Groupe de travail Réseau
Request for Comments : 4158
Catégorie : Information

M. Cooper, Orion Security Solutions
Y. Dzambasow, A&N Associates
P. Hesse, Gemini Security Solutions
S. Joseph, Van Dyke Technologies
R. Nicholas, BAE Systems

Traduction Claude Brière de L'Isle
septembre 2005

Infrastructure de clés publiques X.509 pour l'Internet : construction du chemin de certification

Statut de ce mémoire
Le présent mémoire apporte des informations pour la communauté de l'Internet. Il ne spécifie aucune sorte de norme de l'Internet. La distribution du présent mémoire n'est soumise à aucune restriction.

Résumé
Le présent document apporte des directives et des recommandations aux développeurs de constructions de chemins de certification de clés publiques X.509 au sein de leurs applications. En suivant les directives et recommandations définies dans ce document, un développeur d'application va plus probablement développer une application à capacité de certificat X.509 robuste qui peut construire des chemins de certification valides dans une large gamme d'environnements de PKI.

Table des Matières

1. Introduction

Les certificats de clé publique [X.509] sont devenus une méthode acceptée pour lier en toute sécurité l'identité d'un individu ou appareil à une clé publique, afin de prendre en charge les opérations de cryptographie à clé publique telles que les vérifications de signature numérique et le chiffrement fondé sur la clé publique. Cependant, avant d'utiliser la clé publique contenue dans un certificat, une application doit d'abord déterminer l'authenticité de ce certificat, et précisément, la validité de tous les certificats qui conduisent à une clé publique de confiance, appelée une ancre de confiance. Par la validation de ce chemin de certification, l'assertion du lien faite entre l'identité et la clé publique dans chacun des certificats peut être retracée jusqu'à une seule ancre de confiance.

Le processus par lequel une application détermine cette authenticité d'un certificat est appelée processus de chemin de certification. Le processus de chemin de certification établit une chaîne de confiance entre une ancre de confiance et un certificat. Cette chaîne de confiance se compose d'une série de certificats appelée un chemin de certification. Un chemin de certification commence par un certificat dont la signature peut être vérifiée en utilisant une ancre de confiance et se termine par le certificat cible. Le traitement de chemin englobe de construire et valider le chemin de certification pour déterminer si un certificat cible est approprié pour être utilisé dans un contexte d'application particulier. Voir plus d'informations au paragraphe 3.2 de la [RFC3280] sur les chemins de certification et la confiance.

1.1 Motivation

De nombreux autres documents (comme la [RFC3280]) couvrent en détails les exigences et les procédures de validation de chemin de certification mais ne discutent pas de la construction de chemin de certification parce que les moyens utilisés pour trouver le chemin n'affectent pas sa validation. Le présent document est donc un effort pour fournir des lignes directrices utiles pour les développeurs de mises en œuvre de construction de chemin de certification.

De plus, le besoin de développer des chemins de certification complexes est croissant. De nombreuses PKI utilisent maintenant des structures complexes (voir au paragraphe 1.5) plutôt que de simples hiérarchies. En outre, certaines entreprises s'éloignent graduellement des listes de confiance remplies avec de nombreuses ancres de confiance, et vont vers une infrastructure avec une ancre de confiance et de nombreuses relations de certificats croisés. Le présent document fournit des informations utiles pour développer des chemins de certification dans ces situations plus compliquées.

1.2 Objet

Le présent document donne des informations et des lignes directrices pour la construction de chemins de certification. Il n'y a pas de spécification d'exigences ou de protocole dans ce document. Le présent document propose de nombreuses options pour effectuer la construction de chemins de certification, par opposition à une seule façon particulière. Le présent document s'appuie sur les expériences des auteurs avec les chemins de certification complexes existants pour offrir des conseils et des recommandations aux développeurs qui intègrent la prise en charge des certificats [X.509] dans leurs applications.

De plus, le présent document suggère d'utiliser une approche effective générale de la construction de chemins qui implique une traversée de première profondeur d'arborescence. Bien que les auteurs estiment que cette approche offre l'équilibre entre la simplicité de conception et des capacités très effectives de construction de chemin neutre à l'égard de l'infrastructure, l'algorithme n'est rien d'autre qu'une approche suggérée. D'autres approches (par exemple, traversés d'arborescence d'abord en largeur) existent et peuvent se montrer plus efficaces sous certaines conditions. La validation de chemin de certification est décrite en détails dans [X.509] et [RFC3280] et n'est pas répétée dans ce document.

Le présent document ne donne pas de directive sur la construction du chemin de certification d'un certificat d'entité d'extrémité à un certificat de mandataire comme décrit dans la [RFC3820].

1.3 Terminologie

Les termes utilisés tout au long du présent document seront employés de la façon suivante :

Construction dans la direction de l'avant : processus de construction d'un chemin de certification du certificat cible à une ancre de confiance. "Vers l'avant" est l'ancien nom de l'élément crossCertificatePair de "issuedToThisCA".

Construction dans la direction inverse : processus de construction d'un chemin de certification d'une ancre de confiance au certificat cible. "Inverse" est l'ancien nom de l'élément crossCertificatePair de "issuedByThisCA".

Certificat : lien numérique qui ne peut pas être contrefait entre une entité désignée et une clé publique.

Graphe de certificat : graphe qui représente la PKI entière (ou toutes les PKI à certification croisée) dans laquelle toutes les entités désignées sont vues comme des nœuds et tous les certificats sont vus comme des arcs entre les nœuds.

Système de traitement de certificats : application ou appareil qui effectue les fonctions de construction de chemin et de validation de chemin de certification.

Autorité de certification (CA) : entité qui produit et gère les certificats.

Chemin de certification : liste ordonnée de certificats commençant par un certificat signé par une ancre de confiance et se terminant par le certificat cible.

Construction de chemin de certification : processus utilisé pour assembler le chemin de certification entre l'ancre de confiance et le certificat cible.

Validation de chemin de certification : processus qui vérifie le lien entre le sujet et la clé publique de sujet défini dans le certificat cible, utilisant une ancre de confiance et un ensemble de contraintes connues.

Liste de révocation de certificat (CRL) : liste signée horodatée qui identifie un ensemble de certificats qui ne sont plus considérés comme valides par le producteur de certificats.

Certificat de signataire de CRL : certificat spécifique qui peut être utilisé pour vérifier la signature sur une CRL produite par, ou au nom d'une CA spécifique.

Certificat croisé : certificat produit par une CA à une autre CA pour les besoins de l'établissement d'une relation de confiance entre les deux CA.

Certification croisée : acte de production de certificats croisés.

Arbre de décision : quand le logiciel de construction de chemins a à choisir entre plusieurs certificats, et doit prendre une décision, c'est la collection des choix possibles.

Répertoire : généralement utilisé pour se référer à un répertoire accessible par LDAP pour des informations de certificats et de PKI. Le terme peut aussi être utilisé de façon générique pour se référer à tout répertoire qui mémorise des certificats.

Entité terminale : détenteur d'une clé privée et du certificat correspondant, dont l'identité est définie comme sujet du certificat. Les entités terminales qui sont des humains sont souvent appelées "souscripteurs".

Indicateur Is-revocation-signer : fanion booléen fourni par le logiciel de construction de chemin. Si il est établi, cela indique que le certificat cible est un certificat de signataire de révocation pour une CA spécifique. Par exemple, si on construit un chemin de certification pour un certificat de signataire de CRL indirect, ce fanion serait établi.

PKI locale : ensemble des composants et des données de PKI (certificats, répertoires, CRL, etc.) qui sont créés et utilisés par l'organisation qui utilise le certificat. En général, ce concept se réfère aux composants qui sont dans une étroite proximité avec l'application qui utilise le certificat. L'hypothèse est que les données locales sont plus aisément accessibles et/ou moins coûteuses à restituer que des données de PKI non locales.

Domaine local : voir PKI locale.

Nœud (dans un graphe de certificat) : collection de certificats ayant des noms distinctifs de sujet identiques.

Protocole d'état de certificat en ligne (OCSP, *Online Certificate Status Protocol*) : protocole Internet utilisé par un client pour obtenir l'état de révocation d'un certificat de la part d'un serveur.

Certificat de signataire de réponse OCSP : certificat spécifique qui peut être utilisé pour vérifier la signature sur une réponse OCSP. Cette réponse peut être fournie par la CA, au nom de la CA, ou par un signataire différent comme déterminé par la politique locale du consommateur d'assertions.

Infrastructure de clé publique (PKI, *Public Key Infrastructure*) : ensemble de matériels, logiciels, personnels, politiques, et procédures utilisé par une CA pour produire et gérer les certificats.

Consommateur d'assertions (RP, *Relying Party*) : application ou entité qui traite les certificats pour les besoins 1) de vérification d'une signature numérique, 2) d'authentification d'une autre entité, ou 3) d'établissement de communications confidentielles.

Certificat de signataire de révocation : se réfère collectivement au certificat de signataire de CRL ou au certificat de signataire de réponse OCSP.

Certificat cible : certificat qui est à valider par un consommateur d'assertions. C'est le "certificat ciblé pour validation". Bien que ce soit fréquemment l'entité d'extrémité ou un nœud feuille dans la structure de PKI, ce pourrait aussi être un certificat de CA si un certificat de CA est à valider. (Par exemple, ce pourrait être pour les besoins de la construction et la validation d'un chemin de certification pour le signataire d'une CRL.)

Confiance (de clé publique) : dans la perspective de ce document, une clé publique est considérée comme de confiance si le certificat qui contient la clé publique peut être validé conformément aux procédures de la [RFC3280].

Liste de confiance : liste des ancres de confiance.

Ancre de confiance : combinaison d'une clé publique de confiance et du nom de l'entité à laquelle appartient la clé privée correspondante.

Certificat d'ancre de confiance : certificat auto signé pour une ancre de confiance qui est utilisée dans le traitement du chemin de certification.

Usager : individu qui utilise un système de traitement de certificat. Le présent document se réfère à certains cas dans lesquels des usagers peuvent ou non être sollicités avec des informations ou demandes, selon la mise en œuvre du système de traitement de certificat.

1.4 Notation

Le présent document utilise quelques notations courantes dans les diagrammes et exemples.
La première est le symbole de la flèche (–>) qui représente la production d'un certificat d'une entité à une autre. Par exemple, si l'entité H veut produire un certificat à l'entité K, c'est noté H–>K.
Parfois, il est nécessaire de spécifier le sujet et le producteur d'un certain certificat. Si l'entité H devait produire un certificat à l'entité K, cela peut être noté K(H).
Ces notations peuvent être combinées pour noter des chemins de certification compliqués comme C(D)–>B(C)–>A(B).

1.5 Vue d'ensemble des structures de PKI

Lors de la vérification de certificats de clé publique [X.509] l'application qui effectue la vérification n'a souvent pas connaissance de l'infrastructure de clé publique (PKI) sous-jacente qui a produit le certificat. Les structures de PKI peuvent aller de structures hiérarchiques très simples à des structures complexes comme des architectures maillées impliquant plusieurs ponts (voir le paragraphe 1.5.4). Ces structures définissent les types de chemins de certification qui peuvent être construits et validés par une application [MINHPKIS]. Ce paragraphe décrit quatre structures de PKI courantes.

1.5.1 Structures hiérarchiques

Une PKI hiérarchique, décrite dans la Figure 1, est celle dans laquelle toutes les entités terminales et les consommateurs d'assertions utilisent une seule "CA racine" comme leur ancre de confiance. Si la hiérarchie a plusieurs niveaux, la CA racine certifie les clés publiques des CA intermédiaires (appelées aussi des CA subordonnées). Ces CA certifient ensuite les

clés publiques des entités finales (des souscripteurs) ou peuvent, dans une grande PKI, certifier d'autres CA. Dans cette architecture, les certificats sont produits seulement dans une direction, et une CA ne certifie jamais une autre CA "supérieure" à elle-même. Normalement, une seule CA supérieure certifie chaque CA.

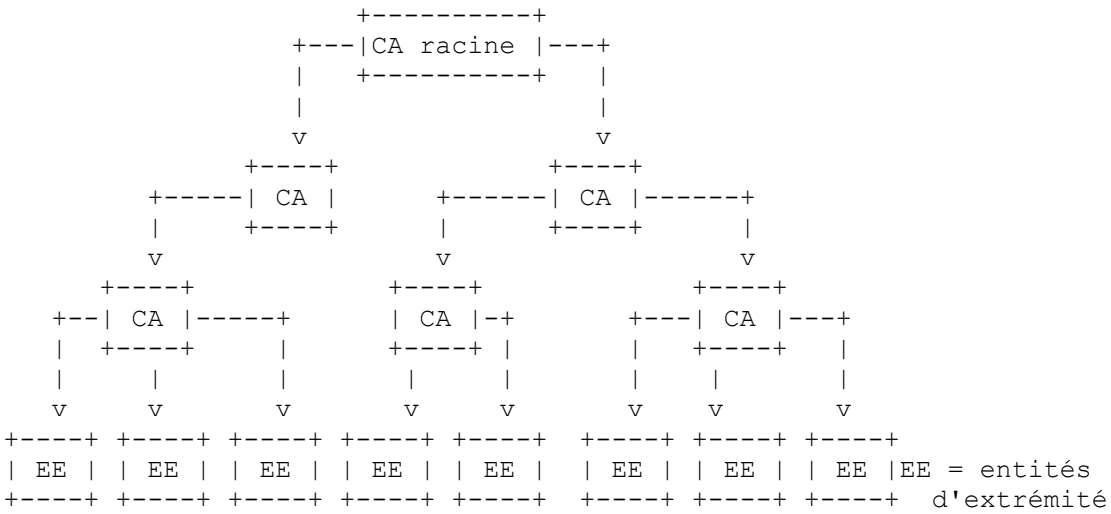


Figure 1 - Exemple de PKI hiérarchique

La construction de chemin de certification dans une PKI hiérarchique est un processus direct qui exige simplement que le consommateur d'assertions réussisse à restituer les certificats de producteur jusqu'à ce que soit localisé un certificat produit par l'ancre de confiance (la "CA racine" dans la Figure 1).

Une variation largement utilisée de la PKI hiérarchique à une seule racine est l'inclusion de plusieurs CA comme ancres de confiance (voir la Figure 2). Ici, les certificats d'entité d'extrémité sont validés en utilisant la même approche qu'avec toute PKI hiérarchique. La différence est qu'un certificat sera accepté si il peut être vérifié par toute ancre de confiance de l'ensemble. Les navigateurs populaires de la Toile utilisent cette approche, et sont munis de listes de confiance contenant des douzaines à jusqu'à plus d'une centaine de CA. Bien que cette approche simplifie la mise en œuvre de formes limitées de vérification de certificat, elle peut aussi introduire certaines vulnérabilités pour la sécurité. Par exemple, l'usager peut avoir une faible idée, ou pas du tout, des politiques ou des pratiques de fonctionnement des diverses ancres de confiance, et peut ne pas savoir quelle racine a été utilisée pour vérifier un certain certificat. De plus, la compromission de toute clé privée de CA de confiance ou l'insertion d'un certificat de CA félon dans la liste de confiance peut compromettre le système entier. À l'inverse, si la liste de confiance est gérée de façon appropriée et garde une taille raisonnable, cela peut être une solution efficace de construction et validation des chemins de certification.

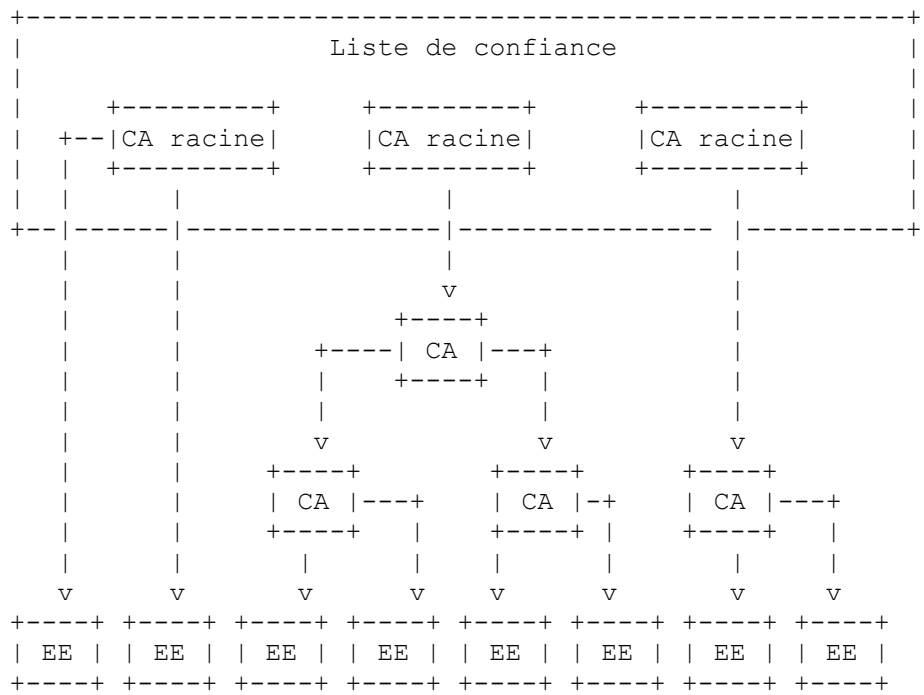


Figure 2 - PKI hiérarchique multi racines

1.5.2 Structures maillées

Dans une PKI de style maillé typique (décrite à la Figure 3) chaque entité d'extrémité fait confiance à la CA qui a produit son ou ses propres certificats. Donc, il n'y a pas de "CA racine" pour la PKI entière. Les CA dans cet environnement ont des relations d'homologues ; elles ne sont ni supérieures ni subordonnées à une autre. Dans un maillage, les CA dans la PKI s'inter certifient. C'est-à-dire que chaque CA produit un certificat pour, et reçoit un certificat, des CA homologues dans la PKI. La figure décrit une PKI maillée qui est pleinement inter certifiée (parfois appelée un maillage complet). Cependant, il est possible de construire et déployer une PKI maillée avec un mélange d'inter certifications unidirectionnelles et bidirectionnelles (appelée un maillage partiel). Les maillages partiels peuvent aussi inclure des CA qui ne sont pas inter certifiées avec les autres CA du maillage.

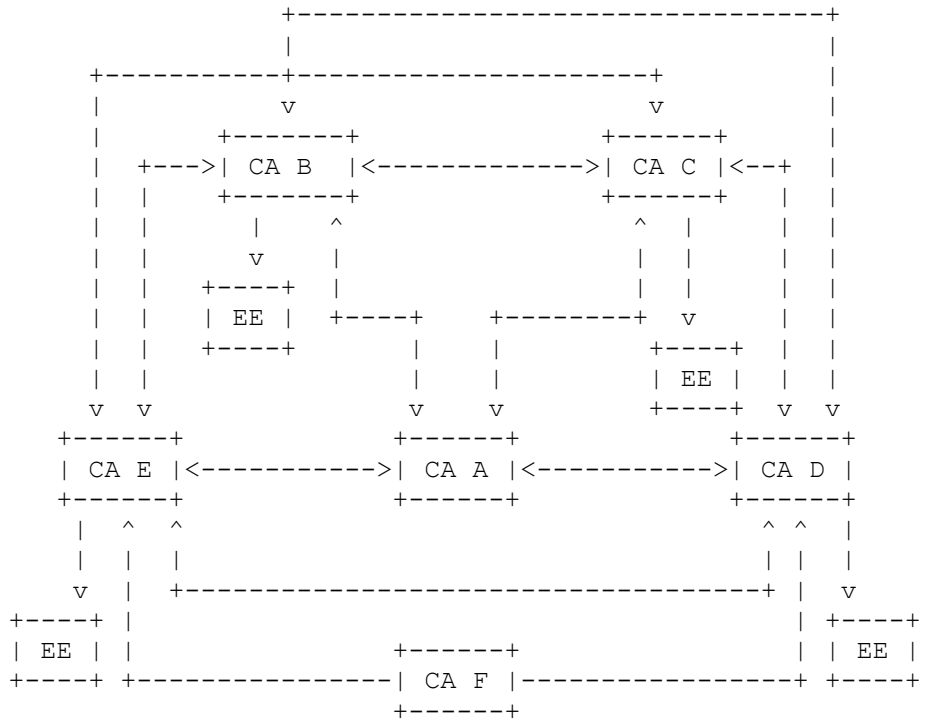


Figure 3 : PKI maillée

La construction de chemin de certification dans une PKI maillée est plus complexe que dans une PKI hiérarchique du fait de l'existence probable de multiples chemins entre l'ancre de confiance d'un consommateur d'assertions et le certificat à vérifier. Ces multiples chemins accroissent la création potentielle de "boucles", "d'impasses", ou de chemins invalides lors de la construction d'un chemin de certification entre une ancre de confiance et un certificat cible. De plus, dans les cas où aucun chemin valide n'existe, le nombre total de chemins traversés par le logiciel de construction de chemin pour conclure que "aucun chemin n'existe" peut devenir excessivement grand. Par exemple, si on ignore tout sauf la structure du graphe, la figure de PKI maillée ci-dessus a 22 certificats de CA non auto produits et un total de 5 092 429 chemins de certification entre la CA F et la EE produits par la CA D sans répéter aucun certificat.

1.5.3 Structures bilatérales à certification croisée

Les PKI peuvent être connectées via l'inter certification pour permettre aux consommateurs d'assertions de vérifier et accepter chacun des certificats produits par les autres PKI. Si les PKI sont hiérarchiques, l'inter certification va normalement être accomplie par chaque CA racine produisant un certificat pour les autres CA racines de la PKI. Il en résulte un environnement légèrement plus complexe, mais encore essentiellement hiérarchique. Si les PKI sont de style maillé, alors une CA au sein de chaque PKI est choisie, plus ou moins arbitrairement, pour établir l'inter certification, créant effectivement une plus grande PKI maillée. La Figure 4 décrit une situation hybride résultant d'une PKI hiérarchique inter certifiant avec une PKI maillée.

