rfc5280, rfc6818

Profile de certification et de liste de révocation de certificat d'infrastructure à clé publique X.509

Introduction

Cette spécification fait partie d'une famille de standards pour les infrastructures à clé publique X.509 pour l'Internet.

Cette spécification profile le format et les sémantiques des certificats et des listes de révocation de certificat pour les PKI Internet. Les procédures sont décrites pour traiter les chemins de certification dans l'environnement Internet. Finalement, les modules ASN.1 sont fournis dans l'appendice pour toutes les structures de données définies ou référencées.

La section "Exigences et hypothèses" décris les pré-requis de PKI Internet et les hypothèses qui affectent le périmètre de ce document.

La section "Aperçu de l'approche" Présentent un modèle d'architecture qui décrivent ses relations aux précédant standards IETF et ISO/IEC/ITU-T. En particulier, la relation de ce document avec les spécification PEM IETF et les document ISO/IEC/ITU-T X.509 sont décris.

La section "Profile de certificat et d'extensions de certificat" profile les certificats X.509 version 3.

La section "Profile de CRL et d'extensions de CRL" profile les CRL X.509 v2.

La section "Validation de chemin de certification" inclus les procédures de validation de chemin de certification. Ces procédures sont basée sur la définition ISO/IEC/ITU-T.

Les procédures pour l'identification et l'encodage des clés publiques et des signatures numérique sont définis dans la rfc3279, la rfc 4055, et la rfc4491. Les implémentations de cette spécification ne sont pas obligés d'utiliser un algorithme crypto-graphique particulier. Cependant, les implémentations conformes qui utilisent les algorithmes identifiés dans la rfc3279,la rfc 4055, et la rfc 4491 doivent identifier en encoder la clé publique et les signatures numérique comme décris dans ces spécifications.

Finalement, 2 appendices sont fournis pour aider les implémenteurs.

Cette spécification obsolète la rfc 3280. Les différences sont résumées ici :

- Support amélioré pour les noms internationalisés, avec des règles pour l'encodage et la comparaison de noms de domaines internationalisés, Identifiant de ressources internationalisés (IRI), et de noms distinct. Ces règles sont alignés avec les règles de comparaison établies dans les rfc courantes, incluant la rfc3490, rfc3987 et la rfc 4518.
- Les section "Issuer" et "Subject" incorporent les conditions pour l'utilisation continue de schémas d'encodage de texte existant qui ont été spécifiés dans la rfc4630. Là où c'est utilisé par les PKI établies, la transition vers UTF8String pourrait poser des problème DOS basés sur les erreurs de chaînage de nom ou des traitement incorrecte de contraintes de nom.
- La section qui spécifie l'extension de certificat privateKeyUsagePeriod de la rfc 3280, mais déprécie son utilisant, a été supprimée. L'utilisation de cette extension standards ISO n'est plus dépréciée ni recommandée pour l'utilisation dans les PKI de l'Internet.
- La section Policy Mappings recommande de marquer l'extension de contraintes de stratégie comme critique. La rfc 3280 nécessitait que l'extension de mappage de stratégie soit marquée non critique.
- La section Policy Constraints nécessite que l'extension de contraintes de stratégie soit critique. la rfc 3280 permettait cette extension d'être critique ou non.
- L'extension de CRL AIA, comme spécifié dans la rfc 4325, a été ajoutée.
- Les sections extensions de CRL et extensions d'entrée de crl clarifient les règles pour manipuler les extensions de CRL et d'entrée de CRL non-reconnues, respectivement.
- La section qui spécifie l'extension d'entrée de CRL holdInstructionCode dans la rfc 3280 a été supprimée.

- L'algorithme de validation de chemin ne suit plus la criticité des extensions de stratégie de certificat dans une chaîne de certificats. Dans la rfc 3280, cette information était retourné à un tier de confiance.
- La section de considérations de sécurité adresse les risques de dépendances circulaires se produisant lors de l'utilisation de schémas https ou similaire dans les points de distribution de CRL, AIA, ou extension d'accès aux informations du sujet.
- La section de considérations de sécurité adresse les risques associés aves l'ambiguïté des noms
- La section de considérations de sécurité référence les rfc 4210 pour les procédures pour signaler les changements dans les opération de CA.

Pré-requis et hypothèses

Le but de cette 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 WWW, les messageries électroniques, l'authentification utilisateur, et IPSec. Afin de soulager certains obstacles à l'utilisation des certificat X.509, cet document définis un profile pour promouvoir le développement de systèmes de gestion de certificat, le développement d'outils applicatifs, et d'interopérabilité déterminé par stratégie.

Un utilisateur de certificat devrait revoir la stratégie de certificat générée par l'autorité de certification avant de faire confiance aux services d'authentification et de non-répudiation associés avec la clé publique dans un certificat particulier. À cette fin, ce standards ne prescrit pas de règles juridiques contraignantes ou de droits.

Vu que les outils d'autorisation supplémentaires et de gestions d'attributs émergent, tels que les certificats d'attributs, 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 appropriés de transport de nombreux attributs authentifiés.

Communication et topologie

Les utilisateurs de certificat vont opérer dans une grande variété d'environnements en respect de leur topologie de communication, spécialement les utilisateurs de messagerie électronique. Ce profile supporte les utilisateurs à faible bande passante, connectivité IP temps-réel, ou connexion à haute disponibilité. En plus, le profile permet la présence de firewall ou autre communication filtrée.

Ce profil n'assume pas le déploiement d'un système d'annuaire X.500 ou LDAP. Le profile n'interdit pas leur utilisation ; cependant, tout moyen de distribuer des certificats et CRL peut être utilisé.

Critère d'acceptabilité

Le but d'une PKI est de répondre aux besoins de fonctions déterministes, d'identification automatisée, d'authentification, de contrôle d'accès, et d'autorisation. Le support pour cet services détermine les attributs contenus dans le certificat aussi bien que les information de contrôle auxiliaire dans le certificat tel que les données de stratégie et contraintes de chemin de certification.

Attentes des utilisateurs

Les utilisateurs de PKI Internet sont des gens et des processus qui utilisent des clients logiciel et sont les sujets nommés dans les certificats. Ces utilisations incluent les lecteurs et rédacteurs de messages électroniques, les clients pour les navigateurs web, les serveurs web, et le gestionnaire de clé pour IPSec dans un routeur. Ce profile reconnaît les limitations des plate-formes que les utilisateurs utilisent et les limitations de la sophistication et de l'attention des utilisateurs eux-même. Cela se manifeste dans la responsabilité minimale de la configuration utilisateur (ex : les clé de CA de confiance, les règles), les contraintes d'utilisation des plate-formes explicites dans le

certificat, les contraintes de chemin de certification qui protège l'utilisateur les actions malicieuses, et les applications qui automatise sensiblement les fonction de validation.

Attentes des administrateurs

Comme avec les attentes des utilisateurs, le profile PKI de l'Internet est structuré pour supporter les individus qui opèrent généralement les CA. Fournir aux administrateurs des choix illimités augmente les chances qu'une erreur subtile d'administrateur de CA résulte en un large compromission. En outre, des choix illimités compliquent grandement le logiciel qui traite et valide les certificats crées par la CA.

Présentation de l'approche

Ceci est une vue simplifiée du modèle architecturel assumé par les infrastructures à clé publique utilisant les spécification X.509 (PKIX). Les composants dans ce modèle sont :

end entity L'utilisateur de certificats PKI et/ou le système utilisateur final qui est le sujet d'un certificat.

CA Autorité de certification

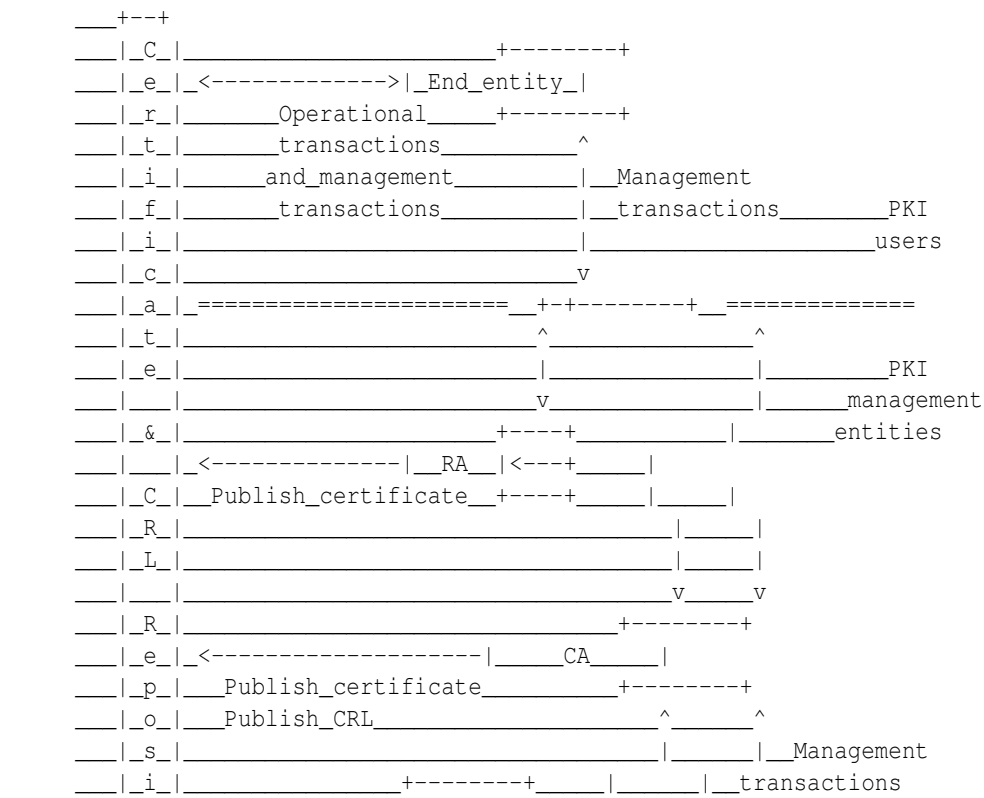
RA Autorité d'enregistrement, ex. un système opérationnel pour lequel une CA délègue certaines fonctions de gestion.

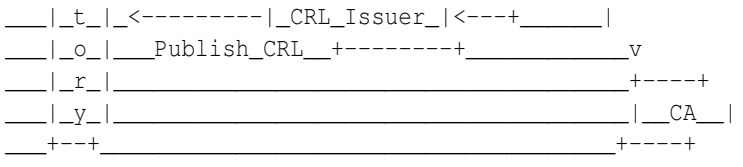
CRL issuer Un système qui génère et signe les CRL

repository Un système ou une collection de systèmes distribués qui stockent les certificats et les CRL et sert de moyen de distribution aux end entity

Les CA sont responsables de l'indication du statut de révocation des certificats qu'elles fournissent. Les informations de statut de révocation peuvent être fournis en utilisant OCSP, CRL, ou d'autres mécanismes. En général, quand les informations de statut de révocation sont fournis en utilisant les CRL, la CA est également le CRL issuer. Cependant, une CA peut déléguer cette responsabilité à une entité différente.

Noter qu'un Attribute Authority (AA) peut également choisir de déléguer la publication des CRL à un CRL issuer.





Certificat X.509 Version 3

Les utilisateurs d'une clé publique nécessitent la confidentialité de la clé privée associée qui est possédée par le bon sujet (personne ou système) avec laquelle un mécanisme de chiffrement ou de signature numérique sera utilisé. Cette confidentialité est obtenue via l'utilisation de certificats à clé publique, qui sont des structures de donnée qui lie des valeurs de clés publique à des sujets. La liaison est affirmée en ayant un CA de confiance qui signe numériquement chaque certificat. La CA peut baser cette affirmation sur les moyens techniques (ex : preuve de possession via un protocole de challenge/réponse), la présentation de la clé privée, ou sur une affirmation par le sujet. Un certificat a une durée de vie limitée, qui est indiquée dans son contenu signé. Parce que la signature du certificat et la du validité peuvent être vérifiés indépendamment par un client, les certificats peuvent être distribués via des communication et des systèmes serveurs non-sûres, et peuvent être mis en cache dans des stockages dans des systèmes non-sûres.

l'ITU-T X.509 ou l'ISO/IEC 9594-8, qui a été publié en 1988 comme partie des recommandations des annuaires X.500, définis un format de certificat standard. Le format de certificat dans le standard 1988 est appelé le format version 1 (v1). Quand X.500 a été révisé en 1993, 2 champs ont été ajoutés, résultant en un format version 2 (v2).

Les rfc PEM, publiés en 1993, incluent les spécifications pour une infrastructure à clé publique basés sur X.509 v1 (rfc 1422). L'expérience a montré que les déploiement rfc 1422 faits avec les formats v1 et v2 étaient défaillant dans de nombreux aspects. Plus important, de nombreux champs étaient nécessaires pour gérer les information que le concept et l'implémentation de PEM a démontré la nécessité. En réponse, l'ISO/IEC, l'ITU-T, et ANSI X9 on développés le format de certificat version 3 (v3). Il étend le format v2 en ajoutant les champs d'extension additionnels. Les types de champ d'extension particulier peuvent être spécifiés dans des standards ou peuvent être définis et enregistrés par une organisation ou une communauté. En Juin 1996, la standardisation du format de base v3 a été complété.

L'ISO/IEC, l'ITU-T, et ANSI X9 ont également développés des extensions standard développés pour l'utilisation dans le champs d'extensions v3. Ces extensions peuvent transmettre des données telles des informations d'identification de sujet additionnels, des information d'attribut de clé, informations de stratégie, et de contraintes de chemin de certification.

Cependant, les extensions standard e l'ISO/IEC, l'ITU-T, et de l'ANSI X9 étaient très générales dans leur utilisation. Pour développer des implémentations intéropérable des système X.509 v3 pour l'Internet, il est nécessaire de spécifier un profile pour utiliser les extensions X.509 v3 adaptés pour l'Internet. C'est un des objectifs de ce document pour spécifier un profile pour WWW, les messages électroniques, et les application IPSec. Les environnements avec les pré-requis additionnels peuvent construire par-dessus ce profil pour le remplacer.

Chemins de certification et confiance

Un utilisateur d'un service de sécurité nécessitant la connaissance d'une clé publique, a besoin généralement d'obtenir et de valider un certificat contenant la clé publique requise. Si l'utilisateur de clé publique ne possède pas une copie fiable de la clé publique de la CA qui a signé le certificat, le nom de la CA et les information liées (tel que la période de validité, ou les contraintes de nom), alors il peut nécessiter un certificat additionnel pour obtenir la clé publique. En général, une chaîne de plusieurs certificats peuvent être nécessaires, comprenant un certificat du propriétaire de la clé publique (l'entité finale) signée par une CA, et 0 ou plusieurs certificats de CA signés par d'autres CA. De telles chaînes, appelées chemin de certification, sont requis parce qu'un utilisateur de clé publique est seulement initialisé avec un nombre limité de clé publique de CA.

Les CA peuvent être configurées de différentes manières pour que les utilisateurs de clé publique soient capable de trouver les chemins de certification. Pour PEM, la rfc 1422 définis une structure hiérarchique rigide de CA. Il y a 3 types d'autorité de certification PEM :

(a) Internet Policy Registration Authority (IPRA) : cette autorité, qui opère sous les auspices de l'Internet Society, agit comme le

niveau racine de la hiérarchie de certification PEM, au niveau 1. Il fournis des certificats uniquement pour le niveau suivant d'autorités, les PCA. Tous les chemins de certification commencent avec l'IPRA.

(b) Policy Certification Authorities (PCAs) : les PCA sont au niveau 2 de la hiérarchie, chaque PCA est certifié par l'IPRA. Un PCA devrait établir et publier un état de sa stratégie, avec respect de certifier des utilisateurs ou des autorités de certification subordonnés. Des PCA distincts ont pour objectifs de satisfaire différents besoins utilisateurs. Par exemple, un PCA peut supporter les besoins de messages électroniques des organisations commerciales, et un autre PCA peut avoir une stratégie plus stricte conçue pour satisfaire les besoins légaux des signatures numériques.

(c) Les autorités de certification (CA) : les CA sont au niveau 3 de la hiérarchie et peuvent également être à des niveaux inférieurs. Ceux de niveau 3 sont certifiés par les PCA. Les CA représentent, par exemple, des organisations particulières, les unités organisationelles particulières, ou des zones géographiques particulières.

La rfc 1422 a en outre une règle de subordination de nom, qui nécessite qu'une CA peut seulement fournir des certificats pour des entités dont les noms sont subordonnés au nom de la CA elle-même, le trust associé avec un chemin de certification PEM est impliqueé par le nom du PCA. La règle de subordination de nom s'assure que les CA sous le PCA sont contraint sensiblement au jeu d'entité subordonnés qu'ils peuvent certifier. Les systèmes d'utilisateurs de certificat sont capable 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 cette version impose de nombreuses restrictions structurelles pour associer clairement les information de stratégie ou restreindre l'utilité des certificats. Ces restrictions incluent :

(a) Un hiérarchie "top-down", avec tous les chemins de certifications commençant depuis IPRA

(b) Une règle de subordination de nom qui restreints les noms des sujets des CA

(c) L'utilisation du concept PCA, qui nécessite la connaissance de tous les PCA individuells pour construire la logique de vérification de la chaîne de certification.

Avec X.509 v3, la plupart des pré-requis adressés par la rfc1422 peuvent être adressés en utilisant les extensions de certificats, sans avoir à restreindre les structures CA utilisées. En particulier, les extensions de certificat liées aux stratégies de certificat évitent les besoins de PCA et les extensions de contraintes évitent les besoins de règle de subordination de nom. En résultat, ce document supporte une architecture plus flexible, incluant :

(a) Les chemins de certification commencent avec une clé publique d'une CA dans le domaine de l'utilisateur, ou avec la clé publique de la racine d'une hiérarchie. Commencer avec la clé publique d'une CA dans le domaine de l'utilisateur a certains avantages. Dans certains environnements, le domaine local est le plus sûr.

(b) Les contraintes de nom peuvent être imposés via l'inclusion explicite d'une extension de contrainte de noms dans un certificat, mais n'est pas requis.

(c) Les extensions de stratégie et les mappages de stratégie remplacent le concept de PCA, qui permet un plus haut degré d'automatisation. L'application peut determiner si le chemin de certification est acceptable basé sur le contenu du certificat au lieu d'un connaissance à priori des PCA. Cela permet d'automatiser le traitement des chemins de certification.

X.509 v3 inclus également une extension qui identifie le sujet d'un certificat comme étant une CA ou une entité finale, réduisant la dépendance sur les informations out-of-band démandé dans PEM.

Cette spécification couvre 2 classes de certificats : les certificats CA et les certificats d'entité finale. Les certificats CA peuvent être divisés en 3 classes : cross-certificats, certificats auto-fournis et certificat auto-signés.

Les Cross-certificats sont des certificats CA dans lequel le fournisseur et les sujets sont des entités différentes. Les Cross-certificats décrivent une relation de confiance entre les 2 CA.

Les certificats auto-fournis sont des certificat CA dans lequel le fournisseur et le sujet sont la même entité. Les certificats auto-fournis sont générés pour supporter les changements dans la stratégie ou les opérations.

Les certificats auto-signés sont des certificat auto-fournis où la signature numérique peut être vérifiée par la clé publique lié dans le certificat. Les certificats auto-signés sont utilisé pour transporter une clé publique à utiliser au début des chemins de certification.

Les certificats d'entité finale sont fournis aux sujets qui ne sont pas autorisés à fournir de certificats.

Révocation

Quand un certificat est fournis, il est prévu pour être utilisé pour toute sa période de validité. Cependant, diverses circonstances peuvent causer un certificat à devenir invalide avant l'expiration de la période de validité. De telles circonstances incluent le changement de nom, changement d'association entre le sujet et la CA (ex : un employé termine sont emploie avec une organisation), et la compromission de la clé privée associée. Dans de telles circonstances, la CA doit révoquer le certificat.

X.509 définis une méthode de révocation de certificat. Cette méthode implique que chaque CA fournis périodiquement une structure de données appelée une liste de statut de révocation. Une CRL est une liste horodatée identifiant les certification révoqués. qui est signée par une CA ou un fournisseur de CA et est librement accessible dans de dépôt publique. Chaque certificat révoqué est identifié dans une CRL par son numéro de série. Quand un système utilisant les certificats utilise un certificat (par exemple pour vérifier la signature numérique d'un utilisateur), ce système ne vérifie pas seulement la signature et la validité du certificat mais récupère également une CRL utilisable et récente et vérifie que le numéro de série n'est pas dans cette CRL

La signification de "utilisable et récente" peut varier avec la stratégie locale, mais signifie généralement la plus récente CRL fournis. Une nouvelle CRL est fournie sur une base régulière. Une entrée est ajoutée à la CRL comme partie de la prochaine mise à jours suivant la notification de la révocation. Une entrée ne doit pas être supprimée de la CRL jusqu'à ce qu'elle apparaisse dans une dans une CRL régulière fournie au-delà de la période de validité du certificat.

Un avantage de cette méthode de révocation est que les CRL peuvent être distribués exactement de la même manière que les certificats eux-mêmes, via des serveurs et des communications non-sûres.

Une limitation de la méthode de révocation CRL, utilisant des communications et des serveurs non sûrs, et que la granularité de temps de révocation est limitée à la période d'émission de la CRL. Par exemple, si une révocation est reportée maintenant, cette révocation ne sera pas notifiée tant qu'une nouvelle CRL n'est pas générée.

Comme avec le format de certificat X.509 v3, pour pouvoir faciliter l'interopérabilité des implémentation, le format de CRL v2 X.509 doit être profilé pour l'utilisation sur l'Internet. C'est un des objectifs de ce document. Cependant, ce profile ne nécessite pas l'émission des CRL. Les formats de message et les protocoles supportant les notification de révocation on-line sont définis dans d'autres spécifications PKIX. Les méthodes on-line de notification de révocation peuvent être applicables dans certains environnements comme une alternative à la CRL X.509. La vérification de révocation On-line peut significativement réduire la latence entre une révocation et la distribution de l'information. Une fois que la CA acceptes une révocation comme authentique et valide, toute requête au service on-line va refléter correctement l'information de révocation.

Cependant, ces méthodes imposent de nouveaux pré-requis de sécurité : le validateur de certificat a besoin de faire confiance au service de validation on-line, alors que le dépôt n'a pas besoin de l'être.

Protocoles opérationnels

Les protocoles opérationnels sont requis pour délivrer les certificats et les CRL (ou les informations de statut) aux système client qui utilisent les certificats. Des dispositions sont nécessaires pour divers moyens de délivrer des certificats et les CRL, incluant les procédures de distribution basées sur LDAP, HTTP, FTP, et X.500. Les protocoles opérationnels supportant ces fonctions sont définies dans d'autres spécification PKIX. Ces spécifications peuvent inclurent les définition de formats de message et des procédures pour supporter tous les environnements opérationnel, incluant les définitions et les référence aux types de contenu MIME appropriés.

Protocoles de gestion

Les protocoles de gestion sont requis pour supporter les interactions on-line entre les utilisateurs de PKI et les entités de gestion. Par exemple, un protocole de gestion peut être utilisé entre une CA et un système client avec lequel une paire de clé est associée, ou entre 2 CA qui se cross-certifient entre-eux. Le jeu de fonctions qui nécessite potentiellement d'être supportés par les protocoles de gestion incluent :

(a) Enregistrement : c'est la procédure par laquelle un utilisateur se fait connaître à la CA (directement, ou via un RA), avec que la

CA ne lui fournisse un ou plusieurs certificats.

(b) Initialisation : Avant qu'un système client puisse opérer de manière sécurisé, il est nécessaire d'installer les clés qui ont la relation appropriée avec les clés stockées ailleurs dans l'infrastructure. Par exemple, le client a besoin d'être initialisée de manière sécurisée avec la clé publique et d'autres informations des CA de confiance, pour être utilisées dans les validation de chemins de certification. De plus, un client a généralement besoin d'être initialisé avec ses propres paires de clés.

(c) Certification : C'est le processus dans lequel une CA fournie un certificat pour une clé publique de l'utilisateur, et retourne ce certificat au système client de l'utilisateur et/ou poste ce certificat dans un dépôt.

(d) Récupération de paire de clé : En option, les clés du client (ex : la clé privée de l'utilisateur) peut être sauvegardée par une CA ou un système de sauvegarde de clé. Si un utilisateur a besoin de récupérer ces clés (par exemple lors de la perte du mot de passe ou la perte de la clé), un protocole d'échange on-line peut être nécessaire pour de telles récupérations.

(e) Mise à jours de paires de clé : toutes les paires de clé ont besoin d'être mises à jours régulièrement, par ex. remplacées avec une nouvelle paire de clé, et de nouveaux certificats fournis.

(f) Demandes de révocation : Une personne autorisée avertit une CA d'une situation anormale nécessitant la révocation d'un certificat.

(g) cross-certification : 2 CA échangent des informations utilisées pour établir une certification croisée. Un cross-certificat est un certificat fournis par une CA à une autre qui contient une clé de signature CA utilisée pour fournir des certificats.

Noter que les protocoles on-line ne sont pas la seul manière d'implémenter les fonction ci-dessus. Pour toutes les fonctions, il y a des méthodes off-line pour accomplir de même résultat, et cette spécification ne mandate pas l'utilisation des protocoles on-line. Par exemple, quand des jetons hardware sont utilisés, beaucoup de fonctions peuvent être faites comme partie de la livraison du jeton physique. De plus, certaines fonctions ci-dessus peuvent être combinées dans un seul protocole d'échange. En particulier, les fonctions d'enregistrement, d'initialisation et de certification peuvent être combinés en un seul protocole d'échange.

