ensemble_de_qualificatifs à P-Q, et régler ensemble_de_politiques_attendues à {P-OID}.
   c. Supprimer le nœud de profondeur n avec la politique_valide de anyPolicy.
4. Si il y a un nœud dans arborescence_des_politiques_valides de profondeur n-1 ou moins sans nœud fils, supprimer ce nœud. Répéter cette étape jusqu'à ce qu'il n'y ait plus de nœud de profondeur n-1 ou moins sans enfant.

Si (1) la valeur de la variable politique_explicite est supérieure à zéro ou (2) arborescence_des_politiques_valides n'est pas NULLE, alors le traitement de chemin a réussi.

6.1.6 Résultats

Si le traitement de chemin réussit, la procédure se termine, retournant une indication de réussite avec une valeur finale de arborescence_des_politiques_valides, la clé_publique_activée, l'algorithme_de_clé_publique_activé, et les paramètres_de_clé_publique_activés.

6.2 Utilisation de l'algorithme de validation de chemin

L'algorithme de validation de chemin décrit le processus de validation d'un seul chemin de certification. Bien que chaque chemin de certification commence par une ancre de confiance spécifique, il n'est pas exigé que toutes les certifications de chemin validées par un système particulier partagent une seule ancre de confiance. Le choix de une ou plusieurs CA de confiance est une décision locale. Un système peut fournir n'importe laquelle de ses CA de confiance comme ancre de confiance pour un chemin particulier. Les entrées à l'algorithme de validation de chemin peuvent être différentes pour chaque chemin. Les entrées utilisées pour traiter un chemin peuvent refléter des exigences spécifiques de l'application ou des limitations à la confiance accordée à une ancre de confiance particulière. Par exemple, une CA de confiance peut n'être de confiance que pour une politique de certificat particulière. Cette restriction peut être exprimée par les entrées à la procédure de validation de chemin.

Une mise en œuvre PEUT augmenter l'algorithme présenté au paragraphe 6.1 pour limiter l'ensemble de certifications de chemin valides qui commencent avec une ancre de confiance particulière. Par exemple, une mise en œuvre PEUT modifier l'algorithme pour appliquer une contrainte de longueur de chemin à une ancre de confiance spécifique durant la phase d'initialisation, ou l'application PEUT exiger la présence d'une forme de nom de remplacement particulière dans le certificat cible, ou l'application PEUT imposer des exigences aux extensions spécifiques de l'application. Donc, l'algorithme de validation de chemin présenté au paragraphe 6.1 définit les conditions minimum pour qu'un chemin soit considéré valide.

Lorsque une CA distribue des certificats auto signés pour spécifier les informations d'ancre de confiance, les extensions de certificat peuvent être utilisées pour spécifier les entrées recommandées à la validation de chemin. Par exemple, une extension Contraintes de politique pourrait être incluse dans le certificat auto-signé pour indiquer que les chemins qui commencent par cette ancre de confiance ne devraient être de confiance que pour les politiques spécifiées. De même, une extension de contraintes de nom pourrait être incluse pour indiquer que les chemins qui commencent par cette ancre de confiance devraient n'être de confiance que pour les espaces de noms spécifiés. L'algorithme de validation de chemin présenté au paragraphe 6.1 ne suppose pas que les informations d'ancre de confiance sont fournies dans les certificats auto signés et ne spécifient pas de règles de traitement pour des informations supplémentaires incluses dans de tels certificats. Les mises en œuvre qui utilisent des certificats auto signés pour spécifier des informations d'ancre de confiance sont libres de traiter ou d'ignorer ces informations.

6.3 Validation de CRL

Ce paragraphe décrit les étapes nécessaires pour déterminer si un certificat est révoqué quand les CRL sont le mécanisme de révocation utilisé par le producteur de certificat. Les mises en œuvre conformes qui prennent en charge les CRL ne sont pas obligées de mettre en œuvre cet algorithme, mais elles DOIVENT être fonctionnellement équivalentes au comportement externe résultant de cette procédure lors du traitement de CRL qui sont produites conformément au présent profil. Tout algorithme peut être utilisé par une mise en œuvre particulière pour autant qu'il déduise le résultat correct.

Cet algorithme suppose que toutes les CRL nécessaires sont disponibles dans une antémémoire locale. De plus, si l'heure de la prochaine mise à jour d'une CRL est passée, l'algorithme suppose un mécanisme pour aller chercher la CRL actuelle et la placer dans l'antémemoire locale de CRL.

Cet algorithme définit un ensemble d'entrées, un ensemble de variables d'état, et des étapes de traitement qui sont effectuées pour chaque certificat sur le chemin. Le résultat de l'algorithme est l'état de révocation du certificat.

6.3.1 Entrées de révocation

Pour prendre en charge le traitement de révocation, l'algorithme exige deux entrées :

(a) certificat : l'algorithme exige le numéro de série du certificat et le nom du producteur pour déterminer si un certificat est sur une CRL particulière. L'extension basicConstraints est utilisée pour déterminer si le certificat fourni est associé à une CA ou à une entité d'extrémité. Si elle est présente, l'algorithme utilise les extensions cRLDistributionPoints et freshestCRL pour déterminer l'état de révocation.

(b) use-deltas : cette entrée booléenne détermine si des CRL delta sont appliquées aux CRL.

6.3.2 Variables d'état d'initialisation et de révocation

Pour prendre en charge le traitement de CRL, l'algorithme exige les variables d'état suivantes :

(a) reasons_mask : cette variable contient l'ensemble des causes de révocation prises en charge par les CRL et les CRL delta traitées jusqu'à présent. Les membres légaux de l'ensemble sont les valeurs de cause de révocation possibles moins les non spécifiées : keyCompromise, cACompromise, affiliationChanged, superseded, cessationOfOperation, certificateHold, privilegeWithdrawn, et aACompromise. La valeur spéciale all-reasons est utilisée pour noter l'ensemble de tous les membres légaux. Cette variable est initialisée à l'ensemble vide.

(b) cert_status : cette variable contient l'état du certificat. Cette variable peut être allouée à une des valeurs suivantes : unspecified, keyCompromise, cACompromise, affiliationChanged, superseded, cessationOfOperation, certificateHold, removeFromCRL, privilegeWithdrawn, aACompromise, la valeur spéciale UNREVOKED, ou la valeur spéciale UNDETERMINED. Cette variable est initialisée à la valeur spéciale UNREVOKED.

(c) interim_reasons_mask : cette variable contient l'ensemble des raisons de révocation prises en charge par la CRL ou CRL delta en cours de traitement.

Note : dans certains environnements, il n'est pas nécessaire de vérifier tous les codes de cause. Par exemple, certains environnements sont seulement concernés par cACompromise et keyCompromise pour les certificats de CA. Cet algorithme vérifie tous les codes de cause. Des variables supplémentaires de traitement et d'état peuvent être nécessaires pour limiter la vérification à un sous ensemble des codes de cause.

6.3.3 Traitement de CRL

Cet algorithme commence par supposer que le certificat n'est pas révoqué. L'algorithme vérifie une ou plusieurs CRL jusqu'à ce que l'état du certificat soit déterminé être révoqué ou que suffisamment de CRL aient été vérifiées pour couvrir tous les codes de cause.

Pour chaque point de distribution (DP) dans l'extension de points de distribution de CRL du certificat, pour chaque CRL correspondante dans l'antémémoire locale de CRL, quand ((reasons_mask n'est pas all-reasons) et (cert_status est UNREVOKED)) effecuer ce qui suit :

(a) Mettre à jour l'antémémoire locale de CRL en obtenant une CRL complète, une CRL delta, ou les deux, comme nécessaire :
  (1) Si l'heure actuelle est après la valeur du champ Prochaine mise à jour de CRL, faire une des choses suivantes :
    (i) Si use-deltas est établi et si le certificat ou la CRL contient la plus fraîche extension de CRL, obtenir une CRL delta avec une prochaine valeur de mise à jour qui soit après l'heure actuelle et puisse être utilisée pour mettre à jour la CRL en antémémoire locale comme spécifié au paragraphe 5.2.4.
    (ii) Mettre à jour l'antémémoire locale de CRL avec une CRL complète actuelle, vérifier que l'heure actuelle est avant la prochaine valeur de mise à jour dans la nouvelle CRL, et continuer le traitement avec la nouvelle CRL. Si use-deltas est établi et que le certificat ou la CRL contient la plus fraîche extension de CRL, obtenir alors la CRL delta actuelle qui peut être utilisée pour mettre à jour la nouvelle CRL complète en antémémoire locale, comme spécifié au paragraphe 5.2.4.
  (2) Si l'heure actuelle est avant la valeur du champ Prochaine mise à jour, use-deltas est établi, et le certificat ou la CRL contient la plus fraîche extension de CRL, obtenir alors la CRL delta actuelle qui peut être utilisée pour mettre à jour la CRL complète en antémémoire locale, comme spécifié au paragraphe 5.2.4.

(b) Vérifier le producteur et la portée de la CRL complète comme suit :
  (1) Si le DP inclut cRLIssuer, vérifier que le champ Producteur dans la CRL complète correspond à cRLIssuer dans le DP et que la CRL complète contient une extension Point de distribution producteur avec le booléen indirectCRL affirmé. Autrement, vérifier que le producteur de CRL correspond au producteur du certificat.
  (2) Si la CRL complète inclut une extension de CRL point de distribution producteur (IDP) vérifier ce qui suit :
    (i) Si le nom du point de distribution est présent dans l'extension de CRL IDP et si le champ distribution est présent dans le DP, vérifier qu'un des noms dans le IDP correspond à un des noms dans le DP. Si le nom de point de distribution est présent dans l'extension de CRL IDP et si le champ distribution est omis du DP, vérifier alors qu'un des noms dans le IDP correspond à un des noms dans le champ cRLIssuer du DP.
    (ii) Si le booléen onlyContainsUserCerts est affirmé dans l'extension de CRL IDP, vérifier que le certificat n'inclut pas l'extension Contraintes de base avec le booléen cA affirmé.
    (iii) Si le booléen onlyContainsCACerts est affirmé dans l'extension de CRL IDP, vérifier que le certificat inclut l'extension Contraintes de base avec le booléen cA affirmé.
    (iv) Vérifier que le booléen onlyContainsAttributeCerts n'est pas affirmé.

(c) Si use-deltas est établi, vérifier le producteur et la portée de la CRL delta comme suit :
  (1) Vérifier que le producteur de CRL delta correspond au producteur de la CRL complète.
  (2) Si la CRL complète inclut une extension de CRL Point de distribution producteur (IDP) vérifier que la CRL delta contient une extension CRL IDP correspondante. Si la CRL complète omet une extension de CRL IDP, vérifier que la CRL delta omet aussi l'extension de CRL IDP.
  (3) Vérifier que l'extension identifiant de clé d'autorité de CRL delta correspond à l'extension identifiant de clé d'autorité de CRL complète.

(d) Calculer le interim_reasons_mask *(gabarit de raisons intermédiaires)* pour cette CRL comme suit :
  (1) Si l'extension de CRL point de distribution producteur (IDP) est présente et inclut onlySomeReasons et si le DP inclut des raisons, établir alors interim_reasons_mask à l'intersection des raisons dans le DP et de onlySomeReasons dans l'extension de CRL IDP.
  (2) Si l'extension de CRL IDP inclut onlySomeReasons mais si le DP omet les raisons, régler alors interim_reasons_mask à la valeur de onlySomeReasons dans l'extension de CRL IDP.
  (3) Si l'extension de CRL IDP n'est pas présente ou omet onlySomeReasons mais si le DP inclut des raisons, régler alors set interim_reasons_mask à la valeur des raisons du DP.

(4) Si l'extension de CRL IDP n'est pas présente ou omet onlySomeReasons et si le DP omet les raisons, régler alors interim_reasons_mask à la valeur spéciale de all-reasons.

(e) Vérifier que interim_reasons_mask inclut une ou plusieurs raisons qui ne sont pas incluses dans le reasons_mask.

(f) Obtenir et valider le chemin de certification pour le producteur de la CRL complète. L'ancre de confiance pour le chemin de certification DOIT être la même que l'ancre de confiance utilisée pour valider le certificat cible. Si une extension d'usage de clé est présente dans le certificat du producteur de CRL, vérifier que le bit cRLSign est établi.

(g) Valider la signature sur la CRL complète en utilisant la clé publique validée dans l'étape (f).

(h) Si use-deltas est établi, valider alors la signature sur la CRL delta en utilisant la clé publique validée dans l'étape (f).

(i) Si use-deltas est établi, chercher alors le certificat sur la CRL delta. Si une entrée est trouvée qui correspond au producteur de certificat et au numéro de série comme décrit au paragraphe 5.3.3, alors établir la variable cert_status à la raison indiquée comme suit :
   (1) Si l'extension d'entrée de CRL code de cause est présente, régler la variable cert_status à la valeur de l'extension d'entrée de CRL code de cause.
   (2) Si l'extension d'entrée de CRL code de cause n'est pas présente, régler la variable cert_status à la valeur non spécifiée.

(j) Si (cert_status est UNREVOKED) chercher alors le certificat sur la CRL complète. Si une entrée est trouvée qui correspond au producteur de certificat et numéro de série comme décrit au paragraphe 5.3.3, régler alors la variable cert_status à la raison indiquée comme décrit dans l'étape (i).

(k) Si (cert_status est removeFromCRL), régler alors cert_status à UNREVOKED.

(l) régler la variable d'état reasons_mask à l'union de sa valeur précédente et de la valeur de la variable d'état interim_reasons_mask.

Si ((reasons_mask est all-reasons) OU (cert_status n'est pas UNREVOKED)) alors le statut de révocation a été déterminé, donc on retourne cert_status.

Si le statut de révocation n'a pas été déterminé, répéter le processus ci-dessus avec toute CRL disponible non spécifiée dans un point de distribution mais produite par le producteur de certificat. Pour le traitement d'une telle CRL, on suppose un DP avec les deux champs raisons et cRLIssuer omis et un nom de point de distribution du producteur de certificat. C'est-à-dire, la séquence de noms dans fullName est générée à partir du champ Producteur de certificat ainsi que l'extension de certificat issuerAltName. Après le traitement de telles CRL, si le statut de révocation n'a toujours pas été déterminé, retourner alors le cert_status UNDETERMINED.

7. Règles de traitement pour les noms internationalisés

Des noms internationalisés peuvent se rencontrer dans de nombreux champs et extensions de certificat et CRL, incluant des noms distinctifs, des noms de domaine internationalisés, des adresses de messagerie électronique, et des identifiants de ressource internationalisés (IRI). La mémorisation, comparaison, et présentation de ces noms exige une attention particulière. Certains caractères peuvent être codés de plusieurs façons. Les mêmes noms pourraient être représentés dans plusieurs codages (par exemple, ASCII ou UTF8). Cette section établit des exigences de conformité pour la mémorisation ou comparaison de chacune de ces formes de noms. Des lignes directrices pour information sont fournies pour certaines de ces formes de nom.

7.1 Noms internationalisés dans les noms distinctifs

La représentation des noms internationalisés dans des noms distinctifs est couverte aux paragraphes 4.1.2.4, Nom de producteur, et 4.1.2.6, Nom de sujet. Des attributs standard de dénomination, comme des noms communs, emploient le type DirectoryString, qui prend en charge les noms internationalisés par divers codages de langage. Les mises en œuvre conformes DOIVENT prendre en charge UTF8String et PrintableString. La RFC 3280 exigeait seulement la comparaison binaire des valeurs d'attribut codées en UTF8String, cependant, la présente spécification exige un traitement de comparaison plus complet. Les mises en œuvre peuvent rencontrer des certificats et CRL avec des noms codés en utilisant TeletexString, BMPString, ou UniversalString, mais leur prise en charge est FACULTATIVE.