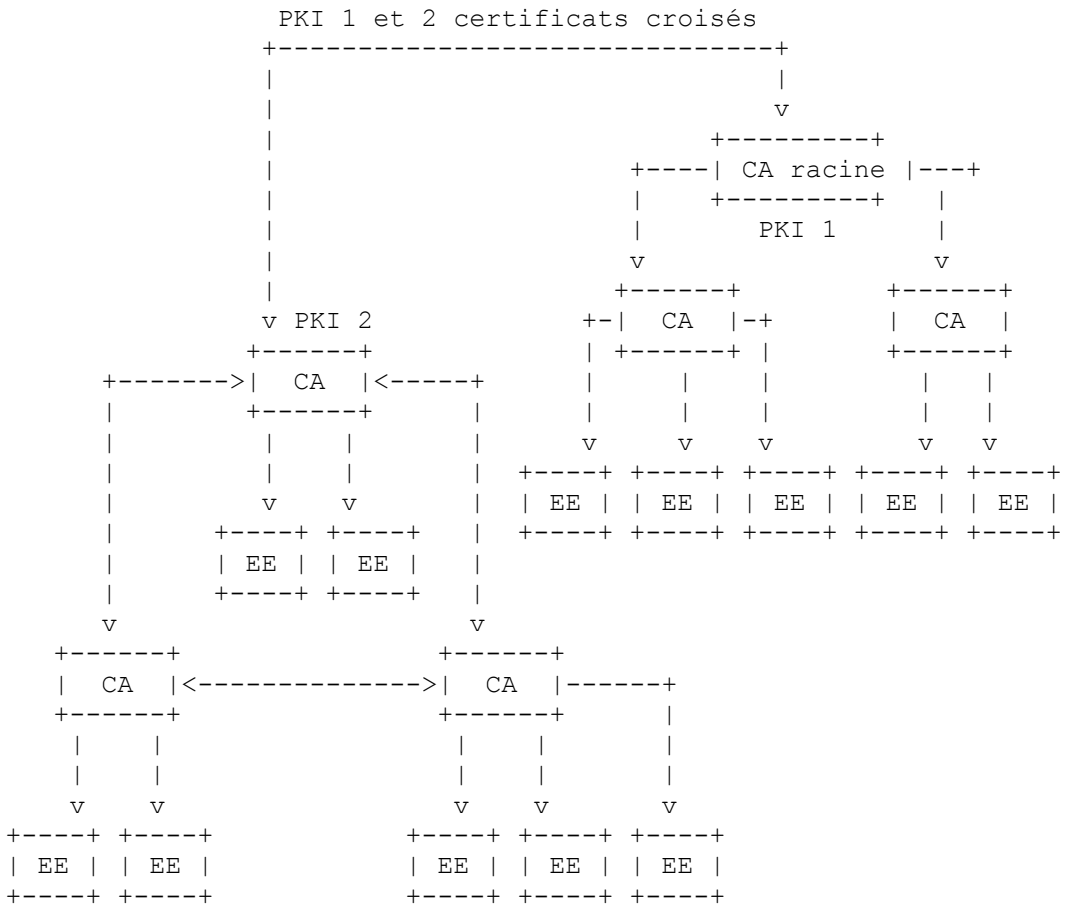


Figure 4 : PKI hybride

Dans les mises en œuvre courantes, cette situation crée le souci que les applications utilisées sous les PKI hiérarchiques n'aient pas de capacités de construction de chemins assez robustes pour traiter ces graphes de certificats plus complexes. Lorsque le nombre de PKI inter certifiées croît, le nombre de relations entre elles croît de façon exponentielle. Deux soucis principaux concernant l'inter certification sont la création de chemins de certification imprévus à travers la confiance transitive, et la dilution de l'assurance quand une PKI de forte assurance avec des politiques de fonctionnement restrictives est inter certifiée avec une PKI qui a des politiques moins restrictives. (Un traitement approprié des politiques de contraintes de nom et de certificat peut aider à atténuer le problème de la dilution d'assurance.)

1.5.4 Structures de pont

Une autre approche de l'interconnexion de PKI est l'utilisation d'une autorité de certification "pont" (BCA, *bridge certification authority*). Une BCA est une connexion pour établir des chemins de confiance parmi plusieurs PKI. La BCA s'inter certifie avec une CA dans chaque PKI participante. Chaque PKI s'inter certifie seulement avec une autre CA (c'est-à-dire, la BCA) et la BCA s'inter certifie seulement une fois avec chaque PKI participante. Par suite, le nombre de relations inter certifiées dans l'environnement ponté croît de façon linéaire avec le nombre de PKI tandis que le nombre de relations inter certifiées dans une architecture maillée croît de façon exponentielle. Cependant, lorsque on connecte les PKI de cette façon, le nombre et la variété des PKI impliquées résulte en un environnement non hiérarchique, comme celui décrit à la Figure 5. (Noter, comme expliqué au paragraphe 2.3, que des PKI non hiérarchiques peuvent être considérées comme hiérarchiques selon la perspective.)

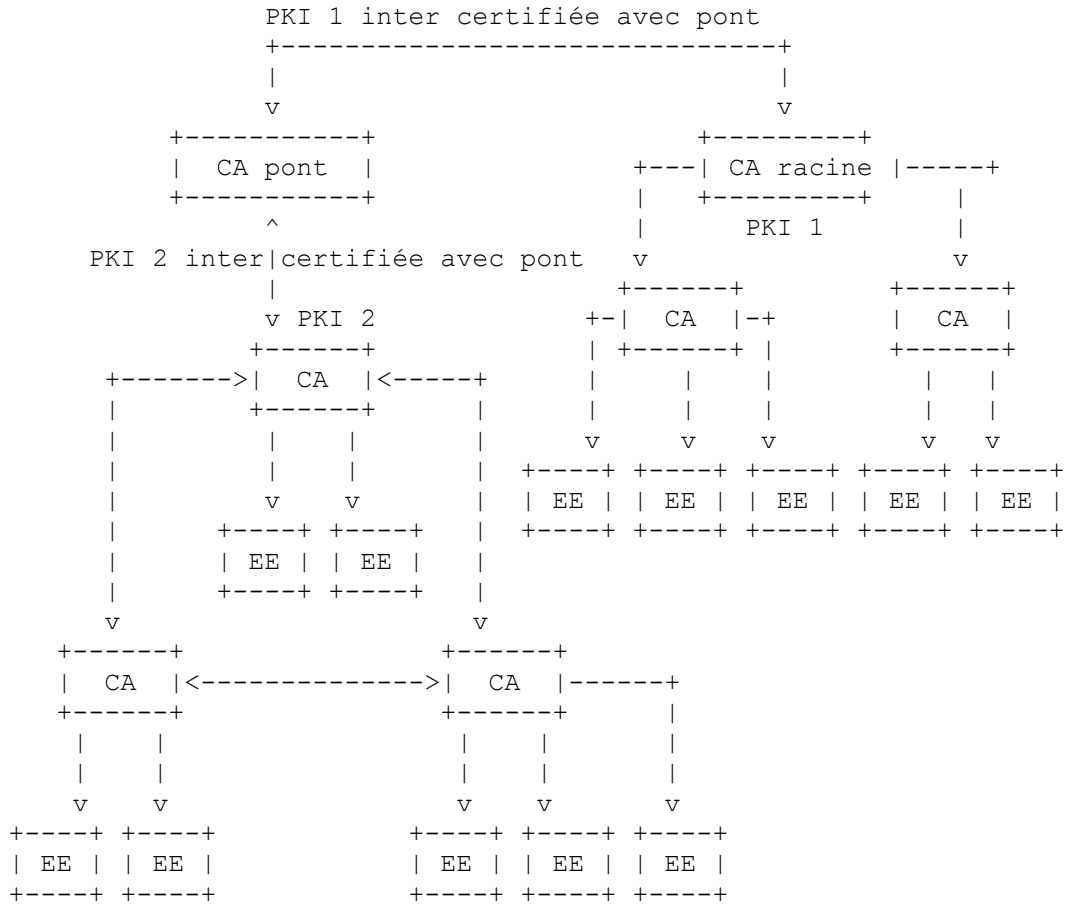


Figure 5 : Certification croisée avec CA pont

1.6 Traitement des structures de pont et de chemin de certification

Les développeurs d'applications de construction à capacité de certificat destinées à une large utilisation sur des secteurs variés sont invités à prendre en compte la prise en charge d'une structure de PKI pont parce que la mise en œuvre de fonctions de traitement de chemin de certification pour prendre en charge une structure de PKI pont requiert la prise en charge de toutes les structures de PKI (par exemple, hiérarchique, maillée, hybride) que le pont peut connecter. Une application qui peut réussir à construire des chemins de certification valides dans toutes les PKI ponts va donc avoir mis en œuvre toute la logique de traitement nécessaire pour la prise en charge des structures de PKI moins compliquées. Donc, si une application prend pleinement en charge la structure de PKI pont, elle peut être déployée dans tout environnement de PKI conforme aux normes et elle va effectuer correctement le traitement de chemin de certification requis.

2. Construction du chemin de certification

La construction du chemin de certification est le processus par lequel le système de traitement de certificat obtient le chemin de certification entre une ancre de confiance et le certificat cible. Différentes mises en œuvre peuvent construire le chemin de certification de façons différentes ; donc, il n'est pas dans les intentions du présent document de recommander une seule "meilleure" façon d'effectuer cette fonction. Des directives sont plutôt fournies sur les questions techniques qui entourent le processus de construction de chemin, et sur les capacités que les mises en œuvre de construction de chemin doivent avoir afin de réussir à construire des chemins de certification, sans égard aux structures de PKI.

2.1 Introduction à la construction du chemin de certification

Un chemin de certification est une liste ordonnée de certificats commençant par un certificat qui peut être validé par une des ancres de confiance du consommateur d'assertions, et se terminant avec le certificat à valider. (Le certificat à valider est appelé le "certificat cible" dans le présent document.) Bien que cela ne soit pas exigé, en pratique, ces ancres de confiance sont normalement mémorisées dans des certificats d'ancre de confiance. Les certificats intermédiaires qui constituent le chemin de certification peuvent être restitués par tous les moyens disponibles à l'application de validation. Ces sources peuvent inclure LDAP, HTTP, SQL, une antémémoire ou magasin de certificats local, ou au titre du protocole de sécurité

lui-même comme c'est la pratique courante avec les messages S/MIME signés et les sessions SSL/TLS.

La Figure 6 montre un exemple d'un chemin de certification. Dans cette figure, les flèches horizontales représentent les certificats, et la notation B(A) signifie un certificat produit à B, signé par A.

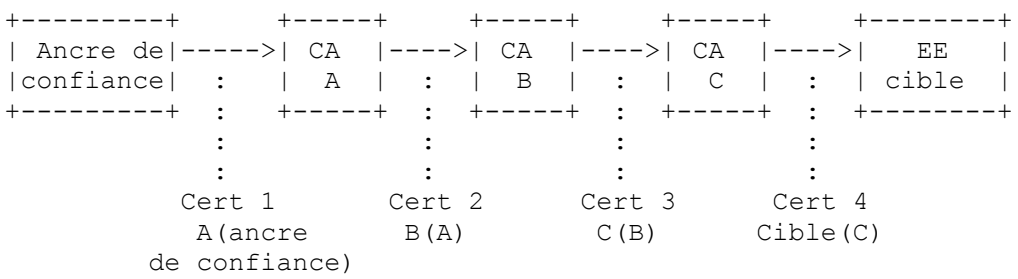


Figure 6 : Exemple de chemin de certification

À la différence de la validation du chemin de certification, la construction du chemin de certification n'est pas visée par les normes qui définissent la sémantique et la structure d'une PKI. C'est parce que la validation d'un chemin de certification n'est pas affectée par la méthode dans laquelle le chemin de certification a été construit. Cependant, la capacité de construire un chemin de certification valide est d'une importance capitale pour les applications qui reposent sur une PKI. Sans chemins de certification valides, les certificats ne peuvent pas être validés conformément à la [RFC3280] et ne peuvent donc pas être de confiance. Donc, la capacité de construire un chemin est aussi importante que celle de le valider correctement.

Il y a de nombreux problèmes qui peuvent compliquer le processus de construction de chemin. Par exemple, la construction d'un chemin à travers un environnement inter certifié pourrait exiger que le module de construction du chemin traverse plusieurs domaines de PKI s'étendant sur plusieurs répertoires, utilisant plusieurs algorithmes, et employant diverses longueurs de clé. Un client de construction de chemin peut aussi avoir besoin de gérer un certain nombre d'ancres de confiance, des entrées de répertoire partiellement remplies (par exemple, des entrées de issuedToThisCA manquantes dans l'attribut crossCertificatePair), d'analyser certaines extensions de certificat (par exemple, authorityInformationAccess) et des attributs de répertoire (par exemple, crossCertificatePair) et des traitements d'erreur comme la détection de boucle.

De plus, un développeur doit décider si il construit des chemins à partir d'une ancre de confiance (dans la direction inverse) vers le certificat cible ou à partir du certificat cible (la direction vers l'avant) vers une ancre de confiance. Certaines mises en œuvre peuvent même décider d'utiliser les deux. Le choix que fait un développeur devrait dépendre de l'environnement et de la PKI sous-jacente pour cet environnement. On trouvera plus d'informations sur ce choix au paragraphe 2.3.

2.2 Critères de la construction de chemin

À partir de ce point, le présent document va discuter des algorithmes et mécanismes spécifiques pour aider les développeurs de mises en œuvre de construction de chemin de certification. Pour donner une justification de ces mécanismes, il est important de noter ce que les auteurs considèrent comme critères pour une mise en œuvre de construction de chemin.

Critère 1 : La mise en œuvre est capable de trouver tous les chemins possibles, excepter les chemins contenant des paires répétées {nom de sujet, clé publique}. Cela signifie que tous les chemins de certification potentiellement valides entre l'ancre de confiance et le certificat cible qui peuvent être des chemins valides peuvent être construits par l'algorithme. Comme exposé au paragraphe 2.4.2, on recommande que les paires {nom de sujet, clé publique} ne soient pas répétées dans les chemins.

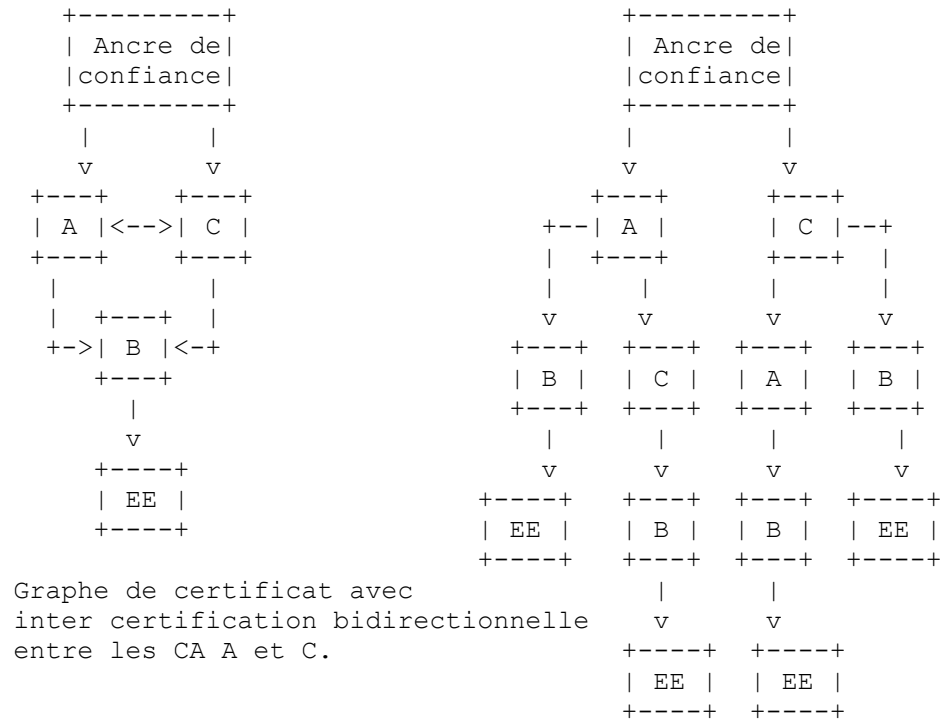
Critère 2 : La mise en œuvre est aussi efficace que possible. Une mise en œuvre de construction de chemin de certification est définie comme celle qui construit des chemins qui vont très probablement être validés conformément à la [RFC3280], avant la construction de chemins qui ne vont probablement pas être validés, en comprenant qu'il n'y a pas de moyen de tenir compte de toutes les configurations et infrastructures possibles. Ce critère est destiné à assurer que les mises en œuvre peuvent produire des informations d'erreur utiles. Si un chemin particulier est entièrement valide excepté pour un seul certificat expiré, c'est très probablement le "bon" chemin. Si d'autres chemins sont développés qui sont invalides pour plusieurs raisons obscures, cela donne peu d'informations utiles.

Les algorithmes et mécanismes discutés désormais sont choisis parce que les auteurs les considèrent comme de bonnes méthodes pour satisfaire les critères ci-dessus.

2.3 Algorithmes de construction de chemin

Il est intuitif pour les gens qui sont familiers du concept de CA pont ou des PKI de type maillé de voir la construction de chemin comme traversant un graphe complexe. Cependant, du plus simple point de vue, écrire un module de construction de chemin peut n'être rien de plus que la traversée d'une arborescence d'extension, même dans un environnement inter

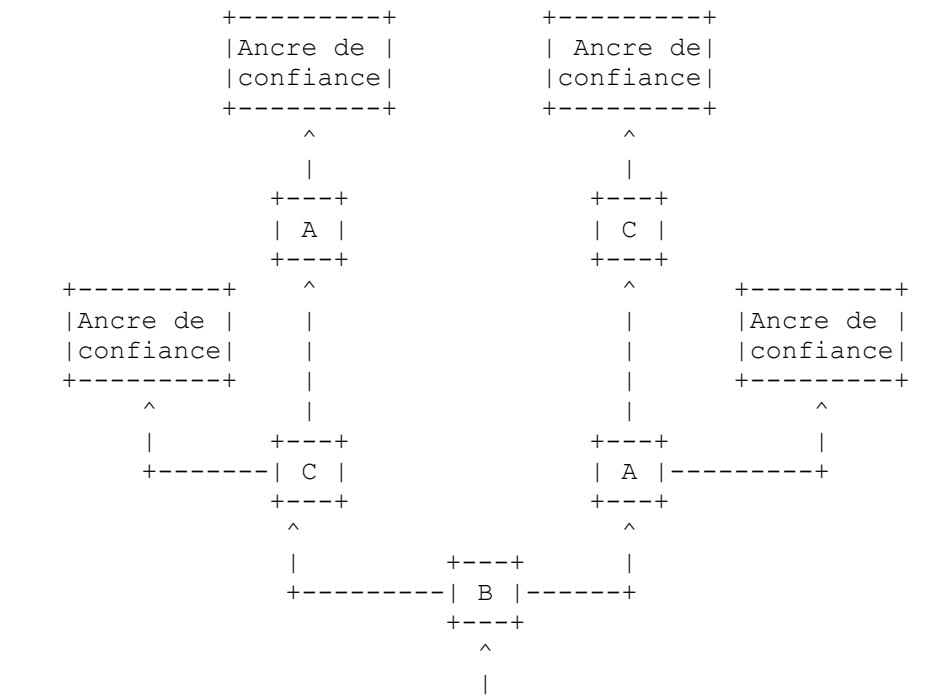
certifié très complexe. Les environnements complexes ainsi que les PKI hiérarchiques peuvent être représentés comme des arborescences parce que les certificats n'ont pas la permission de se répéter dans un chemin. Si les certificats pouvaient se répéter, des boucles pourraient être formées de sorte que le nombre de chemins et le nombre de certificats dans un chemin augmenteraient tous deux sans limites (par exemple, A produit à B, B produit à C, et C produit à A). La Figure 7 ci-dessous illustre ce concept du point de vue de l'ancre de confiance.



Le même graphe de certificat rendu comme arborescence – façon dont le logiciel de construction de chemin pourrait le voir

Figure 7 : Graphe de certificat simple – vu de l'arborescence d'ancre

Vues dans cette perspective, toutes les PKI ressemblent à des hiérarchies émanant de l'ancre de confiance. Une infrastructure peut être décrite de cette façon sans considération de sa complexité. Dans la Figure 8, le même graphe est décrit depuis l'entité d'extrémité (EE) (le certificat cible dans cet exemple). Il apparaîtrait de cette façon si la construction est dans la direction avant (en partant de l'EE ou de la cible). Dans cet exemple, sans connaître les particularités des certificats, il apparaît à première vue que la construction à partir de l'EE a une plus petite arborescence de décision que la construction à partir de l'ancre de confiance. Bien qu'il soit vrai qu'il y a moins de nœuds dans l'arborescence, ce n'est pas nécessairement plus efficace dans cet exemple.



Le même graphe de certificat rendu comme arborescence mais depuis l'entité d'extrémité plutôt que l'ancre de confiance.

Figure 8 : Graphe de certificat – vu du certificat cible

Supposons qu'un algorithme de construction de chemin n'a pas effectué d'optimisations. C'est-à-dire que l'algorithme est seulement capable de détecter que le certificat courant dans l'arborescence a été produit par l'ancre de confiance (TA, *Trust Anchor*), ou qu'il a produit le certificat cible (EE). D'après l'arborescence ci-dessus, la construction à partir du certificat cible va exiger de passer par deux certificats intermédiaires avant de rencontrer un certificat produit par l'ancre de confiance 100 % du temps (par exemple, EE s'enchaîne à B, qui s'enchaîne à C, qui est produit par l'ancre de confiance). Le module de construction de chemin n'enchaînerait pas C à A parce qu'il peut reconnaître que C a un certificat produit par l'ancre de confiance.

D'un autre côté, dans la première arborescence (Figure 7 : vue depuis l'ancre) il y a une probabilité à 50 % de construction d'un chemin plus long que nécessaire (par exemple, TA à A à C à B à EE plutôt que le plus court de TA à A à B à EE). Cependant, même dans notre exemple simpliste, le logiciel de construction de chemin, en A, pourrait être conçu pour reconnaître que le nom distinctif (DN, *distinguished name*) de sujet de B correspond au DN du producteur de l'EE. Étant donnée cette optimisation, le constructeur pourrait préférer B à C. (Le DN de sujet de B correspond à celui du producteur de l'EE tandis que le DN de sujet de C ne le fait pas.) Donc, pour cet exemple, en supposant que les éléments issuedByThisCA (inverse) et issuedToThisCA (vers l'avant) ont été bien remplis dans le répertoire et que notre module de construction de chemin a mis en œuvre la méthode d'optimisation de correspondance de DN susmentionnée, la construction de chemin à partir de l'ancre de confiance ou à partir du certificat cible pourrait être rendue à peu près équivalente. Une liste des méthodes d'optimisation possibles est fournie plus loin dans ce document.

Un exemple plus compliqué est créé lorsque le logiciel de construction de chemin rencontre une situation où il y a plusieurs certificats entre lesquels choisir lors de la construction d'un chemin. On se réfère à cela comme à de grandes arborescences de décision, ou à une situation très étalée. Cela peut se produire si une mise en œuvre a plusieurs ancres de confiance entre lesquelles choisir, et est construite dans la direction inverse (à partir de l'ancre de confiance). Ou cela peut se produire dans l'une ou l'autre direction si une CA pont se rencontre. Les grandes arborescences de décision sont l'ennemi de l'efficacité du logiciel de construction de chemin. Pour combattre ce problème, les mises en œuvre devraient prendre des décisions prudentes sur la direction de la construction de chemin, et devraient utiliser des optimisations comme celles discutées au paragraphe 3.1 lorsque confrontées à une grande arborescence de décision.