Les séries PKIX des spécifications définissent un jeu de formats de messages standard pour supporter les fonction ci-dessus. Les protocoles pour transporter ces messages dans différents environnement (ex : email, transfert de fichier, et www) sont décris dans ces spécifications.

Profile de certificat et d'extensions de certificat

Cette section présente un profile pour les certificats à clé publique qui favorisent l'interopérabilité et une PKI ré-utilisable. Cette section est basée sur le format de certificat X.509 v3 et les extensions de certificats standards définis dans X.509. Les document ISO/IEC et ITU-T utilisent la version 1997 de ASN.1 ; alors que ce document utilise la syntaxe ASN.1 1988, le certificat et les extensions standard encodés sont équivalents. Cette section définis également des extensions privées nécessaires pour supporter une PKI pour la communauté Internet.

Les certificats peuvent être utilisés dans une grande variété d'applications et d'environnements couvrant un large spectre d'interopérabilité et de pré-requis opérationnel et d'assurance. Le but de ce document est d'établir une base commune pour des applications générique nécessitant une grande interopérabilité et des besoins spéciaux limités. En particulier, l'accent sera mis sur le support de l'utilisation des certificats X.509 v3 pour les messages électronique, IPSec, et les applications web.

Champs de certificat de base

La syntaxe de base des certificats X.509 v3 est comme suit. Pour le calcule de la signature, les données à signer sont encodés en utilisant DER. L'encodage DER ASN.1 est un système d'encodage de tag, longueur, et valeur pour chaque élément.

```
Certificate ::= SEQUENCE {
 tbsCertificate TBSCertificate,
 signatureAlgorithm AlgorithmIdentifier,
 signatureValue BIT STRING }

TBSCertificate ::= SEQUENCE {
 version [0] EXPLICIT Version DEFAULT v1,
 serialNumber CertificateSerialNumber,
 signature AlgorithmIdentifier,
 issuer Name,
```

```
 validity Validity,
 subject Name,
 subjectPublicKeyInfo SubjectPublicKeyInfo,
 issuerUniqueID [1] IMPLICIT UniqueIdentifier OPTIONAL,
  - Si présent, version doit être v2 ou v3.
 subjectUniqueID [2] IMPLICIT UniqueIdentifier OPTIONAL,
  - Si présent, version doit être v2 ou v3
 extensions [3] EXPLICIT Extensions OPTIONAL
  - Si présent, version doit être v2 ou v3
}

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

CertificateSerialNumber ::= INTEGER

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

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

UniqueIdentifier ::= BIT STRING

SubjectPublicKeyInfo ::= SEQUENCE {
 algorithm AlgorithmIdentifier,
 subjectPublicKey BIT STRING }

Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension

Extension ::= SEQUENCE {
 extnID OBJECT IDENTIFIER,
 critical BOOLEAN DEFAULT FALSE,
 extnValue OCTET STRING
   - contient l'encodé DER d'une valeur ASN.1 correspondant au type d'extension identifié par extnID
}
```

Champs de certificat

Certificate est une séquence de 3 champs requis. Les champs sont décris en détail dans les section suivantes.

tbsCertificate Le champ contient les noms du sujet et du fournisseur, une clé publique associée avec le sujet, une période de validité, et d'autres informations associées. Les champs sont décris plus bas. tbsCertificate inclue généralement des extensions, qui sont décris plus bas également.

signatureAlgorithm Le champ signatureAlgorithm contient l'identifiant pour l'algorithme cryptographique utilisé par la CA pour signer ce certificat. les rfc 3279, rfc 4055, et rfc 4491 listent les algorithmes de signature supportés, mais d'autres algorithmes peuvent être également supportés.

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

```
AlgorithmIdentifier ::= SEQUENCE {
 algorithm OBJECT IDENTIFIER,
 parameters ANY DEFINED BY algorithm OPTIONAL }
```

L'identifiant d'algorithme est utilisé pour identifier un algorithme cryptographique. L'identifiant d'objet identifie l'algorithme (tel que DSA avec SHA-1). Le contenu des paramètres optionnels varient en fonction de l'algorithme identifié. Ce champ doit contenir le même identifiant d'algorithme que le champ signature dans la séquence tbsCertificate.

signatureValue Le champ signatureValue contient une signature numérique calculée sur l'encodé DER ASN.1 de tbsCertificate, qui est utilisé en entrée de l'algorithme de la fonction de signature. Cette valeur de signature est encodée en chaîne de bits et inclus dans le champ signature. Les détails de ce processus sont spécifiés pour chaque algorithme listés dans les rfc3279, rfc4055, et rfc4491.

En générant cette signature, une CA certifie la validité de l'information dans le champ tbsCertificate. En particulier, la CA certifie la liaison entre la clé publique et le sujet du certificat.

TBSCertificate

La séquence TBSCertificate contient les informations associées avec le sujet du certificat et la CA qui l'a fournie. Tous TBSCertificate contient les noms du sujet et du fournisseur, une clé publique associée avec le sujet, une période de validité, un numéro de version, et un numéro de série ; certains peuvent optionnellement contenir des champs d'identifiant unique. Le reste de cette section décris la syntaxe et les sémantiques de ces champs. Un TBSCertificate inclus généralement des extensions. Les extensions pour les PKI de l'Internet sont décris plus bas.

Version

Ce champ décris la version du certificat encodé. Quand les extensions sont utilisée, comme prévu dans ce profile, la version doit être 3 (valeur 2). Si aucune extension n'est présente, mais qu'un UniqueIdentifier est présent, la version devrait être 2 (valeur 1) ; cependant, la version peut être 3. Si seul les champs de base sont présents, la version devrait être 1 (la valeur est omise du certificat) ; cependant, la version peut être 2 ou 3.

Les implémentations devraient être préparés à accepter n'importe quel numéro de version. Au minimum, les implémentations conformes doivent reconnaître la version 3. La génération de certificats version 2 n'est pas prévue pour les implémentations de ce profile.

Serial Number

Le numéro de série doit être un entier positif assigné par la CA pour chaque certificat. Il doit être unique pour chaque certificat fournis par une CA donnée (ex : le nom du fournisseur et le numéro de série identifie un certificat de manière unique). Les CA doivent forcer le serialNumber à être un entier non-négatif.

Les numéros de série peuvent être prévus pour contenir des entiers long. Les utilisateurs de certificat doivent être capable de manipules des valeurs jusqu'à 20 octets. Les CA conformes ne doivent pas utiliser de valeur serialNumber supérieur à 20 octets.

Note : les CA non-conforme peuvent fournir des certificats avec des numéro de série négatifs ou 0. Les utilisateurs de certificat devraient être préparés à manipuler de tels certificats.

Signature

Ce champ contient l'identifiant d'algorithme pour 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 Certificate. Le contenu du champ optionnel parameters varie en fonction de l'algorithme.

Issuer

Le champ issuer identifie l'entité qui a signé et fournis le certificat. Le champ issuer doit contenir un dn non-vide. Le champ issuer est définis comme nom de type X.501. Le nom est définis par les structures ASN.1 suivantes :

```
Name ::= CHOICE { - un seul possible -
 rdnSequence RDNSequence }

RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

RelativeDistinguishedName ::=
 SET SIZE (1..MAX) OF AttributeTypeAndValue

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

AttributeType ::= OBJECT IDENTIFIER

AttributeValue ::= ANY - DEFINED BY AttributeType

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

Le nom décris un nom hiérarchique composé d'attributs, tel que le nom du pays, et les valeurs correspondantes, tel que US. Le type du composant AttributeValue est déterminé par le AttributeType ; en général il sera un DirectoryString.

Le type DirectoryString est définis comme choix de PrintableString, TeletexString, BMPString, UTF8String, et UniversalString. Les CA conformes à ce profile doivent utiliser l'encodage de DirectoryString soit PrintableString soit UTF8String, avec 2 exceptions. Quand les CA on précédemment fournis des certificats avec les champs issuer avec attributs encodés en utilisant TeletexString, BMPString, ou UniversalString, alors la CA peut continuer à utiliser ces encodages de DirectoryString pour préserver la compatibilité en arrière. Également, les nouvelles CA qui sont ajoutée à un domaine où des CA existantes fournissent des certificat avec les champs issuer avec des attribut utilisant TeletexString, BMPString, ou UniversalString peuvent encoder les attributs qu'ils partagent avec les CA existantes en utilisant les même encodages que les CA existantes.

Comme mentionné plus haut, les dn sont composés d'attributs. Cette spécification ne restreint pas le jeu de type d'attribut qui peuvent apparaître dans le nom. Cependant, les implémentations conformes doivent être préparées à recevoir des certificats avec des noms de fournisseur contenant le jeu de type d'attributs définis ci-dessous. Cette spécification recommande le support pour des type d'attributs additionnels.

Les jeux standard d'attributs ont été définis dans X.520. Les Implémentations de cette spécification doit être préparée à recevoir les types d'attribut standard suivant dans le noms de fournisseur et du sujet :

- country
- organization
- organizational unit
- distinguished name qualifier
- state or province name
- common name (ex : "Susan Housley")
- serial number

En plus, les implémentations de cette spécification devraient être préparés à recevoir le type d'attributs standard suivant dans les noms de fournisseur et de sujet :

- locality
- title
- surname
- given name
- initials
- pseudonym
- generation qualifier (ex : "Jr.", "3rd", ou "IV")

La syntaxe et les identifiant d'objets associés pour ces types d'attribut sont fournis dans les modules ASN.1 dans l'appendice A. En plus, les implémentations de cette spécification doivent être préparées à recevoir l'attribut domainComponent, comme définis dans la rfc4519. Le DNS founis un système de label de ressource hiérarchique. Cet attribut fournis un mécanisme pour les organisations qui souhaitent utiliser les DN en parallèle à leurs noms DNS. Ce n'est pas un remplacement pour le composant dNSName des extensions de nom alternative. Les implémentations ne sont pas obligés de convertir de tels noms en noms DNS. La syntaxe et OID associé pour cet type d'attribut sont fournis dans les modules ASN.1 dans l'appendice 1. Les règles pour l'encodage des noms de domaines internationalisés à utiliser avec le type d'attribut domainComponent sont spécifiés plus bas.

Les utilisateurs de certificat doivent être préparés à traiter les champs issuer et subject pour effectuer un chaînage de nom pour la validation de chemin de certification. Le chaînage de noms est effectué en faisant correspondre le dn issuer dans un certificat avec le sujet dans un certificat CA. Les règles pour comparer les dn sont spécifiés plus loin dans ce document. Si les noms dans les champ issuer et subject dans un certificat correspond aux règles spécifiés plus bas, le certificat est auto-signé.

Validity

La période de validité de certificat est un intervalle de temps durant lequel la CA garantie qu'elle va maintenir les informations sur le statut du certificat. Le champ est représenté comme séquence de 2 dates : la date à laquelle la période de validité commence (notBefore) et la date à laquelle la période de validité se termine (notAfter). Ces 2 dates peuvent être encodés en UTCTime ou GeneralizedTime.

Les CA conformes à ce profile doivent toujours encoder les dates de validité de certificat jusqu'à l'année 2049 en UTCTime ; les de date en 2050 ou ultérieur en GeneralizedTime. Les applications conformes doivent être capable de traiter les dates de validité qui sont encodés en UTCTime ou en GeneralizedTime.

La période de validité pour un certificat est la période de temps depuis notBefore jusqu'à notAfter, inclus.

Dans certaines situations, les périphériques donnent des certificats pour lesquels aucune date d'expiration correcte ne peut être assignée. Par exemple, un périphérique pourrait fournir un certificat qui lie son modèle et un numéro de série à sa clé publique ; un tel certificat est prévu pour être utilisé pour toute la durée de vie du périphérique.

Pour indiquer qu'un certificat n'a pas de date d'expiration, notAfter devrait être assigné à la valeur GeneralizedTime 99991231235959Z.

Quand le fournisseur n'est pas capable de maintenir les informations de statut jusqu'à la date notAfter (incluant le cas où notAfter vaut 99991231235959Z), le fournisseur doit s'assurer qu'aucun chemin de certification valide n'existe pour le certificat après que la maintenance du statut d'information soit terminée. Cela peut être fait par l'expiration ou la révocation de tous les certificats CA contenant la clé publique utilisée pour vérifier la signature dans le certificat et d'arrêter d'utiliser la clé publique utilisée pour vérifier la signature dans le certificat.

UTCTime

Le type de temps universelle UTCTime, est un type ASN.1 standard prévu pour représenter les dates et le temps. UTCTime spécifie l'année via 2 chiffres et le temps est spécifié avec une précision d'une minute ou une seconde. UTCTime inclus soit Z (pour Zulu, ou Greenwich Mean Time) ou un temps différentiel.

Pour ce profile, les valeurs UTCTime doivent être exprimées en GMT (Zulu) et doivent inclurent les secondes (YYMMDDHHMMSSZ), même que le nombre de secondes est 0. Les systèmes conforme doivent interpréter le champ année comme suit :

- Si YY est supérieur ou égale à 50, l'année devrait être interprété comme 19YY.
- Si YY est inférieur à 50, l'année dervait être interprété comme 20YY.

GeneralizedTime

Le type de temps généralisé, GeneralizedTime, est un type ASN.1 standard pour des représentation à précision variable du temps. Optionnellement, le champ GeneralizedTime peut inclure une représentation du temps différentiel entre GMT et la local.

Pour ce profile, les valeurs GeneralizedTime doivent être exprimées en GMT (Zulu) et doivent inclure les secondes (YYYYMMDDHHMMSSZ), même si le nombre de secondes est 0. Les valeurs GeneralizedTime ne doivent pas inclure les fractions de secondes.

Subject

Le champ subject identifie l'entité associée avec la clé publique stockée dans le champ de clé publique. Le nom du sujet peut être placé dans le champ subject et/ou dans l'extension SubjectAltName. Si le sujet est une CA (ex : l'extension de contraintes de base et présent et la valeur de cA est TRUE), alors le champ subject doit être renseigné avec un dn non-vide corrspondant au contenu du champs issuer dans tous les certificats émis par la CA. Si le sujet est un fournisseur de CRL (ex : l'extension utilisation de clé est présent et la valeur de cRLSign est TRUE), alors le champ subject doit être populé avec un dn non-vide correspondant au contenu du champ issuer dans toutes les CRL émise par le fournisseur de CRL. Si les informations de nommage du sujet sont présent seulement dans l'extension subjectAltName (ex : une clé liée seulement à une adresses email ou une URI), alors le sujet doit être une séquence vide et l'extension subjectAltName doit être critique.

Quand il est non-vide, le champ subject doit contenir un dn X.500. Le dn doit être unique pour chaque entité sujet certifié par une CA comme définie dans le champ issuer. Une CA peut fournir plus d'un certificat avec le même dn à la même entité sujet.

Le champ subject est définis comme nom de type X.501. Les pré-requis d'implémentation pour ce champ sont ceux définis pour le champ issuer. Les implémentations de cette spécification doivent être préparés à recevoir des noms de sujet contenant les types d'attributs requis pour le champ issuer. Les implémentations de cette spécification devraient être préparés à recevoir des noms de sujet contenant les types d'attributs recommandés pour le champ issuer. La syntaxe et les identifiant d'objet associés pour ces types d'attributs sont fournis dans les modules ASN.1 en appendice 1.

Les implémentations de cette spécification peuvent utiliser les règles de comparaison pour traiter les types d'attributs non-familier (ex : pour le chaînage de noms) dont les valeurs d'attributs utilisent une des options d'encodage de DirectoryString. Les comparaisons binaires devraient être utilisées quand les type d'attribut non-familiers incluent des valeurs d'attribut avec des options d'encodage autre que ceux trouvés dans DirectoryString. Cela permet aux implémentations de traiter les certificats avec des attributs non-familier dans le sujet.

En encodant des valeurs d'attribut de type DirectoryString, les CA conformes doivent utiliser PrintableString ou UTF8String, avec les exceptions suivant :

(a) Quand le sujet du certificat est une CA, le champ subject doit être encodé de la même manière que s'il est encodé dans le champ issuer dans tous les certificat qu'elle fournis. Donc, si la CA encode les attributs dans les champs issuer des certificat qu'elle fournis en utilisant TeletexString, BMPString, ou UniversalString, alors le champ subject des certificats fournis par cette CA doit utiliser le même encodage.

(b) Quand le sujet du certificat est un fournisseur de CRL, le champ subject doit être encodé de la même manière qu'il est encodé dans le champ issuer dans toutes les CRL fournies par le fournisseur de CRL.

(c) TeletexString, BMPString, et UniversalString sont inclus pour la compatibilité en arrière, et ne devraient pas être utilisés pour les certificats pour les nouveaux sujets. Cependant, ces types peuvent être utilisés dans les certificats où le nom a été précédemment établit, incluant les cas dans lequel un nouveau certificat est fournis à un nouveau sujet où les attributs à encoder ont été

précédemment établis dans les certificats fournis à d'autres sujets. Les certificats utilisateur devraient être préparés à recevoir les certificats avec ces types.

Les implémentations anciennes existent où une adresse de messagerie électronique est embarquée dans le sujet comme emailAddress (rfc2985). La valeur d'attribut pour emailAddress est de type IA5String pour permettre l'inclusion du caractère '@' qui ne fait pas partie du jeu de caractère PrintableString. Les valeurs de l'attribut emailAddress sont sont pas sensibles à la casse.

Les implémentations conformes générant de nouveaux certificat avec des adresse email doivent utiliser le rfc822Name dans l'extension de nom alternative du sujet pour décrire de telles identités. L'inclusion simultanée de l'attribut emailAddress dans le sujet pour supporter les anciennes implémentations est dépréciée, mais permise.

Subject Public Key Info

Ce champ est utilisé pour gérer la clé publique et identifier l'algorithme avec lequel la clé est utilisée (ex : RSA, DSA, ou DH). L'algorithme est identifié en utilisant la structure AlgorithmIdentifier. Les identifiant d'objet pour les algorithmes supportés et les méthodes pour l'encodage de la clé publique (clé publique et paramètres) sont spécifiés dans les rfc3279, rfc4055, et rfc4491.

Unique Identifiers

Ces champs doivent seulement apparaître si la version est 2 ou 3. Les identifiant uniques du sujet et du fournisseur sont présent dans le certificat pour gérer la possibilité de réutiliser les noms du sujet et/ou fournisseur dans le temps. Ce profile recommande que les noms ne soient pas réutilisés pour différentes entités et que les certificats Internet n'utilisent pas les identifiant uniques. Les CA conformes à ce profile ne doivent pas générer de certificats avec des identifiants uniques. Les applications conformes à ce profile doivent être capable de lire les certificats qui incluent des identifiants uniques, mais il n'y a pas de pré-requis de traitement associés avec les identifiant uniques.

Extensions

Ce champ doit seulement apparaître si la version est 3. Si présent, ce champ est une séquence d'une ou plusieurs extensions de certificat.

Extensions de certificat

Les extension définies pour les certificats X.509 v3 fournissent une méthode pour associer des attributs additionnels avec les utilisateurs ou les clés publiques et pour gérer les relations entre les CA. Le format de certificat X.509 v3 permet également aux communautés de définir des extensions privées pour gérer l'unicité des informations de ces communautés. Chaque extension dans un certificat peut être soit critique soit non-critique. Un système utilisant les certificats doit rejeter le certificat s'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 section suivantes présentent les extensions recommandées utilisée avec les certificats Internet et les emplacements standards pour les informations.

Les communautés peuvent choisir d'utiliser des extensions additionnels, cependant, la prudence doit être de mise en adoptant des extension critiques dans les certificats qui peuvent empêcher leur utilisation dans un contexte général.

Chaque extension inclus un OID et une structure ASN.1. Quand une extension apparaît dans un certificat, l'OID apparaît comme champ extnID et l'a structure encodée DER ASN.1 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. Une extension inclus le booléen critical, avec une valeur par défaut à FALSE. Le texte pour chaque extension spécifie les valeurs acceptable pour le champ critical pour les CA conformes à ce profile.

Les CA conformes doivent supporter les identifiant de clé, les contraintes de base, l'utilisation de clé, et les stratégie de certificat. Si les CA fournissent des certificats avec une séquence vide pour le champ sujet, la CA doit supporter l'extension de noms alternatifs du sujet. Le support pour les extensions restantes est optionnel. Les CA conformes peuvent supporter les extension qui ne sont pas identifiés dans cette spécification ; les fournisseurs de certificats sont prévenus que marquer de telles extensions comme critique peut inhiber l'interopérabilité.

Au minimum, les applications conformes à ce profile doivent reconnaître les extensions suivantes : utilisation de clé, stratégie de certificat, contraintes de base, contraintes de nom, contraintes de stratégie, utilisation de clé étendue, et inhibit anyPolicy. En plus, les applications conformes à ce profile devraient reconnaître les extensions d'identifiant de clé du sujet et de l'autorité, et les mappages de stratégie.

Extensions standard

Cette section identifie les extensions de certificat standard définis dans X.509 pour l'utilisation dans les PKI de l'Internet. Chaque extension est associée avec un OID définis dans X.509. Ces OID sont membres de id-ce arc, qui est définis :

```
id-ce OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) ds(5) 29 }
```

Authority Key Identifier

L'extension d'identifiant de clé de l'autorité fournis un moyen d'identifier la clé publique correspondant à la clé privée utilisée pour signer un certificat. Cette extension est utilisée où un fournisseur a plusieurs clé de signature (soit dû à plusieurs paires de clé concurrentes, soit à cause d'un changement de clé). L'identification peut être basée sur l'identifiant de clé (le subject key identifier dans le certificat du fournisseur) ou le nom du fournisseur et un numéro de série.

Le champ keyIdentifier de l'extension authorityKeyIdentifier doit être inclus dans tous les certificats générés par les CA conformes pour faciliter la construction de chemin de certification. Il y a une exception : là où une CA distribue sa clé publique sous la forme d'un certificat auto-signé, l'identifiant de clé d'autorité peut être omise. La signature d'un certificat auto-signé est générée avec la clé privée associée avec la clé publique du sujet du certificat. Dans ce cas, le sujet et les identifiants de clé d'autorité seraient identiques, mais seul l'identifiant de clé du sujet est nécessaire pour construire le chemin de validation.

La valeur du champ keyIdentifier devrait être dérivé de la clé publique utilisée pour vérifier la signature du certificat ou une méthode qui génère des valeurs uniques. 2 méthodes communes pour générer des identifiant de clé depuis une clé publique sont décris dans la section suivante. Là où un identifiant de clé n'a pas été précédemment établis, cette spécification recommande l'utilisation d'une de ces méthodes pour générer les keyIdentifiers ou l'utilisation de méthodes similaires qui utilisent un algorithme de hashage différent. Là où un identifiant de clé a été précédemment établis, la CA devrait utiliser l'identifiant établis précédemment.

Ce profile recommande le support pour la méthode d'identifiant de clé par tous les utilisateurs de certificat. Les CA conformes doivent marquer cette extension comme non-critique.

```
id-ce-authorityKeyIdentifier OBJECT IDENTIFIER ::= { id-ce 35 }

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

KeyIdentifier ::= OCTET STRING
```

Subject Key Identifier

L'extension d'identifiant de clé du sujet fournis un moyen d'identifier les certificats qui contiennent une clé publique particulière.

Pour simplifier la construction de chemin de certification, cette extension doit apparaître dans tous certificats CA conformes, c'est à dire, tous les certificats incluant l'extension de contraintes de base où la valeur de cA est TRUE. Dans les certificats de CA conformes, la valeur de l'identifiant de clé du sujet doit être la valeur placée dans le champ d'identifiant de clé de l'extension de clé d'autorité des certificats fournis par le sujet de ce certificat. Les applications ne sont pas obligées de vérifier que les identifiants de clé correspondent en validant le chemin de certification.

Pour les certificats de CA, les identifiants de clé du sujet devraient être dérivés de la clé publique ou une méthode qui génère des valeurs uniques. 2 méthodes communes pour générer des identifiant de clé depuis la clé publique sont :

(1) keyIdentifier est composé d'un hash SHA-1 160-bits de la valeur de la chaîne de bits subjectPublicKey (excluant le tag, longueur et le nombre de bits non-utilisé).

(2) keyIdentifier est composé d'un champ de type 4-bits avec la valeur 0100, suivis par au moins 60bits de hash SHA-1 de la valeur de la chaîne de bits subjectPublicKey (excluant le tag, longueur et le nombre de bits non-utilisé).

D'autres méthodes pour générer des nombres uniques sont également acceptables.

Pour un certificat d'entité finale, l'extension d'identifiant de clé du sujet fournis un moyen d'identifier les certificats contenant la clé publique particulière utilisée dans une application. Quand une entité finale a obtenu plusieurs certificats, spécialement depuis plusieurs CA, l'identifiant de clé du sujet fournis un moyen de rapidement identifier le jeu de certificats contenant une clé publique particulière. Pour assister les applications à identifier le bon certificat, cette extension devrait être incluse dans tous les certificats d'entité finale.

Pour les certificats d'entité finale, les identifiants de clé du sujet devraient être dérivés de la clé publique. 2 méthodes communes pour générer des identifiants de clé depuis la clé publique sont identifiés ci-dessus.

Quand un identifiant de clé n'a pas été établis précédemment, cette spécification recommande l'utilisation d'une de ces méthodes pour générer les keyIdentifiers ou utiliser une méthodes similaire qui utilise un algorithme de hash différent. Quand un identifiant de clé a été précédemment établis, la CA devrait utilise cet identifiant.

Les CA conformes doivent marquer cette extension comme non-critique.