Les mises en œuvre conformes DOIVENT utiliser le profil LDAP StringPrep (incluant un traitement des espaces non significatives) comme spécifié dans la [RFC4518], comme base de comparaison des attributs de nom distinctif codés en PrintableString ou UTF8String. Les mises en œuvre conformes DOIVENT prendre en charge les comparaisons utilisant caseIgnoreMatch. La prise en charge des types d'attribut qui utilisent d'autres règles de comparaison d'égalité est facultative.

Avant de comparer les noms en utilisant la règle de correspondance caseIgnoreMatch, les mises en œuvre conformes DOIVENT effectuer l'algorithme de préparation de chaîne en six étapes décrit dans la [RFC4518] pour chaque attribut de type DirectoryString, avec les précisions suivantes :

* Dans l'étape 2, Transposition, la transposition devra inclure le repli de casse comme spécifié dans l'Appendice B.2 de la [RFC3454].
* Dans l'étape 6, Suppression des caractères non significatifs, effectuer la compression d'espaces, comme spécifié au paragraphe 2.6.1, Traitement des espaces non significatives, de la [RFC4518].

Quand on effectue l'algorithme de préparation de chaîne, les attributs DOIVENT être traités comme des valeurs mémorisées.

Les comparaisons des attributs domainComponent DOIVENT être effectuées comme spécifié au paragraphe 7.3.

Deux attributs de dénomination correspondent si les types d'attributs sont les mêmes et si les valeurs des attributs sont une correspondance exacte après le traitement avec l'algorithme de préparation de chaîne. Deux noms distinctifs relatifs RDN1 et RDN2 correspondent si ils ont le même nombre d'attributs de désignation et si pour chaque attribut de désignation dans RDN1 il y a un attribut de désignation correspondant dans RDN2. Deux noms distinctifs DN1 et DN2 correspondent si ils ont le même nombre de RDN, si pour chaque RDN dans DN1 il y a un RDN correspondant dans DN2, et si les RDN correspondants apparaissent dans le même ordre dans les deux DN. Un nom distinctif DN1 est dans la sous arborescence définie par le nom distinctif DN2 si DN1 contient au moins autant de RDN que DN2, et si DN1 et DN2 correspondent quand les RDN en queue dans DN1 sont ignorés.

7.2 Noms de domaine internationalisés dans GeneralName

Des noms de domaine internationalisés (IDN, *Internationalized Domain Name*) peuvent être inclus dans des certificats et CRL dans les extensions subjectAltName et issuerAltName, l'extension contraintes de nom, l'extension accès aux informations d'autorité, l'extension accès aux informations de sujet, l'extension points de distribution de CRL, et l'extension point de distribution producteur. Chacune de ces extensions utilise le type GeneralName ; un choix dans GeneralName est le champ dNSName, qui est défini comme type IA5String.

IA5String est limité au jeu de caractères ASCII. Pour s'accommoder des noms de domaine internationalisés dans la structure actuelle, les mises en œuvre conformes DOIVENT convertir les noms de domaine internationalisés au format de codage comptatible ASCII (ACE, *ASCII Compatible Encoding*) comme spécifié à la Section 4 de la RFC 3490 avant la mémorisation dans le champ dNSName. Précisément, les mises en œuvre conformes DOIVENT effectuer l'opération de conversion spécifiée à la Section 4 de la RFC 3490, avec les précisions suivantes :

* dans l'étape 1, le nom de domaine DEVRA être considéré comme une "chaîne mémorisée". C'est-à-dire, le fanion AllowUnassigned NE DEVRA PAS être établi ;
* dans l'étape 3, établir le fanion appelé "UseSTD3ASCIIRules" ;
* dans l'étape 4, traiter chaque étiquette avec l'opération "ToASCII" ; et
* dans l'étape 5, changer tous les séparateurs d'étiquette en U+002E (point).

Quand on compare les noms DNS pour égalité, les mises en œuvre conformes DOIVENT effectuer une correspondance exacte insensible à la casse sur le nom DNS entier. Quand elles évaluent les contraintes de nom, les mises en œuvre conformes DOIVENT effectuer une correspondance exacte insensible à la casse étiquette par étiquette. Comme noté au paragraphe 4.2.1.10, tout nom DNS qui peut être construit en ajoutant des étiquettes au côté gauche du nom de domaine donné comme la contrainte est considéré entrer dans la sous-arborescence indiquée.

Les mises en œuvre devraient convertir les IDN en Unicode avant l'affichage. Précisément, les mises en œuvre conformes devraient effectuer l'opération de conversion spécifiée à la Section 4 de la RFC 3490, avec les précisions suivantes :

* dans l'étape 1, le nom de domaine DEVRA être considéré comme une "chaîne mémorisée". C'est-à-dire, le fanion AllowUnassigned NE DEVRA PAS être établi ;
* dans l'étape 3, établir le fanion appelé "UseSTD3ASCIIRules" ;
* dans l'étape 4, traiter chaque étiquette avec l'opération "ToUnicode" ; et
* sauter l'étape 5.

Note : les mises en œuvre DOIVENT permettre des exigences accrue d'espaces pour les IDN. Une étiquette IDN ACE va commencer par les quatre caractères supplémentaires "xn--" et peut exiger jusqu'à cinq caractères ASCII pour spécifier un seul caractère international.

7.3 Noms de domaine internationalisés dans les noms distinctifs

Les noms de domaines peuvent aussi être représentés comme des noms distinctifs utilisant des composants de domaine dans le champ sujet, le champ producteur, l'extension subjectAltName, ou l'extension issuerAltName. Comme avec le dNSName dans le type GeneralName, la valeur de cet attribut est définie comme une IA5String. Chaque attribut domainComponent représente une seule étiquette. Pour représenter une étiquette provenant d'un IDN dans le nom distinctif, la mise en œuvre DOIT effectuer la conversion d'étiquette "ToASCII" spécifiée au paragraphe 4.1 de la RFC 3490. L'étiquette DEVRA être considérée comme une "chaîne mémorisée". C'est-à-dire, le fanion AllowUnassigned NE DEVRA PAS être établi.

Les mises en œuvre conformes devront effectuer une correspondance exacte insensible à la casse lors de la comparaison des attributs domainComponent dans les noms distinctifs, comme décrit au paragraphe 7.2.

Les mises en œuvre devraient convertir les étiquettes ACE en Unicode avant l'affichage. Précisément, les mises en œuvre conformes devraient effectuer l'opération de conversion "ToUnicode" spécifiée, comme décrit au paragraphe 7.2, sur chaque étiquette ACE avant d'afficher le nom.

7.4 Identifiants de ressource internationalisée

Les identifiants de ressource internationalisées (IRI, *Internationalized Resource Identifier*) sont les compléments internationalisés de l'identifiant de ressource universel (URI, *Uniform Resource Identifier*). Les IRI sont des séquences de caractères Unicode, tandis que les URI sont des séquences de caractères du jeu de caractères ASCII. La [RFC3987] définit une transposition des IRI en URI. Alors que les IRI ne sont pas codés directement dans un champ ou extension de certificat, leurs URI transposés peuvent être inclus dans les certificats et les CRL. Les URI peuvent apparaître dans les extensions subjectAltName et issuerAltName, les extensions contraintes de nom, l'extension accès aux informations d'autorité, l'extension accès aux informations de sujet, l'extension point de distribution producteur, et l'extension points de distribution de CRL. Chacune de ces extensions utilise le type GeneralName ; les URI sont codés dans le champ uniformResourceIdentifier dans GeneralName, qui est défini comme type IA5String.

Pour accommoder les IRI dans la structure actuelle, les mises en œuvre conformes DOIVENT transposer les IRI en URI comme spécifié au paragraphe 3.1 de la [RFC3987], avec les précisions suivantes :

* dans l'étape 1, générer une séquence de caractères UCS à partir du format original d'IRI pour normaliser conformément à la NFC comme spécifié dans la variante b (normalisation selon la NFC) ;
* effectuer l'étape 2 en utilisant le résultat de l'étape 1.

Les mises en œuvre NE DOIVENT PAS convertir le composant ireg-name avant d'effectuer l'étape 2.

Avant que les URI puissent être comparés, les mises en œuvre conformes DOIVENT effectuer une combinaison des techniques de normalisation fondées sur la syntaxe et de celles fondées sur le schéma décrites dans la [RFC3987]. Précisément, les mises en œuvre conformes DOIVENT préparer les URI pour la comparaison comme suit :

Étape 1 : Lorque les IRI permettent l'usage des IDN, ces noms DOIVENT être convertis en codage compatible ASCII comme spécifié au paragraphe 7.2.
Étape 2 : Le schéma et l'hôte sont normalisés en minuscules, comme décrit au paragraphe 5.3.2.1 de la [RFC3987].
Étape 3 : On effectue la normalisation en codage de pourcentage, comme spécifié au paragraphe 5.3.2.3 de la [RFC3987].
Étape 4 : On effectue la normalisation de segment de chemin, comme spécifié au paragraphe 5.3.2.4 de la [RFC3987].
Étape 5 : Si elle le reconnaît, la mise en œuvre DOIT effectuer la normalisation fondée sur le schéma comme spécifié au paragraphe 5.3.3 de la [RFC3987].

Les mises en œuvre conformes DOIVENT reconnaître et effectuer la normalisation fondée sur le schéma pour les schémas suivants : ldap, http, https, et ftp. Si le schéma n'est pas reconnu, l'étape 5 est omise.

Quand elles comparent les URI pour équivalence, les mises en œuvre conformes devront effectuer une correspondance exacte sensible à la casse.

Les mises en œuvre devraient convertir les URI en Unicode avant l'affichage. Précisément, les mises en œuvre conformes devraient effectuer l'opération de conversion spécifiée au paragraphe 3.2 de la [RFC3987].

7.5. Adresses de messagerie électronique internationalisées

Des adresses de messagerie électronique peuvent être incluses dans des certificats et CRL dans des extensions subjectAltName et issuerAltName, des extensions contraintes de nom, des extensions accès aux informations d'autorité, des extensions accès aux informations de sujet, des extensions point de distribution producteur, ou des extension points de distribution de CRL. Chacune de ces extensions utilise la construction GeneralName ; GeneralName inclut le choix rfc822Name, qui est défini comme type IA5String. Pour accommoder les adresses de messagerie électronique avec des noms de domaine internationalisés en utilisant la structure actuelle, les mises en œuvre conformes DOIVENT convertir les adresses en une représentation ASCII.

Lorsque la partie hôte (le domaine de la boîte aux lettres) contient un nom internationalisé, le nom de domaine DOIT être converti d'un IDN en le format de codage compatible ASCII (ACE) comme spécifié au paragraphe 7.2.

Deux adresses de messagerie électronique sont considérées correspondre si :
1) la partie locale de chaque nom est une correspondance exacte, ET
2) la partie hôte de chaque nom correspond en utilisant une comparaison ASCII insensible à la casse.

Les mises en œuvre devraient convertir la partie hôte des adresses de messagerie internationalisée spécifiée dans ces extensions à Unicode avant l'affichage. Précisément, les mises en œuvre conformes devraient effectuer la conversion de la partie hôte de la boîte aux lettres comme décrit au paragraphe 7.2.

8. Considérations pour la sécurité

La majorité de la présente spécification est dédiée au format et au contenu des certificats et des CRL. Comme les certificats et les CRL sont signés numériquement, aucun service de protection de l'intégrité suplémentaire n'est nécessaire. Ni les certificats ni les CRL n'ont besoin d'être gardés secrets, et un accès sans restriction et anonyme aux certificats et CRL n'a pas d'implications pour la sécurité.

Cependant, des facteurs de sécurité hors de la portée de la présente spécification vont affecter l'assurance fournie aux utilisateurs de certificats. Cette section souligne des problèmes critiques à considérer par les mises en œuvre, les administrateurs, et les utilisateurs.

Les procédures effectuées par les CA et RA pour valider le lien de l'identité du sujet à leur clé publique affecte largement l'assurance qui devrait être accordée au certificat. Les parties consommatrices pourraient souhaiter revoir la déclaration de pratique de certification de la CA. Ceci est particulièrement important quand elles produisent des certificats aux autres CA.

L'utilisation d'une seule paire de clés à la fois pour la signature et d'autres obbjets est vivement déconseillée. L'utilisation de paires de clés séparées pour la signature et la gestion de clé procure plusieurs avantages aux utilisateurs. Les ramifications associées à la perte ou la divulgation d'une clé de signature sont différentes de la perte ou de la divulgation d'une clé de gestion de clé. Utiliser des paires de clé séparées permet une réponse équilibrée et souple. De même, différentes périodes de validité ou longueurs de clé pour chaque paire de clés peuvent être appropriées dans certains environnements d'application. Malheureusement, certaines applications traditionnelles (par exemple, la couche de connexion sécurisée (SSL, *Secure Sockets Layer*)) utilisent une seule paire de clés pour la signature et la gestion de clé.

La protection fournie par les clés privées est un facteur de sécurité critique. À petite échelle, la défaillance des utilisateurs à protéger leurs clés privées va permettre à un attaquant de se faire passer pour eux ou à déchiffrer leurs informations personnelles. À plus grande échelle, la compromission de la clé de signature privée d'une CA peut avoir un effet catastrophique. Si un attaquant obtient la clé privée sans se faire remarquer, il peut produire des certificats et des CRL bogués. L'existence de certificats et CRL bogués va saper la confiance dans le système. Si une telle compromission est détectée, tous les certificats produits à la CA compromise DOIVENT être révoqués, empêchant les services entre ses utilisateurs et ceux des autres CA. La reconstruction après une telle compromission va être problématique, de sorte que il est conseillé aux CA de mettre en œuvre une combinaison de mesures techniques fortes (par exemple, des modules de chiffrement résistant à l'altération) et des procédures de gestion appropriées (par exemple, la séparation des tâches) pour éviter de tels incidents.

La perte de la clé de signature privée d'une CA peut aussi être problématique. La CA ne va pas être capable de produire des CRL ou d'effectuer le changement normal de clés. Les CA DEVRAIENT maintenir des sauvegardes sûres pour les clés de signature. La sécurité de la procédure de sauvegarde de clé est un facteur critique pour éviter la compromission de clé.

La disponibilité et la fraîcheur des informations de révocation affectent le degré d'assurance qui devrait être porté à un certificat. Bien que les certificats expirent naturellement, des événements peuvent survenir durant leur vie naturelle qui

rompent le lien entre le sujet et la clé publique. Si les informations de révocation ne sont pas en temps utile ou sont indisponibles, l'assurance associée au lien est clairement réduite. Les parties consommatrices pourraient n'être pas capables de traiter chaque extension critique qui peut apparaître dans une CRL. Les CA DEVRAIENT prendre un soin particulier à ne rendre les informations de révocation disponible qu'à travers les CRL qui contiennent des extensions critiques, en particulirer si la prise en charge de ces extensions n'est pas rendue obligatoire par le présent profil. Par exemple, si les informations de révocation sont fournies en utilisant une combinaison de CRL delta et de CRL complètes, et si les CRL delta sont produites plus fréquemment que les CRL complètes, les parties consommatrices qui ne peuvent pas traiter les extensions critiques relatives au traitement des CRL delta ne vont pas être capables d'obtenir les plus récentes informations de révocation. Autrement, si une CRL complète est produite chaque fois qu'une CRL delta est produite, des informations de révocation à jour vont être disponibles en temps utile à toutes les parties consommatrices. De même, les mises en œuvre du mécanisme de validation du chemin de certification décrites à la Section 6 qui omettent la vérification de révocation fournissent moins d'assurance que celles qui la prennent en charge.

L'algorithme de validation de chemin de certification dépend de la connaissance certaine de la clé publiques (et autres informations) sur une ou plusieurs CA de confiance. La décision de faire confiance à une CA est une importante décision car elle détermine en fin de compte la confiance accordée à un certificat. La distribution authentifiée des clés publiques d'une CA de confiance (généralement sous la forme d'un certificat "auto-signé") est un processus de sécurité critique hors bande qui sort du domaine d'application de la présente spécification.