Sans égard à l'approche de la construction de chemin pour tout algorithme de construction de chemin, des cas peuvent être construits qui font que l'algorithme fonctionne mal. Les questions suivantes devraient aider un développeur à décider à partir de quelle direction construire les chemins de certification pour leur application :

1) Qu'est ce qui est exigé pour s'accommoder de l'environnement local de PKI et des environnements de PKI avec lesquels l'interopérabilité sera exigée ?
   a. Si on utilise un répertoire, celui-ci est il conforme à la [RFC2587] (précisément, les certificats croisés issuedToThisCA [vers l'avant] et/ou les attributs cACertificate sont-ils complètement remplis dans le répertoire) ? Si oui, on est capable de construire dans la direction vers l'avant.
   b. Si on utilise un répertoire, celui-ci contient il tous les certificats croisés issuedByThisCA (inverse) dans l'attribut crossCertificatePair, ou autrement, tous les certificats produites de chaque CA sont ils disponibles via d'autres moyens ? Si oui, il est possible de construire dans la direction inverse. Noter que la [RFC2587] n'exige pas que les certificats croisés issuedByThisCA (inverse) soient remplis ; si ils sont absents il ne sera pas possible de construire seulement dans la direction inverse.
   c. Les producteurs de certificats sont ils disponibles via des moyens autres qu'un répertoire (par exemple, l'extension authorityInformationAccess est elle présente et remplie dans tous les certificats) ? Si oui, on est capable de construire dans la direction vers l'avant.
2) Combien d'ancres de confiance le logiciel de construction et de validation de chemin va t-il utiliser ?
   a. Y a t-il (ou y aura t-il) plusieurs ancres de confiance dans la PKI locale ? Si oui, la construction de chemin vers l'avant peut offrir de meilleures performances.
   b. Le logiciel de construction et de validation de chemin a t-il besoin de faire confiance aux ancres de confiance provenant des PKI qui ne remplissent pas les certificats croisés inverses pour toutes les CA intermédiaires ? Si non, et si la PKI locale remplit les certificats croisés inverses, la construction de chemin inverse est une option.

2.4 Comment construire un chemin de certification

Comme on l'a exposé au paragraphe précédent, la construction de chemin est essentiellement une traversée d'arborescence. Il est très facile de voir que c'est vrai par un simple exemple, mais que ce passe t-il avec un exemple plus compliqué ? Avant d'examiner un scénario plus compliqué, on peut s'intéresser aux boucles et à ce qui constitue une boucle dans un chemin de certification. La Recommandation [X.509] spécifie que le même certificat ne doit pas se répéter dans un chemin.

Au sens strict, cela fonctionne bien car il n'est pas possible de créer une boucle sans fin sans répéter un ou plusieurs certificats dans le chemin. Cependant, cette exigence ne réussit pas à bien fonctionner dans les environnement de PKI pontées.

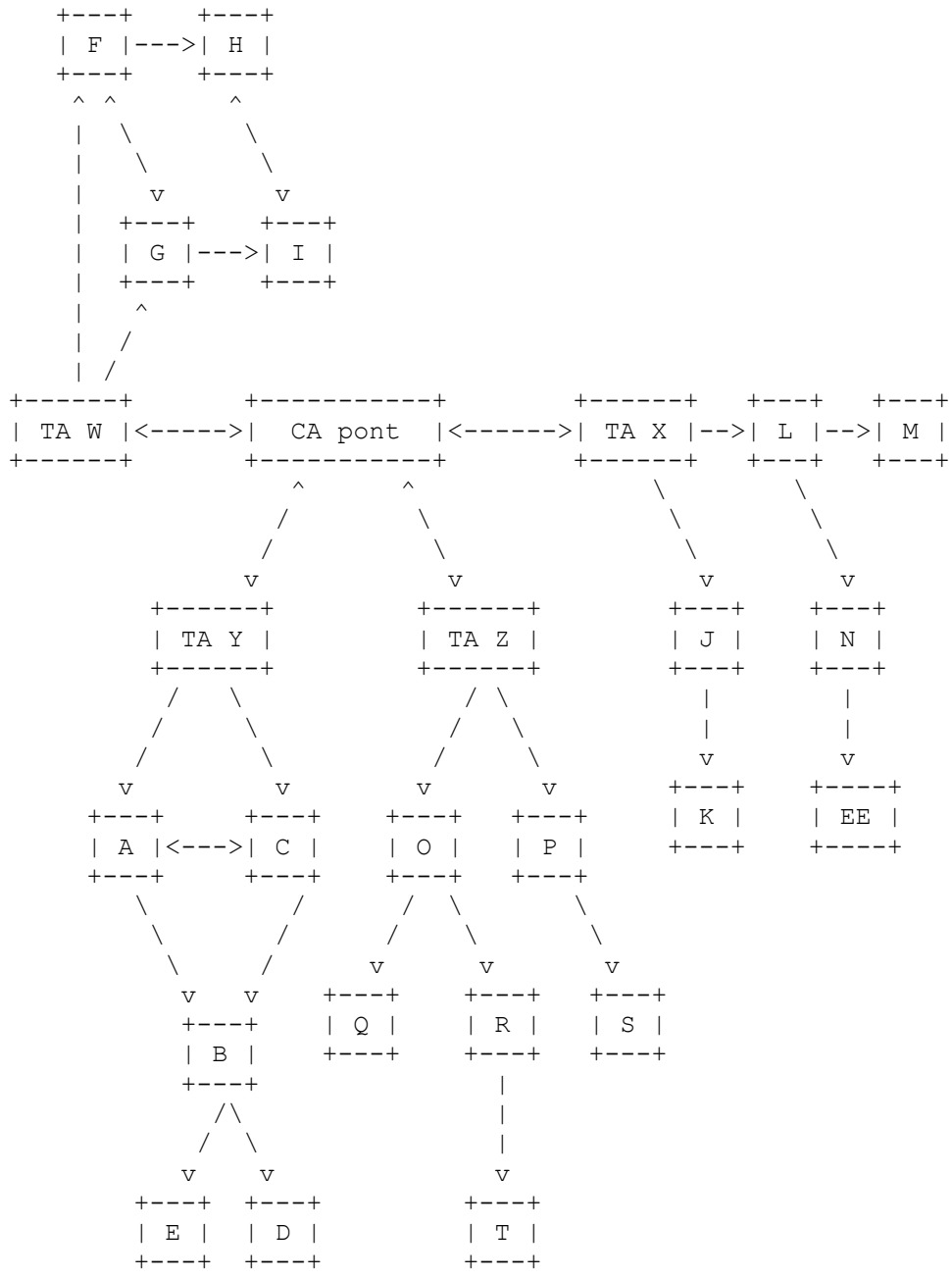


Figure 9 : Quatre PKI pontées

La Figure 9 montre quatre autorités de certification racines inter certifiées avec une CA pontée (BCA, *Bridge CA*). Alors que plusieurs ancres de confiance sont montrées dans la figure, nos exemples considèrent tous la TA Z comme l'ancre de confiance. Les autres ancres de confiance servent des consommateurs d'assertions différents. Par la construction des chemins de certification à travers la BCA, la confiance peut être étendue à travers les quatre infrastructures. Dans la Figure 9, la BCA a quatre certificats qui lui sont produits ; un issu de chacune des ancres de confiance du graphe. Si ils sont mémorisés dans le système de répertoire de la BCA, les quatre certificats produits à la BCA vont être mémorisés dans l'entrée issuedToThisCA (vers l'avant) des quatre différentes structures crossCertificatePair. La BCA a aussi produit quatre certificats, un à chacune des ancres de confiance. Si ils sont mémorisés dans le système de répertoire de la BCA, ces certificats seraient mémorisés dans l'entrée issuedByThisCA (inverse) des mêmes quatre structures crossCertificatePair. (Noter que les certificats croisés sont mémorisés comme des paires correspondantes dans l'attribut crossCertificatePair. Par exemple, une structure crossCertificatePair peut contenir à la fois A(B) et B(A), mais pas contenir A(C) et B(A).) Les quatre structures crossCertificatePair vont alors être mémorisées dans l'entrée de répertoire de BCA dans l'attribut crossCertificatePair.

2.4.1 Répétition de certificat

La Recommandation [X.509] exige que les certificats ne soient pas répétés lors de la construction des chemins. Par exemple, à partir de la figure ci-dessus, on ne construit pas le chemin TA Z->BCA->Y->A->C->A->C->B->D. Non seulement la répétition n'est pas nécessaire pour construire le chemin de Z à D, mais aussi il faut réutiliser un certificat (celui produit de C à A) ce qui rend le chemin non conforme à [X.509].
Qu'en est-il du chemin suivant de TA Z à EE ? TA Z->BCA->Y->BCA->W->BCA->X->L->N->EE
À la différence du premier exemple, ce chemin n'exige pas d'un développeur qu'il répète un certificat ; donc, il est conforme à [X.509]. Chacun des certificats de la BCA est produit d'une source différente et est donc un certificat différent. Supposons maintenant que la PKI en bas à droite sur la Figure 9 ait une double flèche entre Y et C, ainsi qu'entre Y et A. Le chemin suivant pourrait alors être construit : TA Z->BCA->Y->A->C->Y->BCA->W->BCA->X->L->N->EE
Un chemin comme celui-ci pourrait devenir arbitrairement complexe et traverser chaque CA inter certifiée CA dans chaque PKI dans un environnement inter certifié tout en restant conforme à [X.509]. En pratique, le chemin ci-dessus n'est pas ce qu'une application voudrait ou aurait besoin normalement de construire pour diverses raisons :

- D'abord, les chemins de certification comme ceux de l'exemple ci-dessus ne sont généralement pas voulus par les concepteurs de PKI et ne devraient pas être nécessaires pour valider un certificat. Si un chemin tortueux comme celui de l'exemple ci-dessus est nécessaire (parce qu'il n'y a pas de chemin simple correspondant) afin de valider un certain certificat, cela indique très probablement une faute dans la conception de la PKI.
- Ensuite, plus le chemin devient long, plus fort est le potentiel de dilution de la confiance sur le chemin de certification. C'est-à-dire qu'avec chaque liaison successive dans l'infrastructure (c'est-à-dire, certification par les CA et inter certification entre CA) on peut considérer qu'une certaine quantité d'assurance est perdue.
- En troisième lieu, plus un chemin est long et compliqué, plus diminue la probabilité qu'il soit validé à cause des contraintes de base, des politiques ou des contraintes de politique, des contraintes de noms, de la disponibilité de CRL, ou même de révocation.
- Enfin, et certainement pas le moins important du point de vue d'un développeur ou utilisateur, les performances. Permettre des chemins comme celui-ci augmente de façon dramatique le nombre de chemins possibles pour chaque certificat dans un environnement de maillage ou d'inter certification. Chaque chemin construit peut exiger un ou plusieurs de ce qui suit : validation des propriétés de certificat, validations intensives de signature de CPU, restitutions de CRL, charge du réseau accrue, et mise en mémoire tampon locale. Éliminer les chemins superflus peut nettement améliorer les performances, en particulier dans le cas où il n'existe pas de chemin.

Il y a un cas particulier qui implique des certificats avec le même nom distinctif mais des codages différents exigés par la [RFC3280]. Ce cas ne devrait pas être considéré comme celui de certificats répétés. Voir plus d'informations au paragraphe 5.4.

2.4.2 Introduction à l'optimisation de la construction de chemin

Comment ces chemins superflus peuvent ils être éliminés ? Plutôt que de seulement interdire que des certificats identiques se répètent, il est recommandé qu'un développeur interdise que la même paire de clé publique et de nom sujet soit répétée. Pour une souplesse maximale, le nom de sujet devrait collectivement inclure tous les noms de sujet de remplacement. En utilisant cette approche, tous les chemins prévus et nécessaires devraient être disponibles, et les chemins en excès et dilués devraient être éliminés. Par exemple, en utilisant cette approche, un seul chemin existe de TA Z à EE dans le diagramme ci-dessus : TA Z->BCA->X->L->N->EE.
Étant donnée la règle de simplification de ne pas répéter les paires de nom de sujet (incluant les noms de sujet de remplacement) et clé publiques, et de n'utiliser que les certificats qui se trouvent dans le cACertificate et l'élément vers l'avant (issuedToThisCA) des attributs crossCertificatePair, la Figure 10 décrit l'arborescence de décision de construction de chemin vers l'avant de EE à tous les nœuds accessibles dans le graphe. C'est le graphe idéal pour un constructeur de chemin qui tente de construire un chemin de TA Z à EE.

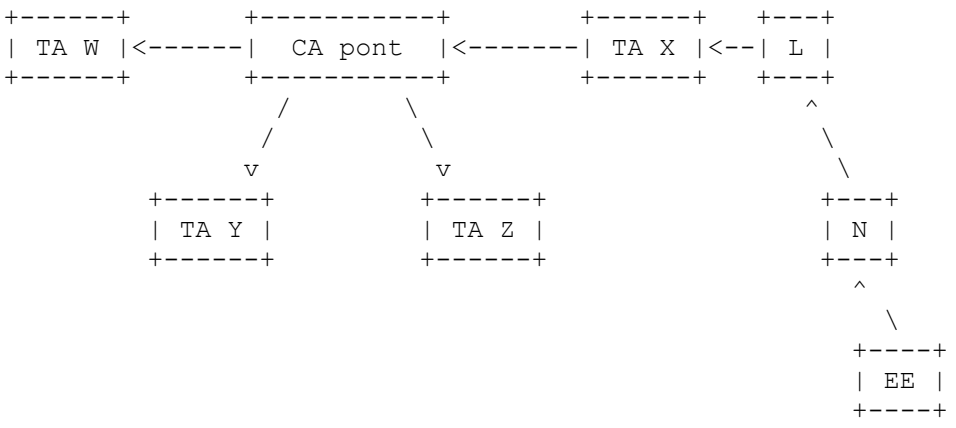


Figure 10 : Arborescence de décision vers l'avant (à partir de l'entité)

Il n'est pas possible de construire des chemins dans la direction vers l'avant dans les infrastructures derrière les CA W, Y, et Z, parce que W, Y, et Z n'ont pas reçu de certificats de leurs CA subordonnées. (Les CA subordonnées sont F et G, A et C, et O et P, respectivement.) Si on souhaite la simplicité et la rapidité, le graphe de la Figure 10 est une façon très attractive de structurer l'algorithme de construction de chemin. Trouver un chemin à partir de EE pour une des quatre ancres de confiance est raisonnablement simple. Autrement, un développeur pourrait choisir de construire dans la direction opposée, en utilisant les certificats croisés inverses à partir de n'importe laquelle des quatre ancres de confiance autour de la BCA. Le graphe de la Figure 11 décrit tous les chemins possibles comme une arborescence émanant de TA Z. (Noter qu'il n'est pas recommandé que les mises en œuvre tentent de déterminer tous les chemins possibles, cela exigerait la restitution et la mémorisation de toutes les données de PKI incluant les certificats et les CRL ! Cet exemple est donné pour montrer la complexité qui peut se rencontrer.)

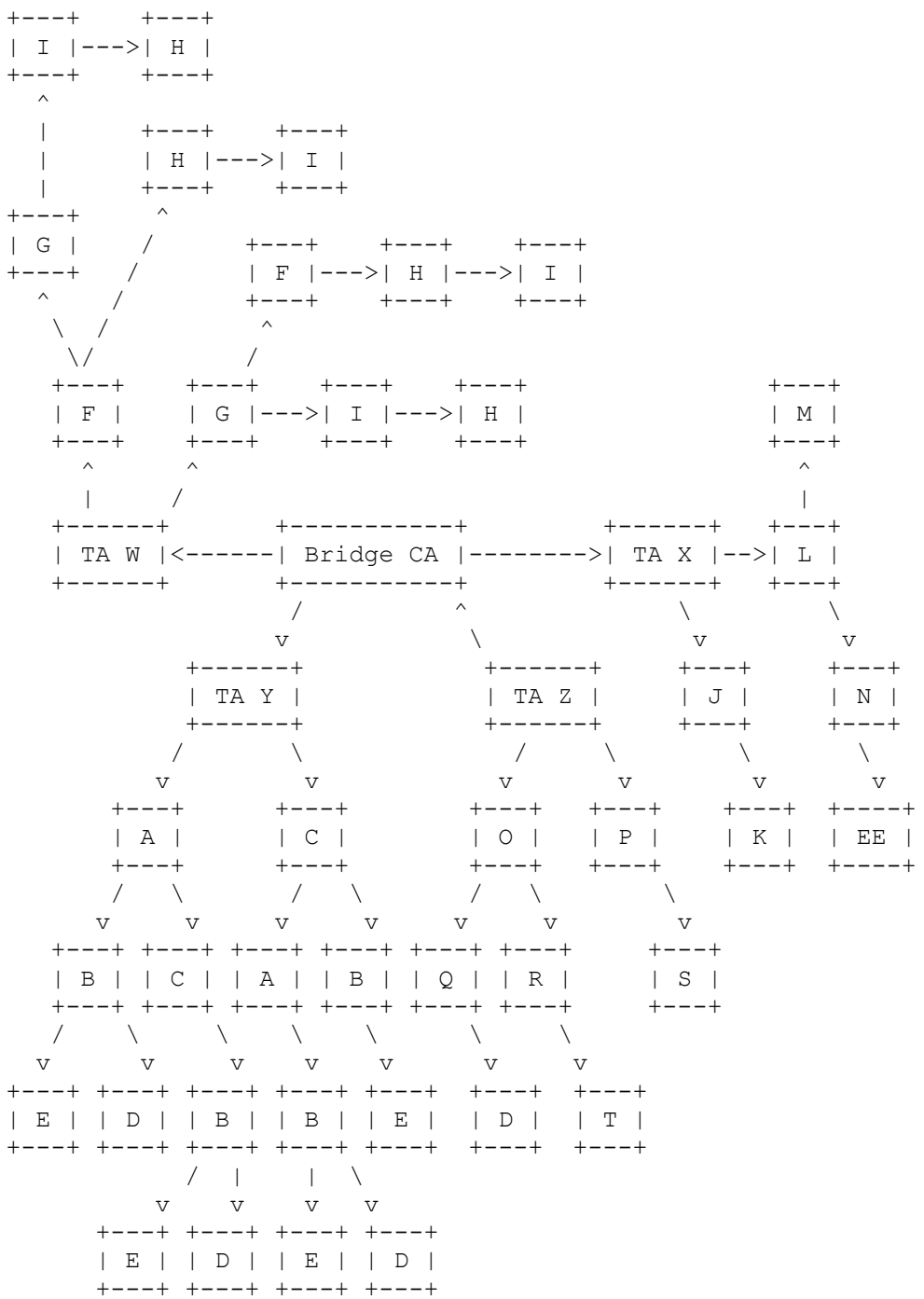


Figure 11 : Arborescence de décision inverse (à partir de l'ancre)

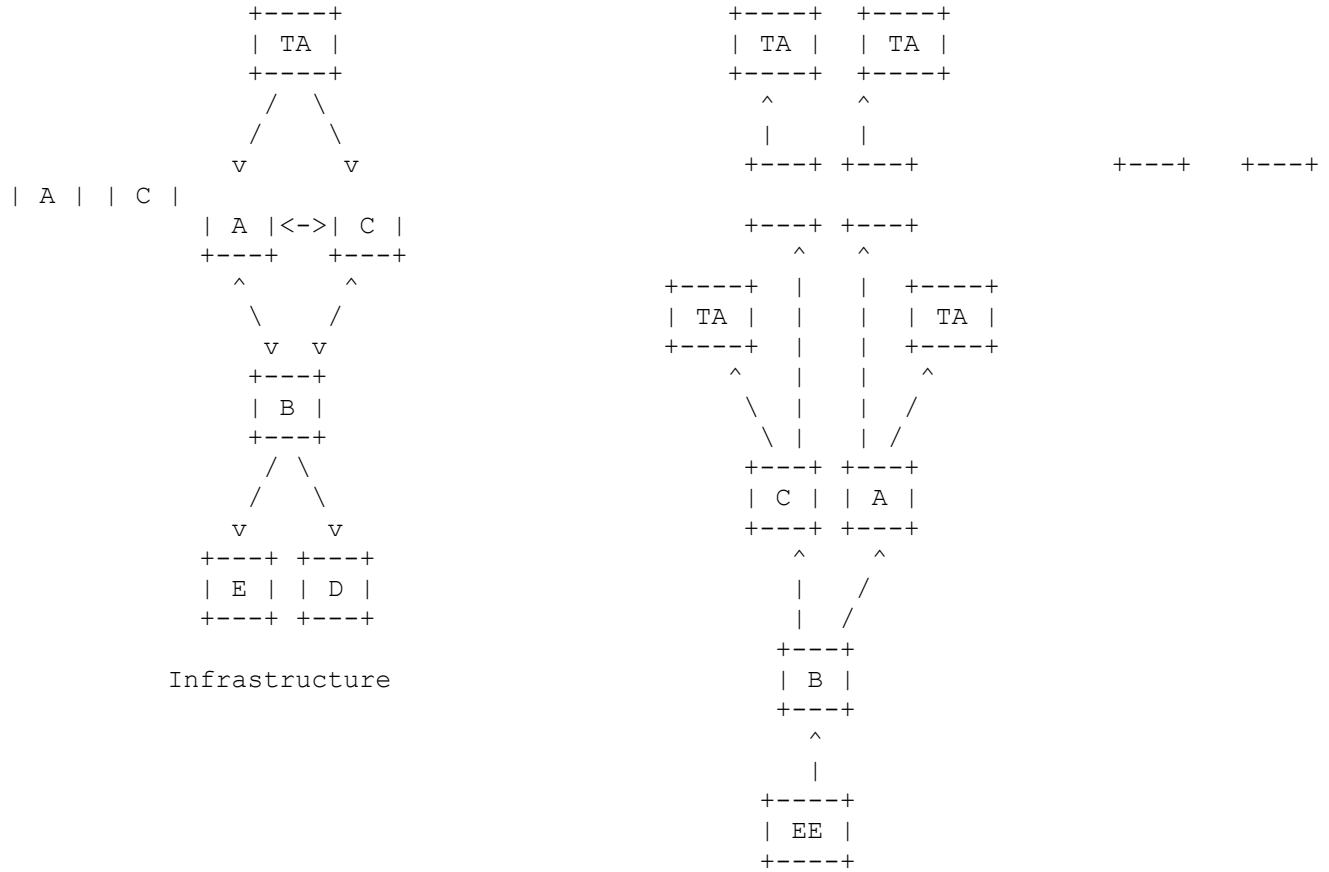
Étant donnée la relative complexité de cette arborescence de décision, il devient clair que faire les bons choix en navigant dans l'arborescence peut faire une grande différence sur la rapidité avec laquelle un chemin valide est retourné. Le logiciel de construction de chemin pourrait éventuellement traverser le graphe entier avant de choisir le plus court chemin : TA Z->BCA->X->L->N->EE. Avec une arborescence de décision comme celle ci-dessus, l'approche de base de traversée de

première profondeur introduit une inefficacité évidente dans le processus de construction de chemin. Pour compenser cela, un module de construction de chemin doit décider non seulement dans quelle direction traverser l'arborescence, mais aussi quelles branches de l'arborescence vont plus probablement donner un chemin valide.