```
id-ce-subjectKeyIdentifier OBJECT IDENTIFIER ::= { id-ce 14 }

SubjectKeyIdentifier ::= KeyIdentifier
```

Key Usage

L'extension d'utilisation de clé définis le but (ex : le chiffrement, la signature, signature de certificat) de la clé contenue dans le certificat. La restriction d'utilisation peut être employée quand une clé qui pourrait être utilisée pour plus d'une opération doit être restreindre. Par exemple, quand une clé RSA devrait être utilisée seulement pour vérifier les signatures sur des objets autre que des certificats à clé publique et les CRL, les bits digitalSignature et/ou nonRepudiation devraient être mis. De même, quand une clé RSA devrait être utilisée seulement pour la gestion de clé, le bit keyEncipherment devrait être mis.

Les CA conformes doivent inclure cette extension dans les certificats qui contiennent des clés publiques qui sont utilisée pour valider les signatures numériques dans d'autre certificats à clé publique ou des CRL. Quand il est présent, les CA conformes devraient marquer cette extension comme critique.

```
id-ce-keyUsage OBJECT IDENTIFIER ::= { id-ce 15 }

KeyUsage ::= BIT STRING {
 digitalSignature (0),
 nonRepudiation (1), - Les éditions récentes de X.509 ont renommé ce bis en contentCommitment
 keyEncipherment (2),
 dataEncipherment (3),
 keyAgreement (4),
 keyCertSign (5),
```

```
cRLSign (6),
encipherOnly (7),
decipherOnly (8) }
```

digitalSignature est mis quand la clé publique du sujet est utilisée pour vérifier les signatures numériques, autre que les signature des certificats et des CRL, telles que celles utilisées dans les services d'authentification d'entité, services d'authentification de l'origine des données, et/ou services d'intégrité.

nonRepudiation est mis quand la clé publique du sujet est utilisée pour vérifier les signatures numérique, autre que les signatures des certificats et des CRL, utilisé pour fournir un service de non-répudiation qui protège contre l'entité signataire niant à tord certaines actions. Dans le cas de conflits ultérieurs, un tiers de confiance peut déterminer l'authenticité des données signées.

keyEncipherment est mis quand la clé publique du sujet est utilisée pour chiffrer des clés secrètes ou privée, par exemple, pour le transport de clé. Par exemple, ce bit devrait être mis quand une clé publique RSA est utilisée pour chiffrer une clé de déchiffrement de contenu symétrique ou une clé privée asymétrique.

dataEncipherment est mis quand la clé publique du sujet est utilisée pour chiffrer directement les donnée brutes de l'utilisateur sans l'utilisation de clé symétrique intermédiaire. Noter que l'utilisation de ce bit est extrêmement peu commun ; la plupart des applications utilisent le transport de clé pour l'agrément de clé pour établir une clé symétrique.

keyAgreement est mis quand la clé publique est utilisée pour la gestion de clé. Par exemple, quand une clé Diffie-Hellman est utilisée pour la gestion de clé, alors ce bit est mis.

keyCertSign Est mis quand la clé publique est utilisée pour vérifier les signatures des certificats à clé publique. Si keyCertSign est mis, alors le bit cA dans l'extension de contrainte de base doit également être mis.

cRLSign est mis quand la clé publique est utilisée pour vérifier les signatures dans les listes de révocation de certificat.

encipherOnly la signification de ce bit est indéfinis en l'absence du bit keyAgreement. Quand ces 2 bits sont mis, la clé publique du sujet peut être utilisée seulement pour chiffrer les données en effectuant un agrément de clé.

decipherOnly la signification de ce bit est indéfinis en l'absence du bit keyAgreement. Quand ces 2 bits sont mis, la clé publique du sujet peut être utilisée seulement pour déchiffrer les données en effectuant un agrément de clé.

Si l'extension keyUsage est présent, alors la clé publique du sujet ne doit pas être utilisée pour vérifier les signatures dans le certificats ou les CRL à moins que les bits correspondants ne soient mis. Si la clé publique du sujet est seulement utilisée pour vérifier les signatures dans les certificats et/ou les CRL, alors les bits digitalSignature et nonRepudiation ne devraient pas être mis. Cependant, digitalSignature et/ou nonRepudiation peuvent être mis en plus de keyCertSign et/ou cRLSign si la clé publique du sujet est utilisée pour vérifier les signatures dans les certificats et/ou les CRL et également dans d'autres objets.

Combiner le bit nonRepudiation dans l'extension de certificat keyUsage avec d'autres bits d'utilisation de clé peut avoir des implications de sécurité en fonction du contexte dans lequel le certificat est utilisé. D'autres distinction entre digitalSignature et nonRepudiation peuvent être fournis dans des stratégies de certificat spécifiques.

Ce profil ne restreins pas les combinaisons de bits qui peuvent être mis dans une extension keyUsage. Cependant, des valeurs appropriées pour keyUsage pour des algorithmes particuliers sont spécifiés dans les rfc3279, rfc4055, et rfc4491. Quand l'extension keyUsage apparaît dans un certificat, au moins un bit doit être mis.

Certificate Policies

L'extension de stratégie de certificat contient une séquence d'un ou plusieurs termes d'informations de stratégie, chacun consistant d'un identifiant d'objet et d'un qualifiant optionnel. Les qualifiants optionnel, qui peuvent être présents, ne sont pas prévus pour changer la définition de la stratégie. Un OID de stratégie de certificat ne doit pas apparaître plus d'une fois dans une extension de stratégie de certificat.

Dans un certificat d'entité finale, ces informations de stratégie indique la stratégie sous laquelle le certificat a été émis et le but pour lequel le certificat peut être utilisé. Dans un certificat CA, ces stratégies limitent le jeu de stratégie pour les chemins de certification qui incluent ce certificat. Quand une CA ne souhaite pas limiter le jeu de stratégies pour les chemins de certification qui incluent ce certificat, elle peut indiquer une stratégie spéciale anyPolicy, qui a la valeur { 2 5 29 32 0 }.

Les applications avec des besoins de stratégie spécifiques sont censés avoir une liste de ces stratégies qu'elles vont accepter et pour comparer les OID de stratégie 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 qualifiant optionnel), ou doivent rejeter le certificat.

Pour promouvoir l'interopérabilité, ce profile recommande que les termes d'informations de stratégie consistent de seulement un OID. Quand un OID seul n'est pas suffisant, ce profile recommande fortement que l'utilisation des qualifiants soient limités à ceux identifiés dans cette section. Quand les qualifiants sont utilisé avec le stratégie spéciale anyPolicy, ils doivent être limités aux qualifiants identifés dans cette section. Seul ces qualifiants retournés en résultat de la validation de chemin sont considérés.

Cette spécification définis 2 types de qualifiants de stratégie à utiliser avec les concepteurs de stratégie de certificat. Les types de qualifiants sont le CPS Pointer et User Notice.

Le CPS Pointer contient un pointer vers un Certification Practice Statement (CPS) publié par la CA. Le pointeur est sous la forme d'un URI. Les pré-requis de traitement pour ce qualifiant sont définis localement. Aucune action n'est mandatée par cette spécification sans regarder la criticité de cette extension.

User notice est prévu pour afficher un tier de confiance quand un certificat est utilisé. Seuls les consignes utilisateurs retournés en résultat de la validation de chemin sont prévus pour l'affichage à l'utilisateur. Si une consigne est dupliqué, seul une copie est affichée. Pour empêcher une telle duplication, ce qualifiant devrait seulement être présent dans les certificats d'entité finaux et les certificats de CA fournis à d'autres organisations.

La consigne utilisateur a 2 champs optionnels : noticeRef et explicitText. Les CA conformes ne devraient pas utiliser l'option noticeRef.

Le champ noticeRef, si 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 un numéro de consigne 1. Dans une implémentation typique, l'application aura un fichier de consigne contenant le jeu de textes dans un fichier et l'affichera. Les messages peuvent être multi-langue.

Le champ explicitText inclus la déclaration textuel directement dans le certificat. Ce champ est une chaîne avec un maximum de 200 caractères. Les CA conformes devraient utiliser l'encodage UTF8String, mais peuvent utiliser IA5String. Les CA conformes ne doivent pas encoder explicitText en VisibleString ou BMPString. La chaîne explicitText ne devrait pas inclure de caractères de contrôle. Encodé en UTF8String, toutes les séquences de caractère devraient être normalisés en accord avec Unicode Normalization form C (NFC).

Si les 2 options sont inclues dans un qualifiant et si l'application peut localiser le texte indiqué par noticeRef, alors ce texte devrait être affiché ; sinon, la chaîne explicitText devrait être affichée.

Note : bien que explicitText a une taille maximum de 200 caractères, certaines CA non-conformes dépassent cette limite. Cependant, les utilisateurs de certificat devrait gérer ce champ avec plus de 200 caractères.

```
id-ce-certificatePolicies OBJECT IDENTIFIER ::= { id-ce 32 }

 anyPolicy OBJECT IDENTIFIER ::= { id-ce-certificatePolicies 0 }

 certificatePolicies ::= SEQUENCE SIZE (1..MAX) OF PolicyInformation

 PolicyInformation ::= SEQUENCE {
  policyIdentifier CertPolicyId,
  policyQualifiers SEQUENCE SIZE (1..MAX) OF
   PolicyQualifierInfo OPTIONAL }

 CertPolicyId ::= OBJECT IDENTIFIER

 PolicyQualifierInfo ::= SEQUENCE {
  policyQualifierId PolicyQualifierId,
  qualifier ANY DEFINED BY policyQualifierId }

 - policyQualifierIds for Internet policy qualifiers
```

```
id-qt OBJECT IDENTIFIER ::= { id-pkix 2 }
id-qt-cps OBJECT IDENTIFIER ::= { id-qt 1 }
id-qt-unotice OBJECT IDENTIFIER ::= { id-qt 2 }

PolicyQualifierId ::= OBJECT IDENTIFIER ( id-qt-cps | id-qt-unotice )

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

CPSuri ::= IA5String

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

NoticeReference ::= SEQUENCE {
 organization DisplayText,
 noticeNumbers SEQUENCE OF INTEGER }

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

Policy Mappings

Cette extension est utilisée dans les certificats CA. Elle liste une ou plusieurs paires d'OID ; chacune incluant un issuerDomainPolicy et un subjectDomainPolicy. Le jumelage indique que la CA émettrice considère son issuerDomainPolicy équivalent au subjectDomainPolicy de la CA.

Les utilisateurs de la CA émettrice peuvent accepter un issuerDomainPolicy pour certaines applications. Le mappage de stratégie définie la liste des stratégies associées avec le sujet CA qui peuvent être acceptés comme comparable à issuerDomainPolicy.

Chaque issuerDomainPolicy nommé dans l'extension de mappage de stratégies devrait également être mis dans une extension de stratégie de certificat dans le même certificat. Les stratégies ne doivent pas être mappés soit pour ou depuis la valeur spéciale anyPolicy.

En général, les stratégies de certificat qui apparaissent dans le champ issuerDomainPolicy de l'extension de mappage de stratégie ne sont pas considérées comme stratégie acceptable pour l'inclusion dans le certificats sous-jacents dans le chemin de certification. Dans certaines circonstances, une CA peut souhaiter mapper une stratégie (p1) à une autre (p2), mais veut que le issuerDomainPolicy (p1) soit considéré comme acceptable pour l'ajout dans les certificat sous-jacents. Cela peut se produire, par exemple, si la CA est dans le processus de transition d'e l'utilisation de la stratégie p1 vers l'utilisation de la stratégie p2 et a des certificats valides qui ont été émis sous chaque stratégie. Une CA peut indiquer cela en incluant 2 mappage de stratégie dans les certificats CA qu'elle fournis. Chaque mappage de stratégie aura un issuerDomainPolicy de p1 ; un mappage de stratégie aura subjectDomainPolicy de p1 et l'autre aura le subjectDomainPolicy de p2.

Cette extension peut être supportée par les CA et/ou les applications. Les CA conformes devraient marquer cette extension comme critique.

```
id-ce-policyMappings OBJECT IDENTIFIER ::= { id-ce 33 }

PolicyMappings ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {
 issuerDomainPolicy CertPolicyId,
 subjectDomainPolicy CertPolicyId }
```

Subject Alternative Name

L'extension subject alternative name permet aux identités d'être liées au sujet du certificat. Ces identités peuvent être inclues en plus ou à la place e l'identité dans le champ subject du certificat. Les options définies incluent une adresse de messagerie électronique, un nom DNS, une adresse IP, et une URI. D'autres options existent, incluant les définitions complètement locales. Plusieurs formes de nom, et plusieurs instances de chaque forme de nom peuvent être inclus. Quand de telles identités sont liées dans un certificat, l'extension subject alternative name (ou issuer alternative name) doit être utilisée ; cependant, un nom DNS peut également être représenté dans le champ subject en utilisant l'attribut domainComponent. Noter que quand de tels noms sont représentés dans le champ subject, les implémentation ne sont pas obligées de les convertir en noms DNS.

Puisque le nom alternatif du sujet est considéré comme lié définitivement à la clé publique, toutes les parties doivent être vérifiés par la CA.

De plus, si la seul identité du sujet inclue dans le certificat est une forme de nom alternative, alors le nom distinct du sujet doit être vide, et l'extension subjectAltName doit être présent. Si le champ subject contient une séquence vide, alors la CA émettrice doit inclure une extension subjectAltName marquée comme critique. En incluant l'extension subjectAltName dans un certificat qui a un sujet non-vide, les CA conformes devraient marque l'extension subjectAltName comme non-critique.

Quand l'extension subjectAltName contient une adresse email, cette adresse doit être stockée dans le rfc822Name. Le format d'un rfc822Name est un Mailbox comme définis dans la rfc 2821. Une Mailbox a la forme "local-part@Domain". Noter qu'une Mailbox n'a pas de phrase (cet un nom commun) avant lui, et n'est pas entouré de "<" et ">". Les règles d'encodage des adresses email qui incluent des noms de domaines internationalisés sont spécifiés plus bas.

Quand l'extension subjectAltName contient une iPAdress, l'adresse doit être stockée dans la chaîne d'octets dans "network byte order", comme spécifié dans la rfc791. Le LSB de chaque octet est le LSB de l'octet correspondant dans l'adresse réseau. Pour les IPv4, comme spécifié dans la rfc791, la chaîne d'octets doit contenir exactement 4 octets. Pour la version IPv6, comme spécifié dans la rfc2460, la chaîne d'octets doit contenir exactement 16 octets.

Quand l'extension subjectAltName contient un nom de domaine, le nom de domaine doit être stocké dans le dNSName (un IA5String). Le nom doit être en "preferred name syntax", comme spécifié dans la rfc1034 et modifié dans la rfc1123. Noter que bien que les lettres majuscules et minuscules sont permises dans les noms de domaines, la casse n'est pas prise en compte. En plus, alors que la chaîne " " est un nom de domaine légal, les extensions subjectAltName avec un dNSName de " " ne doivent pas être utilisé. Finalement, l'utilisation de la représentation DNS pour les adresses mail ne doivent pas être utilisées, de telles identités sont encodées en rfc822Name. Les règles pour l'encodage des noms de domaine internationalisés sont spécifiés plus bas.

Quand l'extension subjectAltName contient une URI, le nom doit être stocké dans un uniformResourceIdentifier (un IA5String). Le nom ne doit pas être une URI relative, et doit suivre la syntaxe URI et les règles d'encodage spécifiés dans la rfc3986. Le nom doit inclure à la fois un schéma (ex : http ou ftp) et une partie spécifique au schéma. Les URI qui incluent une autorité doivent inclure un nom pleinement qualifié ou une adresse IP comme hôte. Les règles pour l'encodage des Internationalized Resource Identifiers (IRI) sont spécifiées plus bas.

Comme spécifié dans la rfc3986, le schema n'est pas sensible à la casse (ex : http est équivalent à HTTP). La partie hôte, si présent, est également insensible à la casse, mais les autres composant de la partie spécifique au schéma peut être sensible à la casse. Les règles pour comparer les URI sont spécifiées plus bas.

Quand l'extension subjectAltName contient un DN dans le directoryName, les règles d'encodage sont les mêmes qui ceux spécifiés pour le champ issuer. Le DN doit être unique pour chaque entité sujet certifiée par une CA comme définie par le champ issuer. Une CA peut émettre plus d'un certificat vec le même DN au même sujet.

Le subjectAltName peut gérer des types de noms additionnel via l'utilisation du champ otherName. Le format et les sémantiques de nom sont indiqués via l'identifiant d'objet dans le champ type-id. Le nom lui-même est acheminé comme valeur de champ dans otherName. Par exemple un OID de nom principal Kerberos peut être encodé comme séquence du Realm et du PrincipalName.

Les noms alternatifs du sujet peuvent être contraints de la même manière que les noms distinct du sujet en utilisant une extension de contrainte de nom.

Si l'extension subjectAltName est présent, la séquence doit contenir au moins une entrée. À la différence du champ subject, les CA

conformes ne doivent pas fournir de certificats avec un subjectAltName contenant des champs GeneralName vides. Par exemple, un rfc822Name est représenté en IA5String. Alors qu'une chaîne vide est un IA5String valide, un tel rfc822Name n'est pas permis par ce profile. Le comportement des client qui rencontrent un tel certificat en traitant un chemin de certification n'est pas définis par ce profile.

Finalement, les sémantiques des noms alternatifs du sujet qui incluent des caractères wilcards ne sont pas adressés par cette spécification. Les applications avec des requis spécifiques peuvent utiliser de tels noms, mais elles doivent définir les sémantiques.

```
id-ce-subjectAltName OBJECT IDENTIFIER ::= { id-ce 17 }

SubjectAltName ::= GeneralNames

GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName

GeneralName ::= CHOICE {
 otherName [0] OtherName,
 rfc822Name [1] IA5String,
 dNSName [2] IA5String,
 x400Address [3] ORAddress,
 directoryName [4] Name,
 ediPartyName [5] EDIPartyName,
 uniformResourceIdentifier [6] IA5String,
 iPAddress [7] OCTET STRING,
 registeredID [8] OBJECT IDENTIFIER }

OtherName ::= SEQUENCE {
 type-id OBJECT IDENTIFIER,
 value [0] EXPLICIT ANY DEFINED BY type-id }

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

Issuer Alternative Name

Comme pour la section précédente, cette extension est utilisée pour associer des identité de style Internet avec des fournisseurs de certificats. Le nom alternatif du fournisseur doit être encodé comme dans la section précédente. Les noms alternatifs du fournisseur ne sont pas traités comme partie de l'algorithme de validation de chemin de certification. (ils ne sont pas utilisé dans le chaînage et les contraintes de noms ne sont pas forcées). Quand présent, les CA conformes devraient marquer cette extension non-critique.

```
id-ce-issuerAltName OBJECT IDENTIFIER ::= { id-ce 18 }

IssuerAltName ::= GeneralNames
```

Subject Directory Attributes

L'extension d'attributs d'annuaires du sujet est utilisée pour transmettre des attributs d'identification (ex : 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.

```
id-ce-subjectDirectoryAttributes OBJECT IDENTIFIER ::= { id-ce 9 }

SubjectDirectoryAttributes ::= SEQUENCE SIZE (1..MAX) OF Attribute
```

Basic Constraints

L'extension de contraintes de base identifie si le sujet du certificat est une CA et la profondeur maximale des chemins de certification valides qui incluent ce certificat.

Le booléen cA indique si la clé publique certifiée peut être utilisée pour vérifier le signatures de certificat. Si le booléen cA n'est pas présent, alors le bit keyCertSign de l'extension d'utilisation de clé ne doit pas être présent. Si l'extension de contraintes de base n'est pas présent dans un certificat v3, ou l'extension est présent mais le booléen cA n'est pas mis, alors la clé publique certifiée ne doit pas être utilisée pour vérifier les signatures de certificat.

Le champ pathLenConstraint a une signification seulement si cA est mis et l'extension d'utilisation de clé, si présent, à le bit keyCertSign mis. Dans ce cas, il donne le nombre maximum de certificats intermédiaires non auto-fournis qui peuvent suivre ce certificat dans le chemin de certification. (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 finale, mais peut être un certificat de CA). Un pathLenConstraint de 0 indique qu'aucun certificat de CA intermédiaire non auto-fournis ne peut suivre dans un chemin de certification valide. Quand il apparaît, le champ pathLenConstraint doit être supérieur ou égal à 0. Quand pathLenConstraint n'apparaît pas, aucune limite n'est imposée.

Les CA conformes doivent inclure cette extension dans tous les certificats CA qui contiennent une clé publique utilisée pour valider les signatures numérique dans les certificat et doivent marquer cette extension comme critique. Cette extension peut apparaître comme critique ou non dans les certificats CA qui contiennent des clés publiques utilisée exclusivement pour d'autres but que la validation des signatures numérique dans les certificats. De tels certificats de CA incluent ceux qui contiennent des clé publique utilisées exclusivement pour valider les signatures numérique dans les CRL et ceux qui contiennent des clés publique de gestion de clé utilisées avec les protocoles d'inscription de certificat.

Les CA ne doivent pas inclure le champ pathLenConstraint sauf si le booléen cA est mis et que l'extension d'utilisation ed clé a le bit keyCertSign mis.

```
id-ce-basicConstraints OBJECT IDENTIFIER ::= { id-ce 19 }

BasicConstraints ::= SEQUENCE {
 cA BOOLEAN DEFAULT FALSE,
 pathLenConstraint INTEGER (0..MAX) OPTIONAL }
```

Name Constraints

L'extension de contrainte de noms, qui doit être utilisé seulement dans un certificat de CA, indique un espace de noms dans lequel tous les noms de sujet dans les certificats sous-jacent dans un chemin de certification doivent être localisés. Les restrictions d'appliquent aux dn du sujet et aux noms alternatifs du sujet. Les restrictions s'appliquent seulement quand la forme de nom spécifié est présente. Si aucun nom du type n'est dans le certificat, le certificat est acceptable.

Les contraintes de nom ne sont pas appliquées aux certificats auto-fournis (à moins que le certificat est un certificat final dans le chemin). (Cela peut empêcher les CA qui utilisent les contraintes de noms d'employer des certificats auto-fournis pour implémenter du remplacement de clé).

Les restrictions sont définies en termes de subtrees de noms permis ou exclus. Tout nom correspondant à une restriction dans le champ excludedSubtrees est invalide sans regarder les informations apparaîssant dans permittedSubtrees. les CA conformes doivent marquer cette extension comme critique et ne devraient pas imposer de contraintes de nom sur les formes de noms x400Address, ediPartyName, ou registeredID. Les CA conformes ne doivent pas fournir de certificats où les contraintes de nom est une séquence vide, le champ permittedSubtrees ou excludedSubtrees doit être présent.

Les applications conformes à ce profile doivent être capable de traiter les contraintes de noms qui sont imposées dans le forme de nom directoryName et devraient être capable de traiter les contraintes de nom qui sont imposées dans les formes de nom rfc822Name, uniformResourceIdentifier, dNSName, et iPAddress. Si une extension de contraintes de nom qui est marqué critique impose des contraintes sur une forme de nom particulier, et une instance de cette forme de nom apparaît dans le champ subject ou dans l'extension subjectAltName. d'un certificat sous-jacent, alors l'application doit traiter la contrainte ou rejeter le certificat.

Dans ce profile, les champs minimum et maximum ne sont pas utilisé pour les formes de nom, donc, le minimum doit être 0, et maximum doit être absent. Cependant, si une application rencontre une extension de contraintes de noms critique qui spécifie d'autres valeurs pour minimum et maximum pour une forme de nom qui apparaît dans un certificat sous-jacent, l'application doit traiter ces champs ou rejeter le certificat.

Pour les URI, la contrainte s'applique à la partie hôte du nom. La contrainte doit être spécifiée en fqdn et peut spécifier un hôte ou un domaine. Quand la contrainte commance avec un point, elle peut être étendue avec un ou plusieurs labels. Par exemple, ".example.com" est satisfait par host.example.com, et my.host.example.com. Cependant, la contrainte .example.con n'est pas satisfait par example.com. Quand la contrainte ne commence pas avec un point, elle spécifie un hôte. Si une contrainte est appliquée à une forme de nom URI et qu'un certificat sous-jacent inclue une extension subjectAltName avec une URI qui n'inclue pas un composant d'autorité avec un nom d'hôte spécifié en fqdn, alors l'application doit rejeter le certificat.

Une contrainte de nom pour les adresses email de l'Internet peut spécifier une boîte mail particulière, toutes les adresses dans un hôte particulier, ou toutes les boîtes mail dans un domaine. Pour indiquer une boîte mail particulière, la contrainte est l'adresse mail complète. Pour indiquer toutes les adresses mail dans un hôte particulier, la contrainte est spécifiée comme nom d'hôte. Pour spécifier toutes les adresse mail dans un domaine, la contrainte est spécifiée avec un point, comme avec une URI (.example.com indique toutes les adresse mail du domaine example.com, mail pas les adresses internet de l'hôte example.com).

Les restrictions des noms DNS sont exprimées en host.example.com. Tous nom DNS qui peut être construit en ajoutant simplement 0 ou plusieurs labels à la partie gauche du nom satisfont la contrainte de nom. Par exemple, www.host.example.com satisfait la contrainte mais pas host1.example.com.

Les anciennes implémentations existent où une adresse mail est embarquée dans le sujet dans une attribut de type emailAddress. Quand les contraintes sont imposée sur la forme de nom rfc822Name, mais que le certificat n'inclut pas une extension subjecAltName, la contrainte rfc822Name doit être appliquée à l'attribut de type emailAdress dans le sujet. La syntaxe ASN.1 pour emailAddress et l'OID correspondant sont fournis en appendix A.

Les restrictions de la forme directoryName doivent être appliqués au champ subject dans le certificat (quand le certificat inclus un champ de sujet non-vide). et à tous noms de type directoryName dans l'extension subjectAltName. Les restrictions de la form x400Address doivent être appliquées à tous les noms de type x400Address dans l'extension subjectAltName.

La syntaxe de iPAddress doit être comme décris précédemment avec les ajouts suivant spécifiquement pour les contraintes de noms. Pour les adresses IPv4, le champ IPv4 de GeneralName doit contenir 8 cotets, encodés dans le style rfc4632 (CIDR) opur représenter une plage d'adresses. Pour les adresses IPv6, le champ iPAddress doit contenir 32 octets encodés similairement. Par exemple, une contrainte de nom un sous-réseaux de classe C 192.0.2.0 est représenté pas les octets C0 00 02 00 FF FF FF 00, représentant la notation CIDR 192.0.2.0/24.

Des règles additionnelles pour l'encodage et le traitement des contraintes de noms sont spécifiées plus bas. La syntaxe et les sémantiques pour les contraintes de noms pour otherName, ediPartyName, et registeredID ne sont pas définies par cette spécification. Cependant, la syntaxe et les sémantiques pour les contraintes de nom pour d'autres formes de nom peuvent être spécifiés dans d'autres documents.