De plus, lorsque une clé compromise ou une défaillance de CA se produit pour une CA de confiance, l'utilisateur va devoir modifier les informations fournies au programme de validation de chemin. Le choix de trop nombreuses CA de confiance rend les informations de CA de confiance difficiles à maintenir. Par ailleurs, le choix d'une seule CA de confiance pourrait limiter les utilisateurs à une communauté fermée d'utilisateurs.

La qualité des mises en œuvre qui traitent les certificats affecte aussi le degré d'assurance fournie. L'algorithme de validation de chemin décrit à la Section 6 s'appuie sur l'intégrité des informations de la CA de confiance, et en particulier sur l'intégrité des clés publiques associées aux CA de confiance. En substituant les clés publiques pour lesquelles il a la clé privée, un attaquant pourrait amener l'utilisateur à accepter de faux certificats.

Le lien entre une clé et le sujet du certificat ne peut pas être plus fort que la mise en œuvre et les algorithmes de module de chiffrement utilisés pour générer la signature. De courtes longueurs de clé ou des algorithmes de hachage faibles vont limiter l'utilité d'un certificat. Les CA sont invitées à noter les avancées de la cryptologie afin qu'elles puissent employer des techniques de chiffrement fortes. De plus, les CA DEVRAIENT refuser de produire des certificats aux CA ou entités d'extrémité qui génèrent des signatures faibles.

Une application non cohérente des règles de comparaison de noms peut résulter en l'acceptation de certifications de chemin X.509 invalides ou au rejet de certifications valides. La série de spécifications X.500 définit les règles de comparaison des noms distinctifs qui exigent une comparaison de chaînes sans égard à la casse, au jeu de caractères, aux sous chaînes d'espace multi-caractères, ou aux espaces en tête et en queue. La présente spécification assouplit ces exigences, exigeant au minimum la prise en charge de la comparaison binaire.

Les CA DOIVENT coder le nom distinctif dans le champ sujet d'un certificat de CA de façon identique au nom distinctif dans le champ Producteur dans les certificats produits par cette CA. Si les CA utilisent des codages différents, les mises en œuvre pourraient échouer à reconnaître les chaînes de noms pour les chemins qui incluent ce certificat. Par conséquent, des chemins valides pouraient être rejetés.

De plus, les contraintes de nom pour les noms distinctifs DOIVENT être déclarées de façon identique au codage utilisé dans le champ sujet ou l'extension subjectAltName. Sinon, les contraintes de nom déclarées comme excludedSubtrees ne vont pas corespondre et des chemins invalides vont être acceptés et des contraintes de nom exprimées comme permittedSubtrees ne vont pas correspondre et des chemins valides vont être rejetés. Pour éviter d'accepter des chemins invalides, les CA DEVRAIENT déclarer les contraintes de nom pour les noms distinctifs comme permittedSubtrees chaque fois que possible.

En général, utiliser l'extension nameConstraints pour contraindre une forme de nom (par exemple, noms DNS) n'offre pas de protection contre l'utilisation d'autres formes de nom (par exemple, des adresses de messagerie électronique).

Bien que X.509 oblige à ce que les noms soient sans ambiguïté, il y a un risque que deux autorités sans relation produisent des certificats et/ou des CRL sous le même nom de producteur. Un moyen pour réduire les problèmes de sécurité relatifs aux collisions de nom de producteur est que les noms de la CA et du producteur de CRL DEVRAIENT être formés d'une façon qui réduise la probabilité de collisions de noms. Les mises en œuvre devraient prendre en compte l'existence possible de plusieurs CA et producteurs de CRL avec le même nom. Au minimum, les mises en œuvre qui valident les CRL DOIVENT s'assurer que le chemin de certification d'un certificat et que le chemin de certification du producteur de CRL utilisés pour valider le certificat se terminent à la même ancre de confiance.

Bien que la partie locale d'une adresse de messagerie électronique soit sensible à la casse [RFC2821], les valeurs d'attribut emailAddress ne sont pas sensibles à la casse [RFC2985]. Par suite, il y a un risque que deux adresses de messagerie différentes soient traitées comme la même adresse quand la règle de correspondance pour l'attribut emailAddress est utilisée, si le serveur de messagerie exploite la sensibilité à la casse des parties locales des boîtes aux lettres. Les mises en œuvre ne devraient pas inclure d'adresse de messagerie dans l'attribut emailAddress si le serveur de messagerie qui héberge l'adresse de messagerie traite la partie locale de l'adresse comme sensible à la casse.

Les mises en œuvre devraient être conscientes des risques impliqués si les points de distribution de CRL ou les extensions d'accès aux informations d'autorité de certificats ou CRL corrompus contiennent des liens sur du code malveillant. Les mises en œuvre devraient toujours suivre les étapes de validation des données restituées pour s'assurer que les données sont correctement formées.

Quand des certificats incluent une extension cRLDistributionPoints avec un URI https ou schéma similaire, des dépendances circulaires peuvent être introduites. Le consommateur d'assertions est forcé d'effectuer une validation de chemin supplémentaire afin d'obtenir la CRL requise pour achever la validation de chemin initiale ! Des conditions circulaires peuvent aussi être créées avec un URI https (ou schéma similaire) dans les extensions authorityInfoAccess ou subjectInfoAccess. Au pire, cette situation peut créer des dépendances insolubles.

Les CA NE DEVRAIENT PAS inclure des URI qui spécifient https, ldaps, ou des schémas similaires dans les extensions. Les CA qui incluent un URI https dans une de ces extensions DOIVENT s'assurer que le certificat du serveur peut être validé sans utiliser les informations qui sont pointées par l'URI. Les parties consommatrices qui choisissent de valider le certificat du serveur quand elles obtiennent les informations pointées par un URI https dans les extensions cRLDistributionPoints, authorityInfoAccess, ou subjectInfoAccess DOIVENT être prêtes à la possibilité que cela résulte en une récurrence illimitée.

Les certificats auto-produits fournissent aux CA un mécanisme automatique pour indiquer des changements dans les opérations de la CA. En particulier, les certificats auto-produits peuvent être utilisés pour mettre en œuvre un changement en douceur d'une paire de clés de CA non compromise à la prochaine. Les procédures détaillées pour "la mise à jour de clé de CA" sont spécifiées dans la [RFC4210], où la CA protège sa nouvelle clé publique en utilisant sa précédente clé privée et vice versa en utilisant deux certificats auto-produits. Les mises en œuvre de client conformes vont traiter le certificat auto-produit et déterminer si les certificats produits sous la nouvelle clé peuvent être de confiance. Les certificats auto-produits PEUVENT être utilisés pour prendre en charge d'autres changements dans les opérations de CA, comme des ajouts à l'ensemble de politique de la CA, en utilisant des procédures similaires.

Certaines mises en œuvre traditionnelles prennent en charge des noms codés dans le jeu de caractères ISO 8859-1 (Latin1String) [ISO8859] mais les étiquette comme des TeletexString. TeletexString code un plus grand jeu de caractères que ISO 8859-1, mais il code certains caractères différemment. Les règles de comparaison de noms spécifiées au paragraphe 7.1 supposent que les TeletexString sont codées comme décrit dans la norme ASN.1. Quand on compare des noms codés en utilisant le jeu de caractères Latin1String, des faux positifs et négatifs sont possibles.

Quand des chaînes sont transposées de représentations internes à des représentations visuelles, parfois deux chaînes différentes vont avoir la même représentation visuelle ou similaire. Cela peut arriver pour de nombreuses raisons différentes, incluant l'utilisation de glyphes similaires et l'utilisation de caractères composés (comme e + ' égal U+00E9, les caractères coréens composés, et les voyelles au dessus de consonnes dans certains langages). Par suite de cette situation, les gens qui font des comparaisons visuelles entre deux noms différents peuvent penser qu'ils sont les mêmes alors qu'en fait ils ne le sont pas. Aussi, les gens peuvent prendre par erreur une chaîne pour une autre. Les producteurs de certificats et les parties consommatrices doivent être conscientes de cette situation.

9. Considérations relatives à l'IANA

Les extensions dans les certificats et les CRL sont identifiées en utilisant des identifiants d'objet. Les objets sont définis dans une archive déléguée par l'IANA au groupe de travail PKIX. Aucune autre action de l'IANA n'est nécessaire pour le présent document ou ses mises à jour prévues.

10. Remerciements

Warwick Ford a participé avec les auteurs à certaines des réunions de l'équipe de conception qui a dirigé le développement du présent document. Les efforts de l'équipe de conception ont été éclairés par les contributions de Matt Crawford, Tom Gindin, Steve Hanna, Stephen Henson, Paul Hoffman, Takashi Ito, Denis Pinkas, et Wen-Cheng Wang.

11. Références

11.1 Références normatives

[RFC0791] J. Postel, éd., "Protocole Internet - Spécification du protocole du programme Internet", STD 5, septembre 1981.

[RFC1034] P. Mockapetris, "Noms de domaines - Concepts et facilités", STD 13, novembre 1987.

[RFC1123] R. Braden, éditeur, "Exigences pour les hôtes Internet – Application et prise en charge", STD 3, octobre 1989.

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

[RFC2460] S. Deering et R. Hinden, "Spécification du protocole Internet, version 6 (IPv6) ", décembre 1998. *(MàJ par 5095, 6564 ; D.S)*

[RFC2585] R. Housley et P. Hoffman, "Protocoles de fonctionnement de l'infrastructure de clé publique X.509 pour l'Internet : FTP et HTTP", mai 1999. *(P.S.)*

[RFC2616] R. Fielding et autres, "Protocole de transfert hypertexte -- HTTP/1.1", juin 1999. *(D.S., MàJ par 2817, 6585)*

[RFC2797] M. Myers, X. Liu, J. Schaad et J. Weinstein, "Messages de gestion de certificat sur CMS", avril 2000. (*Obsolète, voir 5272, P.S*)

[RFC2821] J. Klensin, éditeur, "Protocole simple de transfert de messagerie", STD 10, avril 2001. *(Obsolète, voir RFC5321)*

[RFC3454] P. Hoffman et M. Blanchet, "Préparation de chaînes internationalisées ("stringprep")", décembre 2002. *(P.S.)*

[RFC3490] P. Faltstrom et autres, "Internationalisation des noms de domaine dans les applications (IDNA)", mars 2003. (*Remplacée par les* RFC5890 et 5891*, D.S.*)

[RFC3629] F. Yergeau, "UTF-8, un format de transformation de la norme ISO 10646", STD 63, novembre 2003.

[RFC3986] T. Berners-Lee, R. Fielding et L. Masinter, "Identifiant de ressource uniforme (URI) : Syntaxe générique", STD 66, janvier 2005.

[RFC3987] M. Duerst et M. Suignard, "Identifiant de ressource internationalisé (IRI)", janvier 2005.

[RFC4516] M. Smith, éd. et T. Howes, "Protocole léger d'accès à un répertoire (LDAP) : Localisateur uniforme de ressource", juin 2006.

[RFC4518] K. Zeilenga, "Protocole léger d'accès à un répertoire (LDAP) : Préparation de chaîne internationalisée", juin 2006.

[RFC4523] K. Zeilenga, "Protocole léger d'accès à un répertoire (LDAP) : Définitions de schémas pou les certificats X.509", juin 2006.

[RFC4632] V. Fuller et T. Li, "Acheminement inter domaine sans classe (CIDR) : Plan d'allocation et d'agrégation des adresses Internet", août 2006. (BCP 122)

[X.680] Recommandation UIT-T X.680 (1997) "Notation numéro un de syntaxe abstraite (ASN.1) - Spécification de la notation de base", (aussi ISO/CEI 8824-1:1998).

[X.690] Recommandation UIT-T X.690 (1997) "Spécification des règles de codage ASN.1 : Règles de codage de base (BER), Règles de codage canonique (CER), et Règles de codage distinctives (DER)", X.690 (1997) (aussi ISO/CEI 8825-1:1998).

11.2 Références pour information

[ISO8859] ISO/CEI 8859-1:1998. "Technologies de l'information – Jeux de caractères graphiques codés sur un seul octet de 8 bits -- Partie 1 : alphabet latin n° 1".

[ISO10646] ISO/CEI 10646:2003. "Technologies de l'information – Jeux de caractères universels codés sur plusieurs octets (UCS)".

[NFC] M. Davis and M. Duerst, "Unicode Standard Annex #15: Unicode Normalization Forms", octobre 2006, <http://www.unicode.org/reports/tr15/ >.

[RFC1422] S. Kent, "Amélioration de la confidentialité pour la messagerie électronique Internet : Partie II – Gestion de clés fondée sur le certificat", février 1993. *(Historique)*

[RFC2277] H. Alvestrand, "Politique de l'IETF en matière de jeux de caractères et de langages", BCP 18, janvier 1998.

[RFC2459] R. Housley, W. Ford, W. Polk et D. Solo, "Profil de certificat d'infrastructure de clé publique X.509 et de CRL pour l'Internet", janvier 1999. (*Obsolète, voir la* RFC3280) *(P.S.)*

[RFC2560] M. Myers, R. Ankney, A. Malpani, S. Galperin et C. Adams, "Protocole d'état de certificat en ligne d'infrastructure de clé publique X.509 pour l'Internet - OCSP", juin 1999. *(P.S.)*

[RFC2985] M. Nystrom et B. Kaliski, "PKCS n° 9 : Classes d'objet et types d'attribut choisis, version 2.0", novembre 2000. *(Information)*

[RFC3161] C. Adams, P. Cain, D. Pinkas et R. Zuccherato, "Protocole d'horodatage (TSP) d'infrastructure de clé publique X.509 pour l'Internet", août 2001.

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

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

[RFC4055] J. Schaad et autres, "Algorithmes et identifiants supplémentaires pour la cryptographie RSA à utiliser dans le profil de certificat d'infrastructure de clé publique X.509 et de liste de révocation de certificat (CRL) pour l'Internet", juin 2005.

[RFC4120] C. Neuman et autres, "Service Kerberos d'authentification de réseau (V5)", juillet 2005. (*MàJ par* RFC4537, RFC5021, RFC6649)

[RFC4210] C. Adams et autres, "Protocole de gestion de certificat (CMP) d'infrastructure de clé publique X.509 pour l'Internet", septembre 2005. (*MàJ par la* RFC6712) *(P.S.)*

[RFC4325] S. Santesson et R. Housley, "Extension de liste de révocation de certificat (CRL) d'accès aux informations sur les autorités d'infrastructure de clé publique X.509 pour l'Internet", décembre 2005. (*Obsolète, voir* RFC5280)

[RFC4491] S. Leontiev et D. Shefanovski, éditeurs, "Utilisation des algorithmes GOST R 34.10-94, GOST R 34.10-2001 et GOST R 34.11-94 avec le profil de certificat d'infrastructure de clé publique X.509 et de liste de révocation de certificat (CRL) pour l'Internet", mai 2006. *(P.S.)*

[RFC4510] K. Zeilenga, éditeur, "Protocole léger d'accès à un répertoire (LDAP) : Descriptif des spécifications techniques", juin 2006.

[RFC4512] K. Zeilenga, "Protocole léger d'accès à un répertoire (LDAP) : Modèle d'informations de répertoires", juin 2006.

[RFC4514] K. Zeilenga, éd., "Protocole léger d'accès à un répertoire (LDAP) : Représentation de chaîne des noms distinctifs", juin 2006.

[RFC4519] A. Sciberras, éd., "Protocole léger d'accès à un répertoire (LDAP) : Schéma pour les applications d'utilisateur", juin 2006.

[RFC4630] R. Housley et S. Santesson, "Mise à jour du traitement DirectoryString dans le profil de certificat d'infrastructure de clé publique X.509 et de liste de révocation de certificat (CRL) pour l'Internet", août 2006.

[X.500] Recommandation UIT-T X.500 (1993) "L'Annuaire – Généralités sur les concepts, modèles et services", (aussi ISO/CEI 9594-1:1994).