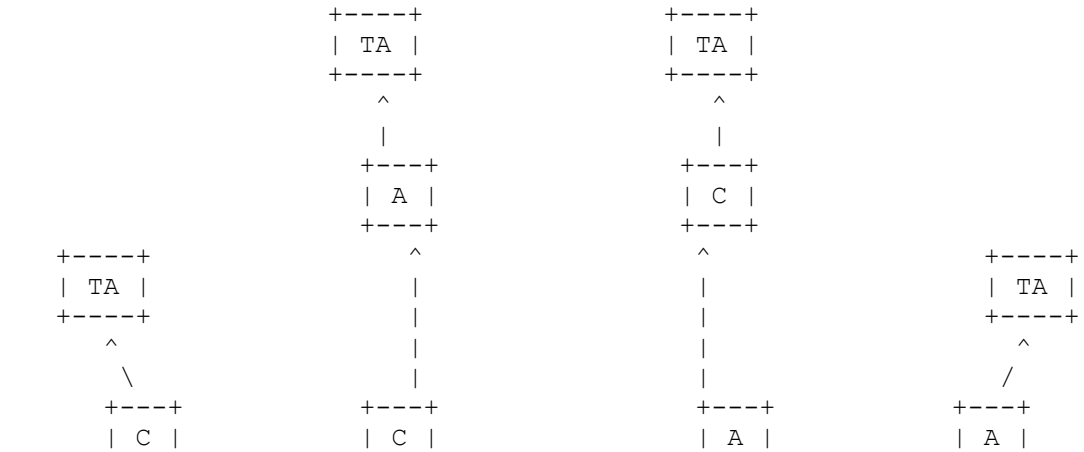
L'algorithme de construction de chemin devient alors idéalement un algorithme de traversée d'arborescence avec des pondérations ou des priorités allouées à chaque point d'embranchement pour guider la prise de décision. Si il est bien conçu, une telle approche va effectivement donner le "meilleur chemin en premier" plus souvent que non. (La terminologie "meilleur chemin en premier" est entre guillemets parce que la définition du "meilleur" chemin peut différer d'une PKI à l'autre. Cela sera déterminé en fin de compte par le développeur, non par ce document.) Trouver le "meilleur chemin en premier" est un effort pour rendre la mise en œuvre efficace, ce qui est un des critères déclarés au paragraphe 2.2.

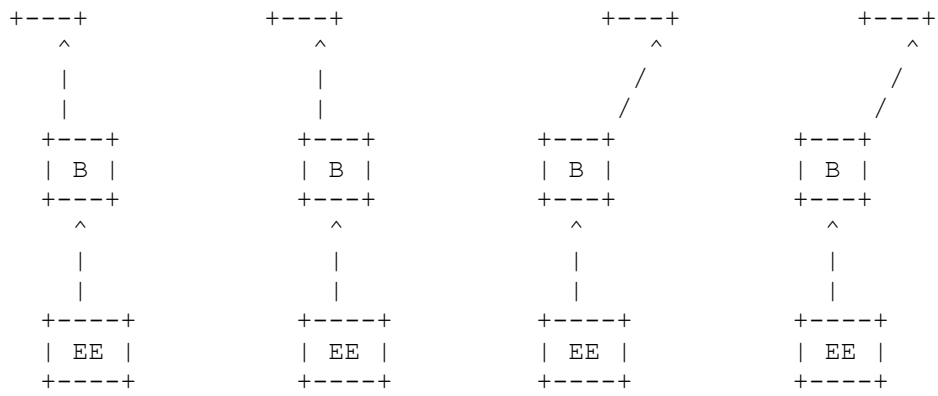
Donc, comment un développeur va t-il faire pour trouver le meilleur chemin en premier ? Avec l'idée simplificatrice de voir la construction de chemin comme une traversée d'arborescence, la construction de chemin pourrait être structurée comme une recherche de première profondeur. Un simple exemple de construction de chemin par traversée d'arborescence de première profondeur est décrit à la Figure 12, sans préférence pour l'ordre de tri.

Note : Les flèches dans la portion inférieure de la figure n'indiquent pas la direction de production des certificats; mais la direction de la traversée de l'arborescence à partir du certificat cible (EE).



La même infrastructure représentée comme arborescence





Tous les chemins possibles de EE à TA en utilisant une traversée d'arborescence de décision de première profondeur

Figure 12 : Construction de chemin avec une traversée d'arborescence de première profondeur

La Figure 12 illustre que quatre chemins possibles existent pour cet exemple. Supposons que le dernier chemin (TA->A->B->EE) soit le seul chemin qui va se valider. Cela pourrait être pour toute combinaison de raisons comme des contraintes de nom, le traitement de politique, les périodes de validité, ou des contraintes de longueur de chemin. Le but d'un composant de construction de chemin efficace est de choisir le quatrième chemin en premier en vérifiant les propriétés des certificats lorsque l'arborescence est traversée. Par exemple, lorsque le logiciel de construction de chemin est à l'entité B dans le graphe, il devrait examiner les deux choix A et C pour déterminer quel certificat est le meilleur choix le plus probable. Un module efficace va conclure que A est le chemin correct le plus probable. Ensuite, en A, le module compare de terminer le chemin à TA, ou de passer à C. Là encore un module efficace va faire le meilleur choix (TA) et par là trouver le "meilleur chemin en premier".

Qu'en est il si le choix entre les certificats de CA n'est pas binaire comme il l'était dans l'exemple précédent ? Qu'en est il si le logiciel de construction de chemin rencontre un point d'embranchement avec un nombre de certificats de CA arbitraire, créant par suite le même nombre arbitraire de branches d'arborescence ? (Cela serait typique d'un style de CA PKI maillée, ou d'une entrée de répertoire de CA pontée, car chacune va avoir plusieurs certificats produits à elle–même par les autres CA.) Cette situation ne change en fait pas du tout l'algorithme, si il est structuré correctement. Dans notre exemple, plutôt que de traiter chaque décision comme binaire (c'est-à-dire, de choisir A ou C) le logiciel de construction de chemin pourrait trier toutes les possibilités disponibles à un certain point d'embranchement, et retenir ensuite le meilleur choix de la liste. Dans le cas où le chemin ne pourrait pas être construit sur le premier choix, le second choix serait alors essayé au prochain passage à ce point de l'arborescence. On continue de suivre ce schéma jusqu'à trouver un chemin, ou que tous les nœuds CA de l'arborescence aient été traversés. Noter que les certificats en tout point de l'arborescence ne devraient être triés qu'au moment où une décision est prise. Précisément, dans l'exemple, le tri de A et C est fait lorsque l'algorithme atteint B. Il n'y a pas de représentation présente en mémoire de l'arborescence des entrées. Tout comme tout autre algorithme de recherche en première profondeur récurrent, la seule information dont l'algorithme ait besoin de garder trace est ce qu'il y a derrière les nœuds (entités) dans l'arborescence sur le chemin en cours, et pour chacun de ces nœuds, quels arcs (certificats) ont déjà été essayés.

2.5 Construction des chemins de certification pour certificats de signataire de révocation

Une considération particulière est donnée à la construction d'un chemin de certification pour le certificat de signataire de révocation parce qu'il peut être ou non le même que le certificat d'autorité de certification. Par exemple, après qu'une CA a effectué un retournement de clé, le nouveau certificat de CA va être le certificat de signataire de CRL, tandis que le vieux certificat de CA est le certificat d'autorité de certification pour les certificats produits précédemment. Dans le cas de CRL indirectes, le certificat de signataire de CRL va contenir un nom et une clé différents de ceux du certificat d'autorité de certification. Dans le cas de OCSP, le certificat de signataire de révocation peut représenter un répondeur OCSP qui n'est pas la même entité que l'autorité de certification.

Lorsque le certificat de signataire de révocation et le certificat d'autorité de certification sont identiques, aucune considération supplémentaire n'est requise du point de vue de la construction de chemin de certification. C'est-à-dire, le chemin de certification construit (et validé) pour le certificat d'autorité de certification peut aussi être utilisé comme chemin de certification pour le certificat de signataire de révocation. Dans ce cas, la signature sur les données de révocation (par exemple, CRL ou réponse OCSP) est vérifiée en utilisant le même certificat, et aucune autre construction de chemin de certification n'est requise. Un algorithme efficace de validation de chemin de certification devrait d'abord essayer toutes les CRL possibles produites par l'autorité de certification pour déterminer si une des CRL (a) couvre le certificat en question, (b) est en cours, et (c) est signée en utilisant la même clé qu'utilisée pour signer le certificat.

Lorsque le certificat de signataire de révocation n'est pas identique au certificat d'autorité de certification, un chemin de

certification doit être construit (et validé) pour le certificat de signataire de révocation. En général, le logiciel de construction de chemin de certification peut construire le chemin comme il le ferait pour tout autre certificat. Cependant, le présent document souligne aussi un peu plus loin des méthodes qui améliorent notablement l'efficacité de la construction de chemin dans le cas du certificat de signataire de révocation.

2.6 Composants de logiciel de construction de chemin suggérés

Il n'y a pas une seule façon de définir une interface à un module de construction de chemin. Il n'est pas dans les intentions de ce document de prescrire une méthode ou sémantique particulières ; il appartient plutôt à la mise en œuvre de décider. Cela peut être fait de nombreuses façons. Par exemple, un module de construction de chemin pourrait construire chaque chemin concevable et retourner la liste entière à l'appelant. Ou le module pourrait construire jusqu'à ce qu'il en trouve juste un qui valide et terminer alors la procédure. Ou il pourrait construire des chemins de façon itérative, selon une validation en dehors du constructeur et des appels successifs au constructeur pour avoir plus de chemins jusqu'à ce que soit trouvé un chemin valide ou que tous les chemins possibles aient été trouvés. Toutes ces approches sont possibles, et chacune d'elles peut offrir des avantages différents à un environnement ou application particuliers.

Sans considération de la sémantique, un module de construction de chemin a besoin de contenir les composants suivants :
1) la logique pour la construction et la traversée du graphe de certificats ;.
2) la logique pour restituer les certificats (et CRL et/ou autres informations d'état de révocation si le chemin doit être validé) nécessaires de la ou des sources disponibles.

En supposant qu'on désire un module plus efficace et agile de construction de chemin, ce qui suit est un bon point de départ pour toute la suite du présent document. Pour qu'un module de construction de chemin tire pleinement parti de toutes les optimisations suggérées dans le présent document, il va avoir besoin de tous les composants cités ci-dessous :
1) Un certificat local et une antémémoire de CRL.
   a. Cela peut être utilisé par tous les composants du certificat ; cela n'a pas besoin d'être spécifique du logiciel de construction de chemin. Une antémémoire locale pourrait résider dans la mémoire, mémorisée dans un système d'exploitation ou un magasin de certificats d'application, mémorisée dans une base de données, ou même mémorisée dans un fichier individuel sur le disque dur. Bien que la mise en œuvre de cette antémémoire sorte du domaine d'application de ce document, quelques considérations de conception sont données ci-dessous.
2) La logique pour la construction et la traversée du graphe/arborescence de certificat.
   a. Cela effectue la fonction de tri pour donner des priorités aux certificats (optimisant par là la construction de chemin) pendant la traversée de l'arborescence.
   b. Il n'est pas nécessaire de construire un graphe complet avant de commencer la construction de chemin. Comme celle-ci peut être mise en œuvre comme une traversée d'arborescence de première profondeur, le constructeur de chemin a seulement besoin de mémoriser la localisation actuelle dans l'arborescence avec les points traversés vers la localisation actuelle. Toute branche terminée peut être éliminée de la mémoire et les branches futures sont découvertes lorsque l'arborescence est traversée.
3) La logique pour restituer les certificats nécessaires à partir de la ou des sources de certificat :
   a. Antémémoire locale.
      - être capable de restituer tous les certificats pour une entité par nom de sujet, ainsi que les certificats individuels par couple producteur, numéro de série ;
      - tracer dans quel attribut de répertoire (incluant issuedToThisCA <forward> et issuedByThisCA <reverse> pour les attributs crossCertificatePair partagés) chaque certificat a été trouvé peut être utile. Cela permet des fonctions comme "ne restituer que les certificats croisés vers l'avant", etc. ;
      - un horodatage de "fraîcheur" (heure d'expiration de l'antémémoire) peut être utilisé pour déterminer quand le répertoire devrait être examiné à nouveau.
   b. Répertoire LDAPv3 pour les certificats et les CRL.
      - Envisager de prendre en charge plusieurs répertoires pour les interrogations générales.
      - Considérer de prendre en charge des connexions LDAP dynamiques pour restituer les CRL en utilisant un URI LDAP [RFC3986] dans l'extension de certificat de point de distribution de CRL.
      - Prise en charge des références LDAP. C'est normalement seulement une question d'activation du fanion approprié dans l'API LDAP.
   c. Prise en charge de HTTP pour les points de distribution de CRL et de l'accès aux informations d'autorité (AIA, *authority information access*).
      - Considérer de prendre en charge HTTPS, mais être conscient que cela peut créer une récurrence sans limite lorsque la mise en œuvre essaye de construire un chemin de certification pour le certificat de serveur si cela à son tour exige une recherche HTTPS supplémentaire.
4) Une antémémoire de chemin de certification qui mémorise les relations validées antérieurement entre certificats. Cette antémémoire devrait inclure :
   a. une date d'expiration configurable pour chaque entrée. Cette date peut être configurée sur la base de facteurs tels que l'expiration des informations utilisées pour déterminer la validité d'une entrée, la bande passante, le niveau d'assurance, l'espace de mémorisation, etc.

b. la prise en charge de la mémorisation des certificats de producteur antérieurement vérifiés pour les relations de certificat de sujet :
   - comme le couple {DN du producteur, numéro de série} identifie de façon univoque le certificat, une paire de ces couples (un pour le producteur et le sujet) est une méthode efficace de mémorisation de ces relations.

c. la prise en charge des chemins et certificats "connus comme mauvais". Une fois qu'un certificat est déterminé comme invalide, les mises en œuvre peuvent décider de ne pas réessayer le développement et la validation du chemin.

2.7 Entrées au module de construction de chemin

La Recommandation [X.509] vise spécifiquement la liste des entrées requises pour la validation de chemin mais ne fait aucune suggestion concernant les entrées utiles pour la construction de chemin. Cependant, étant donné que le but de la construction de chemin est de trouver des chemins de certification qui vont être validés, il s'ensuit que les mêmes entrées utilisées pour la validation pourront être utilisées pour optimiser la construction de chemin.

2.7.1 Entrées exigées

Laissant de côté les informations de configuration telles que les localisations de répertoire ou d'antémémoire, les entrées suivantes sont requises pour le processus de construction de chemin de certification :

1) Certificat cible : celui qui doit être validé. C'est un point d'extrémité pour le chemin. (Il est aussi possible de fournir des informations utilisées pour restituer un certificat pour une cible, plutôt que le certificat lui-même.)
2) Liste de confiance : C'est l'autre point d'extrémité du chemin, qui peut consister en :
   a. Certificat de CA de confiance
   b. Clés de confiance et DN ; un certificat n'est pas nécessairement requis.

2.7.2 Entrées facultatives

En plus des entrées citées au paragraphe précédent les entrées facultatives suivantes peuvent aussi être utiles pour optimiser la construction de chemin. Cependant, si le logiciel de construction de chemin tire parti de toutes les méthodes d'optimisation décrites plus loin dans ce document, toutes les entrées facultatives suivantes seront requises.

1) Time (T) : la durée pendant laquelle le certificat est valide (par exemple, si on valide une signature datant d'il y a un an, T est nécessaire pour construire un chemin valide)
   a. si il n'est pas inclus comme entrée, le logiciel de construction de chemin devrait toujours prendre T égal à l'heure système en cours.
2) Indicateur Initial-inhibit-policy-mapping
3) Indicateur Initial-require-explicit-policy
4) Indicateur Initial-any-policy-inhibit
5) Ensemble initial de politiques acceptables pour l'utilisateur
6) Manipulateurs d'erreur (rappels ou classes virtuelles)
7) Manipulateurs pour extensions de certificat personnalisé
8) Indicateur Is-revocation-provider
   a. IMPORTANT : Lors de la construction d'un chemin de certification pour un certificat de répondeur OCSP spécifié au titre de la configuration locale, ce fanion ne devrait pas être établi. Il est établi lors de la construction d'un chemin de certification pour un certificat de signataire de CRL ou pour un certificat de signataire de répondeur OCSP découvert en utilisant les informations certifiées dans une extension de certificat authorityInformationAccess.
9) Le chemin de certification complet pour l'autorité de certification (si Is-revocation-provider est établi)
10) Collection de certificats qui peuvent être utiles dans la construction du chemin
11) Collection de listes de révocation de certificats et/ou autres données de révocation.

Les deux derniers éléments sont affaire de convenance. Autrement, les certificats et informations de révocation pourraient être placés dans une mémoire tampon locale accessible au module de construction de chemin avant de tenter de construire un chemin.

3. Optimisation de la construction de chemin

Cette section recommande des méthodes pour optimiser les processus de construction de chemin.

3.1 Construction d'un chemin optimisé

La construction de chemin peut être optimisée en triant les certificats à chaque point de décision (à chaque nœud dans

l'arborescence) et en choisissant ensuite le certificat le plus prometteur non encore sélectionné, comme décrit au paragraphe 2.4.2. Ce processus continue jusqu'à ce que le chemin se termine. C'est en gros équivalent au concept de création d'une arborescence à bordure pondérée, où les bordures sont représentées par les certificats et les nœuds représentent les DN sujets. Cependant, à la différence du concept de bordure pondérée, un constructeur de chemin de certification n'a pas besoin d'avoir tout le graphe disponible pour fonctionner efficacement. De plus, le constructeur de chemin peut être sans état par rapport aux nœuds du graphe non présents dans le chemin en cours, de sorte que l'ensemble de données de travail peut être relativement petit.

Le concept de "sans état" par rapport aux nœuds qui ne sont pas dans le chemin courant est instrumental pour l'utilisation du tri des optimisations mentionnées dans ce document. Initialement, il peut sembler que faire le tri d'un certain groupe de certificats pour une CA une fois et ensuite de préserver l'ordre de tri pour l'utilisation ultérieure serait une façon efficace d'écrire la construction de chemin. Cependant, conserver cet état peut rapidement éliminer l'efficacité donnée par le tri. Considérons le diagramme suivant :

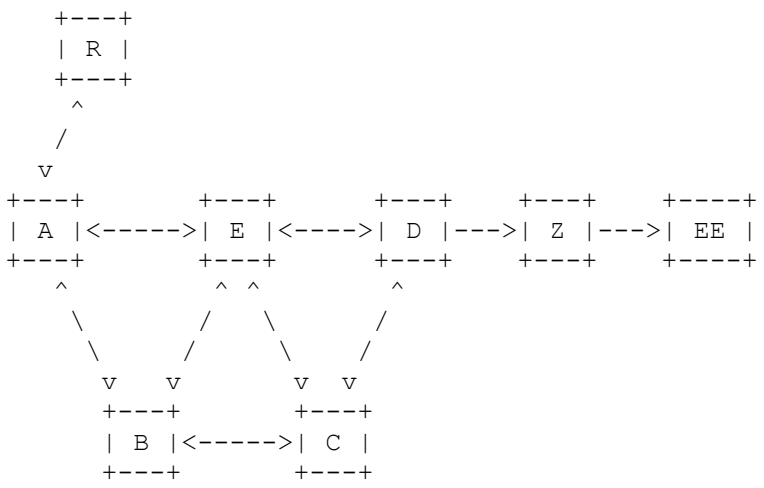


Figure 13 : Exemple d'optimisation de contruction de chemin

Dans cet exemple, la construction du chemin est dans la direction vers l'avant (depuis la cible) pour un chemin entre R et EE. Le constructeur du chemin a aussi opté en faveur de la répétition du nom de sujet et de la clé. (Cela permet plusieurs traversées à travers toutes les CA inter certifiées, créant assez de complexité dans ce petit exemple pour illustrer une conservation d'état appropriée. Noter qu'un exemple d'une complexité similaire pourrait être montré en utilisant plusieurs clés pour chaque entité et en interdisant la répétition.)

La première étape est simple ; le constructeur bâtit le chemin Z(D)->EE(Z). Ensuite il ajoute D et doit prendre une décision entre ceux certificats. (Choisir entre D(C) ou D(E)). Le constructeur trie ensuite les deux choix dans l'ordre de priorité. Le tri est partiellement fondé sur ce qui est actuellement dans le chemin.

Supposons que l'ordre que choisit le constructeur soit [D(E), D(C)].
Le chemin courant est maintenant D(E)->Z(D)->EE(Z). Actuellement, le constructeur a trois nœuds dans le graphe (EE, Z, et D) et devrait conserver l'état, incluant l'ordre de tri des certificats à D, quand il ajoute le prochain nœud, E. Quand E est ajouté, le constructeur a maintenant quatre certificats à trier : E(A), E(B), E(C), et E(D). Dans ce cas, l'exemple de constructeur opte pour l'ordre [E(C), E(B), E(A), E(D)]. Le chemin courant est maintenant E(C)->D(E)-> Z(D)->EE(Z) et le chemin a quatre nœuds : EE, Z, D, et E.

À l'ajout du cinquième nœud, C, le constructeur trie les certificats (C(B), C(D), et C(E)) à C, et choisit C(E). Le chemin est maintenant C(E)->E(C)->D(E)->Z(D)->EE(Z) et le chemin a cinq nœuds : EE, Z, D, E, et C.

Le constructeur se retrouve maintenant au nœud E avec quatre certificats. Si le constructeur devait utiliser l'ordre de tri précédent venant de la première rencontre avec E, il aurait [E(C), E(B), E(A), E(D)]. Dans le contexte du chemin en cours, cet ordre peut être inapproprié. Pour commencer, le certificat E(C) est déjà dans le chemin de sorte qu'il ne mérite certainement pas la première place.

La meilleurs façon de traiter cette situation est que le constructeur de chemin traite cette instance de E comme un nouveau (sixième) nœud dans l'arborescence. En d'autres termes, il n'y a pas d'informations d'état pour cette nouvelle instance de E – elle est traitée juste comme tout autre nouveau nœud. Les certificats au nouveau nœud sont triés sur la base du contenu actuel du chemin et le premier certificat est alors choisi. Par exemple, le constructeur peut examiner E(B) et noter qu'il contient une contrainte de nom interdisant "C". À ce point de l'arborescence de décision, E(B) ne pourrait pas être ajouté au chemin et produire un résultat valide car "C" est déjà dans le chemin. Par suite, le certificat E(B) devrait être placé au bas

de la liste de priorités.

Autrement, E(B) pourrait être éliminé de ce nouveau nœud dans l'arborescence. Il est très important de voir que ce certificat est éliminé seulement à ce nœud et seulement pour le chemin courant. Si la construction de chemin échoue en C et retraverse l'arborescence jusqu'à la première instance de E, E(B) pourrait quand même produire un chemin valide qui ne comporte pas C ; précisément R->A->B->E->D->Z->EE. Donc l'état à tout nœud ne devrait pas altérer l'état des nœuds précédents ou suivants. (Sauf pour l'attribution de priorités aux certificats dans les nœuds suivants.)

Dans cet exemple, le constructeur devrait aussi noter que E(C) est déjà dans le chemin et devrait le faire durer ou l'éliminer de ce nœud car les certificats ne doivent pas être répétés dans un chemin.

Si le constructeur élimine les deux certificats E(B) et E(C) à ce nœud, il lui reste seulement à choisir entre E(A) et E(D). Le chemin a maintenant six nœuds : EE, Z, D, E(1), C, et E(2). E(1) a quatre certificats, et E(2) deux, que le constructeur trie pour donner [E(A), E(D)]. Le chemin courant est maintenant E(A)->C(E)->E(C)->D(E)-> Z(D)->EE(Z). A(R) va être trouvé quand le septième nœud est ajouté au chemin et que le chemin se termine parce que une des ancres de confiance a été trouvée.