```
id-ce-nameConstraints OBJECT IDENTIFIER ::= { id-ce 30 }

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

 GeneralSubtrees ::= SEQUENCE SIZE (1..MAX) OF GeneralSubtree

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

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

Policy Constraints

L'extension de contrainte de stratégie peut être utilisée dans les certificats fournis aux CA. Cette extension contraint la validation de chemin de 2 manières. Elle peut être utilisée pour interdire le mappage de stratégie ou nécessiter que chaque certificat dans le chemin contienne un identifiant de stratégie acceptable.

Si le champ inhibitPolicyMapping est présent, la valeur indique le nombre de certificats additionnels qui peuvent apparaître dans le chemin avant que le mappage de stratégie ne soit plus permis. Par exemple, une valeur indique que le mappage de stratégie peut être traitée dans les certificats fournis par le sujet de ce certificat, mais pas dans les certificats additionnels dans le chemin.

Si le champ requireExplicitPolicy est présent, sa valeur indique le nombre de certificats additionnels qui peuvent apparaître dans le chemin avant qu'une stratégie explicite soit requise pour tous le chemin. Quand une stratégie explicite est requise, il est nécessaire pour tous les certificats dans le chemin de contenir un identifiant de stratégie acceptable dans l'extension de stratégie de certificat. Un identifiant de stratégie acceptable est l'identifiant d'une stratégie requise par l'utilisateur du chemin de certification ou l'identifiant d'une stratégie qui a été déclarée équivalente via le mappage de stratégie.

Les applications conformes doivent être capable de traiter le champ requireExplicitPolicy et devraient être capable de traiter le champ inhibitPolicyMapping. Les application qui supportent le champ inhibitPolicyMapping doivent également implémenter le support pour l'extension policyMappings. Si l'extension policyConstraints est marquée critique et que le champ inhibitPolicyMapping est présent, les applications qui n'implémentent pas le support pour inhibitPolicyMapping doivent rejeter le certificat.

Les CA conformes ne doivent pas fournir de certificats où la contrainte de stratégie est une séquence vide. inhibitPolicyMapping ou requireExplicitPolicy doit être présent. Le comportement des clients qui rencontrent un champ de contrainte de stratégie vide n'est pas adressé dans ce document.

Les CA conformes doivent marquer cette extension critique.

```
id-ce-policyConstraints OBJECT IDENTIFIER ::= { id-ce 36 }

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

SkipCerts ::= INTEGER (0..MAX)
```

Extended Key Usage

Cette extension indique un ou plusieurs buts pour lequel la clé publique certifiées peut être utilisée, en plus ou à la place de l'extension d'utilisation de clé. En général, cette extension apparaît seulement dans les certificats d'entité finales. Cette extension est définie :

```
id-ce-extKeyUsage OBJECT IDENTIFIER ::= { id-ce 37 }

ExtKeyUsageSyntax ::= SEQUENCE SIZE (1..MAX) OF KeyPurposeId

KeyPurposeId ::= OBJECT IDENTIFIER
```

Les identifiants d'objet utilisés pour identifier le but de clé doivent être assignés en accord avec les recommandations de l'IANA ou l'ITU-T X.660. Cette extension peut être critique ou non.

Si l'extension est présente, alors le certificat doit seulement être utilisé pour un des buts indiqué. Si plusieurs buts sont indiqués l'application n'a pas besoin de reconnaître tous les buts indiqués, tant que le but prévus est présent. Les applications utilisant les certificats peuvent nécessiter que l'extension d'utilisation de clé soit présente et qu'un but particulier soit indiqué pour que le certificat soit accepté par l'application.

Si une CA inclus les utilisations de clé pour satisfaire de telles applications, mais ne souhaite pas restreindre les utilisations de clé, la CA

peut inclure le KeyPurposeId spécicial anyExtendedKeyUsage en plus des buts particuliers requis par les applications. Les CA conformes ne devraient pas maquer cette extension critique si anyExtendedKeyUsage est présent. Les applications qui nécessitent la présence d'un but particulier peuvent rejeter les certificats qui incluent anyExtendedKeyUsage mais par l'OID particulier attendus pour l'application.

Si un certificat contient à la fois l'extension d'utilisation de clé et une extension d'utilisation de clé étendue, les 2 extensions doivent être traitées indépendamment et le certificat doit seulement être utilisé pour un but consistant avec les 2 extensions. S'il n'y a pas de but consistant avec les 2 extensions, alors le certificat ne doit pas être utilisé.

Les buts d'utilisation de clé suivants sont définis :

```
anyExtendedKeyUsage OBJECT IDENTIFIER ::= { id-ce-extKeyUsage 0 }

id-kp OBJECT IDENTIFIER ::= { id-pkix 3 }

id-kp-serverAuth OBJECT IDENTIFIER ::= { id-kp 1 }
 - Authentification serveur www TLS
 - Utilisations de clé qui peuvent être consistant: digitalSignature, keyEncipherment ou keyAgreement

id-kp-clientAuth OBJECT IDENTIFIER ::= { id-kp 2 }
 - Authentification client www TLS
 - Utilisations de clé qui peuvent être consistant: digitalSignature, et/ou keyAgreement
 TLS WWW client authentication

id-kp-codeSigning OBJECT IDENTIFIER ::= { id-kp 3 }
 - Signature de codes exécutables téléchargeables
 - Utilisations de clé qui peuvent être consistant: digitalSignature

id-kp-emailProtection OBJECT IDENTIFIER ::= { id-kp 4 }
 - Protection d'email
 - Utilisations de clé qui peuvent être consistant: digitalSignature, nonRepudiation, et/ou (keyEncipherment
ou keyAgreement)

id-kp-timeStamping OBJECT IDENTIFIER ::= { id-kp 8 }
 - Lier le hash d'un objet à une date
 - Utilisations de clé qui peuvent être consistant: digitalSignature et/ou nonRepudiation

id-kp-OCSPSigning OBJECT IDENTIFIER ::= { id-kp 9 }
 - Signature de réponses OCSP
 - Utilisations de clé qui peuvent être consistant: digitalSignature et/ou nonRepudiation
```

CRL Distribution Points

L'extension de point de distribution de CRL identifie comment les informations de CRL sont obtenus. L'extension devrait être non-critique, mais ce profile recommande le support pour cette extension par les CA et les application.

L'extension cRLDistributionPoints est une séquence de DistributionPoint. Un DistributionPoint consiste de 3 champs, chacun d'entre-eux est optionnels : distributionPoint, reasons, et cRLIssuer. Bien que chacun de ces champs est optionnel, un DistributionPoint ne doit pas consister du seul champ reasons ; distributionPoint ou cRLIssuer doit être présent. Si le fournisseur de certificat n'est pas le fournisseur de CRL, alors le champ cRLIssuer doit être présent et contient le nom du fournisseur de CRL. Si le fournisseur de certificat est également le fournisseur de CRL, les CA conformes doivent omettre le champ cRLIssuer et doivent inclure le champ distributionPoint.

Quand le champ distributionPoint est présent, il contient soit une séquence de noms ou une simple valeur, nameRelativeToCRLIssuer. Si le DistributionPointName contient plusieurs valeurs, chaque nom décris un mécanisme différent pour obtenir la même CRL. Par exemple, la même CRL pourrait être disponible via LDAP et HTTP.

Si le champ distributionPoint contient un directoryName, l'entrée pour ce directoryName contient la CRL courante pour les raisons associées et la CRL est fournie par le cRLIssuer associé. La CRL peut être stockée dans soit certificateRevocationList ou authorityRevocationList. La CRL est obtenus par l'application en fonction de la configuration locale du serveur d'annuaire. Le protocole que l'application utilise pour accéder à l'annuaire et un choix local.

Si DistributionPointName contient un nom général de type URI, les sémantiques suivante doivent être assumées : l'URI est un pointer vers la CRL courante pour les raisons associées et sera fournis par le cRLIssuer associé. Quand les schémas d'URI HTTP ou FTP sont utilisés, l'URI doit pointer vers une CRL simple encodé DER comme spécifié dans la rfc2585. Les implémentations de serveur HTTP accèdés via l'URI devraient spécifier le type de media application/pkix-crl dans le champ d'en-tête content-type de la réponse.

Quand le schéma d'URI LDAP est utilisé, l'URI doit inclure un champ dn contenant le nom distinct de l'entrée maintenant la CRL, doit inclure un simple attrdesc qui contient une description d'attribut appropriée pour l'attribut qui maintient la CRL (rfc4523), et devrait inclur un hôte (ex : <ldap ://ldap.example.com/cn=example%20CA,dc=example,dc=com ?certificateRevocationList ;binary>). Omettre l'hôte (ex : <ldap :///cn=CA,dc=example,dc=com ?authorityRevocationList ;binary>) implique de s'appuyer sur la connaissance du client du serveur approprié. Quand présent, DistributionPointName devrait inclure au moins une URI LDAP ou HTTP.

Si DistributionPointName contient un simple valeur nameRelativeToCRLIssuer, la valeur fournis un fragment de nom distinct. Le fragment est ajouté au nom distinct X.500 du fournisseur de CRL pour obtenir le nom du point de distribution. Si le champ cRLIssuer dans le DistributionPoint est présent, alors le fragment est ajouté au nom distinct qu'il contient, sinon le fragment est ajouté au nom distinct du fournisseur de certificat. Les CA conformes ne devraient pas utiliser nameRelativeToCRLIssuer pour spécifier les noms des points de distributions. Le DistributionPointName ne doit pas utiliser un nameRelativeToCRLIssuer relatif quand cRLIssuer contient plus d'un nom distinct.

Si DistributionPoint omet le champ reasons, la CRL doit inclure des informations de révocation pour toutes les raisons. Ce profile recommande d'éviter la segmentation de CRL par code de raison. Quand une CA conforme inclus une extension cRLDistributionPoints dans un certificat, elle doit inclure au moins un DistributionPoint qui pointe vers une CRL qui couvre le certificat pour toutes les raisons.

cRLIssuer identifie l'entité qui signe et fournis la CRL. Si présent, cRLIssuer doit seulement contenir le nom distinct du champ issuer de la CRL pointé par le DistributionPoint. L'encodage du nom dans le champ cRLIssuer doit être exactement le même que l'encodage dans le champ issuer de la CRL. Si le champ cRLIssuer est inclus et le dn dans ce champ ne correspond par à une entrée X.500 ou LDAP où la CRL est localisée, alors les CA conformes doivent inclure le champ distributionPoint.

```
id-ce-cRLDistributionPoints OBJECT IDENTIFIER ::= { id-ce 31 }

CRLDistributionPoints ::= SEQUENCE SIZE (1..MAX) OF DistributionPoint

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

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

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

Inhibit anyPolicy

L'extension inhibit anyPolicy peut être utilisée dans les certificats fournis aux CA. Elle indique que l'OID spéciale anyPolicy { 2 5 29 32 0 }, n'est pas considérée comme une correspondance explicite pour les autres stratégies de certificat excepté quand elle apparaît dans un certificat CA auto-fournis intermédiaire. La valeur indique le nombre de certificats non auto-fournis additionnels qui peuvent apparaître dans le chemin avant que anyPolicy ne soit plus permis. Par exemple, une valeur de 1 indique que anyPolicy peut être traitée dans les certificats fournis par le sujet de ce certificat, mais pas dans les certificats additionnels dans le chemin.

Les CA conformes doivent marquer cette extension critique

```
id-ce-inhibitAnyPolicy OBJECT IDENTIFIER ::= { id-ce 54 }

InhibitAnyPolicy ::= SkipCerts

SkipCerts ::= INTEGER (0..MAX)
```

Freshest CRL

l'extension freshest CRL identifie comment les informations de CRL delta sont obtenus. Cette extension doit être marquée non critique. La même syntaxe est utilisée pour cette extension et l'extension cRLDistributionPoints. Les même conventions d'appliquent aux 2 extensions

```
id-ce-freshestCRL OBJECT IDENTIFIER ::= { id-ce 46 }

FreshestCRL ::= CRLDistributionPoints
```

Extensions Internet privées

Cette section définis 2 extensions à utiliser dans les infrastructures à clé publique de l'Internet. Ces extensions peuvent être utilisées pour diriger les applications vers des informations en ligne sur le fournisseur ou le sujet. Chaque extension contient une séquence de méthodes d'accès et d'emplacements d'accès. La méthode d'accès est un identifiant d'objet qui indique le type d'informations qui sont disponibles. L'emplacement d'accès est un GeneralName qui spécifie implicitement l'emplacement et le format des informations et la méthode pour obtenir ces informations.

Les identifiants d'objet sont définis pour les extensions privées. Les identifiants d'objet associés avec les extensions privées sont définies sous l'arc id-pe dans l'arc id-pkix. Toutes extensions futures définies pour la PKI Internet seront définis également sous cet arc id-pe.

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

id-pe OBJECT IDENTIFIER ::= { id-pkix 1 }
```

Authority Information Access

L'extension d'accès aux informations d'autorité indique comment accéder aux informations et services pour le fournisseur du certificat dans lequel l'extension apparaît. Les informations et services peuvent inclure des services de validation en ligne et les données de stratégie de CA. (L'emplacement des CRL n'est pas spécifié dans cette extension). Cette extension peut être inclus dans les certificat d'entité finale ou de CA. Les CA conformes doivent marquer cette extension non-critique.

```
id-pe-authorityInfoAccess OBJECT IDENTIFIER ::= { id-pe 1 }
```

```
AuthorityInfoAccessSyntax ::=
 SEQUENCE SIZE (1..MAX) OF AccessDescription

AccessDescription ::= SEQUENCE {
 accessMethod OBJECT IDENTIFIER,
 accessLocation GeneralName }

id-ad OBJECT IDENTIFIER ::= { id-pkix 48 }

id-ad-caIssuers OBJECT IDENTIFIER ::= { id-ad 2 }

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

Chaque entrée dans la séquence AuthorityInfoAccessSyntax décris le format et l'emplacement d'informations additionnelles fournies par le fournisseur du certificat dans lequel cette extension apparaît. Le type et le format des informations sont spécifiées par le champ accessMethod ; le champ accessLocation spécifie l'emplacement des informations. Le mécanisme de récupération peut être déduit par accessMethod ou spécifié par accessLocation.

Ce profile définis 2 OID d'accessMethod : id-ad-caIssuers et id-ad-ocsp.

Dans un certificat à clé publique, l'OID id-ad-caIssuers est utilisé quand les informations additionnelles listent les certificats qui ont été fournis à la CA qui a fournis le certificat contenant cette extension. La description de fournisseurs CA référencés est destiné à aider les utilisateurs de certificat dans la sélection d'un chemin de certification qui se termine au point validé par le certificat utilisateur.

Quand id-ad-caIssuers apparaît comme accessMethod, le champ accessLocation décris le serveur de description référencé et le protocole d'accès pour obtenir la description référencée. Le champ accessLocation est définis en GeneralName, qui peut prendre de nombreuses formes.

Quand accessLocation est un directoryName, l'information est obtenu par l'application en fonction du serveur d'annuaire configuré localement. L'entrée pour le directoryName contient les certificats CA dans les attributs crossCertificatePair et/ou cACertificat comme spécifié dans la rf4523. Le protocole que les applications utilisent pour accéder à l'annuaire est une décision locale.

Quand l'information est disponible via LDAP, accessLocation devrait être un uniformResourceIdentifier. L'URI LDAP doit inclure un champ dn contenant le dn de l'entrée maintenant les certificats, doit inclure un champ attribute qui maintient la liste des descriptions d'attributs appropriés pour les attributs qui maintiennent les certificats encodés DER ou les paire le cross-certificats, et devraient inclure un hôte (ex <ldap ://ldap.example.com/cn=CA,dc=example,dc=com ?cACertificate ;binary,crossCertificatePair ;binary>). Omettre l'hôte (ex : <ldap :///cn=exampleCA,dc=example,dc=com ?cACertificate ;binary>) implique que le client connaisse le serveur approprié à contacter.

Quand les informations sont disponibles via HTTP ou FTP, accessLocation doit être un uniformResourceIdentifier et l'URI doit pointer sur soit un simple certificat encodé DER comme spécifié dans la rfc2585, ou une collection de certificats dans message CMS "certs-only" encodé DER ou BER, comme spécifié dans la rfc2797.

Les applications conformes qui supportent HTTP ou FTP pour accéder aux certificats doivent être capable d'accepter les certificats encodés DER individuels et devraient être capable d'accepter les message CMS "certs-only".

Les implémentations serveur HTTP accédés via l'URI devraient spécifier le type de média application/pkix-cert (rfc2585) dans l'en-tête content-type de la réponse pour un simple certificat encodé DER et devrait spécifier le type de média application/pkcs7-mime (rfc2797) dans le champ d'en-tête content-type de la réponse pour les messages CMS "certs-only". Pour FTP, le nom d'un fichier qui contient un simple certificat encodé DER devrait avoir le suffix ".cer" (rfc2585), et le nom d'un fichier qui contient un message CMS "certs-only" devrait avoir l'extension ".p7c" (rfc2797). Les clients peuvent utiliser le type de média ou l'extension de fichier, mais ne devraient pas dépendre uniquement de la présence du type de média ou de l'extension de fichier correcte dans la réponse du serveur.

Les sémantiques pour d'autres formes de noms d'accessLocation id-ad-caIssuers ne sont pas définis.

Une extension authorityInfoAccess peut inclure plusieurs instances de l'accessMethod id-ad-caIssuers. Les différentes instances peut spécifier différentes méthodes d'accès à la même information ou peut pointer vers différentes informations. Quant l'accessMethod

id-ad-caIssuers est utilisé, au moins une instance devrait spécifier un accessLocation qui est une URI HTTP ou LDAP.

L'OID id-ad-ocsp est utilisé quand les informations de révocation pour le certificat contenant cette extension est disponible en utilisant OCSP (rfc2560). Quand id-ad-ocsp apparaît comme accessMethod, le champ accessLocation est l'emplacement du répondeur OCSP, en utilisant les conventions spécifiées dans la rfc2560.

Des descripteurs d'accès additionnels peuvent être définis dans d'autres spécification PKIX.

Subject Information Access

L'extension d'accès aux informations du 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 une donnée de stratégie CA. Quand le sujet est une entité finale, les informations décrivent le type de services offerts et comment y accéder. Dans ce cas, le contenu de cette extension est définis dans les spécifications du protocole pour les services supportés. Cette extension peut être inclue dans des certificats CA ou d'entité finale. Les CA conformes doivent marquer cette extension non-critique.

```
id-pe-subjectInfoAccess OBJECT IDENTIFIER ::= { id-pe 11 }

SubjectInfoAccessSyntax ::=
 SEQUENCE SIZE (1..MAX) OF AccessDescription

AccessDescription ::= SEQUENCE {
 accessMethod OBJECT IDENTIFIER,
 accessLocation GeneralName }
```

Chaque entrée dans la séquence SubjectInfoAccessSyntax décris le format et l'emplacement des informations additionnelles fournies par le sujet du certificat dans lequel cette extension apparaît. Le type et le format de l'information sont spécifiés par le champ accessMethod ; le champ accessLocation spécifie l'emplacement de l'information. Le mécanisme de récupération peut être implicite via accessMethod ou accessLocation.

Ce profile définis une méthode d'accès à utiliser quand le sujet est une CA et une méthode d'acces utilisée quand le sujet est une entité finale. Des méthodes d'accès additionnels peuvent être définis dans de futures spécification de protocole pour d'autres services.

L'OID id-ad-caRepository est utilisé quand le sujet est une CA qui publie les certificat quelle fournis dans un annuaire. Le champ accessLocation est définis en GeneralName, qui peut prendre de nombreuses formes.

Quand accessLocation est un directoryName, l'information est obtenue par l'application depuis le serveur d'annuaire configuré localement. Quand l'extension est utilisée pour pointer vers des certificats CA, l'entrée pour le directoryName contient les certificats CA dans les attributs crossCertificatePair et/ou cACertificat comme spécifiés dans la rfc4523. Le protocole que l'application utilise pour accéder à l'annuaire est un choix local.

Quand les informations sont disponible via LDAP, accessLocation devrait être un uniformResourceIdentifier. L'URI ldap doit inclure un champ dn contenant le dn de l'entrée maintenant les certificats, doit inclure un champ attribute qui liste les descriptions d'attributs appropriés pour les attributs qui maintiennent les certificats encodés DER ou les paires de cross-certificats, et devraient inclure un hôte. (ex : <ldap ://ldap.example.com/cn=CA,dc=example,dc=com ?cACertificate ;binary,crossCertificatePair ;binary>). Omettre l'hôte implique que le client connaît le serveur approprié à contacter.

Quand l'information est disponible via HTTP ou FTP, accessLocation doit être un uniformResourceIdentifier et l'URI doit pointer soit vers un simple certificat encodé DER ou une collection de certificats dans un message CMS "certs-only" encodés DER ou BER.

Les applications conformes qui supportent HTTP ou FTP pour accéder aux certificats doivent être capable d'accepter les certificat encodés DER individuels et devraient être capable d'accepter les message CMS "certs-only".

Les implémentations de serveur HTTP accédés via l'URI devraient spécifier le type de média application/pkix-cert (rfc2585) dans l'en-tête content-type de la réponse pour un simple certificat encodé DER et devrait spécifier le type de média application/pkcs7-mime (rfc2797) dans le champ d'en-tête content-type de la réponse pour les messages CMS "certs-only". Pour FTP, le nom d'un fichier qui contient un simple certificat encodé DER devrait avoir le suffix ".cer" (rfc2585), et le nom d'un fichier qui contient un message CMS "certs-only" devrait avoir l'extension ".p7c" (rfc2797). Les clients peuvent utiliser le type de média ou l'extension de fichier, mais ne devraient pas dépendre uniquement de la présence du type de média ou de l'extension de fichier correcte dans la réponse du serveur.

Les sémantiques pour d'autres formes de nom accessLocation id-ad-caRepository ne sont pas définies.

Une extension subjectInfoAccess peut inclure plusieurs instances de accessMethod id-ad-caRepository. Les instances différentes peuvent spécifier des méthodes différentes pour accéder à la même information ou peuvent pointer vers différentes informations. Quand accessMethod id-ad-caRepository est utilisé, au moins une instance devrait spécifier un accessLocation qui est une URI HTTP ou LDAP.

L'OID id-ad-timeStamping est utilisé quand le sujet offre des services de timestamping utilisant le TSP (rfc3161). Quand les services de timestamping sont disponibles via HTTP ou FTP, accessLocation doit être un uniformResourceIdentifier. Quand les services de timestamping sont disponibles via les messages électroniques, accessLocation doit être un rfc822Name. Quand les services de timestamping sont disponibles en utilisant TCP/IP, les formes de noms dNSName ou iPAddress peuvent être utilisés. Les sémantiques d'autres formes de nom de accessLocation (quand accessMethod est id-ad-timeStamping) ne sont pas définis par cette spécification.

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

```
id-ad OBJECT IDENTIFIER ::= { id-pkix 48 }

id-ad-caRepository OBJECT IDENTIFIER ::= { id-ad 5 }

id-ad-timeStamping OBJECT IDENTIFIER ::= { id-ad 3 }
```

Profile de CRL et d'extensions de CRL

Comme discuté plus haut, un des but de ce profile de CRL X.509 v2 est de favoriser la création d'une ICP Internet interopérable et réutilisable. Pour cela, des lignes directrices pour l'utilisation des extensions sont spécifiée, et certaines hypothèses sont faites sur la nature des informations inclues dans la CRL.

Les CRL peuvent être utilisée dans un grande variété d'applications et d'environnement couvrant un large spectre de pré-requis opérationnels et d'assurance. Ce profile établis une ligne de base commune pour les applications génériques nécessitant une grande interopérabilité. Ce profile définis un jeu d'informations qui peuvent être prévus dans toutes les CRL. Également, le profile définis des emplacements communs dans la CRL pour les attributs fréquemment utilisés et les représentations communes pour ces attributs.

Les fournisseurs de CRL fournissent les CRL. Le fournisseur de CRL est soit la CA ou une entité qui a été autorisée par la CA pour fournir les CRL. Les CA publient les CRL pour fournir des informations de statut sur les certificats qu'elles émettent. Cependant, une CA peut déléguer cette responsabilité à une autre autorité de confiance.

Chaque CRL a un périmètre particulier. Le scope de CRL est le jeu de certificats qui peuvent apparaître dans une CRL donnée. Par exemple, le scope pourrait être "tous les certificats émis par la CA X", "Tous les certificats de CA fournis par la CA X", "tous les certificats émis par la CA X qui ont été révoqués pour une raison de compromission de clé et de CA compromise", ou un jeu de certificats basés sur des informations locales arbitraire, tels que "tous les certificats fournis aux employés NIST localisé à Boulder.

Une CRL complète liste tous les certificat non-expirés, dans son scope, qui ont été révoqués pour une des raisons de révocation couverts par le scope de la CRL. Une CRL pleine et complète liste tous les certificats non-expirés fournis par une CA qui a été révoqué pour une raison. (Noter que vu que les CA et les fournisseurs de CRL sont identifiés par nom, le scope d'une CRL n'est pas affectée par la clé utilisée pour signer la CRL ou les clés utilisées pour signer les certificats).

Si le scope de la CRL inclus un ou plusieurs certificats fournis par une entité autre que le fournisseur de CRL, alors c'est une CRL indirecte. Le scope d'une CRL indirect peut être limitée aux certificats fournis par une simple CA ou peut inclure les certificats fournis par

plusieurs CA. Si le fournisseur de la CRL indirecte est une CA, alors le scope de la CRL indirecte peut également inclure les certificats inclus par le fournisseur de cette CRL.

Le fournisseur de CRL peut également générer des CRL delta. Une CRL delta liste seulement les certificats, dans son scope, dont le statut de révocation a changé depuis l'émission de la dernière CRL de complète référencée. Le scope d'une CRL delta doit être la même que a CRL de base qu'elle référence.

Ce profile définis une extension de CRL Internet privée mais ne définis pas d'extensions d'entrée de CRL privée.

Les environnements avec de besoins spéciaux ou additionnels peuvent se baser sur ce profile ou le remplacer.

Les CA conformes ne sont pas obligés de fournir des CRL si d'autre mécanismes de statut de révocation ou de certificat sont fournis. Quand les CRL sont fournis, les CRL doivent être des CRL v2, doivent inclure la date à laquelle la prochaine CRL sera fournie dans le champ nextUpdate, doivent inclure l'extension de numéro de CRL, et doivent inclure l'extension d'identifiant de clé d'autorité. Les applications conformes qui supportent les CRL doivent traiter les CRL v1 et v2 qui fournissent des informations de révocation pour tous les certificats fournis par une CA. Les applications conformes ne sont pas obligés de supporter le traitement des CRL delta, des CRL indirect, ou des CRL avec un scope autre que tous les certificats fournis par une CA.

Champs de CRL

La syntaxe de CRL X.509 v2 est la suivante. Pour le calcule de signature, la donnée qui est signée est encodée ASN.1 DER. Un encodage ASN.1 DER est un système d'encodage ayant un tag, une longueur, et une valeur pour chaque élément.