[X.501] Recommandation UIT-T X.501 (1993) "L'Annuaire -- Modèles," (aussi ISO/CEI 9594-2:1994).

[X.509] Recommandation UIT-T X.509 (2005) | ISO/CEI 9594-8:2005, "Technologies de l'information – Interconnexion des systèmes ouverts - L'annuaire : Cadres de clé publique et de certificat d'attribut".

[X.520] Recommandation UIT-T X.520 (2005) | ISO/CEI 9594-6:2005, "Technologies de l'information – Interconnexion des systèmes ouverts - L'annuaire : Types d'attribut choisis".

[X.660] Recommandation UIT-T X.660 (2004) | ISO/CEI 9834-1:2005, "Technologies de l'information - Interconnexion des systèmes ouverts - Procédures pour le fonctionnement des autorités d'enregistrement OSI : procédures générales, et arcs sommitaux de l'arborescence d'identifiants d'objet ASN.1".

[X.683] Recommandation UIT-T X.683 (2002) | ISO/CEI 8824-4:2002, "Technologies de l'information - Notation numéro un de syntaxe abstraite (ASN.1) : paramètres des spécifications ASN.1".

[X9.55] ANSI X9.55-1997, "Public Key Cryptography for the Financial Services Industry: Extensions to Public Key Certificates and Certificate Revocation Lists", janvier 1997.

Appendice A. Structures et OID en pseudo ASN.1

Le présent appendice décrit les objets de données utilisés par les composants PKI conformes dans une syntaxe "de style ASN.1". Cette syntaxe est une hybridation des syntaxes ASN.1 de 1988 et de 1993. La syntaxe ASN.1 de 1988 est augmentée par les types UNIVERSEL 1993 UniversalString, BMPString, et UTF8String.

La syntaxe ASN.1 ne permet pas l'inclusion de déclarations de type dans le module ASN.1, et l'ASN.1 1993 standard ne permet pas l'utilisation des nouveaux types UNIVERSEL dans les modules qui utilisent la syntaxe de 1988. Il en résulte que ce module ne se conforme à aucune version de la norme ASN.1.

Le présent appendice peut être converti en ASN.1 1988 en remplaçant les définitions pour les types UNIVERSEL par le fourre tout "ANY" de 1988.

A.1 Module explicitement étiqueté, syntaxe de 1988

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

ÉTIQUETTES EXPLICITES DE DÉFINITION ::=

DÉBUT

-- EXPORTE TOUT --

-- IMPORTE RIEN --

-- Types UNIVERSAL définis dans l'ASN.1 1993 et 1998 et exigés par la présente spécification

UniversalString ::= [UNIVERSAL 28] CHAINE D'OCTETS IMPLICITE
    -- UniversalString est défini dans ASN.1:1993

```
BMPString ::= [UNIVERSAL 30] CHAINE D'OCTETS IMPLICITE
    -- BMPString est le sous type de UniversalString et modèlise le plan multilingue de base de ISO/CEI 10646

UTF8String ::= [UNIVERSAL 12] CHAINE D'OCTETS IMPLICITE
    -- Le contenu de ce type se conforme à la RFC 3629.

-- OID spécifiques de PKIX

IDENTIFIANT D'OBJET id-pkix ::= { iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5)
pkix(7) }

-- arcs PKIX

IDENTIFIANT D'OBJET id-pe ::= { id-pkix 1 }          -- arc pour extensions privées de certificat
IDENTIFIANT D'OBJET id-qt ::= { id-pkix 2 }          -- arc pour types de qualificatifs de politique
IDENTIFIANT D'OBJET id-kp ::= { id-pkix 3 }          -- arc pour les OID d'objet de clé étendus
IDENTIFIANT D'OBJET id-ad ::= { id-pkix 48 }         -- arc pour les descripteurs d'accès

-- policyQualifierIds pour qualificatifs de politique Internet

IDENTIFIANT D'OBJET id-qt-cps ::= { id-qt 1 }        -- OID pour qualificatif CPS
IDENTIFIANT D'OBJET id-qt-unotice ::= { id-qt 2 }    -- OID pour qulificatif de remarque d'utilisateur

-- Définitions de descripteur d'accès

IDENTIFIANT D'OBJET id-ad-ocsp ::= { id-ad 1 }
IDENTIFIANT D'OBJET id-ad-caIssuers ::= { id-ad 2 }
IDENTIFIANT D'OBJET id-ad-timeStamping ::= { id-ad 3 }
IDENTIFIANT D'OBJET id-ad-caRepository ::= { id-ad 5 }

-- Types de données d'attribut

Attribute  ::= SEQUENCE {
    type         AttributeType,
    valeurs      ENSEMBLE DE AttributeValue }          -- au moins une valeur est exigée

AttributeType ::= IDENTIFIANT D'OBJET

AttributeValue ::= ANY -- DÉFINI PAR AttributeType

AttributeTypeAndValue   ::= SEQUENCE {
     type     AttributeType,
     valeur   AttributeValue }

-- attributs de désignation suggérés : la définition de l'ensemble d'objet d'information suivant peut être augmentée pour
satifaire des exigences locales. Noter que supprimer des membres de l'ensemble peut empêcher l'interopérabilité avec les
mises en œuvre conformes.
-- présentées en paires : le AttributeType suivi par la définition de type pour l'arc AttributeValue correspondant pour les
attributs de dénomination standard.

IDENTIFIANT D'OBJET id-at ::= { joint-iso-ccitt(2) ds(5) 4 }

-- Attributs de dénomination de type X520name

id-at-name                AttributeType ::= { id-at 41 }
id-at-surname             AttributeType ::= { id-at  4 }
id-at-givenName           AttributeType ::= { id-at 42 }
id-at-initials            AttributeType ::= { id-at 43 }
id-at-generationQualifier AttributeType ::= { id-at 44 }

-- Attributs de dénomination de type X520Name :
-- X520name ::= DirectoryString (TAILLE (1..ub-name))
```

```
-- Étendu pour éviter un type paramétré :
X520name ::= CHOIX {
     teletexString      TeletexString    (TAILLE (1..ub-name)),
     printableString    PrintableString (TAILLE (1..ub-name)),
     universalString  UniversalString (TAILLE (1..ub-name)),
     utf8String         UTF8String       (TAILLE (1..ub-name)),
     bmpString          BMPString        (TAILLE (1..ub-name)) }

-- Attributs de dénomination de type X520CommonName :
id-at-commonName        AttributeType ::= { id-at 3 }

-- Attributs de dénomination de type X520CommonName :
--   X520CommonName ::= DirectoryName (TAILLE (1..ub-common-name))
--
-- Étendu pour éviter un type paramétré :
X520CommonName ::= CHOIX {
     teletexString      TeletexString     (TAILLE (1..ub-common-name)),
     printableString    PrintableString  (TAILLE (1..ub-common-name)),
     universalString    UniversalString (TAILLE (1..ub-common-name)),
     utf8String         UTF8String       (TAILLE (1..ub-common-name)),
     bmpString          BMPString        (TAILLE (1..ub-common-name)) }

-- Attributs de dénomination de type X520LocalityName
id-at-localityName      AttributeType ::= { id-at 7 }

-- Attributs de dénomination de type X520LocalityName:
--   X520LocalityName ::= DirectoryName (TAILLE (1..ub-locality-name))
--
-- Étendu pour éviter un type paramétré :
X520LocalityName ::= CHOIX {
     teletexString      TeletexString    (TAILLE (1..ub-locality-name)),
     printableString    PrintableString (TAILLE (1..ub-locality-name)),
     universalString    UniversalString (TAILLE (1..ub-locality-name)),
     utf8String         UTF8String       (TAILLE (1..ub-locality-name)),
     bmpString          BMPString        (TAILLE (1..ub-locality-name)) }

-- Attributs de dénomination de type X520StateOrProvinceName
id-at-stateOrProvinceName AttributeType ::= { id-at 8 }

-- Attributs de dénomination de type X520StateOrProvinceName:
-- X520StateOrProvinceName ::= DirectoryName (TAILLE (1..ub-state-name))
--
-- Étendu pour éviter un type paramétré :
X520StateOrProvinceName ::= CHOIX {
     teletexString      TeletexString    (TAILLE (1..ub-state-name)),
     printableString    PrintableString (TAILLE (1..ub-state-name)),
     universalString    UniversalString (TAILLE (1..ub-state-name)),
     utf8String         UTF8String       (TAILLE (1..ub-state-name)),
     bmpString          BMPString        (TAILLE (1..ub-state-name)) }

-- Attributs de dénomination de type X520OrganizationName
id-at-organizationName  AttributeType ::= { id-at 10 }

-- Attributs de dénomination de type X520OrganizationName:
--   X520OrganizationName ::= DirectoryName (TAILLE (1..ub-organization-name))
--
-- Étendu pour éviter un type paramétré :
X520OrganizationName ::= CHOIX {
     teletexString      TeletexString     (TAILLE (1..ub-organization-name)),
     printableString    PrintableString  (TAILLE (1..ub-organization-name)),
     universalString    UniversalString (TAILLE (1..ub-organization-name)),
     utf8String         UTF8String       (TAILLE (1..ub-organization-name)),
     bmpString          BMPString        (TAILLE (1..ub-organization-name))  }
```

```
-- Attributs de dénomination de type X520OrganizationalUnitName
id-at-organizationalUnitName AttributeType ::= { id-at 11 }

-- Attributs de dénomination de type X520OrganizationalUnitName:
--   X520OrganizationalUnitName ::= DirectoryName (TAILLE (1..ub-organizational-unit-name))
--
-- Étendu pour éviter un type paramétré :
X520OrganizationalUnitName ::= CHOIX {
     teletexString      TeletexString     (TAILLE (1..ub-organizational-unit-name)),
     printableString    PrintableString   (TAILLE (1..ub-organizational-unit-name)),
     universalString    UniversalString (TAILLE (1..ub-organizational-unit-name)),
     utf8String         UTF8String        (TAILLE (1..ub-organizational-unit-name)),
     bmpString          BMPString         (TAILLE (1..ub-organizational-unit-name)) }

-- Attributs de dénomination de type X520Title
id-at-title            AttributeType ::= { id-at 12 }

-- Attributs de dénomination de type X520Title :
--   X520Title ::= DirectoryName (TAILLE (1..ub-title))
--
-- Étendu pour éviter un type paramétré :
X520Title ::= CHOIX {
     teletexString      TeletexString     (TAILLE (1..ub-title)),
     printableString    PrintableString   (TAILLE (1..ub-title)),
     universalString    UniversalString (TAILLE (1..ub-title)),
     utf8String         UTF8String        (TAILLE (1..ub-title)),
     bmpString          BMPString         (TAILLE (1..ub-title)) }

-- Attributs de dénomination de type X520dnQualifier
id-at-dnQualifier      AttributeType ::= { id-at 46 }

X520dnQualifier ::= PrintableString

-- Attributs de dénomination de type X520countryName (digraphe provenant de IS 3166) :
id-at-countryName      AttributeType ::= { id-at 6 }

X520countryName ::=     PrintableString (TAILLE (2))

-- Attributs de dénomination de type X520SerialNumber :
id-at-serialNumber     AttributeType ::= { id-at 5 }

X520SerialNumber ::=    PrintableString (TAILLE (1..ub-serial-number))

-- Attributs de dénomination de type X520Pseudonym :
id-at-pseudonym        AttributeType ::= { id-at 65 }

-- Attributs de dénomination de type X520Pseudonym :
--   X520Pseudonym ::= DirectoryName (TAILLE (1..ub-pseudonym))
--
-- Étendu pour éviter un type paramétré :
X520Pseudonym ::= CHOIX {
  teletexString      TeletexString     (TAILLE (1..ub-pseudonym)),
  printableString    PrintableString   (TAILLE (1..ub-pseudonym)),
  universalString    UniversalString (TAILLE (1..ub-pseudonym)),
  utf8String         UTF8String        (TAILLE (1..ub-pseudonym)),
  bmpString          BMPString         (TAILLE (1..ub-pseudonym)) }

-- Attributs de dénomination de type DomainComponent (provenant de la RFC 4519) :
id-domainComponent   AttributeType ::= { 0 9 2342 19200300 100 1 25 }

DomainComponent ::=  IA5String
```

```
-- Attributs traditionnels

IDENTIFIANT D'OBJET pkcs-9 ::= { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) 9 }

id-emailAddress      AttributeType ::= { pkcs-9 1 }

EmailAddress ::=     IA5String (TAILLE (1..ub-emailaddress-length))

-- Types de données de désignation --

Name ::= CHOIX {                                        -- une seule possibilité actuellement --
     rdnSequence  RDNSequence }

RDNSequence ::= SEQUENCE DE RelativeDistinguishedName

DistinguishedName ::=   RDNSequence

RelativeDistinguishedName ::= SET TAILLE (1..MAX) DE AttributeTypeAndValue

-- Type DirectoryString --

DirectoryString ::= CHOIX {
     teletexString        TeletexString    (TAILLE (1..MAX)),
     printableString      PrintableString (TAILLE (1..MAX)),
     universalString     UniversalString (TAILLE (1..MAX)),
     utf8String           UTF8String       (TAILLE (1..MAX)),
     bmpString            BMPString        (TAILLE (1..MAX)) }

-- Les structures spécifiques de certificat et de CRL commencent ici.

Certificate  ::=  SEQUENCE  {
    tbsCertificate           TBSCertificate,
    signatureAlgorithm   AlgorithmIdentifier,
    signature                CHAINE BINAIRE  }

TBSCertificate  ::=  SEQUENCE  {
    version            [0]  Version DEFAULT v1,
    serialNumber         CertificateSerialNumber,
    signature            AlgorithmIdentifier,
    issuer               Name,
    validity             Validity,
    subject              Name,
    subjectPublicKeyInfo SubjectPublicKeyInfo,
    issuerUniqueID   [1]  IMPLICIT UniqueIdentifier FACULTATIF,   -- Si présent, version DOIT être v2 ou v3
    subjectUniqueID [2]  IMPLICIT UniqueIdentifier FACULTATIF,    -- Si présent, version DOIT être v2 ou v3
    extensions           [3]  Extensions FACULTATIF                -- Si présent, version DOIT être v3 --  }

Version ::= ENTIER  {  v1(0), v2(1), v3(2)  }

CertificateSerialNumber ::= ENTIER

Validity ::= SEQUENCE {
    notBefore    Time,
    notAfter      Time  }

Time ::= CHOIX {
    utcTime         UTCTime,
    generalTime    GeneralizedTime }

UniqueIdentifier  ::=  CHAINE BINAIRE

SubjectPublicKeyInfo  ::=  SEQUENCE  {
    algorithm              AlgorithmIdentifier,
```

```
    subjectPublicKey     CHAINE BINAIRE  }

Extensions ::= TAILLE DE SEQUENCE (1..MAX) DE Extension

Extension ::= SEQUENCE  {
    extnID        IDENTIFIANT D'OBJET,
    critical      BOOLEAN DEFAULT FAUX,
    extnValue   CHAINE D'OCTETS
            -- contient le codage en DER d'une valeur ASN.1 correspondant au type d'extension identifié par extnID
    }

-- Structures de CRL

CertificateList  ::=  SEQUENCE  {
    tbsCertList             TBSCertList,
    signatureAlgorithm   AlgorithmIdentifier,
    signature                CHAINE BINAIRE  }

TBSCertList  ::=  SEQUENCE  {
    version              Version FACULTATIF,              -- si présent, DOIT être v2
    signature            AlgorithmIdentifier,
    issuer               Name,
    thisUpdate           Time,
    nextUpdate           Time FACULTATIF,
    revokedCertificates  SEQUENCE DE SEQUENCE  {
        userCertificate    CertificateSerialNumber,
        revocationDate     Time,
        crlEntryExtensions  Extensions FACULTATIF           -- si présent, version DOIT être v2
                    }  FACULTATIF,
    crlExtensions          [0] Extensions FACULTATIF }      -- si présent, version DOIT être v2

-- Version, Time, CertificateSerialNumber, et Extensions ont été définis précédemment dans la structure de certificat

AlgorithmIdentifier  ::=  SEQUENCE  {
    algorithm             IDENTIFIANT D'OBJET,
    parameters            ANY DEFINED BY algorithm FACULTATIF  }
   -- contient une valeur du type enregistré à utiliser avec la valeur d'identifiant d'objet d'algorithme

-- La syntaxe d'adresse X.400 commence ici

ORAddress ::= SEQUENCE {
  built-in-standard-attributes            BuiltInStandardAttributes,
  built-in-domain-defined-attributes  BuiltInDomainDefinedAttributes FACULTATIF,
-- voir aussi teletex-domain-defined-attributes
  extension-attributes                    ExtensionAttributes FACULTATIF }

-- Attributs standard incorporés

BuiltInStandardAttributes ::= SEQUENCE {
  country-name                      CountryName FACULTATIF,
  administration-domain-name    AdministrationDomainName FACULTATIF,
  network-address               [0] IMPLICIT NetworkAddress FACULTATIF,
-- voir aussi extended-network-address
  terminal-identifier           [1] IMPLICIT TerminalIdentifier FACULTATIF,
  private-domain-name           [2] PrivateDomainName FACULTATIF,
  organization-name             [3] IMPLICIT OrganizationName FACULTATIF,
-- voir aussi teletex-organization-name
  numeric-user-identifier       [4] IMPLICIT NumericUserIdentifier FACULTATIF,
  personal-name                 [5] IMPLICIT PersonalName FACULTATIF,
-- voir aussi teletex-personal-name
  organizational-unit-names     [6] IMPLICIT OrganizationalUnitNames FACULTATIF }
-- voir aussi teletex-organizational-unit-names
```

```
CountryName ::= [APPLICATION 1] CHOIX {
  x121-dcc-code           NumericString (TAILLE (ub-country-name-numeric-length)),
  iso-3166-alpha2-code  PrintableString (TAILLE (ub-country-name-alpha-length)) }

AdministrationDomainName ::= [APPLICATION 2] CHOIX {
  numeric   NumericString   (TAILLE (0..ub-domain-name-length)),
  printable  PrintableString   (TAILLE (0..ub-domain-name-length)) }

NetworkAddress ::= X121Address                                  -- voir aussi extended-network-address

X121Address ::= NumericString (TAILLE (1..ub-x121-address-length))

TerminalIdentifier ::= PrintableString (TAILLE (1..ub-terminal-id-length))

PrivateDomainName ::= CHOIX {
  numeric   NumericString   (TAILLE (1..ub-domain-name-length)),
  printable  PrintableString   (TAILLE (1..ub-domain-name-length)) }

OrganizationName ::= PrintableString (TAILLE (1..ub-organization-name-length))
-- voir aussi teletex-organization-name

NumericUserIdentifier ::= NumericString (TAILLE (1..ub-numeric-user-id-length))

PersonalName ::= SET {
  surname               [0] IMPLICIT PrintableString (TAILLE (1..ub-surname-length)),
  given-name            [1] IMPLICIT PrintableString (TAILLE (1..ub-given-name-length)) FACULTATIF,
  initials              [2] IMPLICIT PrintableString (TAILLE (1..ub-initials-length)) FACULTATIF,
  generation-qualifier [3] IMPLICIT PrintableString (TAILLE (1..ub-generation-qualifier-length)) FACULTATIF }
- voir aussi teletex-personal-name

OrganizationalUnitNames ::= TAILLE DE SEQUENCE (1..ub-organizational-units) DE OrganizationalUnitName
-- voir aussi teletex-organizational-unit-names

OrganizationalUnitName ::= PrintableString (TAILLE (1..ub-organizational-unit-name-length))

-- Attributs incorporés définis par le domaine

BuiltInDomainDefinedAttributes ::= TAILLE DE SEQUENCE (1..ub-domain-defined-attributes) DE
            BuiltInDomainDefinedAttribute

BuiltInDomainDefinedAttribute ::= SEQUENCE {
  type PrintableString   (TAILLE (1..ub-domain-defined-attribute-type-length)),
  value PrintableString (TAILLE (1..ub-domain-defined-attribute-value-length)) }

-- Attributs d'extension

ExtensionAttributes ::= SET TAILLE (1..ub-extension-attributes) DE ExtensionAttribute

ExtensionAttribute ::=  SEQUENCE {
  extension-attribute-type   [0] IMPLICIT ENTIER (0..ub-extension-attributes),
  extension-attribute-value [1] ANY DEFINED BY extension-attribute-type }

-- Types d'extension et valeurs d'attributs

common-name ENTIER ::= 1

CommonName ::= PrintableString (TAILLE (1..ub-common-name-length))

teletex-common-name ENTIER ::= 2

TeletexCommonName ::= TeletexString (TAILLE (1..ub-common-name-length))

teletex-organization-name ENTIER ::= 3
```

```
TeletexOrganizationName ::= TeletexString (TAILLE (1..ub-organization-name-length))

teletex-personal-name ENTIER ::= 4

TeletexPersonalName ::= SET {
  surname               [0] IMPLICIT TeletexString (TAILLE (1..ub-surname-length)),
  given-name            [1] IMPLICIT TeletexString (TAILLE (1..ub-given-name-length)) FACULTATIF,
  initials              [2] IMPLICIT TeletexString (TAILLE (1..ub-initials-length)) FACULTATIF,
  generation-qualifier [3] IMPLICIT TeletexString (TAILLE (1..ub-generation-qualifier-length)) FACULTATIF }

teletex-organizational-unit-names ENTIER ::= 5

TeletexOrganizationalUnitNames ::= TAILLE DE SEQUENCE
    (1..ub-organizational-units) DE TeletexOrganizationalUnitName

TeletexOrganizationalUnitName ::= TeletexString (TAILLE (1..ub-organizational-unit-name-length))

pds-name ENTIER ::= 7

PDSName ::= PrintableString (TAILLE (1..ub-pds-name-length))

physical-delivery-country-name ENTIER ::= 8

PhysicalDeliveryCountryName ::= CHOIX {
  x121-dcc-code NumericString          (TAILLE (ub-country-name-numeric-length)),
  iso-3166-alpha2-code PrintableString (TAILLE (ub-country-name-alpha-length)) }

postal-code ENTIER ::= 9

PostalCode ::= CHOIX {
  numeric-code   NumericString (TAILLE (1..ub-postal-code-length)),
  printable-code  PrintableString (TAILLE (1..ub-postal-code-length)) }

physical-delivery-office-name ENTIER ::= 10

PhysicalDeliveryOfficeName ::= PDSParameter

physical-delivery-office-number ENTIER ::= 11

PhysicalDeliveryOfficeNumber ::= PDSParameter

extension-OR-address-components ENTIER ::= 12

ExtensionORAddressComponents ::= PDSParameter

physical-delivery-personal-name ENTIER ::= 13

PhysicalDeliveryPersonalName ::= PDSParameter

physical-delivery-organization-name ENTIER ::= 14

PhysicalDeliveryOrganizationName ::= PDSParameter

extension-physical-delivery-address-components ENTIER ::= 15

ExtensionPhysicalDeliveryAddressComponents ::= PDSParameter

unformatted-postal-address ENTIER ::= 16

UnformattedPostalAddress ::= SET {
  printable-address TAILLE DE SEQUENCE (1..ub-pds-physical-address-lines)
      DE PrintableString (TAILLE (1..ub-pds-parameter-length)) FACULTATIF,
```

```
  teletex-string TeletexString (TAILLE (1..ub-unformatted-address-length)) FACULTATIF }

street-address ENTIER ::= 17

StreetAddress ::= PDSParameter

post-office-box-address ENTIER ::= 18

PostOfficeBoxAddress ::= PDSParameter

poste-restante-address ENTIER ::= 19

PosteRestanteAddress ::= PDSParameter

unique-postal-name ENTIER ::= 20

UniquePostalName ::= PDSParameter

local-postal-attributes ENTIER ::= 21

LocalPostalAttributes ::= PDSParameter

PDSParameter ::= SET {
  printable-string PrintableString (TAILLE(1..ub-pds-parameter-length)) FACULTATIF,
  teletex-string TeletexString      (TAILLE(1..ub-pds-parameter-length)) FACULTATIF }

extended-network-address ENTIER ::= 22

ExtendedNetworkAddress ::= CHOIX {
  e163-4-address SEQUENCE {
    number       [0] IMPLICIT NumericString (TAILLE (1..ub-e163-4-number-length)),
    sub-address  [1] IMPLICIT NumericString (TAILLE (1..ub-e163-4-sub-address-length)) FACULTATIF },
  psap-address   [0] IMPLICIT PresentationAddress }

PresentationAddress ::= SEQUENCE {
   pSelector      [0] EXPLICIT CHAINE D'OCTETS FACULTATIF,
   sSelector      [1] EXPLICIT CHAINE D'OCTETS FACULTATIF,
   tSelector      [2] EXPLICIT CHAINE D'OCTETS FACULTATIF,
   nAddresses     [3] EXPLICIT SET TAILLE (1..MAX) DE CHAINE D'OCTETS }

terminal-type  ENTIER ::= 23

TerminalType ::= ENTIER {
  telex          (3),
  teletex        (4),
  g3-facsimile (5),
  g4-facsimile (6),
  ia5-terminal (7),
  videotex       (8) } (0..ub-integer-options)

-- Attributs d'extension définis par le domaine

teletex-domain-defined-attributes ENTIER ::= 6

TeletexDomainDefinedAttributes ::= TAILLE DE SEQUENCE
  (1..ub-domain-defined-attributes) DE TeletexDomainDefinedAttribute

TeletexDomainDefinedAttribute ::= SEQUENCE {
      type TeletexString (TAILLE (1..ub-domain-defined-attribute-type-length)),
      value TeletexString (TAILLE (1..ub-domain-defined-attribute-value-length)) }
-- Les spécifications de limites supérieures DOIVENT être regardées comme obligatoires d'après l'Annexe B de la
Recommandation UIT-T X.411, Définition des limites supérieurs de paramètre MTS
```

```
-- Limites supérieures
ub-name ENTIER ::= 32768
ub-common-name ENTIER ::= 64
ub-locality-name ENTIER ::= 128
ub-state-name ENTIER ::= 128
ub-organization-name ENTIER ::= 64
ub-organizational-unit-name ENTIER ::= 64
ub-title ENTIER ::= 64
ub-serial-number ENTIER ::= 64
ub-match ENTIER ::= 128
ub-emailaddress-length ENTIER ::= 255
ub-common-name-length ENTIER ::= 64
ub-country-name-alpha-length ENTIER ::= 2
ub-country-name-numeric-length ENTIER ::= 3
ub-domain-defined-attributes ENTIER ::= 4
ub-domain-defined-attribute-type-length ENTIER ::= 8
ub-domain-defined-attribute-value-length ENTIER ::= 128
ub-domain-name-length ENTIER ::= 16
ub-extension-attributes ENTIER ::= 256
ub-e163-4-number-length ENTIER ::= 15
ub-e163-4-sub-address-length ENTIER ::= 40
ub-generation-qualifier-length ENTIER ::= 3
ub-given-name-length ENTIER ::= 16
ub-initials-length ENTIER ::= 5
ub-integer-options ENTIER ::= 256
ub-numeric-user-id-length ENTIER ::= 32
ub-organization-name-length ENTIER ::= 64
ub-organizational-unit-name-length ENTIER ::= 32
ub-organizational-units ENTIER ::= 4
ub-pds-name-length ENTIER ::= 16
ub-pds-parameter-length ENTIER ::= 30
ub-pds-physical-address-lines ENTIER ::= 6
ub-postal-code-length ENTIER ::= 16
ub-pseudonym ENTIER ::= 128
ub-surname-length ENTIER ::= 40
ub-terminal-id-length ENTIER ::= 24
ub-unformatted-address-length ENTIER ::= 180
ub-x121-address-length ENTIER ::= 16

-- Note : les limites supérieures sur les types de chaînes, comme TeletexString, sont mesurées en caractères. Sauf
PrintableString ou IA5String, un nombre d'octets significativement supérieur va être nécessaire pour contenir une telle
valeur. Au minimum, 16 octets, ou deux fois la limite supérieure spécifiée, selon ce qui est le plus grand, devrait être
permis pour TeletexString. Pour UTF8String ou UniversalString au moins quatre fois la limite supérieure devrait être
permis.

FIN
```