Si le premier chemin échoue à la validation, le constructeur de chemin va quand même avoir les sept nœuds et les informations d'état associées pour travailler. À la prochaine itération, le constructeur de chemin est capable de retraverser l'arborescence jusqu'à un point de décision provisoire, tel que A, et de choisir le prochain certificat dans la liste triée à A. Dans cet exemple, ce serait A(B). (A(R) a déjà été essayé.) Ce serait une impasse, et le constructeur retraverse jusqu'au prochain point de décision, E(2) où il va essayer D(E). Ce processus se répète jusqu'à ce que les traversées reviennent jusqu'à EE ou qu'un chemin valide soit trouvé. Si la traversée d'arborescence retourne à EE, tous les chemins possibles ont été épuisés et le constructeur peut conclure qu'aucun chemin valide n'existe.

Cette approche de tri des certificats afin d'optimiser la construction de chemin va donner de meilleurs résultats que de ne pas optimiser la traversée de l'arborescence. Cependant, le processus de construction de chemin peut être encore aminci en éliminant des certificats, et par suite, des branches entières de l'arborescence, lorsque les chemins sont construits.

3.2 Tri ou élimination

Considérons une situation de construction d'un chemin dans laquelle trois certificats de CA sont trouvés pour un certain certificat cible et que des priorités doivent leur être attribuées. Lorsque les certificats sont examinés, comme dans l'exemple précédent, une des trois a une contrainte de nom qui va invalider le chemin construit jusqu'alors. Lors du tri des trois certificats, celui-la va certainement se trouver en dernière place. Cependant, le logiciel de construction de chemin pourrait décider que cette condition élimine le certificat de toute considération à ce point du graphe, réduisant ainsi le nombre de choix de certificat de 33 % à ce point.

Note : Il est important de comprendre que l'élimination d'un certificat ne s'applique qu'à un seul point de décision durant la traversée d'arborescence. Le même certificat peut apparaître à nouveau à un autre point de l'arborescence ; à ce point il peut être ou non éliminé. Le paragraphe précédent détaille un exemple de ce comportement.

L'élimination de certificats pourrait éventuellement éliminer la traversée d'une grande infrastructure qui n'a jamais conduit à unechemin valide, en consommant beaucoup de temps. La question de trier ou éliminer est une de celles qui opposent la souplesse de l'interface logicielle à l'efficacité.

Pour être clair, si on élimine les chemins invalides lors de la construction, en retournant seulement ceux qui sont probablement valides, le résultat final sera un module de construction de chemin efficace. L'inconvénient en est que sauf si le logiciel le tolère, l'application appelante ne sera pas capable de voir ce qui ne va pas. L'utilisateur peut seulement voir le message d'erreur "Pas de chemin de certification trouvé" qui n'est pas très révélateur.

D'un autre côté, le module de construction de chemin pourrait opter pour n'exclure aucun chemin de certification. Le logiciel de construction de chemin pourrait alors retourner tous les chemins qu'il peut construire à partir du graphe de certificats. Il appartient alors au moteur de validation de déterminer lesquels sont valides et ceux qui ne le sont pas. L'utilisateur ou l'application appelante peut alors avoir tous les détails sur pourquoi chaque chemin échoue à la validation. L'inconvénient est évidemment celui des performances, car une application ou utilisateur final peut attendre assez longtemps pendant que les PKI inter certifiées sont parcourues afin de construire des chemins qui ne seront jamais validés.

Aucune option n'est une approche très désirable. Une option donne de bonnes performances pour les utilisateurs, qui en bénéficient. L'autre option permet aux administrateurs de diagnostiquer les problèmes avec la PKI, le répertoire, ou le logiciel. Voici quelques recommandations pour un moyen terme sur ce problème.

D'abord, il est fortement recommandé aux développeurs de sortir un enregistrement détaillé des informations à partir du

logiciel de construction de chemin. L'enregistrement devrait explicitement indiquer chaque choix que fait le constructeur et pourquoi. Il devrait clairement identifier quels certificats sont trouvés et utilisés à chaque étape de la construction de chemin. Si on veille à produire un enregistrement utile, les administrateurs de PKI et le personnel du bureau d'assistance auront d'amples informations pour diagnostiquer un problème de la PKI. Idéalement, il y aurait un mécanisme pour activer et désactiver cet enregistrement, afin qu'il ne tourne pas tout le temps. De plus, il est recommandé que l'enregistrement contienne des informations pour qu'un développeur ou essayeur puisse recréer les chemins essayés par le logiciel de construction de chemin, pour assister les diagnostics et les essais.

Deuxièmement, il est désirable de retourner quelque chose d'utile pour l'utilisateur. L'approche la plus facile est probablement de mettre en œuvre un module de construction de chemin en "mode duel". Dans le premier mode [mode 1], le logiciel élimine tous les chemins qui ne vont pas se valider, ce qui le rend très efficace. Dans le second mode [mode 2], toutes les méthodes de tri sont encore appliquées, mais aucun chemin n'est éliminé sur la base des méthodes de tri. Avoir le mode duel permet au module d'échouer d'abord à trouver un chemin valide, mais retourne encore un chemin invalide (en supposant qu'il en existe un) en passant au second mode assez longtemps pour générer un seul chemin. Cela donne un moyen terme – le logiciel est très rapide, mais en retournant quand même quelque chose qui donne à l'utilisateur une erreur plus spécifique que "pas de chemin trouvé".

Troisièmement, il peut être utile de n'exclure aucun chemin, mais plutôt de limiter le nombre de chemins qui peuvent être construits avec une entrée particulière. En supposant que le module de construction de chemin est conçu pour retourner "le meilleur chemin en premier", les chemins qui ont le plus de chances de se valider seront retournés avant que cette limite soit atteinte. Une fois la limite atteinte, le module peut arrêter la construction de chemins, fournissant une réponse plus rapide à l'appelant que celle qui construit tous les chemins possibles.

Finalement, le développeur détermine comment traiter le compromis entre efficacité et fourniture des informations. Un développeur pourrait choisir le moyen terme d'opter pour mettre en œuvre des optimisations comme les règles d'élimination et pas d'autres. Un développeur pourrait valider les signatures de certificat, ou même vérifier l'état de révocation tout en construisant le chemin, et prendre ensuite ses décisions sur la base du résultat de ces vérifications pour savoir si il faut éliminer le certificat en question.

Le présent document suggère l'approche suivante :

1) Pendant la construction de chemins, éliminer tous les certificats qui ne satisfont pas toutes les exigences de validation de chemin avec les exceptions suivantes :
   a. ne pas vérifier l'état de révocation si cela exige une recherche dans un répertoire ou un accès réseau ;
   b. ne pas vérifier les signatures numériques (voir des considérations supplémentaires au paragraphe 8.1, "Considérations générales pour la construction d'un chemin de certification") ;
   c. ne pas vérifier quelque chose qui ne peut pas être vérifié au titre du processus itératif de traversée de l'arborescence ;
   d. créer un enregistrement détaillé, si ce dispositif est activé ;.
   e. si on ne peut pas trouver de chemin, le constructeur de chemin passe au "mode 2" et permet la construction d'un seul mauvais chemin ;
   i. retourner le chemin avec un indicateur d'échec, ainsi qu'une information d'erreur détaillant pourquoi le chemin est mauvais.

2) Si la construction de chemin réussit, valider le chemin conformément à [X.509] et [RFC3280] avec les recommandations suivantes :
   a. Pour améliorer les performances, ne pas revérifier les éléments déjà vérifiés par le constructeur de chemin. (Noter que si des chemins pré remplis sont fournis au système de construction de chemin, le chemin entier doit être complètement revalidé.)
   b. Si la validation du chemin a échoué, invoquer à nouveau le constructeur de chemin pour construire un autre chemin.
   - Toujours mémoriser les information d'erreur et de chemin de la première itération et retourner ceci à l'utilisateur pour le cas où aucun chemin valide ne serait trouvé. Comme le logiciel de construction de chemin a été conçu pour retourner le "meilleur chemin en premier", ce chemin devrait être montré à l'utilisateur.

Comme mentionné plus haut, le présent document recommande que les développeurs ne valident pas les signatures numériques qui ne vérifient pas l'état de révocation au titre du processus de construction de chemin. Cette recommandation se fonde sur deux hypothèses sur la PKI et son usage. D'abord, les signatures dans une PKI active sont généralement bonnes. Comme la validation de signature est coûteuse en termes de temps de traitement, il vaut mieux retarder la vérification de signature jusqu'à ce qu'un chemin complet soit trouvé et vérifier ensuite les signatures sur chaque certificat dans le chemin de certification en commençant par l'ancre de confiance (voir au paragraphe 8.1). Ensuite, il est très rare dans les environnements d'application normaux de rencontrer un certificat révoqué ; donc, la plupart des certificats validés ne seront pas révoqués. Par suite, il vaut mieux retarder la restitution des CRL ou autres informations d'état de révocation jusqu'à ce qu'un chemin complet ait été trouvé. Cela réduit la probabilité de restituer des informations d'état de révocation inutiles pendant la construction des chemins.

3.3 Représentation de l'arbre de décision

Il y a une multitude de façons de mettre en œuvre la construction de chemins de certification et autant de façons de représenter l'arborescence de décisions en mémoire.

La méthode décrite ci-dessous est une approche qui fonctionnera bien avec les méthodes d'optimisation énumérées plus loin. Bien que cette approche soit la meilleure que les auteurs de ce document aient mise en œuvre, elle n'est en aucune façon la seule. Les développeurs devraient adapter cette approche à leurs propres exigences ou peuvent trouver qu'une autre approche convient mieux à leur environnement, langage de programmation, ou style de programmation.

3.3.1 Représentation de nœud pour les entités CA

Un "nœud" dans le graphe de certification est une collection de certificats de CA avec des DN sujets identiques. Au minimum, pour chaque nœud, afin de pleinement mettre en œuvre les optimisations à suivre, le module de construction de chemin va avoir besoin d'être capable de garder trace des informations suivantes :

1. Les certificats contenus dans le nœud.
2. L'ordre de tri des certificats.
3. L'indicateur de certificat "courant".
4. L'ensemble actuel de politiques (il peut être partagé en ensembles d'autorité et de contraintes d'utilisateur, si désiré.) ; il est suggéré que l'encapsulation de l'ensemble de politiques dans un objet avec la logique de manipulation de l'ensemble comme les intersections à effectuer, les transpositions, etc., va simplifier la mise en œuvre.
5. Les indicateurs (requireExplicitPolicy, inhibitPolicyMapping, anyPolicyInhibit) et les valeurs correspondantes de skipCert.
6. Une méthode pour indiquer quels certificats sont éliminés ou retirés du nœud. Si les nœuds sont recrées à partir de l'antémémoire à la demande, il peut être plus simple de retirer les certificats éliminés du nœud.
7. Un indicateur"next" qui pointe sur le prochain nœud dans le chemin actuel.
8. Un indicateur "previous" qui pointe sur le nœud précédent dans le chemin actuel.

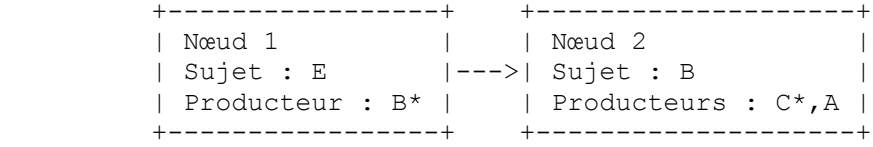
3.3.2 Utilisation des nœuds pour itération sur tous les chemins

Dans la forme la plus simple, un nœud est créé, les certificats sont triés, le prochain DN sujet requis est déterminé à partir du premier certificat, et un nouveau nœud est rattaché au chemin de certification via l'indicateur "next" (numéro 7 ci-dessus). Ce processus continue jusqu'à ce que le chemin se termine. (Noter que les certificats d'entité d'extrémité peuvent ne pas contenir de DN sujets comme permis par la [RFC3280]. Comme par définition les certificats d'entité d'extrémité ne produisent pas de certificat, cela n'a pas d'impact sur le procès.)

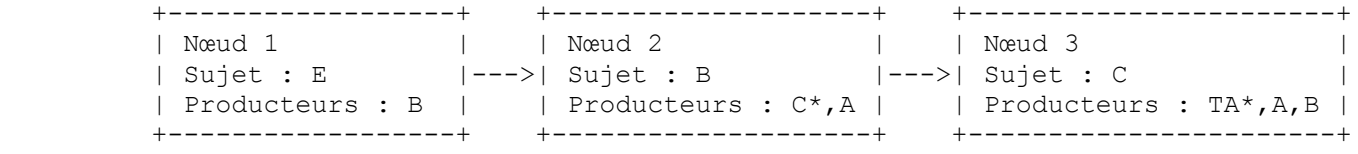
En se souvenant que l'algorithme suivant est conçu pour être mis en œuvre en utilisant la récurrence, on considère l'exemple de la Figure 12 et on suppose que le seul chemin dans le diagramme valide pour E est TA->A-> B->E:

Si notre module de construction de chemin est la construction d'un chemin dans la direction vers l'avant pour E, un nœud est d'abord créé pour E. Il n'y a pas de certificats à trier parce qu'un seul certificat existe, donc toutes les valeurs initiales sont chargées dans le nœud à partir de E. Par exemple, l'ensemble de politiques est extrait du certificat et mémorisé dans le nœud.

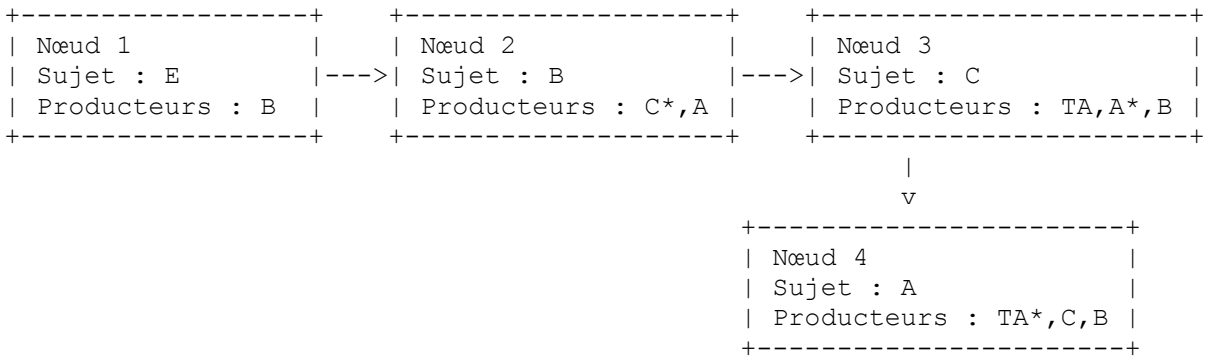
Ensuite, le DN producteur (B) est lu de E, et un nouveau nœud est créé pour B contenant les deux certificats produits à B -- B(A) et B(C). Les règles de tri sont appliquées à ces deux certificats et l'algorithme de tri retourne B(C);B(A). Cet ordre de tri est mémorisé et l'indicateur courant est réglé à B(C). Les indicateurs sont établis et les ensembles de politique sont calculés dans la mesure du possible par rapport à B(C). Le diagramme qui suit illustre l'état courant avec le certificat en cours indiqué avec une "*".



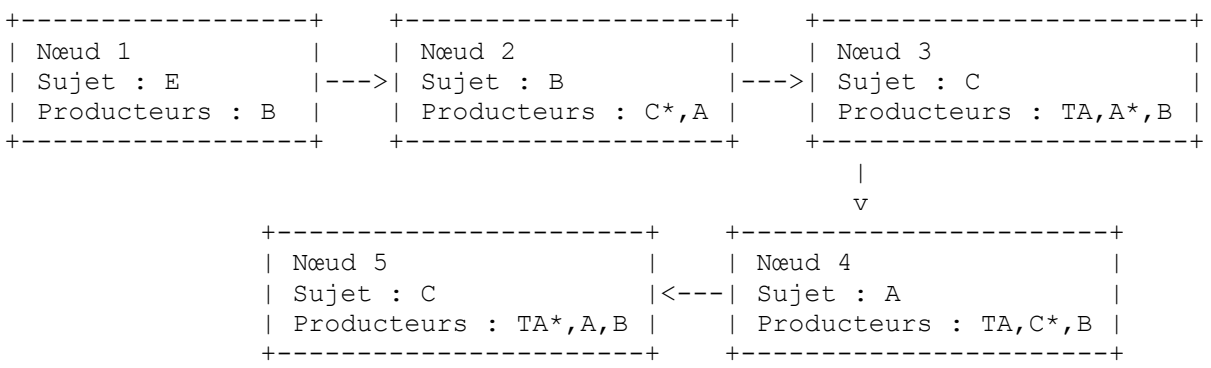
Ensuite, un nœud est créé pour C et les trois certificats lui sont ajoutés. L'algorithme de tri retourne les certificats triés dans l'ordre suivant : C(TA);C(A);C(B)



Reconnaissant que l'ancre de confiance a été trouvée, le chemin (TA->C->B->E) est validé mais échoue. (On se souvient que le seul chemin valide se trouve être TA->A->B->E.) le module de construction de chemin passe maintenant l'indicateur de certificat en cours dans le nœud 3 à C(A), et ajoute le nœud pour A.

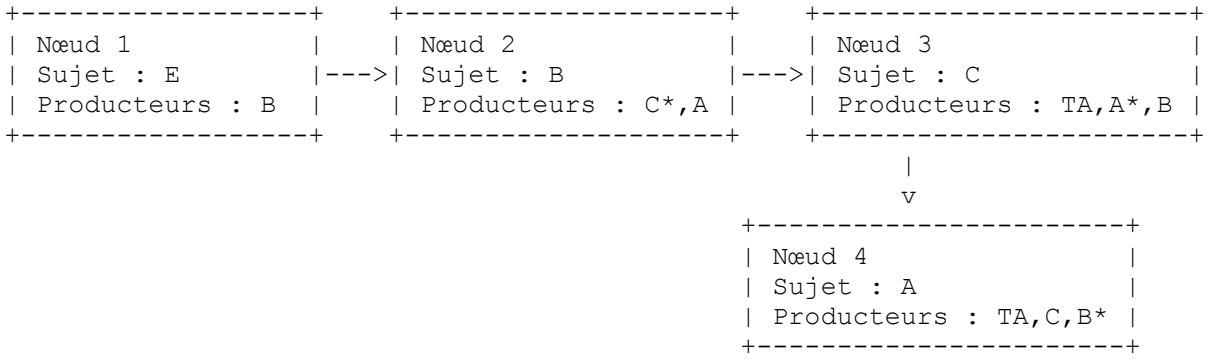


Le chemin TA->A->C->B->E est validé et échoue. Le module de construction de chemin passe maintenant l'indicateur courant dans le nœud 4 à A(C) et ajoute un nœud pour C.



À cette jonction, la décision de permettre la répétition de nom et de clé vient au premier plan. Si le module de construction de chemin de certification NE permet PAS la répétition de nom et de clé, il n'y a pas de certificat dans le nœud 5 qui puisse être utilisé. (C et la clé publique correspondante sont déjà dans le chemin au nœud 3.) À ce point, le nœud 5 est retiré du chemin en cours et l'indicateur courant de certificat sur le nœud 4 est déplacé à A(B).

Si à la place, le module interdit seulement la répétition de certificats, C(A) est éliminé du nœud 5 car il est utilisé au nœud 3, et la construction de chemin continue d'abord en validant TA->C->A-> C->B->E, et ensuite en continuant d'essayer de construire des chemins à travers C(B). Après que cela aussi échoue à fournir un chemin valide, le nœud 5 est retiré du chemin courant et l'indicateur de certificat en cours sur le nœud 4 est passé à A(B).



Un nouveau nœud 5 est maintenant créé pour B. Tout comme avec le nœud 5 précédent, si on ne répète pas les noms et clés, B n'offre pas non plus de certificat qui puisse être utilisé (B et la clé publique de B sont utilisés dans le nœud 2) de sorte que le nouveau nœud 5 est aussi retiré du chemin. À ce point, tous les certificats dans le nœud 4 ont maintenant été essayés, de sorte que le nœud 4 est retiré du chemin, et l'indicateur courant sur le nœud 3 est passé à C(B).

Comme ci-dessus, si on permet la répétition de nom et de clé, B(C) est retiré du nouveau nœud 5 (B(C) est déjà utilisé dans le nœud 3) et les chemins sont tentés à travers le certificat restant B(A). Après que cela a échoué, on est conduit à retirer le

nœud 5 du chemin. À ce point, tous les certificats dans le nœud 4 ont maintenant été essayés, de sorte que le nœud 4 est retiré du chemin, et l'indicateur en cours sur le nœud 3 est passé à C(B).

Ce processus continue jusqu'à ce que tous les certificats dans le nœud 1 (si il se trouve qu'il y en ait plus d'un) aient été essayés, ou jusqu'à ce qu'un chemin valide ait été trouvé. Une fois le processus terminé et au cas où aucun chemin valide n'a été trouvé, on peut conclure qu'aucun chemin ne peut être trouvé de E à TA.

3.4 Mise en œuvre de l'optimisation de construction de chemin

Le paragraphe qui suit décrit les méthodes qui peuvent être utilisées pour optimiser le processus de construction de chemin de certification en triant les certificats. L'optimisation comme décrite plus tôt cherche à donner des priorités dans une liste de certificats, donnant effectivement des priorités (pondération) aux branches du graphe/arborescence. Les méthodes d'optimisation peuvent être utilisées pour allouer une marque cumulative à chaque certificat. Le processus de marquage des certificats revient à peser chaque certificat à l'égard des méthodes d'optimisation qu'un développeur choisit de mettre en œuvre, et ensuite d'ajouter la marque pour chaque pesée à une marque cumulative pour chaque certificat. Après avoir achevé cela pour chaque certificat d'un point d'embranchement donné de l'arborescence des décisions du constructeur, les certificats peuvent être triés afin que le certificat à la plus forte marque soit choisi en premier, le second plus fort est choisi en second, etc.

Par exemple, supposons que le constructeur de chemins ait seulement ces deux simples méthodes de tri :
1) Si le certificat a un identifiant de clé de sujet, + 5 à la marque.
2) Si le certificat a un identifiant de clé d'autorité, + 10 à la marque.

Et il examine ensuite trois certificats :
1) Produit par CA 1 ; a un identifiant de clé d'autorité ; la marque est 10.
2) Produit par CA 2 ; a un identifiant de clé de sujet ; la marque est 5.
3) Produit par CA 1 ; a un identifiant de clé de sujet et un identifiant de clé d'autorité ; la marque est 15.

Les trois certificats sont triés en ordre descendant en commençant par la plus grosse marque : 3, 1, et 2. Le logiciel de construction de chemin devrait d'abord essayer la construction du chemin par le certificat 3. Ensuite, il devrait essayer le certificat 1. Enfin, il devrait essayer la construction d'un chemin par le certificat 2.

Les méthodes d'optimisation suivantes spécifient des essais que les développeurs peuvent choisir d'effectuer, mais on ne suggère pas les marques pour ces méthodes. Les développeurs devraient plutôt évaluer chaque méthode par rapport à l'environnement dans lequel l'application va fonctionner, et allouer des pondérations à chacune conformément au logiciel de construction de chemin. De plus, beaucoup de méthodes d'optimisation ne sont pas de nature binaire. Certaines sont à trois valeurs, et certaines conviendront bien pour des échelles glissantes ou exponentielles. Finalement, celui qui met en œuvre décide des mérites relatifs de chaque optimisation par rapport à son propre logiciel ou infrastructure.