```
CertificateList ::= SEQUENCE {
 tbsCertList TBSCertList,
 signatureAlgorithm AlgorithmIdentifier,
 signatureValue BIT STRING }

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

- Version, Time, CertificateSerialNumber, et les Extensions
- sont tous définis dans la section "champs de certificat de base"
- AlgorithmIdentifier est décris dans la section signatureAlgorithm
```

Champs CertificateList

CertificateList est une séquence de 3 champs requis. Les champs sont décris en détail ci-dessous.

tbsCertList

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

signatureAlgorithm

Le champ signatureAlgorithme contient l'identifiant pour l'algorithme utilisé par le fournisseur de CRL pour signer le CertificateList. Le champ est de type AlgorithmIdentifier, mais d'autres algorithmes de signatures peuvent être supportés. Ce champ doit contenir le même identifiant d'algorithme que le champ signature dans la séquence tbsCertList.

signatureValue

Le champ signatureValue contient une signature numérique calculée sur le tbsCertList encodé DER ASN.1, qui est utilisé en entrée de la fonction de signature. Cette valeur de signature est encodée en chaîne de bits et inclus dans le champ signatureValue de la CRL. Les détails de ce processus sont spécifiés pour chacun des algorithmes supportés dans les rfc3279, rfc4055, et rfc 4491.

Les CA qui sont également fournisseurs de CRL peuvent utiliser une seul clé privée pour signer numériquement les certificats et les CRL, ou peuvent utiliser des clé privées séparées. Quand des clés privées séparés sont utilisées, chaque clé publique associée avec les clés ces clés privée sont placées dans un certificat séparé, une avec le bit keyCertSign mis dans l'extension d'utilisation de clé, et une avec le bit cRLSign mis dans l'extension d'utilisation de clé. Quand des clés privée séparés sont utilisés, les certificats émis par la CA contiennent un seul identifiant de clé d'autorité, et les CRL correspondantes contiennent un identifiant de clé d'autorité différent.

L'utilisation de certificats CA séparés pour la validation des signature de certificat et de CRL peut offrir de caractéristiques de sécurité améliorés ; cependant, cela impose une charge dans les applications, et peut limiter l'interopérabilité. De nombreuses application construisent un chemin de certification, puis le valide. La vérification de CRL nécessite un chemin de certification séparé pour la validation de la signature de la CRL. Les applications qui vérifient la CRL doivent supporter la validation de chemin de certification quand les certificats et les CRL sont signés numériquement avec la même clé privée, et devraient supporter la validation de chemin de certification quand les certificats et les CRL sont signés numériquement avec des clé différentes.

Certificate List To Be Signed

La liste de certificat signée, ou TBSCertList, est une séquence de champs requis et optionnels. Les champs requis identifient le fournisseur de la CRL, l'algorithme utilisé pour signer la CRL, et la date d'émission de la CRL.

Les champs optionnels incluent la date à laquel le fournisseur de CRL va émettre la prochaine CRL, les listes de certificats révoqués, et les extensions de CRL. La liste de certificats révoqués est optionnel pour supporter le cas où une CA n'a pas de certificats révoqué. Ce profile nécessite que les fournisseurs de CRL conformes incluent le champ nextUpdate et les extensions de CRL d'identifiant de clé d'autorité et de numéro.

Version

Ce champ optionnel décris la version de la CRL encodée. Quand les extensions sont utilisées, comme requis par ce profile, ce champ doit

être présent et doit spécifier la version 2 (valeur 1).

Signature

Ce champ contient l'identifiant d'algorithme pour l'algorithme utilisé pour signier la CRL. la rfc3279, rfc4055, et rfc 4491 listent les OID pour les algorithmes de signature les plus populaires utilisés dans les PKI de l'Internet.

Issuer Name

Le nom du fournisseur identifie l'entité qui a signé et fournis la CRL. L'identité du fournisseur est placé dans le champ issuer. Les formes de noms Alternatives peuvent également apparaître dans l'extension issuerAltName. Le champ issuer doit contenir un DN X.500 non-vide. Le champ issuer est définis comme nom de type X.501, et doit suivre les rêgles d'encodage pour le champs du nom du fournisseur dans le certificat.

This Update

Ce champ indiques la date d'émission de cette CRL. thisUpdate peut être encodé en UTCTime ou GeneralizedTime. Les fournisseurs de CRL conformes à ce profile doivent encoder thisUpdate en UTCTime pour les dates avant 2050. Les fournisseurs de CRL conformes à ce profile doivent encoder thisUpdate en GeneralizedTime pour des dates après 2049. Les application conformes doivent être capable de traiter les dates qui sont encodé en UTCTime ou GeneralizedTime.

Quand encodé en UTCTime, thisUpdate doit être spécifié et interprété comme définis plus haut dans ce document (voir section UTCTime). Quand encodé en GeneralizedTime, thisUpdate doit être spécifié et interprété comme définis plus haut dans ce document (voir section GeneralizedTime).

Next Update

Ce champ indique la date à laquelle la prochaine CRL sera fournie. La prochaine CRL pourrait être fournie avant cette date, mais ne sera pas fournis après cette date. Les fournisseurs de CRL devraient fournir les CRL avec un temps nextUpdate égale à ou supérieurs à tous les CRL précédentes. nextUpdate peut être encodé en UTCTime ou GeneralizedTime.

Les fournisseurs de CRL conformes doivent inclure le champ nextUpdate dans toutes les CRL. Noter que la syntaxe ASN.1 de TBSCertList décris ce champ comme optionnel, qui est consistant avec la structure ASN.1 définis dans X.509. Le comportement des clients traitant les CRL qui omettent nextUpdate n'est pas définis par ce profile.

Les fournisseurs de CRL conformes à ce profile doivent encoder nextUpdate en UTCTime pour les dates avant 2050 et en GeneralizedTime pour les dates après 2049. Les application conformes doivent être capable de traiter les dates encodés en UTCTime ou GeneralizedTime.

Quand encodé en UTCTime, nextUpdate doit être spécifié et interprété comme définis plus haut dans ce document (voir section UTCTime). Quand encodé en GeneralizedTime, nextUpdate doit être spécifié et interprété comme définis plus haut dans ce document (voir section GeneralizedTime).

Revoked Certificates

Quand il n'y a pas de certificats révoqués, la liste des certificats révoqué doit être absent. Sinon, les certificats révoqués sont listés par leur numéro de série. Les certificats révoqués par la CA sont identifiés de manière unique par ce numéro de série. La date à laquelle la révocation se produit est spécifiée. La date pour revocationDate doit être exprimée comme décris dans la section thisUpdate. Des informations additionnelles peuvent être fournis dans les extensions d'entrée de CRL.

Extensions

Ce champ peut seulement apparaître si la version est 2. Si présent, ce champ est une séquence d'une ou plusieurs extensions de CRL.

Extensions de CRL

Les extensions définis par ANSI X9, ISO/IEC, et l'ITU-T pour les CRL X.509 v2 fournissent des méthodes pour associer des attributs additionnels avec les CRL. Le format de CRL X.509 v2 permet également des extensions privées. Chaque extension dans une CRL peut être conçue comme critique ou non. Si une CRL contient une extension critique que l'application en 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 reconnus. Les sections suivantes présentent ces extensions utilisées avec les CRL de l'internet. Les communautés peuvent choisir d'inclure des extensions dans les CRL qui ne sont pas définies dans cette spécification. Cependant, une attention particulière doit être faite pour l'adoptions d'extensions critiques dans les CRL qui pourraient être utilisées dans un contexte général.

Les fournisseurs de CRL conformes doivent inclure les extensions d'identifiant de clé d'autorité et de numéro de CRL dans toutes les CRL émises.

Authority Key Identifier

L'extension d'identifiant de clé d'autorité fournis un moyen d'identifier la clé publique correspondante à la clé privée utilisée pour signer une CRL. L'identification peut être basée sur l'identifiant de clé (l'identifiant de clé du sujet dans le certificat du signataire de la CRL) ou le nom et le numéro de série du fournisseur. Cette extension est spécialement utile où un fournisseur a plus d'une clé de signature, soit dû à plusieurs paires de clé concurrentes, ou dû à un changement.

Les fournisseurs de CRL conformes doivent utiliser la méthode d'identifiant de clé, et doivent inclure cette extension dans toutes les CRL émise. La syntaxe pour cette extension de CRL est définie dans la section Authority Key Identifier.

Issuer Alternative Name

L'extension de nom alternatif de fournisseur permet d'associer des identités alternatives avec le fournisseur de CRL. Les options définies incluent une adresse de messagerie électronique (rfc822Name), un nom DNS, une adresse IP, et une URI. Plusieurs instances d'une forme de nom et plusieurs formes de noms peuvent être inclus. Quand de telles identités sont utilisées, l'extension de nom alternatif de fournisseur doit être utilisé ; cependant, un nom DNS peut être représenté dans le champ issuer en utilisant l'attribut domainComponent décris dans la section Issuer.

Les fournisseurs de CRL conformes devraient marquer l'extension issuerAltName comme non-critique. L'OID et la syntaxe pour cette extension de CRL sont définis dans la section Issuer Alternative Name précédente.

CRL Number

Le numéro de CRL est une extension de CRL non-critique qui transmet un numéro de séquence croissant monotone pour un scope de CRL et un fournisseur de CRL donné. Cette extension permet aux utilisateurs de facilement déterminer quand une CRL particulière remplace une autre CRL. Les numéros de CRL supportent également l'identification de CRL complètes et de CRL delta complémentaires. Les fournisseurs de CRL conformes à ce profile doivent inclure cette extension dans toutes les CRL et doivent marquer cette extension non-critique.

Si un fournisseur de CRL génère des CRL delta en plus de CRL complètes pour un scope donné, les CRL complète et delta doivent partager une séquence de numéro. Si une CRL delta et une CRL couvrant le même scope sont émis en même temps, elles doivent avoir le même numéro de CRL et fournir la même information de révocation. C'est à dire, la combinaison de la CRL delta et une CRL complète acceptable doivent fournir la même information de révocation que la CRL complète.

Si un fournisseur de CRL génère 2 CRL (2 CRL complètes, 2 CRL delta, ou une CRL complète et une CRL delta) pour le même scope à des moments différents, les 2 CRL ne doivent pas avoir le même numéro de CRL, c'est à dire, si le champ thisUpdate dans les 2 CRL ne sont pas identiques, les numéros de CRL doivent être différents.

En considération, les numéro de CRL peuvent être de long entiers. Les vérificateurs de CRL doivent être capable de gérer des valeur CRLNumber jusqu'à 20 octets. Les fournisseurs de CRL conformes ne doivent pas utiliser de valeurs CRLNumber plus long que 20 octets.

```
id-ce-cRLNumber OBJECT IDENTIFIER ::= { id-ce 20 }

CRLNumber ::= INTEGER (0..MAX)
```

Delta CRL Indicator

L'indicateur de CRL delta est une extension de CRL critique qui identifie qu'une CRL est une CRL delta. Les CRL delta contiennent des mises à jours d'information de révocation précédemment distribuée. L'utilisation de CRL delta peut significativement réduire la charge réseaux et le temps de traitement dans certains environnements. Les CRL delta sont généralement plus petites que les CRL qu'elles mettent à jours, donc les applications qui obtiennent les CRL delta consomment moins de bande passante que les applications qui obtiennent les CRL complètes correspondantes. Les applications qui stockent les informations de révocation dans un format autre que la structure CRL peuvent ajouter les nouvelles information de révocation dans la base locale sans retraiter les informations.

L'extension d'indication de CRL delta contient une valeur simple de type BaseCRLNumber. Le numéro de CRL identifie la CRL, complète pour un scope donné, qui a été utilisé comme point de départ dans la génération de cette CRL delta. Un fournisseur de CRL conforme doit publier la CRL de base référencée ou une CRL complète. La CRL delta contient toutes les mises à jours de statut de révocation pour ce même scope. La combinaison d'une CRL delta et de la CRL de base référencée est équivalente à une CRL complète, pour le périmètre applicable, au moment de la publication de la CRL delta.

Quand un fournisseur de CRL conforme génère une CRL delta, la CRL delta doit inclure une extension d'indication de CRL delta.

Quand une CRL delta est émise, elle doit couvrir le même jeu de raisons et le même jeu de certificats qui sont couverts par la CRL de base qu'elle référence. C'est à dire que le scope de la CRL delta doit être le même que le scope de la CRL complète référencée comme base. La CRL de base référencée et la CRL delta doivent omettre l'extension de point de distribution d'émission ou doivent contenir des extensions de point de distribution d'émission identique. De plus, le fournisseur de CRL doit utiliser la même clé privée pour signer la CRL delta et toute CRL complète qui peut être utilisée pour les mises à jours.

Une application qui supporte les CRL delta peuvent construire une CRL qui est complète pour un scope donné en combinant une CRL delta pour ce scope avec soit une CRL émise qui est complète pour ce scope ou une CRL construite localement, qui est complète pour ce scope.

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 de numéro de CRL trouvé dans la CRL delta utilisée dans sa construction. En plus, le CRL construite localement résultante a les temps thisUpdate et nextUpdate spécifiés dans les champs correspondant de la CRL delta utilisés dans

sa construction. En plus, la CRL construite localement hérite du point de distribution d'émission de la CRL delta.

Une CRL complète et une CRL delta peuvent être combinés si les 4 conditions suivantes sont satisfaites :

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

(b) La CRL complète et la CRL delta ont le même périmètre. Les 2 CRL ont le même périmètre si une des conditions suivantes est satisfaite :

(1) L'extension issuingDistributionPoint est omis de la CRL complète et de la CRL delta

(2) L'extension issuingDistributionPoint est présente dans la CRL complète et la CRL delta est sont identiques.

(c) Le numéro de CRL de la CRL complète est égale ou supérieur au BaseCRLNumber spécifié dans la CRL delta. C'est à dire que la CRL complète contient (au minimum) toutes les informations de révocation maintenus 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 fournisseurs de CRL doivent s'assurer que la combinaison d'une CRL delta et toute CRL complète appropriée réflète précisément le statut de révocation. Le fournisseur e CRL doit inclure une entrée dans la CRL delta pour chaque certificat dans le scope de la CRL delta dont le status a changé depuis la génération de la CRL de base référencée :

(a) Si le certificat est révoqué pour une raison inclue dans le scope de la CRL, liste le certificat révoqué.

(b) Si le certificat est valide et a été listé dans la CRL de base référencée ou toute CRL suivantes avec un code de raison certificatHold est inclus dans le scope de la CRL, liste le certificat avec le code de raison removeFromCRL.

(c) Si le certificat est révoqué pour une raison hors du scope de la CRL, mais que le certificat a été listé dans la CRL de base référencée ou toute CRL suivante avec un code de raison inclus dans le scope de cette CRL, liste le certificat comme révoqué mais omet le code de raison.

(d) Si le certificat est révoqué pour une raison hors du scope de la CRL est que le certificat n'est ni listé dans la CRL de référence ni dans les CRL suivantes avec un code un code de raison inclus dans le scope de cette CRL, ne pas lister le certificat dans cette CRL.

Le statut d'un certificat est considéré comme ayant changé s'il est révoqué (pour n'importe quelle raison, incluant certificateHold), s'il est libéré de l'attente, ou si sa raison de révocation change.

Il est approprié de lister un certificat avec un code de raison removeFromCRL dans une CRL delta même si le certificat n'était pas en attente dans la CRL de base référencée. Si le certificat a été placé en attente dans une CRL émise après la base mais avant cette CRL delta puis relachée, il doit être listé dans le CRL delta avec une raison de révocation removeFromCRL.

Un fournisseur de CRL peut optionnellement lister un certificat dans une CRL delta avec le code de raison removeFromCRL si le temps notAfter spécifié dans le certificat précède le temps thisUpdate spécifié dans la CRL delta et que le certificat était listé dans la CRL de base référencée. ou dans toute CRL émise après la base mais avant cette CRL delta.

Si une révocation de certificat apparaît dans une CRL delta, il est possible pour que la période de validité du certificat expire avant la prochaine CRL complète pour le même scope. Dans ce cas, La notification de révocation doit être inclue dans toutes les CRL delta suivantes jusqu'à ce que la notification de révocation soit inclue dans au moins une CRL complète émise pour ce scope.

Une application qui supporte les CRL delta doit être capable de construire une CRL complète courante en combinant une CRL complète et la CRL delta la plus récente. Une application qui supporte les CRL delta peut également être capable de construire une CRL complète courante en combinant une CRL complète construite localement et la CRL delta courante. Une CRL delta est considérée courante si le date courante est entre les temps contenus dans thisUpdate et nextUpdtae. Sous certaines circonstances, le fournisseur de CRL peut publier une ou plusieurs CRL avant le temps indiqué par le champ nextUpdate. Si plus d'une CRL delta courante est rencontré pour un scope donné, l'application devrait être considérée celle ayant la valeur thisUpdate la plus récente comme courante.

```
id-ce-deltaCRLIndicator OBJECT IDENTIFIER ::= { id-ce 27 }

BaseCRLNumber ::= CRLNumber
```

Issuing Distribution Point

Le point de distribution d'émission est une extension de CRL critique qui identifie le point de distribution de CRL et le scope pour une CRL particulière, et indique si la CRL couvre la révocation des certificat d'entités finaux uniquement, des certificats CA uniquement, ou un jeu limité de codes de raisons. Bien que cette extension soit critique, les implémentations conformes ne sont pas obligés de supporter cette extension. Cependant, les implémentations qui ne supportent pas cette extension doivent soit traiter le statut des certificats non listés dans cette CRL comme inconnu ou localiser une autre CRL qui ne contient pas d'extensions critique non-reconnues.

La CRL est signée en utilisant la clé privée du fournisseur. Les points de distribution de CRL n'ont pas leur propre paires de clé. Si la CRL est stockée dans un annuaire X.500, elle est stockée dans l'entrée d'annuaire correspondant au point de distribution e CRL, qui peut être différent de l'entrée d'annuaire du fournisseur de CRL.

Les codes de raison associées avec un point de distribution doivent être spécifiés dans onlySomeReasons. Si onlySomeReasons n'apparaît pas, le point de distribution doit contenir les révocation pour tous les codes de raison. Les CA peuvent utiliser les points de distribution de CRL pour partitionner la CRL sur la base de révocation de routine et de compromission. Dans ce cas, les révocations avec le code de raison keyCompromise, cACompromise, et aACompromise apparaissent dans un point de distribution, et les révocations avec d'autres codes de raison apparaissent dans un autre point de distribution.

Si une CRL inclus une extension issuingDistributionPoint avec onlySomeReasons présent, alors tout certificat dans le scope de la CRL qui est révoqué doit être assigné à un code de raison autre qui unspecified. La raison de révocation assignées est utilisée pour déterminer dans quelle CRL lister le certificat révoqué, cependant, il n'est pas requis d'inclure l'extension d'entrée de CRL reasonCode dans l'entrée de CRL correspondant.

La syntaxe et les sémantiques pour le champ distributionPoint sont les même que pour le champ distributionPoint dans l'extension cRLDistributionPoints. Si le champ distributionPoint est présent, alors il doit inclure au moins un des noms du distributionPoint correspondant de l'extension cRLDistributionPoints de tout certificat qui est dans le scope de cette CRL. L'encodage identique doit être utilisé dans les champs distributionPoint du certificat et de la CRL.

Si le champ distributionPoint est absent, le CRL doit contenir des entrées pour tous les certificats révoqués non-expirés fournis par le fournisseur de CRL, dans le scope de la CRL.

Si le scope de la CRL inclus uniquement les certificats émis par le fournisseur de CRL, alors le booléen indirectCRL doit être FALSE. Sinon, si le scope de la CRL inclus des certificats fournis par une ou plusieurs autres autorités, indirectCRL doit être à TRUE. L'autorité responsable pour chaque entrée est indiquée par le fournisseur de certificat de l'extension d'entrée de CRL.

Si le scope de la CRL inclus uniquement des certificats à clé publique d'entité final, alors onlyContainsUserCerts doit être à TRUE. Si le scope de la CRL inclus seulement des certificats CA, alors onlyContainsCACerts doit être à TRUE. Si onlyContainsUserCerts ou onlyContainsCACerts est à TRUE, alors le scope de la CRL ne doit pas inclure de certificats version 1 ou version 2. Les fournisseurs de CRL conformes doivent définir le booléen onlyContainsAttributeCerts à FALSE.

Les fournisseurs de CRL conformes ne doivent pas émettre de CRL où l'encodage DER de l'extension de point de distribution du fournisseur est une séquence vide. C'est à dire que si onlyContainsUserCerts, onlyContainsCACerts, indirectCRL, et onlyContainsAttributeCerts sont à FALSE, alors soit le champ distributionPoint soit le champ onlySomeReasons doit être présent.

```
id-ce-issuingDistributionPoint OBJECT IDENTIFIER ::= { id-ce 28 }

IssuingDistributionPoint ::= SEQUENCE {
 distributionPoint [0] DistributionPointName OPTIONAL,
 onlyContainsUserCerts [1] BOOLEAN DEFAULT FALSE,
 onlyContainsCACerts [2] BOOLEAN DEFAULT FALSE,
 onlySomeReasons [3] ReasonFlags OPTIONAL,
 indirectCRL [4] BOOLEAN DEFAULT FALSE,
 onlyContainsAttributeCerts [5] BOOLEAN DEFAULT FALSE }
```

- au moins un parmis onlyContainsUserCerts, onlyContainsCACerts, et onlyContainsAttributeCerts doit être à TRUE.

Freshest CRL

L'extension freshest CRL identifie comment les informations de CRL delta pour cette CRL complète est obtenue. Les fournisseurs conformes doivent marquer cette extension 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. Cependant, seul le champ de point de distribution est significatif dans ce contexte. Les champs raison et cRLIssuer doivent être omis de cette extension de CRL.

Chaque nom de point de distribution fournis l'emplacement auquel une CRL delta pour cette CRL complète peut être trouvée. Le scope de ces CRL delta doit être le même que le scope de la CRL complète. Le contenu de cette extension de CRL est seulement utilisé pour localiser les CRL delta ; le contenu n'est pas utilisé pour valider la CRL ou les CRL delta.