A.2 Module étiqueté implicitement, syntaxe de 1988

```
PKIX1Implicit88 { iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-
pkix1-implicit(19) }

ÉTIQUETTES IMPLICITES DE DÉFINITION ::=

DÉBUT

-- EXPORTE TOUT --

IMPORTE
  id-pe, id-kp, id-qt-unotice, id-qt-cps,
-- supprimer les lignes suivantes si des types "new" sont pris en charge --
     BMPString, UTF8String,                                    --fin des types "new" --
```

```
    ORAddress, Name, RelativeDistinguishedName,
    CertificateSerialNumber, Attribute, DirectoryString
DE PKIX1Explicit88 { iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
id-pkix1-explicit(18) };

-- Arc ISO pour extensions de certificat et CRL standard

IDENTIFIANT D'OBJET id-ce ::= {joint-iso-ccitt(2) ds(5) 29}

-- OID et syntaxe d'identifiant de clé d'autorité

IDENTIFIANT D'OBJET id-ce-authorityKeyIdentifier ::=  { id-ce 35 }

AuthorityKeyIdentifier ::= SEQUENCE {
   keyIdentifier             [0] KeyIdentifier          FACULTATIF,
   authorityCertIssuer       [1] GeneralNames           FACULTATIF,
   authorityCertSerialNumber [2] CertificateSerialNumber  FACULTATIF }
-- authorityCertIssuer et authorityCertSerialNumber DOIVENT toux deux être présents ou absents

KeyIdentifier ::= CHAINE D'OCTETS

-- OID et syntaxe d'identifiant de clé de sujet

IDENTIFIANT D'OBJET id-ce-subjectKeyIdentifier ::=  { id-ce 14 }

SubjectKeyIdentifier ::= KeyIdentifier

-- OID et syntaxe d'extension d'usage de clé

IDENTIFIANT D'OBJET id-ce-keyUsage ::=  { id-ce 15 }

KeyUsage ::= CHAINE BINAIRE {
    digitalSignature      (0),
    nonRepudiation        (1),          -- les récentes éditions de X.509 ont changé le nom de ce bit en contentCommitment
    keyEncipherment       (2),
    dataEncipherment      (3),
    keyAgreement          (4),
    keyCertSign           (5),
    cRLSign               (6),
    encipherOnly          (7),
    decipherOnly          (8) }

-- OID et syntaxe d'extension de période d'usage de clé privée

IDENTIFIANT D'OBJET id-ce-privateKeyUsagePeriod ::=  { id-ce 16 }

PrivateKeyUsagePeriod ::= SEQUENCE {
    notBefore     [0]     GeneralizedTime FACULTATIF,
    notAfter      [1]     GeneralizedTime FACULTATIF }
-- un de notBefore ou de notAfter DOIT être présent

-- OID et syntaxe d'extension de politiques de certificat

IDENTIFIANT D'OBJET id-ce-certificatePolicies ::=  { id-ce 32 }

IDENTIFIANT D'OBJET anyPolicy ::= { id-ce-certificatePolicies 0 }

CertificatePolicies ::= TAILLE DE SEQUENCE (1..MAX) DE PolicyInformation

PolicyInformation ::= SEQUENCE {
    policyIdentifier    CertPolicyId,
    policyQualifiers   TAILLE DE SEQUENCE (1..MAX) DE PolicyQualifierInfo FACULTATIF }
```

```
CertPolicyId ::= IDENTIFIANT D'OBJET

PolicyQualifierInfo ::= SEQUENCE {
    policyQualifierId  PolicyQualifierId,
    qualifier              ANY DEFINED BY policyQualifierId }

--Les mises en œuvre qui reconnaissent des qualificatifs de politique supplémentaires DOIVENT augmenter la définition
suivante pour PolicyQualifierId

PolicyQualifierId ::= IDENTIFIANT D'OBJET ( id-qt-cps | id-qt-unotice )

-- Qualificatif de pointeur CPS

CPSuri ::= IA5String

-- Qualificatif de notice d'utilisateur

UserNotice ::= SEQUENCE {
    noticeRef       NoticeReference FACULTATIF,
    explicitText    DisplayText       FACULTATIF }

NoticeReference ::= SEQUENCE {
    organization       DisplayText,
    noticeNumbers    SEQUENCE DE ENTIER }

DisplayText ::= CHOIX {
    ia5String         IA5String       (TAILLE (1..200)),
    visibleString    VisibleString   (TAILLE (1..200)),
    bmpString        BMPString      (TAILLE (1..200)),
    utf8String        UTF8String    (TAILLE (1..200)) }

-- OID et syntaxe d'extension de transposition de politique

IDENTIFIANT D'OBJET id-ce-policyMappings ::=  { id-ce 33 }

PolicyMappings ::= TAILLE DE SEQUENCE (1..MAX) DE SEQUENCE {
    issuerDomainPolicy      CertPolicyId,
    subjectDomainPolicy     CertPolicyId }

-- OID et syntaxe d'extension de nom de remplacement de sujet

IDENTIFIANT D'OBJET id-ce-subjectAltName ::=  { id-ce 17 }

SubjectAltName ::= GeneralNames

GeneralNames ::= TAILLE DE SEQUENCE (1..MAX) DE GeneralName

GeneralName ::= CHOIX {
    otherName                       [0]  AnotherName,
    rfc822Name                      [1]  IA5String,
    dNSName                         [2]  IA5String,
    x400Address                     [3]  ORAddress,
    directoryName                   [4]  Name,
    ediPartyName                    [5]  EDIPartyName,
    uniformResourceIdentifier  [6]  IA5String,
    iPAddress                       [7]  CHAINE D'OCTETS,
    registeredID                    [8]  IDENTIFIANT D'OBJET }

-- AnotherName remplace OTHER-NAME ::= TYPE-IDENTIFIER, car TYPE-IDENTIFIER n'est pas accepté dans la
syntaxe ASN.1 de 1988.

AnotherName ::= SEQUENCE {
    type-id    IDENTIFIANT D'OBJET,
```

```
    value     [0] EXPLICIT ANY DEFINED BY type-id }

EDIPartyName ::= SEQUENCE {
    nameAssigner        [0]  DirectoryString FACULTATIF,
    partyName           [1]  DirectoryString }

--OID et syntaxe d'extension de nom de remplacement du producteur

IDENTIFIANT D'OBJET id-ce-issuerAltName ::=  { id-ce 18 }

IssuerAltName ::= GeneralNames

IDENTIFIANT D'OBJET id-ce-subjectDirectoryAttributes ::=  { id-ce 9 }

SubjectDirectoryAttributes ::= TAILLE DE SEQUENCE (1..MAX) DE Attribute

-- OID et syntaxe d'extension Contraintes de base

IDENTIFIANT D'OBJET id-ce-basicConstraints ::=  { id-ce 19 }

BasicConstraints ::= SEQUENCE {
    cA                      BOOLEAN DEFAULT FAUX,
    pathLenConstraint       ENTIER (0..MAX) FACULTATIF }

-- OID et syntaxe d'extension Contraintes de nom

IDENTIFIANT D'OBJET id-ce-nameConstraints ::=  { id-ce 30 }

NameConstraints ::= SEQUENCE {
    permittedSubtrees       [0]    GeneralSubtrees FACULTATIF,
    excludedSubtrees        [1]    GeneralSubtrees FACULTATIF }

GeneralSubtrees ::= TAILLE DE SEQUENCE (1..MAX) DE GeneralSubtree

GeneralSubtree ::= SEQUENCE {
    base                GeneralName,
    minimum         [0]    BaseDistance DEFAULT 0,
    maximum         [1]    BaseDistance FACULTATIF }

BaseDistance ::= ENTIER (0..MAX)

-- OID et syntaxe d'extension Contraintes de politique

IDENTIFIANT D'OBJET id-ce-policyConstraints ::=  { id-ce 36 }

PolicyConstraints ::= SEQUENCE {
    requireExplicitPolicy    [0]    SkipCerts FACULTATIF,
    inhibitPolicyMapping     [1]    SkipCerts FACULTATIF }

SkipCerts ::= ENTIER (0..MAX)

-- OID et syntaxe d'extension Points de distribution de CRL

IDENTIFIANT D'OBJET id-ce-cRLDistributionPoints ::= {id-ce 31}

CRLDistributionPoints ::= TAILLE DE SEQUENCE (1..MAX) DE DistributionPoint

DistributionPoint ::= SEQUENCE {
    distributionPoint    [0]    DistributionPointName FACULTATIF,
    reasons              [1]    ReasonFlags FACULTATIF,
    cRLIssuer            [2]    GeneralNames FACULTATIF }

DistributionPointName ::= CHOIX {
```

```
    fullName                    [0]    GeneralNames,
    nameRelativeToCRLIssuer [1]    RelativeDistinguishedName }

ReasonFlags ::= CHAINE BINAIRE {
    unused                  (0),
    keyCompromise           (1),
    cACompromise            (2),
    affiliationChanged      (3),
    superseded              (4),
    cessationOfOperation    (5),
    certificateHold         (6),
    privilegeWithdrawn      (7),
    aACompromise            (8) }

-- OID et syntaxe étendue d'extension d'usage de clé

IDENTIFIANT D'OBJET id-ce-extKeyUsage ::= {id-ce 37}

ExtKeyUsageSyntax ::= TAILLE DE SEQUENCE (1..MAX) DE KeyPurposeId

KeyPurposeId ::= IDENTIFIANT D'OBJET

-- Utilisations permises de clé non spécifiée

IDENTIFIANT D'OBJET anyExtendedKeyUsage ::= { id-ce-extKeyUsage 0 }

-- OID étendus d'objet de clé

IDENTIFIANT D'OBJET id-kp-serverAuth ::= { id-kp 1 }
IDENTIFIANT D'OBJET id-kp-clientAuth ::= { id-kp 2 }
IDENTIFIANT D'OBJET id-kp-codeSigning ::= { id-kp 3 }
IDENTIFIANT D'OBJET id-kp-emailProtection IDENTIFIANT D'OBJET ::= { id-kp 4 }
IDENTIFIANT D'OBJET id-kp-timeStamping IDENTIFIANT D'OBJET ::= { id-kp 8 }
IDENTIFIANT D'OBJET id-kp-OCSPSigning IDENTIFIANT D'OBJET ::= { id-kp 9 }

-- OID et syntaxe de Inhiber toute politique

IDENTIFIANT D'OBJET id-ce-inhibitAnyPolicy ::=  { id-ce 54 }

InhibitAnyPolicy ::= SkipCerts

-- OID et syntaxe de fraîcheur de CRL (delta)

IDENTIFIANT D'OBJET id-ce-freshestCRL ::=  { id-ce 46 }

FreshestCRL ::= CRLDistributionPoints

-- Accès aux informations d'autorité

IDENTIFIANT D'OBJET id-pe-authorityInfoAccess ::= { id-pe 1 }

AuthorityInfoAccessSyntax ::= TAILLE DE SEQUENCE (1..MAX) DE AccessDescription

AccessDescription  ::=  SEQUENCE {
      accessMethod        IDENTIFIANT D'OBJET,
      accessLocation      GeneralName  }

-- Accès aux informations de sujet

IDENTIFIANT D'OBJET id-pe-subjectInfoAccess ::= { id-pe 11 }

SubjectInfoAccessSyntax ::= TAILLE DE SEQUENCE (1..MAX) DE AccessDescription
```

```
-- OID et syntaxe d'extension Numéro de CRL

IDENTIFIANT D'OBJET id-ce-cRLNumber ::= { id-ce 20 }

CRLNumber ::= ENTIER (0..MAX)

-- OID et syntaxe d'extension Point de distribution producteur

IDENTIFIANT D'OBJET id-ce-issuingDistributionPoint ::= { id-ce 28 }

IssuingDistributionPoint ::= SEQUENCE {
    distributionPoint          [0] DistributionPointName FACULTATIF,
    onlyContainsUserCerts      [1] BOOLEAN DEFAULT FAUX,
    onlyContainsCACerts        [2] BOOLEAN DEFAULT FAUX,
    onlySomeReasons            [3] ReasonFlags FACULTATIF,
    indirectCRL                [4] BOOLEAN DEFAULT FAUX,
    onlyContainsAttributeCerts [5] BOOLEAN DEFAULT FAUX }
-- Au plus un de onlyContainsUserCerts, onlyContainsCACerts, et onlyContainsAttributeCerts peut être réglé à VRAI.

IDENTIFIANT D'OBJET id-ce-deltaCRLIndicator ::= { id-ce 27 }

BaseCRLNumber ::= CRLNumber

-- OID et syntaxe d'extension Code de cause

IDENTIFIANT D'OBJET id-ce-cRLReasons ::= { id-ce 21 }

CRLReason ::= ENUMERATED {
    unspecified             (0),
    keyCompromise           (1),
    cACompromise            (2),
    affiliationChanged      (3),
    superseded              (4),
    cessationOfOperation    (5),
    certificateHold         (6),
    removeFromCRL           (8),
    privilegeWithdrawn      (9),
    aACompromise           (10) }

-- OID et syntaxe d'extension de producteur de certificat d'entrée de CRL

IDENTIFIANT D'OBJET id-ce-certificateIssuer ::= { id-ce 29 }

CertificateIssuer ::= GeneralNames

-- OID et syntaxe d'extension Garde d'instruction

IDENTIFIANT D'OBJET id-ce-holdInstructionCode ::= { id-ce 23 }

HoldInstructionCode ::= IDENTIFIANT D'OBJET

--Arc ANSI x9 de holdinstruction

IDENTIFIANT D'OBJET holdInstruction ::= {joint-iso-itu-t(2) member-body(2) us(840) x9cm(10040) 2}

-- holdinstructions ANSI X9

IDENTIFIANT D'OBJET id-holdinstruction-none ::= {holdInstruction 1}        -- déconseillé

IDENTIFIANT D'OBJET id-holdinstruction-callissuer ::= {holdInstruction 2}

IDENTIFIANT D'OBJET id-holdinstruction-reject ::= {holdInstruction 3}
```