Au delà des marques pour chaque méthode, de nombreuses méthodes peuvent être utilisées pour éliminer des branches durant la traversée d'arborescence plutôt que simplement les marquer et les pondérer. Tous les cas où des certificats peuvent être éliminés sur la base d'une méthode d'optimisation sont notés avec les descriptions des méthodes.

Beaucoup des méthodes de tri décrites ci-dessous se fondent sur ce qui a été perçu par les auteurs comme commun dans les PKI. Beaucoup des méthodes visent à accélérer la construction de chemin pour la PKI courante, mais il y a des cas où presque toutes les méthodes de tri pourraient conduire à une construction de chemin inefficace. Le comportement désiré est que bien qu'une méthode puisse mener l'algorithme dans la mauvaise direction pour une certaine situation ou configuration, les méthodes restantes vont surmonter la ou les méthodes erratiques et envoyer la traversée de chemin sur la branche correcte de l'arborescence plus souvent que pas. Cela ne sera certainement pas vrai pour tous les environnements et configurations, et ces méthodes peuvent devoir être infléchies pour plus d'optimisation dans l'environnement cible de fonctionnement de l'application.

On notera enfin que la liste que contient le présent document n'est pas entendue comme exhaustive. Un développeur peut souhaiter définir des méthodes de tri supplémentaires si l'environnement de fonctionnement en impose le besoin.

3.5 Méthodes choisies pour trier les certificats

Le lecteur ne devrait pas tirer de conclusions spécifiques sur les mérites ou marques relatifs de chacune des méthodes suivantes sur la base de l'ordre de leur apparition. Le mérite relatif de tout critère de tri dépend entièrement des spécificités de l'environnement de fonctionnement. Pour presque toutes les méthodes, un exemple peut être créé pour démontrer que la méthode est efficace et un contre exemple pourrait être conçu pour démontrer qu'elle est inefficace.

Chaque méthode de tri est indépendante et peut (ou non) être utilisée pour allouer des marques supplémentaires à chaque

certificat essayé. La mise en œuvre décide quelles méthodes utiliser et quelle pondération lui affecter. Comme noté plus haut, cette liste n'est pas exhaustive.

De plus, le chaînage de nom (qui signifie que le nom de sujet du producteur du certificat correspond au nom de producteur du certificat produit) n'est pas traité comme une méthode de tri car l'adhésion à cela est exigée pour construire l'arborescence des décisions auxquelles ces méthodes seront appliquées. Aussi non visée dans les méthodes de tri est la prévention de répétition des certificats. Les constructeurs de chemin devraient traiter le chaînage de nom et la répétition de certificat sans égard à l'approche d'optimisation.

Chaque description de méthode de tri spécifie si la méthode peut être utilisée pour éliminer des certificats, le nombre de valeurs numériques possibles (pondération de tri) pour la méthode, les composants du paragraphe 2.6 qui sont requis pour mettre en œuvre la méthode, les descriptions de méthode vers l'avant et en sens inverse, et finalement une justification de l'inclusion de la méthode.

Pour ce qui est de l'élimination des certificats, il est important de comprendre que les certificats sont éliminés seulement à un certain point de décision pour de nombreuses méthodes. Par exemple, le chemin construit jusqu'au certificat X peut être invalidé à cause de contraintes de noms par l'ajout du certificat Y. À ce point de décision seulement, Y pourrait être éliminé de la suite du processus. À un point de décision futur, tout en construisant ce même chemin, l'ajout de Y peut ne pas invalider le chemin.

Pour certaines autres méthodes de tri, des certificats pourraient être éliminés entièrement du processus. Par exemple, des certificats avec des algorithmes de signature non acceptés ne pourraient être inclus dans aucun chemin ni validés. Bien que le constructeur de chemin puisse certainement être conçu pour fonctionner de cette façon, il est suffisant de toujours n'éliminer les certificats que pour un certain point de décision quelle qu'en soit la cause.

3.5.1 basicConstraints est présent et cA égal Vrai

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : binaire.
Composants exigés : aucun
Méthode vers l'avant : les certificats où basicConstraints est présent et cA=VRAI, ou ceux désignés comme certificats de CA hors bande ont priorité. Les certificats sans basicConstraints, avec basicConstraints et cA=FAUX, ou ceux qui ne sont pas désignés comme certificat de CA hors bande peuvent être éliminés ou ont une priorité zéro.
Méthode inverse : la même que vers l'avant sauf à l'égard des certificats d'entité d'extrémité à la terminaison du chemin.
Justification : selon la [RFC3280], basicConstraints est exigé avec cA=VRAI dans tous les certificats de CA, ou doit être vérifié via un mécanisme hors bande. Un chemin valide ne peut pas être construit si cette condition n'est pas satisfaite.

3.5.2 Algorithmes de signature reconnus

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : binaire.
Composants exigés : aucun.
Méthode vers l'avant : les certificats contenant des algorithmes de signature et de clé publique reconnus [RFC3279] ont priorité.
Méthode inverse : la même que vers l'avant.
Justification : si le logiciel de construction de chemin n'est pas capable de traiter les signatures associées au certificat, le chemin de certification ne peut pas être validé.

3.5.3 keyUsage est correct

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : binaire.
Composants exigés : aucun.
Méthode vers l'avant : Si keyUsage est présent, les certificats avec keyCertSign établi ont une priorité de 100 %. Si keyUsage est présent et keyCertSign n'est pas établi, le certificat peut être éliminé ou avoir une priorité zéro. Tous les autres ont une priorité zéro.
Méthode inverse : la même que vers l'avant sauf à l'égard des certificats d'entité d'extrémité à la terminaison du chemin.
Justification : un chemin de certification valide ne peut pas être construit à travers un certificat de CA qui a un keyUsage inapproprié. Noter qu'il n'est pas obligé que digitalSignature soit établi dans un certificat de CA.

3.5.4 Time (T) tombe dans la période de validité du certificat

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : binaire
Composants exigés : aucun
Méthode vers l'avant : les certificats qui contiennent l'heure requise (T) dans sa période de validité ont une priorité de 100 %. Autrement, le certificat est éliminé ou a la priorité zéro.
Méthode inverse : la même que vers l'avant.
Justification : un chemin de certification valide ne peut pas être construit si T tombe en dehors de la période de validité du certificat.

Note : Une attention particulière devrait être portée à retourner une erreur significative à l'appelant, en particulier au cas où le certificat cible ne satisfait pas ce critère, si cette méthode de tri est utilisée pour l'élimination (par exemple, le certificat est expiré ou n'est pas encore valide).

3.5.5 Le certificat a déjà été validé

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : binaire.
Composants exigés : antémémoire de chemin de certification.
Méthode vers l'avant : un certificat qui est présent dans l'antémémoire du chemin de certification a la priorité.
Méthode inverse : ne s'applique pas. (La validité d'un certificat par rapport à une validité inconnue ne dit rien sur la direction correcte dans l'arborescence des décisions. En d'autres termes, savoir la validité d'un certificat de CA n'indique pas que la cible va plus probablement se trouver sur ce chemin plutôt qu'un autre.)
Justification : les certificats dans l'antémémoire du chemin ont été validés précédemment. En supposant que les contraintes initiales n'ont pas changé, il est très probable que le chemin de ce certificat à une ancre de confiance est encore valide. (Des changements aux contraintes initiales peuvent être cause qu'un certificat précédemment considéré comme valide ne le soit plus.)

Note : Il est important que les éléments dans l'antémémoire du chemin aient une durée de vie appropriée. Par exemple, il pourrait être inapproprié de mettre en antémémoire une relation au delà de la période pendant laquelle la CRL qui s'y rapporte va être de confiance pour l'application. Il est aussi critique de considérer les certificats et les CRL qui sont plus loin sur le chemin quand on établit les durées de vie en antémémoire. Par exemple, si le certificat du producteur expire dans dix jours, mais si le certificat produit est valide pendant 20 jours, mettre en antémémoire la relation au delà de 10 jours serait inapproprié.

3.5.6 Signatures déjà vérifiées

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : binaire
Composants exigés : antémémoire de chemin.
Méthode vers l'avant : Si une relation précédemment vérifiée existe dans l'antémémoire de chemin entre le certificat sujet et une clé publique présente dans un ou plusieurs certificats de producteur, tous les certificats contenant ladite clé publique ont une priorité supérieure. Les autres certificats peuvent être éliminés ou réglés à la priorité zéro.
Méthode inverse : Si une relation de mauvaise signature existe entre des certificats, cette relation peut être utilisée pour éliminer des certificats potentiels de l'arborescence des décisions. Rien ne peut être conclu sur la probabilité de trouver un certain certificat cible en descendant une branche contre une autre en utilisant des relations de bonne signature connues.
Justification : Si la clé publique dans un certificat (A) a été précédemment utilisée pour vérifier une signature sur un second certificat (B), tout certificat contenant la même clé que (A) peut être utilisé pour vérifier la signature sur (B). De même, un certificat qui ne contient pas la même clé que (A) ne peut pas être utilisé pour vérifier la signature sur (B). Cette méthode de direction vers l'avant est particulièrement forte pour multiplier les CA inter certifiées après qu'un retournement de clé s'est produit.

3.5.7 Contraintes de longueur de chemin

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : binaire.
Composants exigés : aucun.
Méthode vers l'avant : Les certificats qui ont des contraintes de base et qui contiennent une contrainte de longueur de chemin qui va invalider le chemin actuel (la longueur courante est connue car le logiciel est construit à partir du certificat cible) peuvent être éliminés ou réglés à la priorité zéro. Autrement, la priorité est 100 %.
Méthode inverse : Cette méthode peut être appliquée en sens inverse. Pour ce faire, le constructeur garde une contrainte de longueur de chemin courant variable et établit ensuite une priorité zéro pour les certificats (ou les élimine) qui

violent cette contrainte.
Justification : un chemin valide ne peut pas être construit si la contrainte de longueur du chemin a été violée.

3.5.8 Contraintes de nom

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : binaire.
Composants exigés : aucun.
Méthode vers l'avant : les certificats qui contiennent des contraintes de nom qui seraient violées par les certificats déjà dans le chemin à ce point reçoivent une priorité de zéro ou sont éliminés.
Méthode inverse : les certificats qui vont permettre un traitement réussi de toute contrainte de nom présente dans le chemin à ce point reçoivent une priorité supérieure.
Justification : un chemin valide ne peut pas être construit si des contraintes de nom sont violées.

3.5.9 Le certificat n'est pas révoqué

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : trois.
Composants exigés : antémémoire de CRL.
Méthode vers l'avant : Si une CRL en cours pour un certificat est présente dans l'antémémoire de CRL, si le numéro de série du certificat n'est pas dans la CRL, le certificat a la priorité. Si le numéro de série du certificat est présent dans la CRL, il a la priorité zéro. Si une réponse OSCP (d'une fraîcheur acceptable) est disponible pour un certificat, et identifie le certificat comme valide, le certificat a la priorité. Si une réponse OCSP est disponible pour un certificat, et identifie le certificat comme invalide, le certificat a la priorité zéro.
Méthode inverse : comme vers l'avant. Autrement, le certificat peut être éliminé si la CRL ou la réponse OCSP est vérifiée. C'est-à-dire, vérifie pleinement la CRL ou la signature et la relation de la réponse OCSP au certificat en question en accord avec la [RFC3280]. Bien que ceci soit viable, la vérification de signature exigée la rend moins attractive qu'une méthode d'élimination. Il est suggéré que cette méthode ne soit utilisée que pour le tri et que les réponses de CRL et OCSP soient validées après la construction de chemin.
Justification : les certificats connus pour n'être pas révoqués peuvent être considérés plus probablement comme valides que les certificats pour lesquels l'état de révocation n'est pas connu. Ceci est de plus justifié si la validation de réponse de CRL ou OCSP est effectuée après la validation de chemin – les réponses de CRL ou OCSP ne sont restituées que lorsque des chemins complets sont trouvés.

Note : Une attention particulière devrait être apportée à permettre que des erreurs significatives soient communiquées à l'appelant, en particulier dans les cas où le certificat cible est révoqué. Si un constructeur de chemin élimine des certificats en utilisant des réponses de CRL ou OCSP, certaines informations d'état devraient être préservées afin qu'une erreur significative puisse être retournée dans le cas où aucun chemin n'est trouvé.

3.5.10 Producteur trouvé dans l'antémémoire de chemin

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : binaire.
Composants exigés : antémémoire de chemin de certification.
Méthode vers l'avant : un certificat dont le producteur a une ou des entrées dans l'antémémoire de chemin a la priorité.
Méthode inverse : ne s'applique pas.
Justification : comme l'antémémoire du chemin contient seulement des entrées pour les certificats qui ont été antérieurement validés jusqu'à une ancre de confiance, il est plus probable que le même chemin ou un nouveau chemin puisse être construit à partir de ce point jusqu'à l'ancre de confiance ou une des ancres de confiance. Pour les certificats dont les producteurs ne se trouvent pas dans l'antémémoire du chemin, rien ne peut être conclu.

Note : Cette méthode n'est pas la même que la méthode nommée "Certificat validé précédemment". Il est possible que cette méthode de tri s'évalue à vrai alors que l'autre méthode pourrait s'évaluer à zéro.

3.5.11 Producteur trouvé dans le protocole d'application

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : binaire.
Composants exigés : antémémoire de chemin de certification.
Méthode vers l'avant : Si le producteur d'un certificat envoyé par la cible par le protocole d'application (SSL/TLS, S/MIME, etc.), correspond au signataire du certificat qu'on cherche, ce certificat a alors priorité.
Méthode inverse : Si le sujet d'un certificat correspond au producteur d'un certificat envoyé par la cible par le protocole d'application (SSL/TLS, S/MIME, etc.) ce certificat a alors priorité.

Justification : Le protocole d'application peut contenir des certificats que l'envoyeur considère valables pour la construction du chemin de certification, et va probablement conduire à un chemin vers le certificat cible.

3.5.12 Confrontation des identifiants de clé (KID)

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : trois.
Composants exigés : aucun.
Méthode vers l'avant : Les certificats dont l'identifiant de clé sujette (SKID, *subject key identifier*) correspond à l'identifiant de clé d'autorité (AKID, *authority key identifier*) du certificat en cours ont la plus haute priorité. Les certificats sans SKID ont une priorité moyenne. Les certificats dont le SKID ne correspond pas à l'AKID du certificat en cours (si tous deux sont présents) ont la priorité zéro. Si le certificat en cours exprime le nom et le numéro de série du producteur dans le AKID, les certificats qui correspondent à ces deux identifiants ont la plus haute priorité. Les certificats qui correspondent seulement au nom du producteur dans le AKID ont une priorité moyenne.
Méthode inverse : Les certificats dont le AKID correspond au SKID du certificat en cours ont la plus haute priorité. Les certificats sans un AKID ont une priorité moyenne. Les certificats dont le AKID ne correspond pas au SKID du certificat en cours (si les deux sont présents) ont la priorité zéro. Si le certificat exprime le nom et le numéro de série du producteur dans le AKID, les certificats qui correspondent à ces deux identifiants dans le certificat en cours ont la plus haute priorité. Les certificats qui correspondent seulement au nom du producteur dans le AKID ont une priorité moyenne.
Justification : la confrontation d'identifiant de clé (KID, *Key Identifier*) est un mécanisme très utile pour guider la construction de chemin (qui est l'objet du certificat) et devrait dont recevoir une pondération élevée.

Note : Bien que sa présence soit exigée par la [RFC3280] il est extrêmement important que les KID soient utilisés seulement comme critère de tri ou comme conseil durant la construction de chemin de certification. Il n'est pas exigé que les KID correspondent durant la validation de chemin de certification et ils ne peuvent pas être utilisés pour éliminer les certificats. Ceci est d'une importance critique pour l'interopération à travers les domaines et les mises en œuvre multi fabricants où les KID peuvent n'être pas calculés de la même façon.

3.5.13 Traitement de politique

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : trois.
Composants exigés : aucun.
Méthode vers l'avant : Les certificats qui satisfont au chaînage de politique vers l'avant ont la priorité. (Voir les détails à la Section 4 intitulée "Chaînage de politique vers l'avant".) Si l'appelant a fourni un ensemble initial de politique et n'a pas établi le fanion "initial-require-explicit", le poids de cette méthode de tri devrait être augmenté. Si le fanion "initial-require-explicit-policy" était établi par l'appelant ou par un certificat, les certificats peuvent être éliminés.
Méthode inverse : Les certificats qui contiennent des transposition de politiques qui vont permettre la réussite de la politique de traitement du chemin jusqu'à ce point ont la priorité. Si l'appelant a fourni un ensemble de politique initial et n'a pas établi le fanion "initial-require-explicit", la pondération de cette méthode de tri devrait être augmentée. Les certificats ne peuvent être éliminés que si "initial-require-explicit" était établi par l'appelant ou si "require-explicit-policy" était établi par un certificat dans le chemin à ce point.
Justification : Dans un environnement qui utilise une politique, les certificats qui réussissent à propager les politiques font plus vraisemblablement partie d'un chemin de certification prévu que ceux qui ne le réussissent pas.

Dans la construction vers l'avant, il est toujours possible qu'un certificat plus proche de l'ancre de confiance établisse l'indicateur require-explicit-policy ; donc donner la préférence aux chemins de certification qui propagent des politiques peut augmenter la probabilité de trouver un chemin valide en premier. Si l'appelant (ou un certificat dans le chemin courant) a spécifié ou établi l'indicateur initial-require-explicit-policy comme vrai, cette méthode de tri peut aussi être utilisée pour éliminer les certificats lors de la construction dans la direction vers l'avant.

Si la construction est à l'inverse, il est toujours possible qu'un certificat plus loin sur le chemin établisse l'indicateur require-explicit-policy ; donc, donner la préférence aux certificats qui propagent les politiques va bien servir dans ce cas. Dans celui où require-explicit-policy est établi par des certificats ou par l'appelant, les certificats peuvent être éliminés par cette méthode.

3.5.14 Intersection des politiques avec l'ensemble de politiques recherché

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : Additif.

Composants exigés : aucun.

Méthode vers l'avant : Les certificats qui affirment des politiques trouvées dans le "initial-acceptable-policy-set" ont priorité. Chaque politique correspondante supplémentaire peut avoir un effet additif sur la marque totale. Autrement, cela peut être binaire ; cela correspond : 1 ou plus, ou cela ne correspond pas, zéro.

Méthode inverse : Les certificats qui affirment des politiques trouvées dans le certificat cible ou transposent des politiques à ceux trouvés dans le certificat cible ont priorité. Chaque politique correspondante supplémentaire pourrait avoir un effet additif sur la marque totale. Autrement, ce pourrait être binaire ; cela correspond, 1 ou plus, ou cela ne correspond pas, zéro.

Justification : Vers l'avant, comme le chemin s'approche de l'ancre de confiance dans un environnement inter certifié, les politiques affirmées dans les certificats de CA vont correspondre à celle du domaine de l'appelant. Comme l'ensemble initial de politique acceptable est spécifié dans le domaine de l'appelant, les correspondances peuvent indiquer que la construction de chemin s'approche au plus près d'une ancre de confiance désirée. Dans la direction inverse, trouver des politiques qui correspondent à celles du certificat cible peut indiquer que le chemin s'approche du domaine cible.

3.5.15 Confrontation de noms distinctifs de point d'extrémité

Peut être utilisé pour éliminer des certificats : non.

Nombre de valeurs possibles : binaire.

Composants exigés : aucun.

Méthode vers l'avant : Les certificats dont le producteur correspond exactement au DN sujet d'ancre de confiance ont priorité.

Méthode inverse : Les certificats dont le sujet correspond exactement au DN de producteur de l'entité cible ont priorité.

Justification : Vers l'avant, si le DN de producteur d'un certificat correspond au DN d'une ancre de confiance [X.501], il peut alors terminer le chemin. Dans la direction, inverse, si le DN sujet du certificat correspond au DN de producteur du certificat cible, il peut être le dernier certificat requis pour compléter le chemin.

3.5.16 Confrontation de noms distinctifs relatifs (RDN)

Peut être utilisé pour éliminer des certificats : non.

Nombre de valeurs possibles : échelle glissante.

Composants exigés : aucun.

Méthode vers l'avant : Les certificats qui correspondent à des RDN plus ordonnés entre le DN du producteur et un DN d'ancre de confiance ont priorité. Lorsque tous les RDN correspondent, cela donne la plus haute priorité.

Méthode inverse : Les certificats avec des DN sujets qui correspondent à plus de RDN avec le DN de producteur de la cible ont une priorité supérieure. Lorsque tous les RDN correspondent, cela donne la plus haute priorité.

Justification : Dans les PKI, les DN sont fréquemment construits dans le style d'une arborescence. Les plus forts nombres de correspondance peuvent indiquer que l'ancre de confiance va être trouvée dans cette direction au sein de l'arborescence. Noter que dans le cas où tous les RDN correspondent [X.501], cette méthode de tri apparaît refléter la précédente. Cependant, cette méthode de tri devrait être capable de produire une pondération de 100 % même si le DN de producteur a plus de RDN que l'ancre de confiance. Le DN de producteur a seulement besoin de contenir tous les RDN (dans l'ordre) de l'ancre de confiance.

Note : Dans le cas où tous les RDN correspondent, cette méthode de tri reflète la fonctionnalité de la précédente. Cela permet que des correspondances partielles soient pondérées différemment des correspondances exactes. De plus, cette méthode peut exiger beaucoup de traitement si beaucoup d'ancres de confiance sont présentes.

3.5.17 Certificats restitués d'un attribut de répertoire cACertificate

Peut être utilisé pour éliminer des certificats : non.

Nombre de valeurs possibles : binaire.

Composants exigés : antémémoire de certificat avec des fanions pour l'attribut duquel le certificat a été restitué et mémorisation/restitution de certificat distant en utilisant un répertoire.

Méthode vers l'avant : Les certificats restitués de l'attribut de répertoire cACertificate ont la priorité sur les certificats restitués de l'attribut crossCertificatePair. (Voir la [RFC2587].)

Méthode inverse : ne s'applique pas.

Justification : L'attribut de répertoire cACertificate contient des certificats produits par des sources locales et des certificats auto produits. En utilisant l'attribut de répertoire cACertificate avant l'attribut crossCertificatePair, l'algorithme de construction de chemin va (selon la configuration locale de PKI) tendre à montrer une préférence pour la PKI locale avant de s'aventurer dans des PKI externes inter certifiées. La plupart des applications de PKI d'aujourd'hui passent la plus grande partie de leur temps à traiter des informations qui proviennent de la PKI locale (celle de l'utilisateur) et la PKI locale est souvent très efficace pour la traversée à cause de la proximité et de la vitesse du réseau.

3.5.18 Algorithmes cohérents de clé publique et de signature

Peut être utilisé pour éliminer des certificats : oui.
Nombre de valeurs possibles : binaire
Composants exigés : aucun.
Méthode vers l'avant : Si la clé publique dans le certificat de producteur correspond à l'algorithme utilisé pour signer le certificat sujet, il a alors priorité. (Les certificats avec des algorithmes de clé publique et de signature qui ne correspondent pas peuvent être éliminés.)
Méthode inverse : Si la clé publique dans le certificat courant correspond à l'algorithme utilisé pour signer le certificat sujet, il a alors priorité. (Les certificats qui ont des algorithmes de clé publique et de signature qui ne correspondent pas peuvent être éliminés.)
Justification : Comme les algorithmes de clé publique et de signature ne sont pas cohérents, la signature sur le certificat sujet ne va pas réussir à se vérifier. Par exemple, si le certificat de producteur contient une clé publique RSA, il ne pourrait pas avoir produit un certificat sujet signé avec l'algorithme DSA avec SHA-1.