```
id-ce-freshestCRL OBJECT IDENTIFIER ::= { id-ce 46 }

 FreshestCRL ::= CRLDistributionPoints
```

Authority Information Access

Cette section définis l'utilisation de l'extension d'accès aux informations d'autorité dans une CRL. La syntaxe et les sémantiques définis pour l'extension de certificat du même nom sont également utilisé pour l'extension de CRL. Cette extension de CRL doit être marquée non-critique.

Quand présent dans une CRL, cette extension doit inclure au moins un AccessDescription spécifiant l'id-ad-caIssuers comme accessMethod. L'OID id-ad-caIssuers est utilisé quand l'information disponible liste les certificats qui peuvent être utilisés pour vérifier la signature dans la CRL (par exemple, les certificats qui ont un nom de sujet qui correspond à la clé privée utilisé pour signer la CRL). Les types de méthode d'accès autre que id-ad-caIssuers ne doivent pas être inclus. Au moins une instance de AccessDescription devrait spécifier un URI accessLocation HTTP ou LDAP.

Quand l'information est disponible via HTTP ou FTP, accessLocation doit être un uniformResourceIdentifier et l'URI doit pointer vers soit un simple certificat encodé DER soit une collection de certificats dans une message CMS "certs-only" encodé en DER ou BER.

Les applications conformes qui supportent HTTP ou FTP pour accéder aux certificats doivent être capable d'accepter les certificats individuels encodés DER et devraient être capable d'accepter les messages CMS "certs-only".

Les implémentations de serveurs HTTP accédés via l'URI devraient spécifier le type de média application/pkix-cert dans le champ dans le champ d'en-tête content-type de la réponse pour un simple certificat encodé DER et devraient spécifier le type de média application/pkcs7-mime dans le champ d'en-tête content-type de la réponse pour les message CMS "certs-only". Pour FTP, le nom d'un fichier qui contient un simple certificat encodé DER devrait avoir un suffix ".cer" et le nom d'un fichier qui contient un message CMS "certs-only" devrait avoir le suffix ".p7c". Les client peuvent utiliser le type de média ou l'extension de fichier comme indicateur de contenu, mais ne devraient pas s'en contenter.

Quand accessLocation est un directoryName, l'information est obtenu par l'application via le serveur configuré localement. Quand une clé publique de CA est utilisée pour valider les signatures dans les certificats et CRL, le certificat de CA désiré est stocké dans les attributs crossCertificatePair et/ou cACertificate. quand différentes clé publique sont utilisées pour valider les signatures dans les certificats et les CRL, le certificat désiré est stocké dans l'attribut userCertificate. Donc, les implémentations qui supportent la forme directoryName de accessLocation doivent être préparés à trouver le certificat nécessaire dans n'importe lequel de ces attributs. Le protocole qu'une application utilise pour accéder à l'annuaire est un choix local.

Quand les informations sont disponibles via LDAP, accessLocation devrait être un uniformResourceIdentifier. L'URI LDAP doit inclure un champ dn contenant de nom distinct de l'entrée maintenant les certificats, doit inclure un champ attributes qui liste les descriptions d'attributs appropriés pour les attributs qui maintiennent les certificats encodés DER ou les paires de cross-certificats, et devrait inclure un hôte (ex : <ldap ://ldap.example.com/cn=CA,dc=example,dc=com ?cACertificate ;binary,crossCertificatePair ;binary>). Omettre l'hôte (ex : <ldap :///cn=exampleCA,dc=example,dc=com ?cACertificate ;binary>) implique que le client connaisse le serveur approprié à contacter.

CRL Entry Extensions

Les extensions d'entrée de CRL définis par l'ISO/IEC, ITU-T, et ANSI X9 pour les CRL X.509 v2 fournissent des méthodes pour associer des attributs additionnels avec des entrées de CRL. Le format de CRL X.509 v2 permet également aux communautés de définir des extensions d'entrée de CRL privées pour gérer les informations spécifiques de ces communautés. Chaque extension dans une entrée de CRL peut être désignée comme critique ou non. Si une CRL contient une extension d'entrée de CRL critique qui 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'entrées de CRL non-critique non reconnus.

Le support pour les extensions d'entrée de CRL définis dans cette spécification est optionnel pour les fournisseurs de CRL et les applications conformes. Cependant, les fournisseurs de CRL devraient inclure des codes de raison et des dates d'invalidité quand cette information est disponible.

Reason Code

Le reasonCode est une extension d'entrée de CR non-critique qui identifie la raison pour la révocation du certificat. Les fournisseurs de CRL sont fortement encouragés à inclure des codes de raison significatifs dans les entrées de CRL. Cependant, l'extension d'entrée de CRL de code de raison devrait être absent au lieu d'utiliser la valeur reasonCode unspecified.

La valeur de reasonCode removeFromCRL peut seulement apparaître dans les CRL delta, et indique qu'un certificat à été supprimé de la CRL soit parce qu'il a expiré, soit parce qu'il n'est plus en attente. Toutes les autres raisons peuvent apparaître dans les CRL et indique que le certificat spécifié devrait être considéré révoqué.

```
id-ce-cRLReasons OBJECT IDENTIFIER ::= { id-ce 21 }

 - reasonCode ::= { CRLReason }

CRLReason ::= ENUMERATED {
 unspecified (0),
 keyCompromise (1),
 cACompromise (2),
 affiliationChanged (3),
 superseded (4),
 cessationOfOperation (5),
 certificateHold (6),
 - La valeur 7 n'est pas utilisée
 removeFromCRL (8),
 privilegeWithdrawn (9),
 aACompromise (10) }
```

Invalidity Date

La date d'invalidité est une extension d'entrée de CRL non-critique qui fournis la date à laquelle on sait, ou on suspecte qui la clé privée a été compromise ou que le certificat est devenu invalide. Cette date peut être antérieur à la date de révocation dans l'entrée de CRL, qui est la date à laquelle la CA a procédé à la révocation. Quand une révocation est postées d'abord par un fournisseur de CRL dans une CRL, la date d'invalidité peut précéder la date d'émission des premières CRL, mais la date de révocation ne devrait pas précéder la date d'émission des premières CRL. Quand cette information est disponible, les fournisseurs de CRL sont fortement encouragés à la partager avec les utilisateurs de CRL.

Les valeurs GeneralizedTime inclus dans ce champ doit être exprimés en GMT, et doivent être spécifiés et interprétés comme définis dans la section "GeneralizedTime".

```
id-ce-invalidityDate OBJECT IDENTIFIER ::= { id-ce 24 }

InvalidityDate ::= GeneralizedTime
```

Certificate Issuer

Cette extension d'entrée de CRL identifie le fournisseur de certificat associé avec une entrée dans une CRL indirecte, c'est à dire, une CRL qui a l'indicateur indirectCRL mis dans sont extension de point de distribution d'émission. Quand présent, l'extension d'entrée de CRL de fournisseur de certificat inclus un ou plusieurs noms du champ issuer et/ou de l'extension de noms alternatif du sujet du certificat qui correspond à l'entrée de la CRL. Si cette extension n'est pas présente dans la première entrée dans une CRL indirecte, le fournisseur de certificat par défaut est le fournisseur de CRL. Dans les entrées suivantes, si cette extension n'est pas présente, le fournisseur de certificat pour l'entrée est la même que pour l'entrée précédente. Ce champ est définis comme suit :

```
id-ce-certificateIssuer OBJECT IDENTIFIER ::= { id-ce 29 }

CertificateIssuer ::= GeneralNames
```

Les fournisseurs de CRL conformes doivent inclure dans cette extension le dn du champ issuer du certificat qui correspond à cette entrée de CRL. L'encodage du DN doit être identique à l'encodage utilisé dans le certificat.

Les fournisseurs de CRL doivent marquer cette extension critique vu que l'implémentation qui ignore cette extension ne peut pas correctement attribuer les entrées de CRL aux certificats. Cette spécification recommande que les implémentations reconnaissent cette extension.

Validation de chemin de certification

Les procédures de validation de chemin de certification pour les PKI de l'Internet sont basés sur l'algorithme fournis dans X.509. Le traitement des chemins de certification vérifie la liaison entrée le nom distinct du sujet et/ou le nom alternatif du sujet et la clé publique du sujet. La liaison est limitée par les contraintes qui sont spécifiées dans les certificats qui comprennent le chemin et les entrées qui sont spécifiés par les tiers de confiance. Les extensions de contraintes de base et de contraintes de stratégie permettent à la logique de traitement de chemin de certification d'automatiser le processus de prise de décision.

Cette section décris un algorithme pour valider les chemins de certification. Les implémentations conformes de cette spécification ne sont pas obligés d'implémenter cet algorithme, mais doivent fournir une fonctionnalité équivalente au fonctionnement résultant de cette procédure. Tout algorithme peut être utilisé par une implémentation particulière tant qu'elle dérive le résultat correcte.

Dans la section suivante, le texte décris la validation basique de chemin . Les chemins valides commencent avec des certificats fournis par une entité de confiance. L'algorithme nécessite la clé publique de la CA, le nom de la CA, et toutes les contraintes sur le jeu de chemins qui peuvent être validés en utilisant cette clé.

La sélection d'une entité de confiance est une décision de stratégie locale : il peut être le CA supérieur dans une PKI hiérarchique, la CA qui a fournis le certificat du vérificateur, ou toute autre CA dans le réseau PKI. La procédure de validation de chemin est la même quelque soit l'entité de confiance. En plus, différentes applications peuvent faire confiance à plusieurs entités, ou peuvent accepter des chemins qui commencent avec un jeu d'entité de confiance.

Validation de chemin basique

Ce texte décris un algorithme pour le traitement de chemin X.509. Une implémentation conforme doit inclure une procédure de traitement

de chemin X.509 qui est fonctionnellement équivalente à la sortie de cet algorithme. Cependant, le support pour certains extensions de certificat traités dans cet algorithme sont optionnels pour les implémentations conformes. Les clients qui ne supportent pas ces extensions peuvent omettre les étapes correspondantes dans l'algorithme de validation de chemin.

Par exemple, les clients ne sont pas obligés de supporter l'extension de mappage de stratégie. Les clients qui ne supportent pas cette extension peuvent omettre les étapes de validation de chemin où le traitement du policy mappings sont traités. Noter que les clients doivent rejeter le certificat s'il contient une extension critique non-supportée.

Bien que les profiles de certificat et de CRL spécifiés dans ce document spécifient des valeurs pour les champs et les extensions de certificat et de CRL considérés appropriés pour les PKI de l'Internet, l'algorithme présenté dans cette section n'est pas limité à accepter les certificats et CRL qui se conforment à cet profiles. Cependant, l'algorithme inclus uniquement les vérifications de chemint de certification valide en accord avec X.509 et n'inclus pas les vérifications de conformité des certificats et CRL avec ce profile. Bien que l'algorithme pourrait être étendu pour inclure des vérifications de conformité avec les profiles de ce document, ce profile recommande de ne pas inclure de telles vérification.

L'algorithme présenté dans cette section valide le certificat en respectant l'horodatage courant. Un application conforme peut également supporter la validation en respectant un point dans le passé. Noter que les mécanismes ne sont pas disponible pour valider un certificat en dehors de sa période de validité.

L'entité de confiance est une entrée de l'algorithme. Il n'y a pas d'obligation que la même entité de confiance soit utilisée pour valider tous les chemins de certification. Plusieurs entités de confiance peuvent être utilisés pour valider différents chemins.

Le but premier de la validation de chemin est de vérifier le lien entre un nom distinct du sujet ou un nom alternatif du sujet et la clé publique du sujet, tel que représenté dans le certificat cible, basé sur la clé publique de l'entité de confiance. Dans la plupart des cas, le certificat cible sera un certificat d'entité finale, mais le certificat cible peut être un certificat de CA tant que la clé publique du sujet est utilisée dans un but autre que vérifier la signature dans certificat à clé publique. Vérifier le lien entre le nom et la clé publique nécessite d'obtenir une séquence de certificats qui supportent ce lien. La procédure effectuée pour obtenir cette séquence de certificat est hors du scope de cette spécification.

Dans ce but, le processus de validation de chemin vérifie, entre autre, qu'un chemin de certification éventuel (une séquence de n certificats) satisfait les conditions suivantes :

(a) Pour tout x dans { 1, ..., n-1 }, le sujet des certificats x est le fournisseur du certificat x+1

(b) le certificat 1 est fournis par l'entité de confiance

(c) Le certificat n est le certificat à valider

(d) Pour tout x dans { 1, ..., n }, le certificat était valide à l'époque en question

Un certificat ne doit pas apparaître plus d'une fois dans une chemin de certification éventuel.

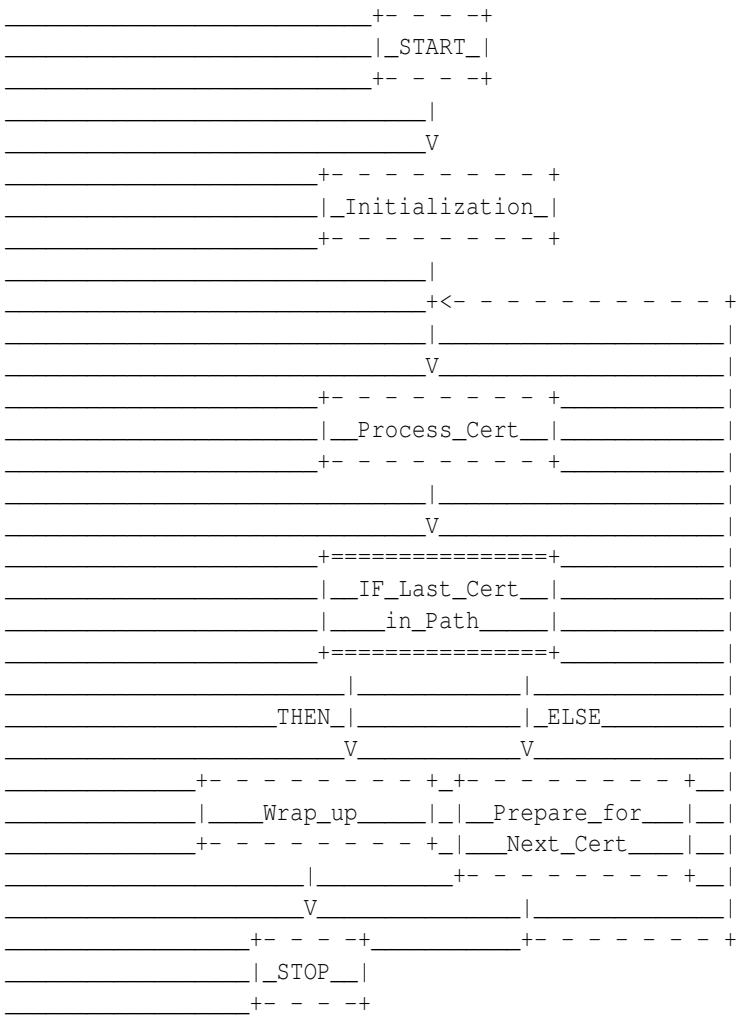
Quand l'entité de confiance est fournis sous la forme d'un certificat auto-signé, ce certificat n'est pas inclus comme partie du chemin de validation éventuel. Les informations sur l'entité de confiance est fournis en entrée de l'algorithme de validation de chemin de certification.

Un chemin de validation de certification particulier ne peut pas, cependant, être approprié pour toutes les applications. Une application peut augmenter cet algorithme pour limiter le jeu de stratégie de certificat qui sont valides pour ce chemin, basé sur l'extension de stratégies de certificat, l'extension de mappage de stratégie, l'extension de contraintes de stratégie, et inhiber l'extension anyPolicy. Pour accomplir cela, l'algorithme de validation de chemin construit une arborescence de stratégie valide. Si le jeu de stratégies de certificat qui sont valides pour ce chemin n'est pas vide, alors le résultat sera une arborescence de stratégie valide de profondeur n, sinon le résultat sera une arborescence de stratégie valide null.

Un certificat est auto-fournis si le même DN apparaît dans les champs sujets et fournisseur. En général, le fournisseur et le sujet des certificat qui constituent un chemin sont différents pour chaque certificat. Cependant, une CA peut fournir un certificat à elle-même pour supporter le changement de clé ou de stratégies de certificat. Ces certificats auto-fournis ne sont pas comptés dans l'évaluation de la longueur de chemin ou les contraintes de nom.

Cette section présente l'algorithme en 4 étapes de base : initialisation, traitement de certificats de base, préparation pour le certificat suivant, et la conclusion. Les étapes 1 et 4 sont effectuées exactement 1 seule fois, l'étape 2 est effectuée pour tous les certificats dans le

chemin, et l'étape 3 est effectuée pour tous les certificats dans le chemin excepté le certificat final. La figure suivante fournis une vue générale de cet algorithme :



Entrées

Cet algorithme assume que les 9 entrées suivantes sont fournis au traitement de chemin :

(a) Une chemin de certification éventuel de longueur n

(b) La date et heure courante

(c) user-initial-policy-set, nommant les stratégies qui sont acceptables pour l'utilisateur de certificat. Le user-initial-policy-set contient la valeur spéciale any-policy si l'utilisateur n'est pas concerné par les stratégies de certificat.

(d) Informations de l'entité de confiance qui incluent :

(1) Le nom du fournisseur de confiance

(2) L'algorithme de clé publique de confiance

(3) La clé publique de confiance

(4) Optionnellement, les paramètres de clé publique associés à la clé publique.

Les informations de l'entité de confiance peuvent être fournis à la procédure de traitement de chemin sous la forme d'un certificat auto-signé. Quand les information de l'entité de confiance sont fournies sous la fourme d'un certificat, le nom dans le champ subject est utilisé comme nom de fournisseur de confiance et le contenu du champ subjectPublicKeyInfo est utilisé comme source de l'algorithme de

clé publique de confiance et la de clé publique de confiance. Les informations d'entité de confiance sont de confiance parce qu'elles ont été délivrées à la procédure de traitement de chemin par une procédure digne de confiance. Si l'algorithme de clé publique de confiance nécessite des paramètres, alors les paramètres sont fournis avec la clé publique de confiance.

(e) initial-policy-mapping-inhibit, qui indique si le mappage de stratégie est permis dans le chemin de certification.

(f) initial-explicit-policy, qui indique si le chemin doit être valide pour au moins une des stratégie de certificat dans le user-initial-policy-set.

(g) initial-any-policy-inhibit, qui indique si l'OID anyPolicy devrait être traité s'il est inclus dans un certificat

(h) initial-permitted-subtrees, qui indique pour chaque type de nom un jeu de sous-arborescences dans lequel tous les noms du sujet dans tous les certificats dans le chemin de certification doivent aller. Cette entrée doit inclure un jeu pour chaque type de nom. Pour chaque type de nom, le jeu peut consister d'une simple sous-arborescence qui inclus tous les noms de ce type de nom ou une ou plusieurs sous-arborescence qui chacune spécifie un sous-jeu de noms de ce type de nom, ou le jeu peut être vide. Si le jeu pour un type de nom est vide, alors le chemin de certification sera considéré invalide dans tous les certificats dans le chemin de certification qui inclus un nom de ce type de nom.

(i) initial-excluded-subtrees, qui indique pour chaque type de nom un jeu de sous-arborescence dans lequel aucun nom du sujet dans les certificats du chemin de certification ne peut aller. Cette entrée inclus un jeu pour chaque type de nom. Pour chaque type de nom, le jeu peut être vide ou peut consister d'une ou plusieurs sous-arborescences qui spécifient chacun un sous-jeu des noms de ce type de nom. Si le jeu pour un type de nom est vide, aucun nom de ce type de nom n'est exclus.

Initialization

Cette phase d'initialisation établis 11 variables d'état basés sur les 9 entrées :

(a) valid_policy_tree : Une arborescence de stratégies de certificats avec leurs qualifiants optionnels ; chaque branche de l'arborescence représente une stratégie valide à ce stade de la validation du chemin de certification. Si des stratégie valides existent à ce stade dans la validation du chemin de certification, la profondeur de l'arbre est égal au nombre de certificats dans la chaîne qui a été traitée. Si des stratégies valides n'existent pas à ce stade, l'arbre est null. Une fois l'arbre mis à NULL, le traitement de stratégie s'arrête.

(1) valid_policy est un simple OID de stratégie représentant une stratégie valide pour le chemin de longueur x.

(2) qualifier_set est un jeu de qualifiants de stratégie associés avec la stratégie valide dans le certificat x.

(3) expected_policy_set contient un ou plusieurs OID de stratégie qui satisferait cette stratégie dans le certificat x+1.

La valeur initiale de valid_policy_tree est un nœud simple avec pour valid_policy anyPolicy, un qualifier_set vide, et un expected_policy_set avec la seule valeur anyPolicy. Ce nœud a une profondeur de 0.

La figure suivante est une représentation graphique de l'état initial de valid_policy_tree. Les figures additionnelles vont utiliser ce format pour décrire les changement dans le valid_policy_tree durant le traitement du chemin.