-- OID et syntaxe d’extension d'entrée de date d’invalidité de CRL

IDENTIFIANT D’OBJET id-ce-invaliditeDate ::= { id-ce 24 }

InvaliditeDate ::= GeneralizedTime

FIN

Appendice B. Notes de l'ASN.1

Les CA DOIVENT forcer le serialNumber à être un entier non négatif, c'est-à-dire que le bit de signe dans le codage en DER de la valeur ENTIER DOIT être zéro. Cela peut être fait en ajoutant un octet `00'H en tête (à gauche) si nécessaire. Cela supprime une ambiguïté potentielle de la transposition entre une chaîne d'octets et une valeur d'entier.

Comme noté au paragraphe 4.1.2.2, on peut s'attendre à ce que les numéros de séries contiennent de longs entiers. Les utilisateurs de certificat DOIVENT être capables de traiter les valeurs de serialNumber jusqu'à 20 octets de long. Les CA conformes NE DOIVENT PAS utiliser de valeurs de serialNumber de plus de 20 octets.

Comme noté au paragraphe 5.2.3, on peut s'attendre à ce que les numéros de CRL contiennent de longs entiers. Les valideurs de CRL DOIVENT être capables de traiter les valeurs de cRLNumber jusqu'à 20 octets de long. Les producteurs de CRL conformes NE DOIVENT PAS utiliser de valeurs de cRLNumber de plus de 20 octets.

La construction "TAILLE DE SEQUENCE (1..MAX) DE" apparaît dans plusieurs constructions ASN.1. Une séquence valide d'ASN.1 va avoir zéro, une ou plusieurs entrées. La construction TAILLE (1..MAX) contraint la séquence à avoir au moins une entrée. MAX indique que la limite supérieure n'est pas spécifiée. Les mises en œuvre sont libres de choisir une limite supérieure qui convient à leur environnement.

Le type de chaîne de caractères PrintableString prend en charge un jeu de caractères Latin très basique : les lettres minuscules 'a' à 'z', les lettres majuscules 'A' à 'Z', les chiffres '0' à '9', onze caractère spéciaux ' = ( ) + , - . / : ? et espace.

Les mises en œuvre devraient noter que les caractères signe à ('@') et souligné ('_') ne sont pas pris en charge par le type ASN.1 PrintableString. Ces caractères apparaissent souvent dans les adresses Internet. Ces adresses DOIVENT être codées en utilisant un type ASN.1 qui les prend en charge. Elles sont généralement codées comme IA5String dans l'attribut emailAddress au sein d'un nom distinctif ou le champ rfc822Name de GeneralName. Les mises en œuvre conformes NE DOIVENT PAS coder les chaînes qui comportent le signe @ ou le caractère souligné comme PrintableString.

Le type de chaîne de caractères TeletexString est a sur-ensemble de PrintableString. TeletexString prend en charge un jeu de caractères Latin très standard (semblable à l’ASCII) : les caractères latins avec des accents non espacés et les caractères japonais.

Les listes de bits nommés sont des CHAINES BINAIRES où les valeurs ont reçu des noms. La présente spécification utilise des listes de bits nommés dans les définitions pour les extensions de certificat d’usage de clé, de points de distribution de CRL, et de fraîcheur de CRL, ainsi que les extensions de fraîcheur de CRL et de point de distribution de CRL. Quand on code en DER une liste de bits nommés, les zéros en queue DOIVENT être omis. C’est-à-dire, la valeur codée se termine avec le dernier bit nommé qui est réglé à un.

Le type de chaîne de caractères UniversalString prend en charge tous les caractères permis par [ISO10646]. ISO 10646 est le jeu de caractères universel codé sur plusieurs octets (UCS).

Le type de chaîne de caractères UTF8String a été introduit dans la version 1997 de l’ASN.1, et UTF8String a été ajouté à la liste des choix pour DirectoryString dans la version 2001 de [X.520]. UTF8String est un type universel et le numéro d’étiquette 12 lui a été alloué. Le contenu de UTF8String a été défini par la RFC 2044 et mis à jour par la RFC 2279, qui a été mise à jour par la [RFC3629].

En anticipation de ces changements, et en conformité avec les bonnes pratiques de l’IETF codifiées dans la [RFC2277], Politique de l’IETF sur les jeux de caractères et les langages, ce document inclut UTF8String comme choix dans DirectoryString et dans le qualificatif de politique de certificat userNotice.

Pour de nombreux types d’attribut définis dans [X.520], AttributeValue utilise le type DirectoryString. Des attributs spécifiés dans l’Appendice A, les attributs name, surname, givenName, initials, generationQualifier, commonName, localityName, stateOrProvinceName, organizationName, organizationalUnitName, title, et pseudonym utilisent tous le type

DirectoryString. X.520 utilise une définition de type paramétré [X.683] de DirectoryString pour spécifier la syntaxe pour chacun de ces attributs. Le paramètre est utilisé pour indiquer la longueur maximum de chaîne permise pour l'attribut. Dans l'Appendice A, afin d'éviter l'utilisation des définitions de type paramétré, le type DirectoryString est écrit dans sa forme étendue pour la définition de chacun de ces types d'attribut. Donc, l'ASN.1 de l'Appendice A décrit la syntaxe de chacun de ces attributs comme étant un CHOIX de TeletexString, PrintableString, UniversalString, UTF8String, et BMPString, avec les contraintes appropriées sur la longueur de chaîne appliquée à chacun des types dans le CHOIX, plutôt que d'utiliser le type ASN.1 DirectoryString pour décrire la syntaxe.