3.5.19 Producteur et nom de sujet similaires

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : échelle glissante.
Composants exigés : aucun.
Méthode vers l'avant : Les certificats rencontrés avec un DN sujet qui correspond avec plus de RDN avec le DN de producteur du certificat cible ont priorité.
Méthode inverse : la même que vers l'avant.
Justification : Comme il est généralement plus efficace de chercher le domaine local avant l'embranchement sur les domaines inter certifiés, utiliser d'abord des certificats avec des noms similaires tend à faire un constructeur de chemin plus efficace. Les certificats croisés issus de domaines externes vont généralement correspondre à moins de RDN (si il y en a) tandis que les certificats dans le domaine local vont fréquemment correspondre à plusieurs RDN.

3.5.20 Certificats dans l'antémémoire de certification

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : trois.
Composants exigés : antémémoire de certificat local et mémorisation/restitution de certificat distant (par exemple, un répertoire LDAP comme répertoire).
Méthode vers l'avant : Un certificat dont le certificat de producteur est présent dans l'antémémoire de certificat et est rempli avec des certificats a une priorité supérieure. Un certificat dont l'entrée de producteur est complètement remplie avec les données courantes (tous les attributs de certificat ont été cherchés dans la période de temporisation) a une priorité supérieure.
Méthode inverse : Si le sujet d'un certificat est présent dans l'antémémoire de certificat et est rempli avec des certificats, il a alors la priorité. Si l'entrée est complètement remplie avec les données courantes (tous les attributs de certificat ont été cherchés pendant la période de temporisation) il a alors la priorité.
Justification : La présence des valeurs de répertoire requises remplies dans l'antémémoire augment la probabilité que tous les certificats et CRL requis nécessaires pour compléter le chemin de ce certificat à l'ancre de confiance (ou cible si la construction est en sens inverse) sont présents dans l'antémémoire provenant d'un chemin antérieur qui est développé, éliminant par là le besoin d'un accès au répertoire pour compléter le chemin. Si aucun chemin ne peut être trouvé, le coût en performance est faible car les certificats n'ont probablement pas été restitués du réseau.

3.5.21 CRL courante trouvée dans l'antémémoire locale

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : binaire.
Composants exigés : antémémoire de CRL.
Méthode vers l'avant : Les certificats ont priorité si l'entrée de CRL du producteur existe et est remplie avec les données courantes dans l'antémémoire de CRL.
Méthode inverse : Les certificats ont priorité si l'entrée de CRL du sujet existe et est remplie avec les données courantes dans l'antémémoire de CRL.
Justification : Si la révocation est vérifiée seulement après qu'un chemin complet a été trouvé, cela indique qu'un chemin complet a été trouvé par cette entité en un point dans le passé, donc un chemin existe encore probablement. Cela aide aussi à réduire les restitutions distantes à ce qui est nécessaire.

3.6 Méthodes de tri des certificats pour les chemins de certification de signataire de révocation

Sauf en utilisant un répondeur OCSP configuré en local ou un autre service d'état de révocation de confiance configuré en local, les informations de révocation de certificat sont supposées être fournies par la PKI qui a produit le certificat. Il s'ensuit que lorsque on construit un chemin de certification pour un certificat de signataire de révocation, il est souhaitable de limiter l'algorithme de construction à la PKI qui a produit le certificat. Les méthodes de tri suivantes cherchent à ordonner les chemins possibles afin que le chemin de certification au signataire de révocation prévu soient trouvé en premier.

Ces méthodes de tri ne sont pas destinées à être utilisées au lieu de celles décrites dans la section précédente ; elles sont plus efficaces lorsque utilisées en conjonction avec celles du paragraphe 3.5. Certains des critères de tri ci-dessous ont des noms identiques à ceux du paragraphe précédant. Cela indique que les critères de tri décrits au paragraphe précédant sont légèrement modifiés lors de la construction du chemin de certification de signataire de révocation.

3.6.1 Ancres de confiance identiques

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : binaire.
Composants exigés : Indicateur Is-revocation-signer et l'ancre de confiance de l'autorité de certification.
Méthode vers l'avant : non applicable.
Méthode inverse : La construction de chemin devrait commencer à partir de la même ancre de confiance qu'utilisée pour valider l'autorité de certification avant d'essayer une autre ancre de confiance. Si une ancre de confiances existe avec une clé publique différente mais un DN sujet identique vers cette ancre de confiance d'autorité de certification, elle devrait être essayée avant celles dont le nom ne concorde pas.
Justification : Les informations de révocation pour un certain certificat devraient être produites par la PKI qui produit le certificat. Donc, la construction d'un chemin à partir d'une ancre de confiance différente de celle de l'autorité de certification n'est pas souhaitable.

3.6.2 Confrontation de noms distinctifs de point d'extrémité

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : binaire.
Composants exigés : Indicateur Is-revocation-signer et ancre de confiance de l'autorité de certification.
Méthode vers l'avant : fonctionne de la même façon que la méthode de tri décrite en 3.5.15, excepté qu'au lieu d'effectuer la confrontation avec les ancres de confiance, la confrontation de DN est effectuée seulement avec le DN d'ancre de confiance utilisé pour valider le certificat de CA.
Méthode inverse : non. Changé pour le chemin de certification du signataire de révocation.
Justification : Les informations de révocation pour un certain certificat devraient être produites par la PKI qui produit le certificat. Donc, la construction d'un chemin à une ancre de confiance différente de celle de la CA n'est pas souhaitable. Cette méthode de tri aide à guider la construction de chemin dans la direction avant vers l'ancre de confiance utilisée pour valider le certificat de CA.

3.6.3 Confrontation de noms distinctifs relatifs (RDN)

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : échelle glissante.
Composants exigés : Indicateur Is-revocation-signer et ancre de confiance de l'autorité de certification.
Méthode vers l'avant : Fonctionne de façon identique à la méthode de tri décrite en 3.5.16 excepté qu'au lieu d'effectuer la confrontation de RDN avec les ancres de confiance, elle est effectuée seulement avec le DN de l'ancre de confiance utilisée pour valider le certificat de CA.
Méthode inverse : non. Changé pour le chemin de certification du signataire de révocation.
Justification : Les informations de révocation pour un certain certificat devraient être produites par la PKI qui produit le certificat. Donc, la construction d'un chemin à une ancre de confiance différente de celle de la CA n'est pas souhaitable. Cette méthode de tri aide à guider la construction de chemin dans la direction avant vers l'ancre de confiance utilisée pour valider le certificat de CA.

3.6.4 Noms intermédiaires identiques

Peut être utilisé pour éliminer des certificats : non.
Nombre de valeurs possibles : binaire.
Composants exigés : Indicateur Is-revocation-signer et chemin de certification complet de l'autorité de certification.
Méthode vers l'avant : Si le DN de producteur dans le certificat correspond au DN de producteur d'un certificat dans le chemin d'autorité de certification, il a une priorité supérieure.

Méthode inverse : Si le DN sujet dans le certificat correspond au DN sujet d'un certificat dans le chemin de l'autorité de certification, il a une priorité supérieure.

Justification : Suivre le même chemin que le certificat devrait empêcher l'algorithme de construction de chemin d'errer dans une direction inappropriée. Noter que cette méthode de tri est indifférente au fait que le certificat soit auto produit. Ceci est avantageux dans cette situation parce qu'il ne serait pas souhaitable de diminuer la priorité d'un certificat de changement de clé.

4. Chaînage de politique vers l'avant

Il serait tentant de sauter à la conclusion que les politiques de certificat offrent peu d'assistance à la construction de chemin lorsque la construction part du certificat cible. Il est aisé de comprendre l'approche de "validation à tout va" à partir de l'ancre de confiance, et c'est beaucoup moins évident qu'une valeur quelconque puisse être déduite dans l'autre direction. Cependant, comme la validation de politique consiste en l'intersection de l'ensemble de politique du producteur avec l'ensemble de politique du sujet et la transposition de l'ensemble du producteur en ensemble du sujet, la validation de politique peut être faite lors de la construction d'un chemin dans la direction vers l'avant aussi bien qu'à l'inverse. C'est simplement une affaire d'inversion de la procédure. C'est-à-dire non pas que c'est idéal comme validation de politique lors de la construction à partir de l'ancre de confiance, mais que cela offre une méthode qui peut être utilisée pour presque éliminer ce qui a longtemps été considéré comme une faiblesse inhérente à la construction dans la direction avant (à partir du certificat cible).

4.1 Intersection simple

La plus basique forme de traitement de politique est l'intersection des ensembles de politique à partir du premier certificat de CA jusqu'au certificat cible. Heureusement, l'intersection des ensembles de politique va toujours donner le même ensemble final sans considération de l'ordre d'intersection. Cela permet le traitement d'intersections d'ensembles de politiques dans l'une ou l'autre direction. Par exemple, si l'ancre de confiance produit un certificat de CA (A) avec les politiques {X,Y,Z}, et si cette CA produit un autre certificat de CA (B) avec les politiques {X,Y}, et si CA B produit alors une troisième certificat de CA (C) avec l'ensemble de politiques {Y,G}, on calcule normalement l'ensemble de politiques à partir de l'ancre de confiance comme suit :

1) Intersection de A{X,Y,Z} avec B{X,Y} pour donner l'ensemble {X,Y}
2) Intersection de ce résultat, {X,Y} avec C{Y,G} pour donner l'ensemble final {Y}

Maintenant il a été montré que le certificat C est bon pour la politique Y.

C'est exactement la même procédure dans l'autre direction, mais en sens inverse :

1) Intersection de C{Y,G} avec B{X,Y} pour donner l'ensemble {Y}
2) Intersection de ce résultat {Y} avec A{X,Y,Z} pour donner l'ensemble final {Y}

Tout comme dans la direction inverse, il a été montré que le certificat C est bon pour la politique Y, mais cette fois dans la direction vers l'avant.

Lors de construction vers l'avant, le traitement de politique est assez semblable à celui du sens inverse -- le logiciel accorde sa préférence aux certificats qui propagent les politiques. Aucune de ces approche ne garantit qu'un chemin ayant des politiques valides sera trouvé, mais les deux aident à guider le chemin dans la direction où il devrait aller pour que les politiques se propagent.

Si l'appelant a fourni un ensemble initial de politiques acceptable, il est moins utile de l'utiliser pour la construction vers l'avant sauf si l'appelant établit aussi inhibit-policy-mapping *(interiction de transposition de politique)*. Dans ce cas, le constructeur de chemin peut contraindre la construction de chemin à propager les politiques qui existent dans le initial-acceptable-policy-set *(ensemble initial de politiques acceptables)*. Cependant, même si le inhibit-policy-mapping n'est pas établi, le initial-policy-set *(ensemble de politiques initial)* peut encore être utilisé pour guider la construction du chemin vers l'avant jusqu'à l'ancre de confiance désirée.

4.2 Transposition de politique

Lorsque une CA produit un certificat dans un autre domaine, un environnement avec des identifiants de politique disparates pour les siens propres, la CA peut faire usage de transpositions de politique pour avoir l'équivalence entre les politiques du domaine local et celles du domaine non local. Si dans l'exemple précédent, A avait inclus une transposition de politique qui transpose X en G dans le certificat qu'elle produit à B, C aurait été bon pour X et Y :

1) Intersection de A{X,Y,Z} avec B{X,Y} pour donner l'ensemble {X,Y}
2) Traiter les transpositions de politique dans le certificat de B (X se transpose en G) pour donner {G,Y} (le même que

{Y,G})
3) Intersection de ce résultat, {G,Y} avec C{Y,G} pour donner l'ensemble final {G,Y}

Comme les politiques sont toujours exprimées dans le domaine du consommateur d'assertions, le certificat C est dit être bon pour {X, Y}, mais pas {Y, G}. C'est parce que "G" ne veut rien dire dans le contexte de l'ancre de confiance qui a produit A sans la transposition de politique.

Lors d'une construction dans la direction avant, les politiques peuvent être "dé transposées" en inversant la procédure de transposition. Cette procédure est limitée par un important aspect : si la transposition de politique s'est produite vers l'avant, il n'y a pas de mécanisme qui permette de savoir à l'avance si un futur ajout au chemin courant va ou non invalider la chaîne de politiques (en supposant qu'il en existe une) en établissant inhibit-policy-mapping. Heureusement, ce fanion est rarement établi en pratique. La procédure suivante est pour le traitement de la transposition de politique vers l'avant :
1) Commencer par l'ensemble de politique de C {Y,G}
2) Appliquer la transposition de politique dans le certificat de B (X se transpose en G) en sens inverse pour donner {Y,X} (le même que {X,Y})
3) Faire l'intersection du résultat {X,Y} avec B{X,Y} pour donner l'ensemble {X,Y}
4) Faire l'intersection de ce résultat, {X,Y}, avec A{X,Y,Z} pour donner l'ensemble final {X,Y}

Tout comme dans la direction inverse, il est déterminé dans le sens avant que le certificat C est bon pour les politiques {X,Y}. Si durant cette procédure, un fanion inhibit-policy-mapping est rencontré, que devrait on faire ? Il est relativement aisé d'en garder aussi trace. Le logiciel conserve simplement un fanion sur toutes les politiques qui ont été propagées comme résultat d'une transposition; tout comme un simple booléen conservé avec les politiques dans l'ensemble. Imaginons maintenant que le certificat produit à A a la contrainte inhibit-policy-mapping exprimée avec une valeur de saut de certificat de zéro.
1) Commencer avec l'ensemble de politiques de C {Y,G}
2) Appliquer la transposition de politique dans le certificat de B et marquer X comme résultant d'une transposition. (X se transpose en G) en sens inverse pour donner {Y,Xm} (le même que {Xm,Y})
3) Faire l'intersection du résultat {Xm,Y} avec B{X,Y} pour donner l'ensemble {Xm,Y}
4) Le certificat de A exprime la contrainte de transposition de politique inhibée, donc on élimine toutes les politiques dans l'ensemble courant qui ont été propagées du fait de la transposition (qui est Xm) pour donner {Y}
5) Faire l'intersection de ce résultat, {Y} avec A{X,Y,Z} pour donner l'ensemble final {Y}

Si dans notre exemple, l'ensemble de politiques se trouve vide à un certain point (et que require-explicit-policy était établi) la construction de chemin pourrait revenir à essayer de traverser une autre branche de l'arborescence. Ceci est analogue à la fonctionnalité de construction de chemin utilisée dans la direction inverse quand l'ensemble de politiques devient vide.

4.3 Pondération du chaînage de politique vers l'avant

En supposant que le module de construction de chemin conserve l'ensemble courant de politiques vers l'avant, des pondérations peuvent être allouées en utilisant la procédure suivante :

1) Pour chaque certificat de CA marqué :
   a. Copier l'ensemble courant de politiques vers l'avant.
   b. Traiter les transpositions de politique dans le certificat de CA dans l'ordre des politiques "dé transposées", si il y en a.
   c. Faire l'intersection de l'ensemble résultant avec les politiques de certificat de CA.

Plus l'ensemble de politiques donné est grand, plus élevée est la marque pour ce certificat de CA.

2) Si un ensemble initial acceptable a été fourni, faire l'intersection de cet ensemble avec l'ensemble résultant pour chaque certificat de CA provenant de (1).

Plus l'ensemble résultant est grand, plus la marque pour ce certificat est élevée.

D'autres schémas de marque peuvent mieux fonctionner si l'environnement de fonctionnement l'impose.

5. Éviter les erreurs de construction de chemin

Cette Section définit certaines erreurs qui peuvent survenir durant le processus de construction de chemin, ainsi que des moyens d'éviter ces erreurs lors du développement des fonctions de construction de chemin.

5.1 Impasses

Lors de la construction de chemins de certification dans une structure de PKI non hiérarchique, un simple algorithme de construction de chemin pourrait échouer prématurément sans trouver un chemin existant à cause d'une "impasse". Considérons l'exemple de la Figure 14.

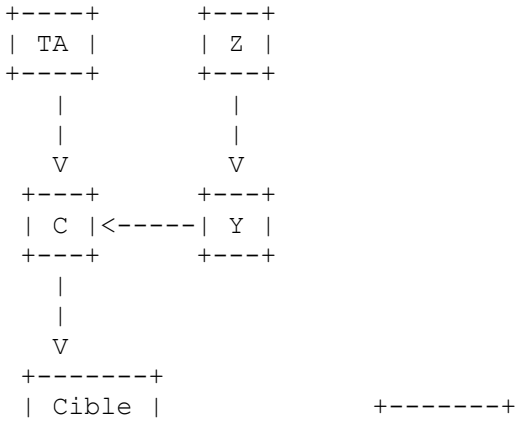


Figure 14 : Exemple d'impasse

Noter que dans l'exemple, C a deux certificats : un produit par Y, et l'autre produit par l'ancre de confiance. Supposons qu'un simple algorithme "trouver le producteur" soit utilisé, et que l'ordre dans lequel le constructeur de chemin a trouvé les certificats soit Cible(C), C(Y), Y(Z), Z(Z). Dans ce cas, Z n'a de certificat produit par aucune autre entité, et donc le processus simpliste de construction de chemin s'arrête. Comme Z n'est pas l'ancre de confiance du consommateur d'assertions, le chemin de certification n'est pas complet, et ne sera pas validé. Cet exemple montre que dans tout sauf la plus simple structure de PKI, une logique de construction de chemin supplémentaire devra traiter les cas où des entités ont produit plusieurs certificats provenant de différents producteurs. L'algorithme de construction de chemin va aussi avoir besoin d'être capable de retraverser l'arborescence des décisions et essayer un autre chemin afin d'être robuste.

5.2 Détection de boucle

Dans une structure de PKI non hiérarchique, un algorithme de construction de chemin peut être pris dans une boucle sans trouver de chemin existant. Considérons l'exemple suivant :

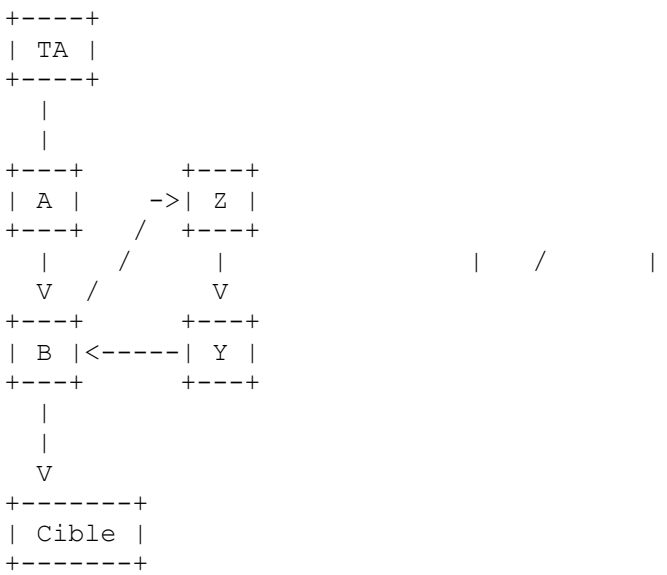


Figure 15 : Exemple de boucle

Supposons que dans cet exemple on utilise le plus simple algorithme "trouver le producteur", et que l'ordre dans lequel les certificats sont restitués soit Cible(B), B(Y), Y(Z), Z(B), B(Y), Y(Z), Z(B), B(Y), ... Une boucle s'est formée qui va être cause que le chemin correct (Cible, B, A) ne sera jamais trouvé. Le système de traitement de certificats va avoir besoin de reconnaître les boucles créées par des certificats dupliqués (qui sont prohibés par [X.509] dans un chemin) avant qu'ils se forment pour permettre au processus de construction de chemin de certification de continuer et trouver des chemins valides.

Les auteurs du présent document recommandent que la détection de boucles non seulement détecte la répétition d'un certificat dans le chemin, mais aussi détecte la présence des mêmes combinaisons de nom de sujet / nom de remplacement de sujet / clé publique sujette survenant deux fois dans le chemin. Une paire nom/clé ne devrait apparaître qu'une seule fois dans le chemin. (Voir au paragraphe 2.4.2 plus d'informations sur le raisonnement derrière cette recommandation.)

5.3 Utilisation des identifiants de clé

Des approches incohérentes et/ou incompatibles de calcul de l'identifiant de clé sujette et d'identifiant de clé d'autorité dans les certificats de clé publique peuvent causer des défaillances dans les algorithmes de construction de chemin de certification qui utilisent ces champs pour identifier les certificats même si des chemins de certification par ailleurs valides peuvent exister. Les mises en œuvre de construction de chemin devraient utiliser les identifiants de clé existants et ne pas tenter de recalculer les identifiants de clé sujettes. Il est extrêmement important que les identifiants de clé ne soient utilisés que comme critères ou indications de tri. Il n'est pas obligé que les KID correspondent durant la validation de chemin de certification et ils ne peuvent pas être utilisés pour éliminer les certificats. Ceci est d'une importance critique pour l'interopération à travers les domaines et les mises en œuvre multi fabricants où les KID peuvent n'être pas calculés de la même façon.

Les mises en œuvre de construction et de traitement de chemin ne devraient pas s'appuyer sur la forme de l'identifiant de clé d'autorité qui utilise le DN d'autorité et le numéro de série comme règle de correspondance restrictive, parce que l'inter certification peut conduire à ce que cette valeur ne corresponde pas aux certificats croisés.

5.4 Codage des noms distinctifs

Le logiciel de construction de chemin de certification ne devrait pas s'appuyer sur les DN qui sont codés comme chaîne imprimable. Bien que fréquemment codés comme chaîne imprimables, les DN peuvent aussi apparaître sous d'autres types, incluant des chaînes BMP ou des chaînes UTF8. Par suite, les systèmes logiciels qui ne savent pas traiter les DN codés comme BMPString et UTF8String peuvent se trouver dans l'incapacité de construire et valider certains chemins de certification.

De plus, les certificats conformes à la [RFC3280] sont obligés de coder les DN comme des chaînes UTF8 depuis le 1er janvier 2004. Le logiciel de construction de chemin de certification devrait être prêt à traiter les certificats à "retournement de nom" comme décrit dans la [RFC3280]. Noter que l'inclusion d'un certificat à "retournement de nom" dans un chemin de certification ne constitue pas une répétition d'un DN et d'une clé. Les mises en œuvre qui incluent le certificat à "retournement de nom" dans le chemin devraient s'assurer que les DN avec de codages différents sont considérés comme dissemblables. (Les mises en œuvre peuvent à la place traiter les DN correspondants de différents codages et n'auront donc pas besoin d'inclure des certificats à "retournement de nom" dans le chemin.)

6. Méthodes de restitution

La construction d'un chemin de certification exige la disponibilité des certificats et des CRL qui constituent le chemin. Il y a de nombreuses méthodes différentes pour obtenir ces certificats et CRL. Cette Section fait une liste de quelques façons courantes d'effectuer cette restitution, ainsi que des approches suggérées pour améliorer les performances. Cette Section n'est pas destinée à fournir une référence complète des méthodes de restitution de certificat et de CRL ou des optimisations qui seraient utiles dans la construction de chemin de certification.