```
_________________+- - - - - - - - +
_________________|___anyPolicy____|___<- - _valid_policy
_________________+- - - - - - - - +
_________________|_______{}_______|___<- - _qualifier_set
_________________+- - - - - - - - +
_________________|__{anyPolicy}___|___<- - _expected_policy_set
_________________+- - - - - - - - +
```

(b) permitted_subtrees : un jeu de noms racine pour chaque type de nom définissant un jeu de sous-arborescence dans lequel tous les noms du sujet dans les certificats suivant dans le chemin de certificat doivent tomber. Cette variable inclus un jeu pour chaque type de nom, et la valeur initiale est initial-permitted-subtrees.

(c) excluded_subtrees : un jeu de noms racines pour chaque type de nom définissant un jeu de sous-arborescence dans lequel aucun nom du sujet dans les certificats suivant dans le chemin de certification ne peut tomber. Cette valiable inclus un jeu pour chaque type de nom, et la valeur initiale est initial-excluded-subtrees.

(d) explicit_policy : Un entier qui indique si un valid_policy_tree non-null est requis. L'entier indique le nombre de certificats non auto-fournis à traiter avant que ce requis soit imposé. Une fois définis, cette variable peut être décrémentée, mais ne peut pas être incrémentée. C'est à dire, si un certificat dans le chemin requière un valid_policy_tree non-null, un certificat ultérieur ne peut pas supprimer ce requis. Si initial-explicit-policy est mis, alors la valeur initiale est 0, sinon la valeur initiale est n+1.

(e) inhibit_anyPolicy : un entier qui indique si l'identifiant de stratégie anyPolicy est considéré comme un match. L'entier indique le nombre de certificats non auto-fournis à traiter avant que l'OID anyPolicy, si indiqué dans un certificat autre qu'un certificat intermédiaire auto-fournis, est ignoré. Une fois mis, cette variable peut être décrémentée, mais ne peut pas être incrémentée. C'est à dire, si un certificat dans le chemin inhibe le traitement de anyPolicy, un certificat ultérieur ne peut pas le permettre. Si initial-any-policy-inhibit est mis, alors la valeur initial est 0, sinon la valeur initial est n+1.

(f) policy_mapping : un entier qui indique si le mappage de stratégie est permis. L'entier indique le nombre de certificat non auto-fournis à traiter avant que le mappage de stratégie soit inhibé. Une fois mis, cette variable peut être décrémentée, mais ne peut pas être incrémenté. C'est à dire,, si un certificat dans le chemin spécifie que le mappage de stratégie n'est pa permis, il ne peut pas écrasé par un certificat ultérieur. Si initial-policy-mapping-inhibit est mis, la valeur initiale est 0, sinon la valeur est n+1.

(h) working_public_key : La clé publique utilisée pour vérifier la signature d'un certificat. working_public_key est initialiée depuis la clé publique de confiance fournis dans les informations de l'entité de confiance.

(i) working_public_key_parameters : Les paramètres associés avec la clé publique courante qui peut être requise pour vérifier une signature (en fonction de l'algorithme). working_public_key_parameters est initialisée avec les paramètres de clé publique de confiance fournis dans les informations de l'entité de confiance.

(j) working_issuer_name : Le dn du fournisseur prévu dans le prochain certificat dans la chaîne. working_issuer_name est initialisé avec le dn du founisseur fournis la les informations de l'entité de confiance.

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

Une fois ces étapes d'initialisation complétées, effectue les étapes de traitement de certificat de base.

Traitement de certificat de base

Les actions de traitement de chemin de base à effectuer pour le certificat i (pour tout i dans [1..n]) sont les suivantes :

(a) Vérifie les informations de certificat de base. Le certificat doit satisfaire chacun des points suivants :

(1) La signature dans le certificat peut être vérifié en utilisant working_public_key_algorithm, working_public_key, et working_public_key_parameters.

(2) La période de validité du certificat inclus l'heure courante

(3) Actuellement, le certificat n'est pas révoqué. Cela peut être déterminé en obtenant le CRL appropriée, par information de statut, ou par des mécanismes tiers.

(4) Le nom du fournisseur de certificat est le working_issuer_name.

(b) Si le certificat i est auto-fournis et n'est pas le certificat final dans le chemin, saute cette étape pour le certificat i. Sinon, vérifie que le nom du sujet est dans un des permitted_subtrees pour les noms distincts X.500, et vérifie que chaque noms alternatifs dans l'extension subjectAltName (critique ou non-critique) est dans un des permitted_subtrees pour ce type de nom.

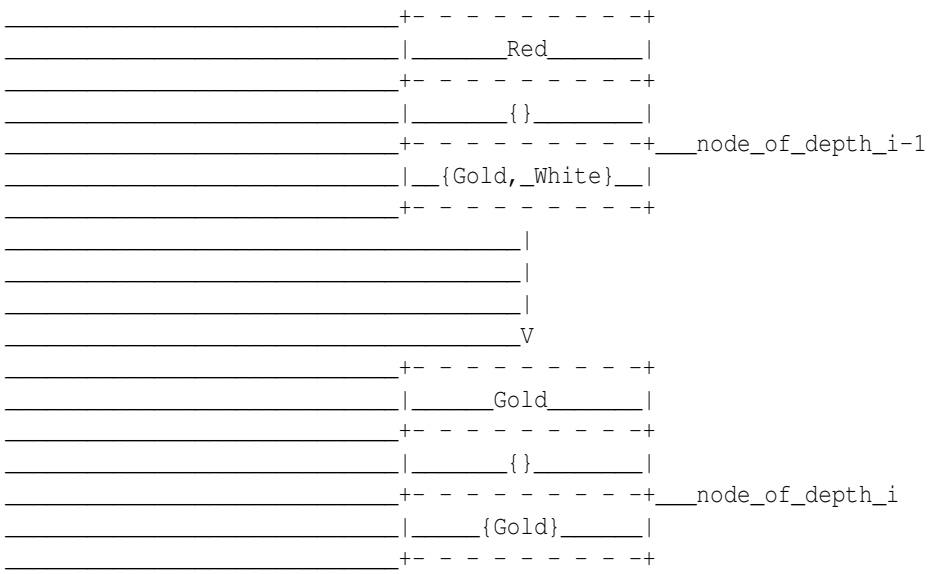
(c) Si le certificat i est auto-fournis et n'est pas le certificat final dans le chemin, saute cette étape pour le certificat i. Sinon, vérifie que le nom du sujet n'est pas dans le excluded_subtrees pour les noms distincts X.500, et vérifie que chaque noms alternatifs n'est pas dans un des excluded_subtrees pour ce type de nom.

(d) Si l'extension de stratégies de certificats est présent dans le certificat et le valid_policy_tree n'est pas null, traite les information de stratégie en effectuant les étapes suivantes dans l'ordre :

(1) Pour chaque stratégie P non égal à anyPolicy dans l'extension de stratégie de certificat, P-OID dénote l'OID pour la stratégie P et P-Q dénote le jeu de qualifiant pour la stratégie P. Effectue les étapes suivantes dans l'ordre :

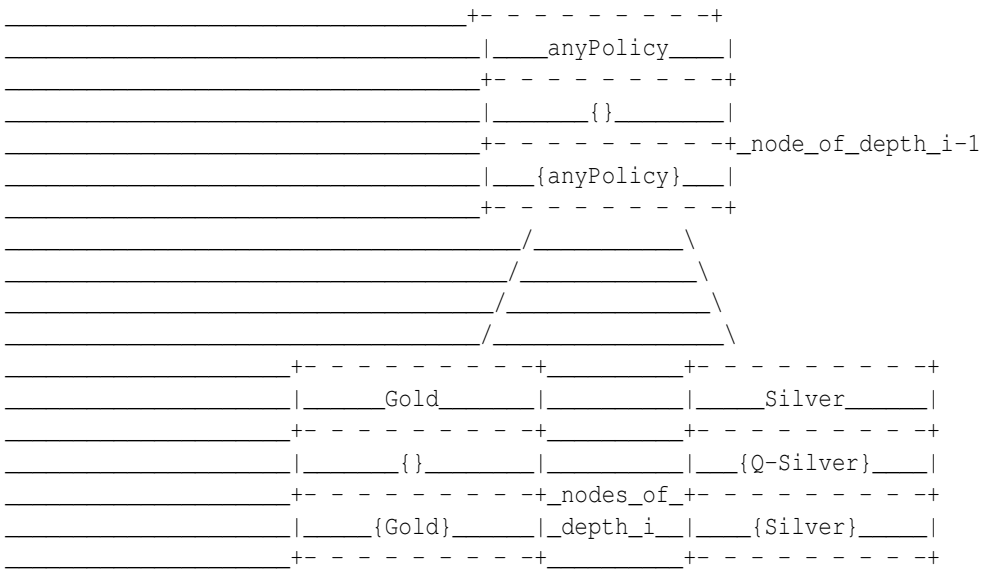
(i) Pour chaque nœud de profondeur i-1 dans le valid_policy_tree où P-OID est dans le expected_policy_set, crée un nœud enfant comme suit : définis valid_policy à P-OID, définis qualifier_set à P-Q, et définis expected_policy_set à {P-OID}.

Par exemple, considérons un valid_policy_tree avec un nœud de profondeur i-1 où expected_policy_set est {Gold, White}. Les stratégies de certificat Gold et Silver apparaissent dans l'extension de stratégie de certificat du certificat i. La stratégie Gold est matchée, mais la stratégie Silver ne l'est pas. Cette règle va générer un nœud enfant de profondeur i pour la stratégie Gold :



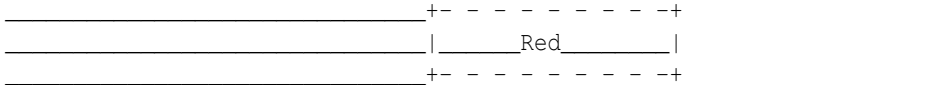
(ii) S'il n'y avait pas de correspondance dans l'étape (i) et valid_policy_tree incluais un nœud de profondeur i-1 avec valid_policy anyPolicy, génère un nœud enfant avec les valeurs suivantes : définis valid_policy à P-OID, définis qualifier_set à P-Q, et définis expected_policy_set à {P-OID}.

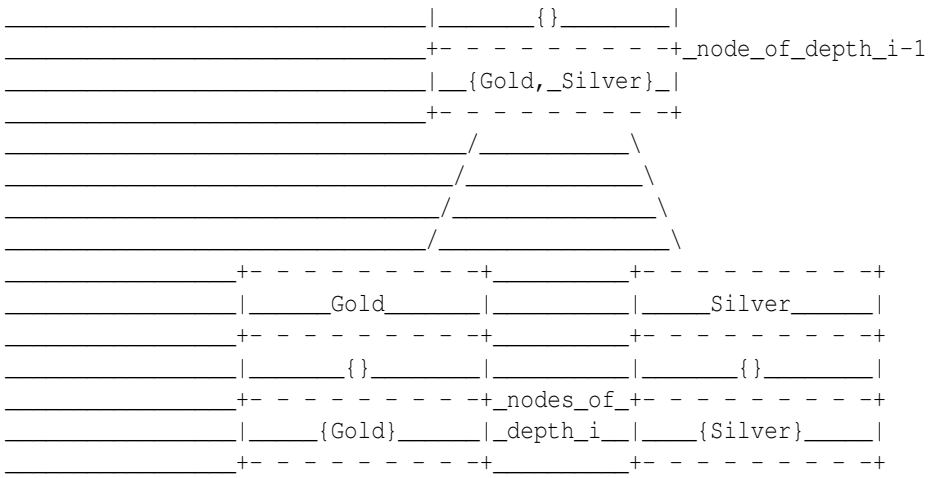
Par exemple, considérons un valid_policy_tree avec un nœud de profondeur i-1 où valid_policy est anyPolicy. Les stratégie de certificat Gold et Silver apparaîssent dans l'extension de stratégies de certificat du certificat i. La stratégie Gold n'a pas de qualifiant, mais la stratégie Silver a le qualifiant Q-Silver. Si Gold et Silver ne correspondent pas dans (i), cette règle va générer 2 nœuds enfants de profondeur i, une pour chaque stratégie :



(2) Si l'extension de stratégies de certificat inclus la stratégie anyPolicy avec le qualifiant AP-Q et soit (a) inhibit_anyPolicy est plus grand que 0 ou (b) i<n et le certificat est auto-fournis, alors : Pour chaque nœud dans valid_policy_tree de profondeur i-1, pour chaque valeur dans expected_policy_set (inclant anyPolicy) qui n'apparaît pas dans un nœud enfant, crée un nœud enfant avec les valeurs suivantes : définis valid_policy à la valeur de expected_policy_set dans le nœud parent, définis qualifier_set à AP-Q, et définis expected_policy_set à la valeur dans valid_policy de ce nœud.

Par exemple, considérons un valid_policy_tree avec un nœud de profondeur i-1 où expected_policy_set est {Gold, Silver}. anyPolicy apparaît dans l'extension de stratégie de certificat i sans qualifiant de stratégie, mais Gold et Silver n'apparaît pas. Cette règle va générer 2 nœuds enfant de profondeur i, une pour chaque stratégie :





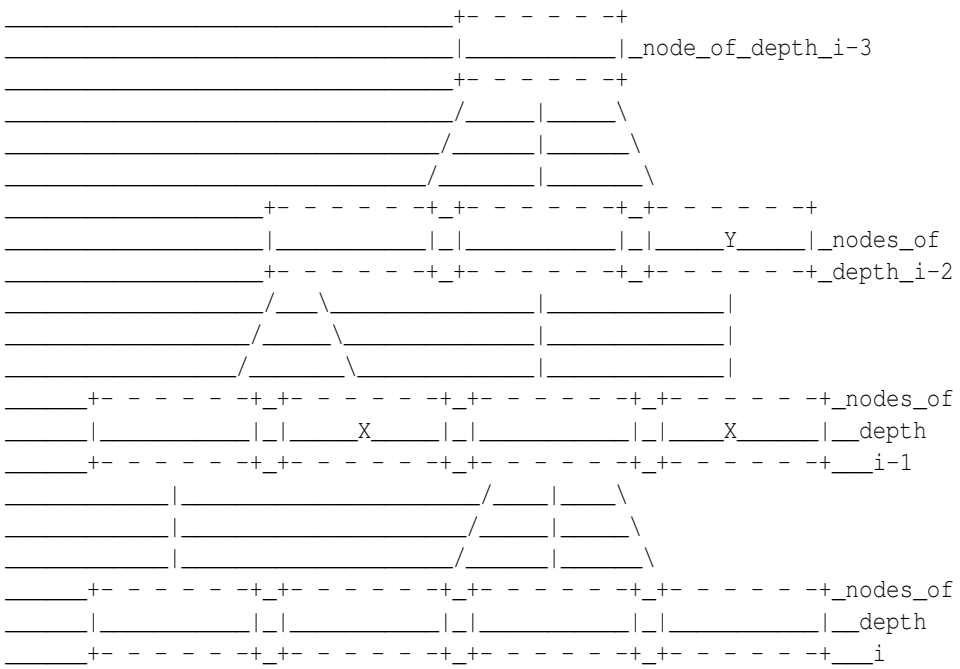
(3) S'il y a un nœud dans valid_policy_tree de profondeur i-1 ou moins sans nœud enfant, supprime ce nœud. Répète cette étape jusqu'à ce qu'il n'y ait plus de nœud de profondeur i-1 ou moins dans enfant. Par exemple, on considère valid_policy_tree ci-dessous. Les 2 nœuds à la profondeur i-1 qui sont marqués avec un 'X' n'ont pas d'enfant, et sont supprimés. Appliquer cette règle à l'arbre résultant va cause le nœud à la profondeur i-2 qui est marqué avec un 'Y' d'être supprimé. Dans l'arbre résultant, il n'y a pas de nœuds de profondeur i-1 ou moins dans enfant, et cette étape est complète.

(e) Si l'extension de stratégies de certificat n'est pas présent, définis valid_policy_tree à NULL.

(f) Vérifie que soit explicit_policy est supérieur à 0 soit valid_policy_tree n'est pas égal à NULL

Si une des étapes (a), (b), (c), ou (f) échoue, la procédure se termine, retournant une erreur indiquant un raison appropriée.

Si i n'est pas égal à n, continue en effectuant le étapes préparatoires listées dans la section suivante. Si i est égal à f, effectue les étapes listés dans la section d'après.



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 de mappage de stratégie est présente, vérifie que la valeur spéciale anyPolicy n'apparaît pas comme un issuerDomainPolicy ou un subjectDomainPolicy.

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

(1) Si la variable policy_mapping est supérieur à 0, pour chaque nœud dans valid_policy_tree de profondeur i où ID-P est le valid_policy, définis expected_policy_set au jeu de valeurs subjectDomainPolicy qui sont spécifiées comme équivalent à ID-P par l'extension de mappage de stratégie.

Si aucun nœud de profondeurs i dans valid_policy_tree n'a un valid_policy à anyPolicy, génère un nœud enfant du nœud de profondeur i-1 qui a un valid_policy de anyPolicy comme suit :

(i) Définis valid_policy à ID-P

(ii) Définis qualifier_set au jeu de qualifiant de stratégie anyPolicy dans l'extension de stratégie de certificat du certificat i

(iii) Définis expected_policy_set au jeu de valeurs subjectDomainPolicy qui sont spécifiés comme équivalent à ID-P par l'extension de mappage de stratégie.

(2) Si policy_mapping est égal à 0 :

(i) Supprime chaque nœud de profondeur i dans valid_policy_tree où ID-P est le valid_policy

(ii) S'il y a un nœud dans valid_policy_tree de profondeur i-1 ou moins sans nœud enfant, supprime ce nœud. Répète cette étape jusqu'à ce qu'il n'y ait plus de nœuds de profondeur i-1 ou moins sans enfant.

(c) Assigne le nom du sujet du certificat au working_issuer_name

(d) Assigne le subjectPublicKey du certificat au working_public_key

(e) Si le champ subjectPublicKeyInfo du certificat contient un champ algorithme avec des paramètres non-null, assigne les paramètres à la variable working_public_key_parameters. Si le champ subjectPublicKeyInfo du certificat contient un champ algorithme avec des paramètres null ou les paramètres sont omis, compare l'algorithme subjectPublicKey au working_public_key_algorithm. Si l'algorithme subjectPublicKey du certificat et le working_public_key_algorithm sont différent, définis working_public_key_parameters à null.

(f) Assigne l'algorithme de subjectPublicKey du certificat à la variable working_public_key_algorithm

(g) Si une extension de contrainte de noms est inclue dans le certificat, modifie permitted_subtrees et excluded_subtrees comme suit :

(1) Si permittedSubtrees est présent dans le certificat, définis la variable d'état permitted_subtrees à l'intersection de sa précédente valeur et de la valeur indiquée dans le champ de l'extension. Si permittedSubtrees n'inclus pas un type de nom particulier, la variable d'état permitted_subtrees est inchangée pour ce type de nom. Par exemple, l'intersection de example.com et foo.example.com est foo.example.com. Et l'intersection de example.com est example.net est un jeu vide.

(2) Si excludedSubtrees est présent dans le certificat, définis la variable d'état excluded_subtrees à l'union de sa valeur précédente valeur et la valeur indiquée dans le champ de l'extension. si excludedSubtrees n'inclus par un type de nom particulier, la variable d'état excluded_subtrees n'est pas changée pour ce type de nom. Par exemple, l'union de l'espace de nom example.con et foo.example.com est example.com. Et l'union de example.com et example.net est les 2 espaces de nom.

(h) Si le certificat i n'est pas auto-fournis :

(1) Si explicit_policy n'est pas 0, décrémente de 1

(2) Si policy_mapping n'est pas 0, décrémente de 1

(3) Si inhibit_anyPolicy n'est pas 0, décrémente de 1

(i) Si une extension de containte de stratégie est inclue dans le certificat, modifie les variables d'état explicit_policy et policy_mapping comme suit :

(1) Si requireExplicitPolicy est présent et est inférieur à explicit_policy, définis explicit_policy à la valeur de requireExplicitPolicy

(2) Si inhibitPolicyMapping est présent et est inférieur à policy_mapping, définis policy_mapping à la valeur de inhibitPolicyMapping

(j) Si l'extension inhibitAnyPolicy est inclus dans le certificat et est inférieur à inhibit_anyPolicy, définis inhibit_anyPolicy à la valeur de inhibitAnyPolicy

(k) Si le certificat i est un certificat v3, vérifie que l'extension basicConstraints est présent et que cA est mis à TRUE. (Si le certificat i est un v1 ou v2, l'application doit vérifier que le certificat i est un certification CA via un moyen externe ou rejeter le certificat. Les implémentations conformes peuvent choisir de rejeter tous les certificat intermédiaire v1 et v2)

(l) Si le certificat n'est pas auto-fournis, vérifie que max_path_length est supérieur à 0 et le décrémente de 1

(m) Si pathLenConstraint est présent dans le certifiat et est inférieur à max_path_length, définis max_path_length à la valeur de pathLenConstraint

(n) Si une extension d'utilisation de clé est présente, vérifie que le bit keyCertSign est mis.

(o) Reconnaît et traite toute autre extension critique présent dans le certificat. Traite tout autre extension non critique présente dans le certificat qui a sont importance dans le traitement du chemin.

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

si (a), (k), (l), (n), et (o) sont complétées avec succès, incrémente i et effectue le traitement de certificat de base spécifié dans la section précédente.

Procédure Wrap-Up

Pour compléter le traitement du certificat cible, effectue les étapes suivantes pour le certificat n :

(a) Si explicit_policy n'est pas 0, décrément explicit_policy de 1

(b) Si une extension de contrainte de stratégie est inclue dans le certificat et requireExplicitPolicy est présent et a une valeur de 0, définis la variable d'état explicit_policy à 0.

(c) Assigne le certificat subjectPublicKey à working_public_key

(d) Si le champ subjectPublicKeyInfo du certificat contient un champ algorithme avec un paramètre non-null, assigne les paramètres à la variable working_public_key_parameters.

Si le champ subjectPublicKeyInfo du certificat contient un champ algorithm avec des paramètre null ou des paramètre omis, compare l'algorithme subjectPublicKey avec working_public_key_algorithm. S'ils sont différents, définis working_public_key_algorithm à nul.

(e) Assigne l'algorithme subjectPublicKey à la variable working_public_key_algorithm.

(f) Reconnaît et traite toute autre extension critique présente dans le certificat n. Traite tout autre extension non critique présente qui a sont importance dans le traitement du chemin.

(g) Calcule l'intersection de valid_policy_tree et user-initial-policy-set, comme suit :

(i) si valid_policy_tree est null, l'intersection est null.

(ii) Si valid_policy_tree n'est pas null et que user-initial-policy-set est any-policy, l'intersection est tout le valid_policy_tree

(iii) Si valid_policy_tree n'est pas null et que user-initial-policy-set est any-policy, calcule l'intersection de valid_policy_tree et de user-initial-policy-set comme suit :

1. Détermine le jeu de nœuds de stratégies dont les nœuds parent ont un valid_policy à anyPolicy. C'est le valid_policy_node_set.

2. Si valid_policy d'un nœud dans valid_policy_node_set n'est pas dans le user-initial-policy-set et n'est pas anyPolicy, supprime ce nœud et tous ses enfants.

3. Si valid_policy_tree inclus un nœuds de profondeur n avec valid_policy anyPolicy et user-initial-policy-set n'est pas any-policy, effectue les étapes suivantes :

a. Définis P-Q au qualifier_set dans le nœud de profondeur n avec valid_policy anyPolicy

b. Pour chaque P-OID dans user-initial-policy-set qui n'est pas le valid_policy d'un nœud dans valid_policy_node_set, crée un nœud enfant dont le parent est le nœud de profondeur n-1 avec valid_policy anyPolicy. Définis les valeurs dans le nœud enfant comme suit : définis valid_policy à P-OID, définis le qualifier_set à P-Q, et définis le jeu expected_policy à {P-OID}.

c. Supprime le nœud de profondeur n avec le valid_policy anyPolicy.

4. S'il y a un nœud dans valid_policy_tree de profondeur n-1 ou moins sans nœud enfant, supprime ce nœud. Répète cette étape jusqu'à ce qu'il n'y ai plus de nœud de profondeur n-1 ou moins sans enfant.

Si (1) la valeur de explicit_policy est supérieur à 0 ou (2) valid_policy_tree n'est pas null, le traitement du chemin a réussit.

Sorties

Si le traitement du chemin réussit, la procédure se termine, retournant une indication de succès avec la valeur finale de valid_policy_tree, working_public_key, working_public_key_algorithm, et working_public_key_parameters.

Utiliser l'algorithme de validation de chemin

L'algorithme de validation de chemin décris le processus de validation d'un simple chemin de certification. Bien que chaque chemin de certification commence avec une ancre de confiance, il n'y a pas de pré-requis pour tous les chemins de certification validés par un système particulier qui partage une seul ancre de confiance. La sélection d'une ou plusieurs CA de confiance comme ancre de confiance est une décision locale. Un système peut fournir une de ses CA de confiance comme ancre de confiance pour un chemin particulier. L'entrée dans l'algorithme de validation de chemin peut être différent pour chaque chemin. Les entrées utilisées pour traiter un chemin peuvent refléter des pré-requis spécifique à une application ou les limitations dans la confiance accordée à une ancre de confiance particulière. Par exemple, une CA de confiance peut seulement être approuvée pour une stratégie de certificat particulier. Cette restriction peut être exprimée via les entrées de la procédure de validation de chemin.

Une implémentation peut augmenter l'algorithme présenté plus haut pour limiter le jeu de chemins de certification valide qui commencent avec une ancre de confiance particulière. Par exemple, une implémentation peut modifier l'algorithme pour appliquer une contrainte le longueur de chemin pour une ancre de confiance spécifique durant la phase d'initialisation, ou l'application peut nécessite la présence d'une forme de nom alternative particulière dans le certificat cible, ou l'application peut imposer des pré-requis dans les extensions spécifiques à l'application. L'algorithme de validation de chemin présenté dans ce document définis les conditions minimum pour qu'un chemin soit considéré valide.

Quand une CA distribut des certificat auto-signés pour spécifier des informations d'ancre de confiance, les extensions de certificat peuvent être utilisé pour spécifier les entrées recommandées à la validation de chemin. Par exemple, une extension de contrainte de stratégie pourrait être inclus dans le certificat auto-signé pour indiquer que le chemin commençant avec cet ancre de confiance devrait être validé seulement pour les stratégies spécifiées. Similairement, une extension de contrainte de nom pourrait être inclus pour indiquer que les chemins commençant avec cet ancre de confiance devraient être validés seulement pour les espaces de nom spécifiés. L'algorithme de validation de chemin présenté dans ce document n'assume pas que les informations de l'ancre de confiance est fournie dans des certificats auto-signés et ne spécifie pas les règles de traitement pour les informations additionnelles inclus dans de tels certificats.

Cependant, la rfc 5914 définis de nombreux formats pour représenter des informations d'ancre de confiance, incluant des certificats auto-signés, et la rfc 5937 fournis un exemple sur comment de telles informations peuvent être utilisées pour initialiser les entrées de validation de chemin. Les implémentations sont libre d'utiliser les informations additionnelles qui sont inclus dans la représentation de l'ancre de confiance, ou des les ignorer.

Validation de CRL

Cette section décris les étapes nécessaires pour déterminer is un certificat est révoqué quand les CRL sont le mécanisme de révocation utilisé par le fournisseur de certificat. Les implémentations conformes qui supportent les CRL ne sont pas obligés d'implémenter cet algorithme, mais elles doivent être fonctionnellement équivalente à cette procédure. Tout algorithme peut être utilisé par une implémentation tant qu'elle dérive un résultat correcte.

Cet algorithme assume que toutes les CRL nécessaire sont disponible dans un cache local. De plus, si le temps nextUpdate d'une CRL a passé, l'algorithme assume un mécanisme pour chercher une CRL courante et la placer dans le cache de CRL local.

Cet algorithme définis un jeu d'entrées, un jeu de variables d'états, et des étapes de traitement qui sont effectués pour chaque certificat dans le chemin. La sortie de l'algorithme est le statut de révocation du certificat.

Entrées de révocation

Pour supporter le traitement de la révocation, l'algorithme nécessite 2 entrées :

(a) certificate : l'algorithme nécessite le numéro de série du certificat et le nom du fournisseur pour déterminer si un certificat est dans une CRL particulière. L'extension basicConstraints est utilisé pour déterminer si le certificat fournis est associé avec un CA ou une entité finale. Si présent, l'algorithme utilise les extensions cRLDistributionPoints et freshestCRL pour déterminer de statut de révocation.

(b) use-delta : ce booléen détermine si des CRL delta sont appliquées aux CRL.

Variables d'état d'initialisation et de révocation

Pour supporter le traitement des CRL, l'algorithme nécessite les variable d'état suivants :

(a) reasons_mask : Cette variable contient le jeu de raisons de révocation supportés par les CRL et les CRL délta. Les membres légaux de ce jeu sont les valeurs de raison possible moins ceux non-spécifiés : La valeur spéciale all-reasons est utilisée pour dénoter le jeu de tous les membres légaux. Cette variable est initialisée à un jeu vide.

(b) cert_status : cette variable contient le statut du certificat. Cette variable peut être assignée 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 UNREVOKED.

(c) interim_reasons_mask : contient le jeu de raisons de révocation supportés par le CRL ou la CRL delta actuellement traitée.

Note : Dans certains environnements, il n'est pas nécessaire de vérifier tous les codes de raison. Par exemple, certains environnement sont seulement concernés par cACompromise et keyCompromise pour les certificats CA. Cet algorithme vérifie tous les codes de raison. Un traitement et des variables d'état additionnel peuvent être nécessaires pour limiter la vérification d'un sous-jeu de codes de raison.

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 soit le statut du certificat est déterminé être révoqué soit que suffisamment de CRL ont été vérifiée pour couvrir tous les codes de raison.

Pour chaque point de distribution dans l'extension de points de distribution de CRL du certificat, pour chaque CRL correspondant dans le cache de CRL local, tant que ((reasons_mask n'est pas all-reasons) et (cert_status est UNREVOKED)), effectue les étapes suivantes :

(a) Met à jour le cache de CRL local en obtenant une CRL complète, une CRL delta, ou les 2 :

(1) Si la date courante est après la valeur du champ nextUpdate de la CRL, effectue une des étapes suivantes :

(i) Si use-deltas et mis et que soit le certificat ou la CRL contient l'extension freshest CRL, obtient une CRL delta avec la valeur nextUpdate qui est après la date courante et peut être utilisée pour mettre à jours la CRL en cache local.

(ii) Met à jour le cache de CRL local avec une CR complète courante, vérifie que la date courante est avant le champ nextUpdate dans la nouvelle CRL. Si use-deltas est mis et soit le certificat soit la CRL contient l'extension freshest CRL, obtient la CRL delta courante qui peut être utilisée pour mettre à jours la nouvelle CRL complète en cache.

(2) Si la date courante est avant la valeur du champ nextUpdate, use-deltas est mis, et soit le certificat ou la CRL contient l'extension freshest CRL, puis obtient la CRL delta courante qui peut être utilisée pour mettre à jour la CRL complète en cache.

(b) Vérifie le fournisseur et le scope de la CRL complète comme suit :

(1) Si le point de distribution (DP) inclus cRLIssuer, alors vérifie que le champ issuer dans la CRL complète correspond au cRLIssuer dans le DP et que la CRL complète contient une extension de point de distribution de fournisseur avec le booléen indirectCRL inséré. Sinon, vérifie que le fournisseur de CRL correspond au fournisseur de certificat.

(2) Si la CRL complète inclus une extension de point de distribution de fournisseur (IDP), vérifie les étapes suivante :

(i) Si le nom du point de distribution est présent dans l'extension de CRL IDP et que le champ de distribution est présent dans le DP, alors vérifie qu'un des noms dans le IDP correspond à un des noms dans le DP. Si le point de distribution est présent dans l'extension de CRL IDP et que le champ distribution est omis du DP, vérifie qu'un des noms dans le IDP correspond à un des noms dans le champ cRLIssuer du DP.