Les mises en œuvre devraient noter que le codage en DER des valeurs ENSEMBLE DE exige le codage dans l'ordre des valeurs. En particulier, ce problème se pose à l'égard des noms distinctifs.

Les mises en œuvre devraient noter que le codage en DER des composants SET ou SEQUENCE dont la valeur est DEFAULT omettent le composant du certificat ou CRL codé. Par exemple, une extension BasicConstraints dont la valeur cA est FAUX omettrait le booléen cA du certificat codé.

Les identifiants d'objet (OID, *Object Identifier*) sont utilisés dans la présente spécification pour identifier les politiques de certificat, clé publique et algorithmes de signature, extensions de certificat, etc. Il n'y a pas de taille maximum pour les OID. La présente spécification rend obligatoire la prise en charge des OID qui ont des éléments avec des valeurs de moins de 2^28, c'est-à-dire, ils DOIVENT être entre 0 et 268 435 455, inclus. Cela permet que chaque élément de l'arc soit représenté dans un seul mot de 32 bits. Les mises en œuvre DOIVENT aussi prendre en charge les OID dont la longueur de la représentation de chaîne en décimal séparé par des points (voir le paragraphe 1.4 de la [RFC4512]) peut être jusqu'à 100 octets (inclus). Les mises en œuvre DOIVENT être capables de traiter des OID avec jusqu'à 20 éléments (inclus). Les CA NE DEVRAIENT PAS produire de certificats qui contiennent des OID qui excèdent cette exigence. De même, les producteurs de CRL NE DEVRAIENT PAS produire des CRL qui contiennent des OID qui excèdent cette exigence.

Les règles spécifiques du contenu pour les valeur de codage de GeneralName dans l'extension nameConstraints diffèrent des règles qui s'appliquent dans les autres extensions. Dans toutes les autres extensions de certificat, CRL, et entrées de CRL spécifiées dans ce document, les règles de codage se conforment aux règles pour le type sous-jacent. Par exemple, les valeurs dans le champ uniformResourceIdentifier doivent contenir un URI valide, comme spécifié dans la [RFC3986]. Les règles spécifiques du contenu pour les valeurs de codage dans l'extension nameConstraints sont spécifiées au paragraphe 4.2.1.10, et ces règles peuvent ne pas se conformer aux règles pour le type sous-jacent. Par exemple, quand le champ uniformResourceIdentifier apparaît dans une extension nameConstraints, il doit contenir un nom DNS (par exemple, "host.exemple.com" ou ".exemple.com") plutôt qu'un URI.

Les développeurs sont avertis que la communauté X.500 a développé une série de règles d'extensibilité. Ces règles déterminent quand une définition ASN.1 peut être changée sans allouer un nouvel identifiant d'objet (OID, *Object Identifier*). Par exemple, au moins deux définitions d'extension incluses dans la [RFC2459], le prédécesseur du présent document de profil, ont des définitions ASN.1 différentes de la présente spécification, mais le même OID est utilisé. Si des éléments inconnus apparaissent dans une extension, et si l'extension n'est pas marquée comme critique, ces éléments inconnus pourraient être ignorés, comme suit :

(a) ignorer toutes les allocations de nom de bit inconnues dans une chaîne binaire ;

(b) ignorer tous les nombres désignés inconnus dans un type ENUMERATED ou ENTIER qui est utilisé dans le style énuméré, pourvu que le nombre se produisent comme un élément facultatif d'un SET ou SEQUENCE ; et

(c) ignorer tous les éléments inconnus dans les SET, à la fin des SEQUENCE, ou dans des CHOIX où le CHOIX est lui-même un élément facultatif d'un SET ou SEQUENCE.

Si une extension contenant des valeurs inattendues est marquée comme critique, la mise en œuvre DOIT rejeter le certificat ou CRL contenant l'extension non reconnue.

Appendice C. Exemples

Cet appendice contient quatre exemples : trois certificats et une CRL. Les deux premiers certificats et la CRL comportent un chemin de certification minimal.

L'Appendice C.1 contient une mémorisation annotée en hexadécimal d'un certificat "auto-signé" produit par une CA dont le nom distinctif est cn=Example CA,dc=example,dc=com. Le certificat contient une clé publique RSA, et est signé par la clé privée RSA correspondante.

L'Appendice C.2 contient une mémorisation annotée en hexadécimal d'un certificat d'entité d'extrémité. Le certificat d'entité d'extrémité contient une clé publique RSA, et est signé par la clé privée correspondant au certificat "auto-signé" de l'Appendice C.1.

L'Appendice C.3 contient une mémorisation annotée en hexadécimal d'un certificat d'entité d'extrémité qui contient une clé publique DSA avec des paramètres, et est signé avec DSA et SHA-1. Ce certificat ne fait pas partie du chemin de certification minimal.

L'Appendice C.4 contient une mémorisation annotée en hexadécimal d'une CRL. La CRL est produite par la CA dont le nom distinctif est cn=Example CA,dc=example,dc=com et la liste des certificats révoqués inclut le certificat d'entité d'extrémité présenté en Appendice C.2.

Les certificats ont été traités en utilisant l'utilitaire dumpasn1 de Peter Gutmann pour générer le résultat. La source de l'utilitaire dumpasn1 est disponible à < http://www.cs.auckland.ac.nz/~pgut001/dumpasn1.c >. Le code binaire pour les certificats et les CRL est disponible à http://csrc.nist.gov/groups/ST/crypto_apps_infra/documents/pkixtools.

Dans les endroits de cet appendice où un nom distinctif est spécifié en utilisant une représentation de chaîne, les chaînes sont formatées en utilisant les règles spécifiées dans la [RFC4514].

C.1 Certificat RSA auto signé

Cet appendice contient une mémorisation annotée en hexadécimal d'un certificat version 3 de 578 octets. Le certificat contient les informations suivantes :
(a) le numéro de série est 17,
(b) le certificat est signé avec RSA et l'algorithme de hachage SHA-1,
(c) le nom distinctif du producteur est cn=Example CA,dc=example,dc=com,
(d) le nom distictif du sujet est cn=Example CA,dc=example,dc=com,
(e) le certificat a été produit le 30 avril 2004 et a expiré le 30 avril 2005,
(f) le certificat contient une clé publique RSA de 1024 bits,
(g) le certificat contient une extension d'identifiant de clé sujette générée en utilisant la méthode (1) du paragraphe 4.2.1.2, et
(h) le certificat est un certificat de CA (comme indiqué par l'extension de contraintes de base).

```
  0  574: SEQUENCE {
  4  423:   SEQUENCE {
  8    3:     [0] {
 10    1:       ENTIER 2
       :     }
 13    1:     ENTIER 17
 16   13:     SEQUENCE {
 18    9:       IDENTIFIANT D'OBJET : sha1withRSAEncryption (1 2 840 113549 1 1 5)
 29    0:       NULL
       :     }
 31   67:     SEQUENCE {
 33   19:       SET {
 35   17:         SEQUENCE {
 37   10:           IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
 49    3:           IA5String 'com'
       :         }
       :       }
 54   23:       SET {
 56   21:         SEQUENCE {
 58   10:           IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
 70    7:           IA5String 'example'
       :         }
       :       }
 79   19:       SET {
 81   17:         SEQUENCE {
 83    3:           IDENTIFIANT D'OBJET commonName (2 5 4 3)
 88   10:           PrintableString 'Example CA'
       :         }
       :       }
```

```
    :     }
100  30:   SEQUENCE {
102  13:     UTCTime 30/04/2004 14:25:34 GMT
117  13:     UTCTime 30/04/2005 14:25:34 GMT
    :     }
132  67:   SEQUENCE {
134  19:     SET {
136  17:       SEQUENCE {
138  10:         IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
150   3:         IA5String 'com'
    :       }
    :     }
155  23:     SET {
157  21:       SEQUENCE {
159  10:         IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
171   7:         IA5String 'example'
    :       }
    :     }
180  19:     SET {
182  17:       SEQUENCE {
184   3:         IDENTIFIANT D'OBJET commonName (2 5 4 3)
189  10:         PrintableString 'Example CA'
    :       }
    :     }
    :   }
201 159:   SEQUENCE {
204  13:     SEQUENCE {
206   9:       IDENTIFIANT D'OBJET : rsaEncryption (1 2 840 113549 1 1 1)
217   0:       NULL
    :     }
219 141:     CHAINE BINAIRE, encapsule {
223 137:       SEQUENCE {
226 129:         ENTIER
    :           00 C2 D7 97 6D 28 70 AA 5B CF 23 2E 80 70 39 EE
    :           DB 6F D5 2D D5 6A 4F 7A 34 2D F9 22 72 47 70 1D
    :           EF 80 E9 CA 30 8C 00 C4 9A 6E 5B 45 B4 6E A5 E6
    :           6C 94 0D FA 91 E9 40 FC 25 9D C7 B7 68 19 56 8F
    :           11 70 6A D7 F1 C9 11 4F 3A 7E 3F 99 8D 6E 76 A5
    :           74 5F 5E A4 55 53 E5 C7 68 36 53 C7 1D 3B 12 A6
    :           85 FE BD 6E A1 CA DF 35 50 AC 08 D7 B9 B4 7E 5C
    :           FE E2 A3 2C D1 23 84 AA 98 C0 9B 66 18 9A 68 47
    :           E9
358   3:         ENTIER 65537
    :       }
    :     }
    :   }
363  66:   [3] {
365  64:     SEQUENCE {
367  29:       SEQUENCE {
369   3:         IDENTIFIANT D'OBJET subjectKeyIdentifier (2 5 29 14)
374  22:         CHAINE D'OCTETS, encapsule {
376  20:           CHAINE D'OCTETS
    :           08 68 AF 85 33 C8 39 4A 7A F8 82 93 8E 70 6A 4A 20 84 2C 32
    :         }
    :       }
398  14:       SEQUENCE {
400   3:         IDENTIFIANT D'OBJET keyUsage (2 5 29 15)
405   1:         BOOLEAN TRUE
408   4:         CHAINE D'OCTETS, encapsule {
410   2:           CHAINE BINAIRE 1 bit inutilisé
    :           '0000011'B
    :         }
    :       }
```

```
414  15:       SEQUENCE {
416   3:        IDENTIFIANT D'OBJET basicConstraints (2 5 29 19)
421   1:        BOOLEAN TRUE
424   5:        CHAINE D'OCTETS, encapsule {
426   3:          SEQUENCE {
428   1:            BOOLEAN TRUE
      :           }
      :         }
      :       }
      :     }
      :   }
      : }
431  13:  SEQUENCE {
433   9:   IDENTIFIANT D'OBJET : sha1withRSAEncryption (1 2 840 113549 1 1 5)
444   0:   NULL
      :   }
446 129:  CHAINE BINAIRE
      :   6C F8 02 74 A6 61 E2 64 04 A6 54 0C 6C 72 13 AD
      :   3C 47 FB F6 65 13 A9 85 90 33 EA 76 A3 26 D9 FC
      :   D1 0E 15 5F 28 B7 EF 93 BF 3C F3 E2 3E 7C B9 52
      :   FC 16 6E 29 AA E1 F4 7A 6F D5 7F EF B3 95 CA F3
      :   66 88 83 4E A1 35 45 84 CB BC 9B B8 C8 AD C5 5E
      :   46 D9 0B 0E 8D 80 E1 33 2B DC BE 2B 92 7E 4A 43
      :   A9 6A EF 8A 63 61 B3 6E 47 38 BE E8 0D A3 67 5D
      :   F3 FA 91 81 3C 92 BB C5 5F 25 25 EB 7C E7 D8 A1
      : }
```

C.2 Certificat d'entité d'extrémité utilisant RSA

Cet appendice contient une mémorisation annotée en hexadécimal d'un certificat de version 3 de 629 octets. Le certificat contient les informations suivantes :

(a) le numéro de série est 18,
(b) le certificat est signé avec RSA et l'algorithme de hachage SHA-1,
(c) le nom distinctif du producteur est cn=Example CA,dc=example,dc=com,
(d) le nom distinctif du sujet est cn=End Entity,dc=example,dc=com,
(e) le certificat était valide du 15 septembre 2004 au 15 mars 2005,
(f) le certificat contient une clé publique RSA de 1024 bits,
(g) le certificat est un certificat d'entité d'extrémité, car l'extension de contraintes de base n'est pas présente,
(h) le certificat contient une extension d'identifiant de clé d'autorité qui correspond à l'identifiant de clé sujette du certificat en appendice C.1, et
(i) le certificat inclut un nom de remplacement : une adresse de messagerie électronique (rfc822Name) de "end.entity@example.com".

```
  0  625: SEQUENCE {
  4  474:   SEQUENCE {
  8    3:    [0] {
 10    1:      ENTIER 2
       :     }
 13    1:    ENTIER 18
 16   13:    SEQUENCE {
 18    9:     IDENTIFIANT D'OBJET : sha1withRSAEncryption (1 2 840 113549 1 1 5)
 29    0:     NULL
       :     }
 31   67:    SEQUENCE {
 33   19:     SET {
 35   17:      SEQUENCE {
 37   10:       IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
 49    3:       IA5String 'com'
       :      }
       :     }
 54   23:     SET {
 56   21:      SEQUENCE {
```

```
 58  10:          IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
 70   7:          IA5String 'example'
      :         }
      :       }
 79  19:       SET {
 81  17:         SEQUENCE {
 83   3:           IDENTIFIANT D'OBJET commonName (2 5 4 3)
 88  10:           PrintableString 'Example CA'
      :         }
      :       }
      :     }
100  30:   SEQUENCE {
102  13:     UTCTime 15/09/2004 11:48:21 GMT
117  13:     UTCTime 15/03/2005 11:48:21 GMT
      :   }
132  67:   SEQUENCE {
134  19:     SET {
136  17:       SEQUENCE {
138  10:         IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
150   3:         IA5String 'com'
      :       }
      :     }
155  23:     SET {
157  21:       SEQUENCE {
159  10:         IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
171   7:         IA5String 'example'
      :       }
      :     }
180  19:     SET {
182  17:       SEQUENCE {
184   3:         IDENTIFIANT D'OBJET commonName (2 5 4 3)
189  10:         PrintableString 'End Entity'
      :       }
      :     }
      :   }
201 159:   SEQUENCE {
204  13:     SEQUENCE {
206   9:       IDENTIFIANT D'OBJET : rsaEncryption (1 2 840 113549 1 1 1)
217   0:       NULL
      :     }
219 141:     CHAINE BINAIRE, encapsule {
223 137:       SEQUENCE {
226 129:         ENTIER
      :           00 E1 6A E4 03 30 97 02 3C F4 10 F3 B5 1E 4D 7F
      :           14 7B F6 F5 D0 78 E9 A4 8A F0 A3 75 EC ED B6 56
      :           96 7F 88 99 85 9A F2 3E 68 77 87 EB 9E D1 9F C0
      :           B4 17 DC AB 89 23 A4 1D 7E 16 23 4C 4F A8 4D F5
      :           31 B8 7C AA E3 1A 49 09 F4 4B 26 DB 27 67 30 82
      :           12 01 4A E9 1A B6 C1 0C 53 8B 6C FC 2F 7A 43 EC
      :           33 36 7E 32 B2 7B D5 AA CF 01 14 C6 12 EC 13 F2
      :           2D 14 7A 8B 21 58 14 13 4C 46 A3 9A F2 16 95 FF
      :           23
358   3:         ENTIER 65537
      :       }
      :     }
      :   }
363 117:   [3] {
365 115:     SEQUENCE {
367  33:       SEQUENCE {
369   3:         IDENTIFIANT D'OBJET subjectAltName (2 5 29 17)
374  26:         CHAINE D'OCTETS, encapsule {
376  24:           SEQUENCE {
378  22:             [1] 'end.entity@example.com'
```

```
     :           }
     :         }
     :       }
402  29:       SEQUENCE {
404   3:        IDENTIFIANT D'OBJET subjectKeyIdentifier (2 5 29 14)
409  22:        CHAINE D'OCTETS, encapsule {
411  20:          CHAINE D'OCTETS
     :           17 7B 92 30 FF 44 D6 66 E1 90 10 22 6C 16 4F C0
     :           8E 41 DD 6D
     :         }
     :       }
433  31:       SEQUENCE {
435   3:        IDENTIFIANT D'OBJET : authorityKeyIdentifier (2 5 29 35)
440  24:        CHAINE D'OCTETS, encapsule {
442  22:          SEQUENCE {
444  20:            [0]
     :             08 68 AF 85 33 C8 39 4A 7A F8 82 93 8E 70 6A
     :             4A 20 84 2C 32
     :           }
     :         }
     :       }
466  14:       SEQUENCE {
468   3:        IDENTIFIANT D'OBJET keyUsage (2 5 29 15)
473   1:        BOOLEAN TRUE
476   4:        CHAINE D'OCTETS, encapsule {
478   2:          CHAINE BINAIRE 6 bits inutilisés
     :           '11'B
     :         }
     :       }
     :     }
     :   }
482  13:  SEQUENCE {
484   9:    IDENTIFIANT D'OBJET : sha1withRSAEncryption (1 2 840 113549 1 1 5)
495   0:    NULL
     :   }
497 129:  CHAINE BINAIRE
     :   00 20 28 34 5B 68 32 01 BB 0A 36 0E AD 71 C5 95
     :   1A E1 04 CF AE AD C7 62 14 A4 1B 36 31 C0 E2 0C
     :   3D D9 1E C0 00 DC 10 A0 BA 85 6F 41 CB 62 7A B7
     :   4C 63 81 26 5E D2 80 45 5E 33 E7 70 45 3B 39 3B
     :   26 4A 9C 3B F2 26 36 69 08 79 BB FB 96 43 77 4B
     :   61 8B A1 AB 91 64 E0 F3 37 61 3C 1A A3 A4 C9 8A
     :   B2 BF 73 D4 4D E4 58 E4 62 EA BC 20 74 92 86 0E
     :   CE 84 60 76 E9 73 BB C7 85 D3 91 45 EA 62 5D CD
     : }
```