(ii) Si le booléen onlyContainsUserCerts est mis dans l'extension de CRL IDP, vérifie que le certificat n'inclus pas l'extension de contrainte de base avec le booléen cA mis.

(iii) Si le booléen onlyContainsCACerts est mis dans l'extension de CRL IDP, vérifie que le certificat inclus l'extension de contraintes de base avec le booléen cA mis.

(iv) Vérifie que le booléen onlyContainsAttributeCerts n'est pas mis.

(c) Si use-deltas est mis, vérifie le fournisseur et le scope de la CRL delta comme suit :

(1) Vérifie que le fournisseur de CRL delta correspond au fournisseur de CRL complète.

(2) Si la CRL complète inclus une extension de CRL IDP, vérifie que la CRL delta contient une extension de CRL IDP correspondant à l'extension de CRL IDP. Si la CRL complète omet une extension de CRL IDP, vérifie que la CRL delta omet également une extension de CRL IDP.

(3) Vérifie que l'extension d'identifiant de clé d'autorité de CRL delta correspond à l'extension d'identifiant de clé d'autorité de la CRL complète.

(d) Calcule interim_reasons_mask pour cette CRL comme suit :

(1) Si l'extension de CRL IDP est présent et inclu onlySomeReasons et que DP inclus des raisons, définis interim_reasons_mask à l'intersection des raisons dans le DP et onlySomeReasons dans l'extension de CRL IDP.

(2) Si l'extension de CRL IDP inclus onlySomeReasons mais que DP omet les raisons, alors définis 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 que DP inclus des raisons, définis interim_reasons_mask à la valeur de reasons DP.

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

(e) Vérifie que interim_reasons_mask inclus une ou plusieurs raisons qui ne sont pas inclus dans le reasons_mask.

(f) Obtient et valide le chemin de certification pour le fournisseur de 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'utilisation de clé est présente dans le certificat du fournisseur de CRL, vérifie que le bit cRLSign est mis.

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

(h) Si use-delta est mis, valide la signature dans la CRL delta en utilisant la clé publique validée dans l'étape (f)

(i) Si use-deltas et mis, alors recherche le certificat dans la CRL delta. Si une entrée est trouvée qui correspond au fournisseur de certificat et numéro de série, met la variable cert_status à la raison indiquée comme suit :

(1) Si l'extension d'entrée de CRL de code de raison est présent, définis la variable cert_status à la valeur de l'extension d'entrée de CRL de code de raison.

(2) Si l'extension d'entrée de CRL de code de raison n'est pas présent, définis va variable cert_status à la valeur unspecified.

(j) Si (cert_status est UNREVOKED), alors cherche le certificat dans la CRL complète. Si une entrée est trouvée qui correspond au fournisseur et numéro de série du certificat, définis la variable cert_status à la raison indiquée comme décris dans l'étape (i).

(k) Si (cert_status est removeFromCRL), définis cert_status à UNREVOKED.

(l) Définis la variable d'état reasons_mask à l'union de ses précédentes valeurs et de la valeur de la variable d'état interim_reasons_mask.

Si ((reason_mask est all-reasons) OU (cert_status n'est pas UNREVOKED)), le status de révocation a été determiné, donc retourne cert_status.

Si le status de révocation n'a pas été déterminé, répète le processus avec toutes les CRL disponible non spécifiées dans un point de distribution, mais fournis par le fournisseur e certificat. Pour le traitement de telles CRL, assume un DP avec champs reasons et cRLIssuer omis et un nom de point de distribution du fournisseur de certifiat. C'est à dire la séquence de noms dans fullName est générée depuis le champ issuer du certificat aussi bien que l'extension issuerAltName du certificat. Après traitement de telles CRL, si le status de révocation n'a pas été déterminé, retourne cert_status à UNDETERMINED.

Traitement des règles pour les noms internationalisés

Les noms internationalisés peuvent être rencontrés dans de nombreux champs et extensions de certificat et de CRL, incluant des noms distinct, noms de domaine internationalisés, adresses de messagerie électronique, et IRI. Le stockage, comparaison, et présentation de tels noms nécessite un attention particulière. Certains caractères peuvent être encodés de plusieurs manières. Les même nom peuvent être représentés dans plusieurs encodage. Cette section établis les pré-requis de conformité pour le stockage ou la comparaison de chacune de ces formes de nom.

Noms internationalisés dans les noms distinct

Les attribut de nommage standard tels les noms communs, employent le type DirectoryString, qui supporte les noms internationalisés via une variété d'encodage de langues. Les implémentation conformes doivent supporter UTF8String et PrintableString. La rfc3280 requière seulement une comparaison binaire des valeurs d'attributs encodés en UTF8String, cependant, cette spécification requière une manipulation plus compréhensible. Les implémentations peuvent rencontrer des certificats et des CRL avec des noms encodés en utilisant TeletexString, BMPString, ou UniversalString, mais leur support est optionnel.

Les implémentations conformes doivent utiliser le profile LDAP StringPrep (rfc4518) comme base pour la comparaison des attributs de noms distinct encodés avec PrintableString ou UTF8String. Les implémentations conforme doivent supporter la comparaison de noms en utilisant caseIgnoreMatch. Le support pour les types d'attributs qui utilisent d'autres règles de correspondance d'égalité est optionnel.

Avant de comparer les noms en utilisant la règle de correspondance caseIgnoreMatch, les implémentation doivent effectuer l'algorithme de conformité en 6 étapes décrit dans la rfc4518 pour chaque attribut de type DirectoryString, avec les clarifications suivantes :

- Dans l'étape 2, le mapparge devrait inclure de cas spécifié dans la rfc3454 Appendix B.2.
- Dans l'étape 6, La suppression des caractères insignifiant, effectue la compression des espaces blanc comme spécifié dans la section Insignificant Space Handling de la rfc 4518.

En effectuant l'algorithme de préparation de chaîne, les attributs doivent être traités comme valeurs stockées.

2 attributs de nommage correspondent si les types d'attributs sont les même et les valeurs des attributs sont un exact match après traitement avec l'algorithme de préparation de chaîne. 2 noms distincts relatifs RDN1 et RDN2 correspondent s'ils ont le même nombre d'attributs de nommage pour chaque attribut de nommage dans RDN1, il y a un attribut de nommage correspondant dans RDN2, et les RDN apparaissent dans le même ordre. Un DN1 est dans le subtree définis dans DN2 si DN1 contient au moins autant de RDN que DN2, et DN1 et DN2 correspondent quand les DNS restants dans DN1 sont ignorés.

Noms de domaine internationalisés dans GeneralName

Les noms de domaine internationalisés (IDN) peuvent être inclus dans les certificats et les CRL dans les extensions subjectAltName et issuerAltName, les extensions de contraintes de nom, extensions d'accès aux informations d'autorité, extensions d'accès aux informations du sujet, extensions de point de distribution de CRL, et les extensions de point de distribution de fournisseur. Chacune de ces extensions utilise le type GeneralName ; Un choix dans GeneralName est le champ dNSName, qui est définis comme type IA5String.

IA5String est limité au jeu de caractères ASCII. Pour adapter les noms de domaine internationalisés dans la structure courante, les implémentations conformes doivent convertir les noms de domaine internationalisés au format ASCII Compatible Encoding (ACE) comme spécifié dans la rfc 3490 avant de stocker le champ dNSName. Spécifiquement, les implémentations conformes doivent effectuer les opérations de conversion spécifiés dans la rfc 3490, avec les clarifications suivantes :

- Dans l'étape 1, le nom de domaine devrait être considéré comme chaîne stockée, c'est à dire que le flag AllowUnassigned n'est pas mis.
- Dans l'étape 3, définis le flag appelé "UseSTD3ASCIIRules"
- Dans l'étape 4, traite chaque label avec l'opération "ToASCII"

- Dans l'étape 5, change tous les séparateurs de label à U+002E.

En comparant les noms DNS pour l'égalité, les implémentations conforme doivent effectuer une correspondance sensible à la casse exacte sur tout le nom DNS. En évaluant les contraintes de nom, les implémentations conformes doivent effectuer une correspondance sensible à la casse exacte sur une base label par label. Comme spécifié dans la section "name constraints", tous nom DNS que peut être construit en ajoutant des labels à gauche du nom de domaine donné comme contrainte est considéré être dans le subtree indiqué.

Les implémentations devraient convertir les IDN en Unicode avant affichage. Spécifiquement, les application conformes devraient effectuer l'opération de conversion spécifiée dans la rfc3490, avec les clarifications suivante :

- Dans l'étape 1, le nom de domaine devrait être considéré comme chaîne stockée, c'est à dire que le flag AllowUnassigned n'est pas mis.
- Dans l'étape 3, définis le flag appelé "UseSTD3ASCIIRules"
- Dans l'étape 4, traite chaque label avec l'opération "ToUnicode"
- Saute l'étape 5.

Note : les implémentations doivent autoriser l'augmentation de l'espace requis pour les IDN. Un label IDN ACE va commencer avec les 4 caractères additionnels "xn–" et peut nécessiter jusqu'à 5 caractères ASCII pour spécifier un simple caractère internationalisé.

Noms de domaine internationalisés dans les noms distincts

Les noms de domaine peuvent également être représentés en noms distinct, en utilisant des composantes domaine dans le champ subject, issuer, l'extension subjectAltName, ou issuerAltName. Comme avec dNSName dans le type GeneralName, la valeur de cet attribut est définis comme IA5String. Chaque attribut domainComponent représente un simple lable. Pour représenter un label depuis un IDN en nom distinct, l'implémentation doit effectuer la conversion de label "ToASCII" spécifié dans la rfc3490. Le label devrait être considéré comme chaîne stockée, c'est à dire que le flag AllowUnassigned n'est pas mis.

Les implémentations conformes devraient convertir les labels ACE en Unicode avant affichage. Spécifiquement, les implémentations conformes devraient effectuer l'opération de conversion "ToUnicode" spécifiée dans la rfc3490, sur chaque label ACE avant affichage.

Identifiants de ressources internationalisés

Les IRI sont le complément internationalisé des URI. Les IRI sont des séquences de caractères Unicode, alors que les URI sont des séquences de caractères ASCII. La rfc3987 définis un mappage des IRI en URI. Bien que les IRI ne sont pas encodés directement dans les champs et extensions de certificat et CRL, leur version mappé URI peuvent êrte inclus dans les certificats et CRL. Les URI peuvent apparaître dans les extensions subjecAltName, issuerAltName, contraintes de noms, accès aux informations de l'autorité, accès aux informations du sujet, point de distribution de fournisseur et de CRL. Chacune de ces extensions utilisent le type GeneralName, les URI sont encodés dans le champ uniformResourceIdentifier dans GeneralName, qui est défini en type IA5String.

Pour représenter les IRI dans la structure courante, les implémentations courantes doivent mapper les IRI en URI comme spécifiés dans la rfc 3987, avec les clarifiations suivantes :

- Dans l'étape 1, génère une séquence de caractère UCS du format IRI original normalisant en accord au NFC comme spécifié dans la variante b
- Effectue l'étape 2 en utilisant la sortie de l'étape 1.

Les implémentations ne doivent pas convertir le composant ireg-name avant d'effectuer l'étape 2.

Avant que les URI puissent être comparés, les implémentations conformes doivent effectuer une combinaison des techniques de normalisation basé sur la syntaxe et sur le schéma décris dans la rfc3987. Spécifiquement, les implémentations conformes doivent préparer les URI pour les comparaisons comme suit :

Étape 1 Quand les IRI permettent l'utilisation d'IDN, ces noms doivent être convertis en ACE comme spécifié plus haut

Étape 2 Le schéma et l'hôte sont normalisés en minuscule comme spécifié dans la rfc3987

Étape 3 Effectue une normalisation percent-encoding, comme spécifié dans la rfc3987

Étape 4 Effectue une normalisation de segment de chemin, comme spécifié dans la rfc3987

Étape 5 Si reconnus, L'implémentation doit effectuer une normalisation basée sur le schéma, comme spécifié dans la rfc3987

Les implémentations conformes doivent reconnaître et effectuer une normalisation basée sur les schéma pour les schéma suivant : ldap, http, https, et ftp. Si le schéma n'est pas reconnu, l'étape 5 est omise.

En comparant les URI pour l'équivalence, les implémentations conformes devraient effectuer une correspondance d'égalité sensible à la casse.. Les implémentations devraient convertir les URI en Unicode avant affichaqe.

Adresses de messagerie électronique internationalisées

Les adresses de messagerie électronique peuvent être inclus dans les certificats et les CRL dans les extension subjecAltName et issuerAltName, contraintes de noms, accès aux information d'autorité et du sujet, et point de distribution du fournisseur et de CRL. chacune de ces extensions utilise la construction GeneralName ; GeneralName inclus rfc822Name, qui est définis comme type IA5String. Pour spécifier des adresses email avec des noms de domaine internationalisés en utilisant la structure courante, les implémentations conforme doivent convertir les adresses en représentation ASCII.

Quand la partie hôte (le domaine de la boîte mail) contient un nom internationalisé, le nom de domaine doit être convertis depuis un IDN en ACE. 2 adresses email correspondent si :

1) La partie locale du chaque nom est un exact match, et

2) la partie hôte de chaque nom correspond en utilisant une comparaison ASCII insensible à la casse.

Les implémentations devraient convertir la partie hôte des adresses email internationalisés spécifiés dans ces extension en Unicode avant affichage. Spécifiquement, les implémentations conformes devraient effectuer la conversion de la partie hôte de la boîte mail comme décris dans la section des IDN dans GeneralName.

Considérations de sécurité

Vu que les certificats et les CRL sont signés numériquement, aucun service d'intégrité additionnel n'est nécessaire, ni de conserver les certificats ou les CRL secrets, et les accès non-restreins et anonymes aux certificats et CRL n'a pas d'implication de sécurité.

Cependant, des facteurs de sécurité en dehors du scope de cette spécification affectent l'assurance fournie par les certificats utilisateurs. Cette section met en avant les problèmes de sécurité hautement critique à considérer par les implémenteurs, administrateurs et les utilisateurs.

L'utilisation d'une simple paire de clé pour la signature et d'autres buts est fortement découragée. L'utilisation de paires de clé séparés pour la signature et la gestion de clé fournis de nombreux bénéfices aux utilisateurs. Les ramifications associées avec la pertes ou la divulgation d'une clé de signature sont différents de la perte ou la divulgation d'une clé de gestion de clé. Utiliser des paires de clé différente permet une réponse balancée et différente. Similairement, différentes périodes de validité ou longueur de clé pour chaque paire de clé peuvent être approprié dans certains environnements. Malheureusement, certaines applications utilisent une simple paire de clé pour la signature et la gestion de clé.

La protection des clés privées conférée aux clés privées est un facteur critique. À petite échelle, les utilisateurs qui ne protègent pas leurs clés privée va permettre à un attaquant de les usurper ou déchiffrer leurs informations personnelles. À plus grande échelle, la clé privée de signature d'une CA compromise peut avoir un effet catastrophique. Si un attaquant obtient une clé privée, il peut émettre de faux certificats et CRL. L'existance de faux certificats et CRL va détruire la confidence dans le système. Si un tel compromis est détecté, tous les certificats émis par la CA compromise doivent être révoqués, bloquant les services entre ses utilisateurs et les utilisateurs d'autres CA. Reconstruire

après un tel compromis sera problématique, donc les CA doivent implémenter une combinaison de mécanismes fort et de procédures de gestion appropriés (ex : séparation des privilèges) pour éviter un incident.

La perte d'une clé de signature privée de CA peut également être problématique. La CA ne sera pas capable de produire de CRL ou d'effectuer des remplacements de clé. Les CA devraient maintenir une sauvegarde sûre pour les clés de signature. La sécurité des procédures de sauvegarde de clé est un facteur critique pour éviter la compromission de clé.

La disponibilité des informations de révocation affecte le degré d'assurance qui est placé dans un certificat. Bien que les certificats expirent naturellement, des évènements peuvent se produire durant sa durée de vie qui supprime la liaison entre le sujet et sa clé publique. Si l'information de révocation n'est pas disponible, l'assurance avec la liaison est clairement réduite. Les parties de confiance peuvent ne pas être capable de traiter toutes les extensions critiques qui peuvent apparaître dans une CRL. Les CA devraient faire très attention créant des informations de révocation uniquement disponibles via des extensions critiques, particulièrement si le support pour ces extensions n'est pas mandaté par ce profile.

Par exemple, si les informations de révocation sont fournis en utilisant une combinaison de CRL delta et le CRL complètes, et que la CRL delta est fournie plus fréquemment que les CRL complètes, les parties de confiance qui ne peuvent pas manipuler les extensions critiques liées au traitement de la CRL delta ne seront pas capable d'obtenir les informations de révocations les plus récentes. Alternativement, il une CRL complète est fournis au même moment qu'une CRL delta, Les informations de révocation seront disponibles à toutes les parties. Similairement, les implémentations du mécanisme de validation de chemin de certification qui omettent la vérification de révocation fournissent moint d'assurance qui celles qui le supporte.

L'algorithme de validation de chemin de certification dépend de la connaissance des clé publiques (et autres informations) sur une ou plusieurs CA de confiance. La décision de la confiance en un CA est une décision importante vu qu'elle détermine la confiance accordée à un certificat. La distribution authentifiée de clés publique de CA (généralement sous la forme de certificat auto-signé) est un processus à part critique qui est en dehors de cette spécification.

En plus, quand une clé est compromise ou qu'une CA en peut plus être de confiance, l'utilisateur doit modifier les informations fournies à la routine de validation de chemin. Sélectionner trop de CA de confiance rend cette maintenance difficile.

La qualité des implémentations qui traitent les certificats affectent également le degré d'assurance fournis. L'algorithme de validation de chemin s'assure de l'intégrité des informations de CA de confiance, et spécialement l'intégrité des clés publiques associées avec ces CA. En substituant les clés publiques pour lequel un attaquant a la clé privée lui permet de faire accepter de faux certificats à un utilisateur.

La liaison entre une clé et le sujet du certificat ne peut pas être plus fort que l'implémentation du module cryptographique et des algorithmes utilisés pour générer la signature. Des longueurs de clé courtes ou des algorithmes de hashage faible limitent l'utilité des certificats. Les CA sont encouragés à noter les progrès en cryptologie afin d'employer des techniques cryptographiques fortes. De plus, les CA devraient refuser de fournir des certificats à de CA ou des entités finales qui génèrent des signatures faibles.

Une application inconsistante de règle de comparaison de nom peut résulter dans l'acceptation de chemins de certification invalides ou le rejet de chemins valides. Les séries de spécification X.500 définissent des règles pour comparer les noms distincts qui nécessitent la comparaison de chaînes sans regarder là casse, le jeu de caractère, espaces blancs multi-caractères, ou espaces blancs en début ou fin. Cette spécification relaxe ces requis, nécessitant le support d'une comparaison binaire au minimum.

Les CA doivent encoder les noms distinct dans le champ sujet d'un certificat CA identiquement au nom distinct dans le champ issuer dans les certificats fournies par cette CA. Si les CA utilisent des encodages différents, les implémentations peuvent échouer à reconnaître les chaînes de nom pour les chemins qui incluent ce certificat. En conséquence, les chemins valides pourraient être rejetés.

En plus, les contraintes de nom pour les noms distincts doivent être identiques à l'encodage utilisé dans le champ subjet ou l'extension subjectAltName. Sinon, les contraintes de noms définis comme excludedSubtrees ne matcheront pas et les chemins invalides seront acceptés et les contraintes de noms exprimés comme permittedSubtrees ne vont pas matcher et les chemins valides seront rejeter. Pour éviter d'accepter les chemins invalides, les CA devraient définir les contraintes de noms pour les noms distincts comme permittedSubtrees si possibles.

En général, utiliser l'extension nameConstraints pour contraindre une forme de nom n'offre pas de protection contre l'utilisation d'autre formes de nom.

Bien que X.509 mandate que les noms soient non ambigus, il y a un risque que 2 autorités non liées fournissent des certificats et/ou des

CRL sous le même nom de fournisseur. Un moyen de réduire ces problèmes et les problèmes de sécurité liés aux collisions de nom de fournisseur, les noms de CA et de fournisseurs de CRL devraient être formé d'une manière qui réduit la probabilité des collisions de noms. Les implémenteurs devraient prendre en compte l'existence possible de multiples CA et fournisseurs de CRL avec le même nom. Au minimum, les implémentations validant les CRL doivent s'assurer que le chemin de certification d'un certificat et du chemin de certification du fournisseur de CRL utilisés pour valider ce certificat se terminent à l'ancre de confiance.

Bien que la partie locale d'une adresse de messagerie électronique est sensible à la casse, les valeurs de l'attribut emailAddress sont insensibles à la casse. Il y un risque que 2 adresses emails différentes soient traitées comme des adresse identiques. Les implémenteurs ne devraient pas inclure une adresses email dans l'attribut emailAddress si le serveur de messagerie traite la partie locale des adresses email comme sensible à la casse.

Les implémenteurs devraient s'assurer des risques liés si les points de distribution de CRL ou les extensions d'accès aux informations d'autorité de certificats ou de CRL corrompus contiennent les liens vers du code malicieux. Les implémenteurs devraient toujours prendre les étapes consistant à valider les données extraites afin de s'assurer que les données sont correctement formées.

Quand les certificats incluent une extension cRLDistributionPoints avec une URI https ou un schéma similaire, des dépendances circulaires peuvent être introduits. Le tier de confiance est forcé d'effectuer une validation de chemin additionnel pour obtenir la CRL requise pour compléter la validation de chemin initiale. Des conditions similaires peuvent égalemente être créés avec une URI https dans les extensions authorityInfoAccess ou subjectInfoAccess.

Les CA ne devraient pas inclure d'URI qui spécifient https, ldaps, ou des schémas similaires dans les extensions. Les CA qui incluent une de ces extension doivent s'assurer que le certificat du serveur peut être validé dans utiliser d'informations pointé par l'URI. Les tiers de confiance qui choisissent de valider le certificat du serveur en obtenant les informations pointés par une URI https dans les extensions cRLDistributionPoints, authorityInfoAccess, ou subjectInfoAccess doivent être préparés pour cette possibilité.

Les certificats auto-fournis fournissent des CA avec un mécanisme automatisé pour indiques les changements dans les opération de la CA. En particulier, les certificat auto-signés peuvent être utilisés pour implémenter un changement de paire de clé non-compromis à un autre. Les procédures détaillée pour la mise à jours de clé de CA sont spécifiés dans la rfc4210, où la CA protège sa nouvelle clé publique en utilisant sa précédente clé privée et vice versa en utilisa 2 certificats auto-fournis. Les implémentations client conformes vont traiter le certificat auto-signé et déterminer si les certificats fournis sous la nouvelle clé peut être trusté. Les certificats auto-fournis peuvent être utilisé pour supporter d'autres changements dans les opérations de CA, tel que l'ajout de stratégies, en utilisant des procédures similaires.

Certaines anciennes implémentations supportent les noms encodés en ISO 8859-1 (Latin1String) mais les tag en TeletexString. TeletexString encode un jeu de caractère plus grand que ISO 8859-1, mais il encode certains caractères différemment. Les règles de comparaison spécifiée plus haut assument que les TeletexStrings sont encodés comme décris dans le standard ASN.1. En comparant les noms encodés en Latin1String, de faux positifs et négatifs sont possibles.

Quand des chaînes sont mappées depuis des représentations interne vers de représentation visuelle, 2 chaînes différentes peuvent parfois avoir la même représentation visuelle. Cela peut se produire pour plusieurs raisons, incluant l'utilisation de glyphes similaires et l'utilisation de caractères composés. Les fournisseurs de certificats et les tiers de confiance doivent gérer cette situation.