C.3 Certificat d'entité d'extrémité utilisant DSA

Cet appendice contient une mémorisation annotée en hexadécimal d'un certificat version 3 de 914 octets. Le certificat contient les informations suivantes :

(a) le numéro de série est 256,
(b) le certificat est signé avec DSA et l'algorithme de hachage SHA-1,
(c) le nom distinctif du producteur est cn=Example DSA CA,dc=example,dc=com,
(d) le nom distinctif du sujet est cn=DSA End Entity,dc=example,dc=com,
(e) le certificat a été produit le 2 mai 2004 et a expiré le 2 mai 2005,
(f) le certificat contient une clé publique DSA de 1024 bits avec des paramètres,
(g) le certificat est un certificat d'entité d'extrémité (non un certificat de CA),
(h) le certificat inclut un nom de sujet de remplacement de "<http://www.example.com/users/DSAendentity.html>" et un nom de producteur de remplacement de "<http://www.example.com>" – tous deux sont des URL,
(i) le certificat inclut une extension d'identifiant de clé d'autorité et une extension de politique de certificat qui spécifient l'OID de politique 2.16.840.1.101.3.2.1.48.9; et

(j) le certificat inclut une extension d'usage de clé critique qui spécifie que la clé publique est destinée à la vérification des signatures numériques.

```
  0  910: SEQUENCE {
  4  846:   SEQUENCE {
  8    3:     [0] {
 10    1:       ENTIER 2
       :     }
 13    2:     ENTIER 256
 17    9:     SEQUENCE {
 19    7:       IDENTIFIANT D'OBJET dsaWithSha1 (1 2 840 10040 4 3)
       :     }
 28   71:     SEQUENCE {
 30   19:       SET {
 32   17:         SEQUENCE {
 34   10:           IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
 46    3:           IA5String 'com'
       :         }
       :       }
 51   23:       SET {
 53   21:         SEQUENCE {
 55   10:           IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
 67    7:           IA5String 'example'
       :         }
       :       }
 76   23:       SET {
 78   21:         SEQUENCE {
 80    3:           IDENTIFIANT D'OBJET commonName (2 5 4 3)
 85   14:           PrintableString 'Example DSA CA'
       :         }
       :       }
       :     }
101   30:     SEQUENCE {
103   13:       UTCTime 02/05/2004 16:47:38 GMT
118   13:       UTCTime 02/05/2005 16:47:38 GMT
       :     }
133   71:     SEQUENCE {
135   19:       SET {
137   17:         SEQUENCE {
139   10:           IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
151    3:           IA5String 'com'
       :         }
       :       }
156   23:       SET {
158   21:         SEQUENCE {
160   10:           IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
172    7:           IA5String 'example'
       :         }
       :       }
181   23:       SET {
183   21:         SEQUENCE {
185    3:           IDENTIFIANT D'OBJET commonName (2 5 4 3)
190   14:           PrintableString 'DSA End Entity'
       :         }
       :       }
       :     }
206  439:     SEQUENCE {
210  300:       SEQUENCE {
214    7:         IDENTIFIANT D'OBJET dsa (1 2 840 10040 4 1)
223  287:         SEQUENCE {
227  129:           ENTIER
       :             00 B6 8B 0F 94 2B 9A CE A5 25 C6 F2 ED FC FB 95
       :             32 AC 01 12 33 B9 E0 1C AD 90 9B BC 48 54 9E F3
```

```
    :          94 77 3C 2C 71 35 55 E6 FE 4F 22 CB D5 D8 3E 89
    :          93 33 4D FC BD 4F 41 64 3E A2 98 70 EC 31 B4 50
    :          DE EB F1 98 28 0A C9 3E 44 B3 FD 22 97 96 83 D0
    :          18 A3 E3 BD 35 5B FF EE A3 21 72 6A 7B 96 DA B9
    :          3F 1E 5A 90 AF 24 D6 20 F0 0D 21 A7 D4 02 B9 1A
    :          FC AC 21 FB 9E 94 9E 4B 42 45 9E 6A B2 48 63 FE
    :          43
359  21:         ENTIER
    :          00 B2 0D B0 B1 01 DF 0C 66 24 FC 13 92 BA 55 F7
    :          7D 57 74 81 E5
382  129:        ENTIER
    :          00 9A BF 46 B1 F5 3F 44 3D C9 A5 65 FB 91 C0 8E
    :          47 F1 0A C3 01 47 C2 44 42 36 A9 92 81 DE 57 C5
    :          E0 68 86 58 00 7B 1F F9 9B 77 A1 C5 10 A5 80 91
    :          78 51 51 3C F6 FC FC CC 46 C6 81 78 92 84 3D F4
    :          93 3D 0C 38 7E 1A 5B 99 4E AB 14 64 F6 0C 21 22
    :          4E 28 08 9C 92 B9 66 9F 40 E8 95 F6 D5 31 2A EF
    :          39 A2 62 C7 B2 6D 9E 58 C4 3A A8 11 81 84 6D AF
    :          F8 B4 19 B4 C2 11 AE D0 22 3B AA 20 7F EE 1E 57
    :          18
    :        }
    :      }
514  132:    CHAINE BINAIRE, encapsule {
518  128:     ENTIER
    :        30 B6 75 F7 7C 20 31 AE 38 BB 7E 0D 2B AB A0 9C
    :        4B DF 20 D5 24 13 3C CD 98 E5 5F 6C B7 C1 BA 4A
    :        BA A9 95 80 53 F0 0D 72 DC 33 37 F4 01 0B F5 04
    :        1F 9D 2E 1F 62 D8 84 3A 9B 25 09 5A 2D C8 46 8E
    :        2B D4 F5 0D 3B C7 2D C6 6C B9 98 C1 25 3A 44 4E
    :        8E CA 95 61 35 7C CE 15 31 5C 23 13 1E A2 05 D1
    :        7A 24 1C CB D3 72 09 90 FF 9B 9D 28 C0 A1 0A EC
    :        46 9F 0D B8 D0 DC D0 18 A6 2B 5E F9 8F B5 95 BE
    :      }
    :    }
649  202:  [3] {
652  199:   SEQUENCE {
655  57:    SEQUENCE {
657  3:      IDENTIFIANT D'OBJET subjectAltName (2 5 29 17)
662  50:     CHAINE D'OCTETS, encapsule {
664  48:       SEQUENCE {
666  46:         [6]
    :          'http://www.example.com/users/DSAendentity.'
    :          'html'
    :         }
    :       }
    :     }
714  33:    SEQUENCE {
716  3:      IDENTIFIANT D'OBJET issuerAltName (2 5 29 18)
721  26:     CHAINE D'OCTETS, encapsule {
723  24:       SEQUENCE {
725  22:         [6] 'http://www.example.com'
    :         }
    :       }
    :     }
749  29:    SEQUENCE {
751  3:      IDENTIFIANT D'OBJET subjectKeyIdentifier (2 5 29 14)
756  22:     CHAINE D'OCTETS, encapsule {
758  20:       CHAINE D'OCTETS
    :        DD 25 66 96 43 AB 78 11 43 44 FE 95 16 F9 D9 B6
    :        B7 02 66 8D
    :       }
    :     }
780  31:    SEQUENCE {
```

```
 782   3:         IDENTIFIANT D'OBJET : authorityKeyIdentifier (2 5 29 35)
 787  24:         CHAINE D'OCTETS, encapsule {
 789  22:           SEQUENCE {
 791  20:             [0]
      :               86 CA A5 22 81 62 EF AD 0A 89 BC AD 72 41 2C
      :               29 49 F4 86 56
      :             }
      :           }
      :         }
 813  23:       SEQUENCE {
 815   3:         IDENTIFIANT D'OBJET certificatePolicies (2 5 29 32)
 820  16:         CHAINE D'OCTETS, encapsule {
 822  14:           SEQUENCE {
 824  12:             SEQUENCE {
 826  10:               IDENTIFIANT D'OBJET '2 16 840 1 101 3 2 1 48 9'
      :             }
      :           }
      :         }
      :       }
 838  14:       SEQUENCE {
 840   3:         IDENTIFIANT D'OBJET keyUsage (2 5 29 15)
 845   1:         BOOLEAN TRUE
 848   4:         CHAINE D'OCTETS, encapsule {
 850   2:           CHAINE BINAIRE 7 bits inutilisés
      :             '1'B (bit 0)
      :           }
      :         }
      :       }
      :     }
      :   }
 854   9:   SEQUENCE {
 856   7:     IDENTIFIANT D'OBJET dsaWithSha1 (1 2 840 10040 4 3)
      :   }
 865  47:   CHAINE BINAIRE, encapsule {
 868  44:     SEQUENCE {
 870  20:       ENTIER
      :         65 57 07 34 DD DC CA CC 5E F4 02 F4 56 42 2C 5E
      :         E1 B3 3B 80
 892  20:       ENTIER
      :         60 F4 31 17 CA F4 CF FF EE F4 08 A7 D9 B2 61 BE
      :         B1 C3 DA BF
      :     }
      :   }
      : }
```

C.4 Liste de révocation de certificat

Cet appendice contient une mémorisation annotée en hexadécimal d'une CRL version 2 avec deux extensions (cRLNumber et authorityKeyIdentifier). La CRL a été produite par cn=Example CA,dc=example,dc=com le 5 février 2005 ; la prochaine production programmée est du 6 février 2005. La CRL inclut un certificat révoqué : numéro de série 18, qui a été révoqué le 19 novembre 2004 suite à une clé compromise. La CRL elle même a le numéro 12, et elle a été signée avec RSA et SHA-1.

```
0  352: SEQUENCE {
4  202:   SEQUENCE {
7    1:     ENTIER 1
0   13:     SEQUENCE {
2    9:       IDENTIFIANT D'OBJET : sha1withRSAEncryption (1 2 840 113549 1 1 5)
23   0:       NULL
      :       }
25   67:     SEQUENCE {
27   19:       SET {
```

```
 29  17:       SEQUENCE {
 31  10:         IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
 43   3:         IA5String 'com'
      :         }
      :       }
 48  23:       SET {
 50  21:         SEQUENCE {
 52  10:           IDENTIFIANT D'OBJET : domainComponent (0 9 2342 19200300 100 1 25)
 64   7:           IA5String 'example'
      :         }
      :       }
 73  19:       SET {
 75  17:         SEQUENCE {
 77   3:           IDENTIFIANT D'OBJET commonName (2 5 4 3)
 82  10:           PrintableString 'Example CA'
      :         }
      :       }
      :     }
 94  13:   UTCTime 05/02/2005 12:00:00 GMT
109  13:   UTCTime 06/02/2005 12:00:00 GMT
124  34:   SEQUENCE {
126  32:     SEQUENCE {
128   1:       ENTIER 18
131  13:       UTCTime 19/11/2004 15:57:03 GMT
146  12:       SEQUENCE {
148  10:         SEQUENCE {
150   3:           IDENTIFIANT D'OBJET cRLReason (2 5 29 21)
155   3:           CHAINE D'OCTETS, encapsule {
157   1:             ENUMERATED 1
      :           }
      :         }
      :       }
      :     }
      :   }
160  47:   [0] {
162  45:     SEQUENCE {
164  31:       SEQUENCE {
166   3:         IDENTIFIANT D'OBJET : authorityKeyIdentifier (2 5 29 35)
171  24:         CHAINE D'OCTETS, encapsule {
173  22:           SEQUENCE {
175  20:             [0] : 08 68 AF 85 33 C8 39 4A 7A F8 82 93 8E 70 6A : 4A 20 84 2C 32
      :           }
      :         }
      :       }
197  10:       SEQUENCE {
199   3:         IDENTIFIANT D'OBJET cRLNumber (2 5 29 20)
204   3:         CHAINE D'OCTETS, encapsule {
206   1:           ENTIER 12
      :         }
      :       }
      :     }
      :   }
209  13:   SEQUENCE {
211   9:     IDENTIFIANT D'OBJET : sha1withRSAEncryption (1 2 840 113549 1 1 5)
222   0:     NULL
      :   }
224 129:   CHAINE BINAIRE
      :     22 DC 18 7D F7 08 CE CC 75 D0 D0 6A 9B AD 10 F4
      :     76 23 B4 81 6E B5 6D BE 0E FB 15 14 6C C8 17 6D
      :     1F EE 90 17 A2 6F 60 E4 BD AA 8C 55 DE 8E 84 6F
      :     92 F8 9F 10 12 27 AF 4A D4 2F 85 E2 36 44 7D AA
      :     A3 4C 25 38 15 FF 00 FD 3E 7E EE 3D 26 12 EB D8
```

```
:    E7 2B 62 E2 2B C3 46 80 EF 78 82 D1 15 C6 D0 9C
:    72 6A CB CE 7A ED 67 99 8B 6E 70 81 7D 43 42 74
:    C1 A6 AF C1 55 17 A2 33 4C D6 06 98 2B A4 FC 2E
:  }
```

Adresse des auteurs

David Cooper
NIST
100 Bureau Drive, Mail Stop 8930
Gaithersburg, MD 20899-8930
USA
mél : david.cooper@nist.gov

Stefan Santesson
Microsoft
One Microsoft Way
Redmond, WA 98052
USA
mél : stefans@microsoft.com

Stephen Farrell
Distributed Systems Group
Computer Science Department
Trinity College Dublin
Ireland
mél : stephen.farrell@cs.tcd.ie

Sharon Boeyen
Entrust
1000 Innovation Drive
Ottawa, Ontario
Canada K2K 3E7
mél : sharon.boeyen@entrust.com

Russell Housley
Vigil Security, LLC
918 Spring Knoll Drive
Herndon, VA 20170
USA
mél : housley@vigilsec.com

Tim Polk
NIST
100 Bureau Drive, Mail Stop 8930
Gaithersburg, MD 20899-8930
USA
mél : wpolk@nist.gov

Déclaration complète de droits de reproduction