6.1 Répertoires utilisant LDAP

La plupart des applications utilisent le protocole léger d'accès à un répertoire (LDAP, *Lightweight Directory Access Protocol*) pour restituer des données de répertoires conformément au modèle de X.500. Les applications peuvent rencontrer des répertoires qui prennent en charge LDAP v2 [RFC1777] ou LDAP v3 [RFC3377].

La spécification LDAP v3 définit une option de restitution d'attribut, l'option "binary". Lorsque spécifiée dans une demande de restitution LDAP, cette option est destinée à forcer le répertoire à ignorer toute représentation fondée sur la chaîne d'informations de répertoire codées en BER, et à envoyer le ou les attributs demandés en format BER. Comme tous les objets de PKI concernés sont des objets codés en BER, l'option "binary" devrait être utilisée. Cependant, tous les répertoires ne prennent pas en charge l'option "binary". Donc, les applications devraient être capables de demander des attributs avec et sans l'option "binary". Par exemple, si une application souhaite restituer l'attribut userCertificate, l'application devrait demander "userCertificate;binary". Si les informations désirées ne sont pas retournées, des mises en œuvre robustes peuvent opter pour demander aussi "userCertificate".

Les attributs suivants devraient être pris en compte par les développeurs d'applications de PKI lors de la restitution de certificat à partir de sources LDAP :

userCertificate : contient des certificats produits par une ou plusieurs autorités de certification avec un DN sujet qui correspond à celui de l'entrée de répertoire. C'est un attribut multi valeurs et toutes les valeurs devraient être reçues et prises en compte durant la construction de chemin. Bien qu'on s'attende normalement à ce que seuls les certificats d'entité d'extrémité soient mémorisés dans cet attribut, (par exemple, c'est l'attribut qu'une application va demander pour trouver le certificat de chiffrement d'une personne) les mises en œuvre peuvent opter pour chercher cet attribut lorsque elles regardent dans les entrées de CA pour rendre leur constructeur de chemin plus robuste. Si il est vide, la redondance ajoutée par son inclusion quand on demande déjà un ou deux des attributs ci-dessous est marginale.

cACertificate : contient les certificats auto produits (si il y en a) et tous certificats produits à cette autorité de certification par d'autres autorités de certification dans le même domaine. (Le domaine dépend de la politique locale.) C'est un attribut multi valeurs et toutes les valeurs devraient être reçues et prises en compte durant la construction de chemin.

crossCertificatePair : Dans les mises en œuvre conformes, crossCertificatePair est utilisé pour contenir tous les certificats, sauf ceux auto produits, produits à cette autorité de certification, ainsi que les certificats produit par son autorité de certification aux autres autorités de certification. Chaque valeur d'attribut est une structure qui contient deux éléments. L'élément issuedToThisCA contient les certificats produits à cette autorité de certification par d'autres autorités de certification. L'élément issuedByThisCA contient les certificats produits par cette autorité de certification aux autres autorités de certification. Les deux éléments de crossCertificatePair sont marqués comme facultatifs dans la définition ASN.1. Si les deux éléments sont présents dans une seule valeur, le nom du producteur dans un certificat doit correspondre au nom de sujet dans l'autre et vice versa, la clé publique sujette dans un certificat devra être capable de vérifier la signature numérique sur l'autre certificat et vice versa. Comme cette technologie a évolué, différents standard ont eu des exigences différentes à l'égard de l'endroit où ces informations peuvent se trouver. Par exemple, le schéma LDAP v2 [RFC2587] déclare que l'élément issuedToThisCA (qu'on a appelé "forward") de l'attribut crossCertificatePair est obligatoire et que l'élément issuedByThisCA (qu'on a appelé "reverse") est facultatif. À l'opposé, le paragraphe 11.2.3 de [X.509] exige que l'élément issuedByThisCA soit présent si la CA produit un certificat à une autre CA si le sujet n'est pas une CA subordonnée dans une hiérarchie. Les répertoires conformes se comportent comme requis par [X.509], mais les mises en œuvre robustes de construction de chemin peuvent vouloir restituer tous les certificats des attributs cACertificate et crossCertificatePair pour s'assurer que tous les certificats d'autorité de certification possibles sont obtenus.

certificateRevocationList : l'attribut certificateRevocationList contient une liste de révocation de certificat (CRL, *certificate revocation list*). Une CRL est définie dans la [RFC3280] comme une liste horodatée qui identifie les certificats révoqués, qui est signée par une CA ou producteur de CRL et qui est gratuitement disponible dans un répertoire public. Chaque certificat révoqué est identifié dans une CRL par son numéro de série de certificat. Il peut y avoir une ou plusieurs CRL dans cet attribut, et les valeurs devraient être traitées conformément à la [RFC3280].

authorityRevocationList : l'attribut authorityRevocationList contient aussi des CRL. Ces CRL contiennent des informations de révocation concernant les certificats produits aux autres CA. Il peut y avoir une ou plusieurs CRL dans cet attribut, et les valeurs devraient être traitées conformément à la [RFC3280].

Les systèmes de traitement de chemin de certification qui prévoient d'être interopérables avec diverses structures de PKI et conceptions de répertoires devraient au minimum être capables de restituer et traiter les attributs userCertificate, cACertificate, crossCertificatePair, certificateRevocationList, et authorityRevocationList provenant des entrées de répertoire.

6.2 Accès aux mémorisations de certificats via HTTP

Une autre méthode possible de restitution de certificat est d'utiliser HTTP comme mécanisme d'interface pour restituer les certificats et CRL des répertoires de PKI. Un document courant PKIX [RFC4387] donne un protocole pour une capacité d'interface à caractère général pour restituer les certificats et CRL des répertoires de PKI. Comme la [RFC4387] est toujours en cours de préparation au moment de la rédaction du présent document, on n'a pas ici les détails de la façon dont ce mécanisme peut être utilisé pour restituer les certificats et les CRL. Se référer plutôt à la [RFC4387] dans sa version actuelle. Les systèmes de traitement de chemin de certification peuvent souhaiter mettre en œuvre la prise en charge de cette capacité d'interface, en particulier si elle va être utilisée dans des environnements qui fournissent un accès fondé sur HTTP aux certificats et aux CRL.

6.3 Accès aux informations d'autorité

L'extension d'accès aux informations d'autorité (AIA, *authority information access*) définie dans la [RFC3280], indique comment accéder aux informations et services de CA pour le producteur du certificat dans lequel l'extension apparaît. Si un

certificat avec une extension AIA contient une accessMethod définie avec l'OID id-ad-caIssuers, l'AIA peut être utilisé pour restituer un ou plusieurs certificats pour la CA qui a produit le certificat contenant l'extension AIA. L'AIA va fournir un identifiant de ressource universel (URI, *uniform resource identifier*) [RFC3986] quand les certificats peuvent être restitués via LDAP, HTTP, ou FTP. L'AIA va fournir un directoryName quand des certificats peuvent être restitués via un protocole d'accès de répertoire (DAP, *directory access protocol*). L'AIA va fournir un rfc822Name quand des certificats peuvent être restitués via la messagerie électronique. De plus, l'AIA peut spécifier la localisation d'un répondeur OCSP [RFC2560] capable de fournir des informations de révocation sur le certificat.

Si elle est présente, l'extension AIA peut fournir aux mises en œuvre de construction de chemin vers l'avant une liaison directe avec un certificat pour le producteur d'un certain certificat. Donc, les mises en œuvre peuvent souhaiter fournir la prise en charge du décodage de l'extension AIA et du traitement des localisateurs LDAP, HTTP, FTP, DAP, ou de messagerie électronique. La prise en charge de AIA est facultative ; les mises en œuvre conformes à la [RFC3280] ne sont pas obligées de remplir l'extension AIA. Cependant, les mises en œuvre et modules de validation de construction de chemin sont vivement encouragées à prendre en charge AIA, en particulier le transport HTTP ; cela assure la possibilité d'utiliser et d'interopérer avec de nombreuses PKI existantes.

6.4 Accès aux informations de sujet

L'extension Accès aux informations de sujet (SIA, *subject information access*) définie dans la [RFC3280], indique comment accéder aux informations et services d'accès pour le sujet du certificat dans lesquels l'extension apparaît. Si un certificat avec une extension SIA contient une accessMethod définie avec l'OID id-ad-caRepository, la SIA peut être utilisée pour localiser un ou plusieurs certificats (et éventuellement des CRL) pour des certificats produits pour des entités par le sujet. La SIA va fournir un identifiant de ressource universel (URI) [RFC3986] lorsque des données peuvent être restituées via LDAP, HTTP, ou FTP. La SIA va fournir un directoryName *(nom de répertoire)* lorsque des données peuvent être restituées via un protocole d'accès de répertoire (DAP). La SIA va fournir un rfc822Name lorsque des données peuvent être restituées via la messagerie électronique.

Si elle est présente, l'extension SIA peut fournir à des mises en œuvre de construction de chemin inverse les certificats requis pour continuer la construction du chemin. Donc, des mises en œuvre peuvent souhaiter fournir la prise en charge du décodage de l'extension SIA et du traitement des localisateurs LDAP, HTTP, FTP, DAP, ou de messagerie électronique. La prise en charge de SIA est facultative ; les mises en œuvre conformes à la [RFC3280] ne sont pas obligées de remplir l'extension SIA. Cependant, les mises en œuvre de modules de construction et de validation de chemin sont vivement encouragées à prendre en charge SIA, en particulier le transport HTTP ; cela assure la possibilité d'utiliser et interopérer avec de nombreuses PKI existantes.

6.5 Points de distribution de CRL

L'extension de points de distribution de CRL (CRLDP, *CRL distribution points*) définie dans la [RFC3280], indique comment accéder aux informations de CRL. Si une extension CRLDP apparaît au sein d'un certificat, la ou les CRL auxquelles se réfère la CRLDP sont généralement les CRL qui vont contenir des informations de révocation pour le certificat. L'extension CRLDP peut pointer sur plusieurs points de distribution à partir desquels les informations de CRL peuvent être obtenues ; le système de traitement de certificats devrait traiter l'extension CRLDP en accord avec la [RFC3280]. Les points de distribution les plus courants contiennent des URI à partir desquels la CRL appropriée peut être téléchargée, et des noms de répertoires, qui peuvent être interrogés pour récupérer les attributs de la CRL dans l'entrée correspondante.

Si elle est présente, CRLDP peut fournir des mises en œuvre de traitement de certificat avec un lien sur les informations de CRL pour un certain certificat. Donc, les mises en œuvre peuvent souhaiter prendre en charge le décodage de l'extension CRLDP et utiliser les informations pour restituer les CRL. La prise en charge de CRLDP est facultative et les mises en œuvre conformes à la [RFC3280] ne sont pas obligées de remplir l'extension CRLDP. Cependant, les mises en œuvre de modules de construction et de validation de chemin sont vivement encouragées à prendre en charge l'extension CRLDP. Au minimum, les développeurs sont encouragés à envisager de prendre en charge les transports LDAP et HTTP ; cela va assurer l'interopérabilité sur une large gamme de PKI existantes.

6.6 Données obtenues via un protocole d'application

De nombreux protocoles d'applications, comme SSL/TLS et S/MIME, permettent à des parties de fournir des certificats et des CRL l'une à l'autre. Les données fournies dans cette méthode sont généralement très précieuses pour le logiciel de construction de chemin (elle fournissent des chemins valides vers l'avant) et devraient être mémorisées et utilisées en conséquence . noter que les certificats auto signés obtenus via un protocole d'application ne sont pas dignes de confiance ; les mises en œuvre ne devraient prendre en compte que les ancres de confiances du consommateur d'assertions lors de la construction de chemins.

6.7 Mécanismes propriétaires

Certains systèmes de production de certificats et systèmes de traitement de certificats peuvent utiliser des mécanismes de restitution propriétaires, comme des pilotes transposés du réseau, des bases de données, ou autres méthodes qui ne sont pas directement référencées via les normes de l'IETF. Les systèmes de traitement de certificats peuvent souhaiter prendre en charge d'autres mécanismes propriétaires, mais ne devraient le faire qu'en plus de la prise en charge des mécanismes de restitution standard comme LDAP, AIA, et CRLDP (sauf à fonctionner dans un environnement clos).

7. Amélioration des performances de restitution

Les performances de restitution peuvent être améliorées par des mécanismes un peu différents, incluant l'utilisation d'antémémoires et en établissant un ordre de restitution spécifique. Cette Section discute quelques méthodes par lesquelles les performances d'un système de traitement de certificats peuvent être améliorées durant la restitution des objets de PKI. Les systèmes de traitement de certificats qui sont de façon persistante très lents durant le traitement n'auront pas la faveur des utilisateurs et mettront du temps à être adoptés dans les organisations. Les systèmes de traitement de certificats sont encouragés à faire leur possible pour réduire les délais associés à la demande et la restitution des données provenant de sources externes.

7.1 Mise en antémémoire

Les systèmes de traitement de certificats qui fonctionnent dans une PKI non hiérarchique vont souvent avoir besoin de récupérer les certificats et les listes de révocation de certificat (CRL) de source extérieure au protocole d'application. Normalement, ces objets sont restitués de répertoires X.500 ou LDAP, un URI Internet [RFC3986], ou quelque autre source non locale. Du fait des délais associés à l'établissement des connexions ainsi que des transferts dans le réseau, les systèmes de traitement de certificats devraient être aussi efficaces que possible lors de la restitution des données provenant de sources externes. Peut-être que la meilleure façon d'améliorer l'efficacité de la restitution est d'utiliser un mécanisme de mise en antémémoire. Les systèmes de traitement de certificats peuvent mettre en antémémoire les données récupérées de sources externes pendant un certain temps, qui ne devrait pas excéder la période d'utilité des données (c'est-à-dire, un certificat arrivé à expiration n'a pas besoin d'être mis en antémémoire). Bien que cela ait un coût en mémoire/consommation de disque par le système, l'avantage en coût et performances de réduire les transmissions réseau est grand. Aussi, les CRL sont souvent produites et disponibles en avance de la date nextUpdate dans la CRL. Les mises en œuvre peuvent souhaiter obtenir ces CRL "plus fraîches" avant que la date nextUpdate soit passée.

Il y a un certain nombre de façons différentes de mettre en œuvre la mise en antémémoire ; les spécificités de ces méthodes peuvent être utilisées comme caractéristiques distinctives entre les systèmes de traitement de certificats. Cependant, les mises en œuvre peuvent souhaiter prendre en compte les éléments suivants lors du développement de systèmes de mise en antémémoire :

- Si des objets de PKI sont mis en antémémoire, le mécanisme de construction de chemin de certification devrait être capable de les examiner et restituer à partir de l'antémémoire durant la construction de chemin. Cela va permettre au système de traitement de certificats de trouver ou éliminer rapidement un ou plusieurs chemins sans exiger un contact externe avec un répertoire ou autre mécanisme de restitution.

- Partager des antémémoires entre plusieurs usagers (via un réseau de zone locale ou LAN) peut être utile si de nombreux utilisateurs dans une organisation effectuent de façon soutenue des opérations de PKI avec une autre organisation.

- Mettre en antémémoire non seulement des objets de PKI (comme des certificats et des CRL) mais aussi des relations entre des objets de PKI (mémoriser un lien entre un certificat et le certificat du producteur) peut être utile. Ces liaisons peuvent ne pas toujours conduire à la relation la plus correcte ou meilleure, mais pourraient représenter un lien qui a fonctionné dans un autre scénario.

- Il est assez utile de mettre en antémémoire les chemins et chemins partiels construits précédemment, parce qu'ils vont fournir des informations sur les réussites et échecs précédents. De plus, si l'antémémoire est protégée contre les modifications non autorisées, la mise en antémémoire de l'état de vérification de validation et de signature des certificats, CRL, et chemins peut aussi être mémorisée.

7.2 Ordre de restitution

Pour optimiser l'efficacité, les systèmes de traitement de certificats sont encouragés à aussi considérer l'ordre dans lequel

les différents objets de PKI sont restitués, ainsi que le mécanisme par lequel ils le sont. Si la mise en antémémoire est utilisée, les antémémoires peuvent être consultées pour les objets de PKI avant de tenter d'autres mécanismes de restitution. Si plusieurs antémémoires sont présentes (comme un disque local et le réseau) les antémémoires peuvent être consultées dans l'ordre dans lequel on peut supposer qu'elles vont retourner leur résultat du plus rapide au plus lent. Par exemple, si un système de traitement de certificats souhaite restituer un certificat avec un DN sujet particulier, le système pourrait d'abord consulter l'antémémoire locale, puis l'antémémoire réseau, puis tenter la restitution d'un répertoire. Les spécificités des types de mécanismes de restitution et leurs coûts relatifs sont l'affaire des mises en œuvre.

En plus des mécanismes de restitution d'ordre, le système de traitement de certificats devrait ordonner les mérites relatifs des différentes sources externes à partir desquelles un objet de PKI peut être restitué. Si l'extension AIA est présente au sein d'un certificat, avec un URI [RFC3986] pour le certificat du producteur, le système de traitement de certificats (si il en est capable) peut souhaiter tenter de restituer d'abord le certificat provenant de l'antémémoire locale et d'ensuite utiliser cet URI (parce qu'il est supposé pointer directement sur le certificat désiré) avant de tenter de restituer les certificats qui peuvent exister dans un répertoire.

Si un répertoire est consulté, il peut être désirable de restituer les attributs dans un ordre particulier. Une structure de PKI très inter certifiée va conduire à de multiples possibilités pour les chemins de certification, ce qui peut signifier de multiples tentatives de validation avent qu'on réussisse à restituer un chemin. Donc, cACertificate et userCertificate (qui contiennent normalement les certificats provenant du même "domaine") pourraient être consultés avant de tenter de restituer les valeurs de crossCertificatePair pour une entrée. Autrement, les trois attributs pourraient tous être restitués en une interrogation, mais les certificats croisés ont été étiquetés à ce titre et seront utilisés seulement après l'épuisement des possibilités provenant de l'attribut cACertificate. La meilleure approche va dépendre de la nature de l'application et de l'environnement de PKI.

7.3 Restitution en parallèle et pré restitution

Une grande partie du présent document se concentre sur un algorithme de construction de chemin qui minimise l'impact des performances de restitution sur le réseau, en empêchant ces restitutions et l'utilisation d'antémémoires. Une autre façon d'améliorer les performances serait de permettre que les restitutions à partir du réseau soient effectuées à l'avance (pré allocations) au moment où d'autres opérations sont effectués (restitution en parallèle). Par exemple, si une application de messagerie électronique reçoit un message électronique signé, elle pourrait télécharger les certificats et CRL requis avant que le receveur voit (ou tente de vérifier) le message. Les mises en œuvre qui fournissent la capacité de restitution et/ou pré allocation en parallèle avec une antémémoire robuste, peuvent conduire à des performances ou un ressenti de l'usager largement améliorés.

8. Considérations sur la sécurité

8.1 Considérations générales pour la construction d'un chemin de certification

Bien que la construction de chemin de certification traite directement des données de PKI qui relèvent de la sécurité, les données de PKI elles-mêmes n'ont pas besoin d'un traitement particulier parce que leur intégrité est protégée par la signature numérique qui leur est appliquée. La seule exception à cela est la protection appropriée des clés publiques de l'ancre de confiance. Celles-ci sont conservées en sûreté et obtenues hors bande (par exemple, pas d'un message électronique ou d'un répertoire) par rapport au module de construction de chemin.

Les plus grands risques de sécurité associés au présent document tournent autour de la réalisation de la validation du chemin de certification pendant la construction du chemins de certification. Il est donc noté ici que la validation pleinement mise en œuvre de chemin de certification conformément à la [RFC3280] et à [X.509] est requise afin que la construction de chemin de certification, la validation de chemin de certification, et du certificat qui utilisent l'application soient sécurisées de façon appropriée. Toutes les considérations pour la sécurité mentionnées à la Section 9 de la [RFC3280] s'appliquent également ici.

De plus, comme avec toute application qui consomme des données provenant de localisations réseau potentiellement non fiables, les composants de construction de chemin de certification devraient être mis en œuvre avec précaution afin de réduire ou éliminer la possibilité d'exploitations malveillantes fondées sur le réseau. Par exemple, une mise en œuvre insuffisante de module de construction de chemin peut ne pas vérifier la longueur de l'URI de CRLDP [RFC3986] avant d'utiliser la fonction strcpy() du langage C pour placer l'adresse dans une mémoire tampon de 1024 octets. Un pirate pourrait utiliser une telle faute pour créer une attaque de débordement de mémoire tampon en codant un code d'assemblage malveillant dans le CRLDP d'un certificat et ensuite utiliser le certificat pour tenter une authentification. Une telle attaque pourrait donner le contrôle au niveau système à l'attaquant et exposer les données sensibles que la PKI était censée protéger.

La construction de chemin peut être utilisée pour monter une attaque de déni de service (DOS). Cela peut se produire si plusieurs demandes simples pouvaient être effectuées pour causer l'exécution par un serveur d'un certain nombre de développements de chemins, chacun consommant du temps et des ressources du serveur. Les serveurs peuvent aider à éviter cela en limitant les ressources qu'ils acceptent de dédier à la construction de chemin, et en étant capables de limiter encore plus ces ressources lorsque la charge est lourde. Les protections standard contre le DOS comme les systèmes qui identifient et bloquent les attaquants peuvent aussi être utiles.

Une attaque de DOS peut être aussi créée en présentant des certificats de CA parasites contenant de très grandes clés publiques. Lorsque le système tente d'utiliser la grande clé publique pour vérifier la signature numérique sur des certificats supplémentaires, un long délai de traitement peut survenir. Cela peut être atténué par une de ces deux stratégies : effectuer des vérifications de signature seulement après qu'un chemin complet est construit, en commençant par l'ancre de confiance ; cela élimine le certificat de CA parasite de la prise en considération avant que la grande clé publique soit utilisée. La seconde stratégie est de reconnaître et simplement rejeter les clés plus longues qu'une certaine taille.

Une attaque de DOS similaire peut se produire avec des très grandes clés publiques dans les certificats d'entité d'extrémité. Si un système utilise la clé publique dans un certificat avant la construction et la validation du chemin de certification de ce certificat, de longs délais de traitement peuvent survenir. Pour atténuer cette menace, la clé publique dans un certificat d'entité d'extrémité ne devrait être utilisée sous aucun prétexte avant qu'un chemin de certification complet soit construit et validé pour ce certificat.

8.2 Considérations spécifiques pour la construction de chemins de certification de signataire de révocation

Si le certificat de signataire de CRL (et le chemin de certification) n'est pas identique au certificat d'autorité de certification (et au chemin de certification) une attention particulière devrait être apportée lors de la construction du chemin de certification de signataire de CRL.

Si une attention particulière n'est pas apportée à la construction d'un chemin de certification de signataire de CRL, ce chemin pourrait être construit de façon telle qu'il se termine par une racine différente ou par un chemin de certification différent à la même racine. Si ce comportement n'est pas empêché, le consommateur d'assertions peut se trouver à vérifier les mauvaises données de révocation, ou même de données substituées dans une intention malveillante, résultant en déni de service ou une brèche de la sécurité.

Par exemple, supposons que le chemin de certification suivant soit construit pour E et soit valide pour une politique de "forte assurance" . A->B->C->E.

Lorsque le sous programme de construction/validation tente de vérifier que E n'est pas révoqué, C se réfère au certificat d'autorité de certification. Le constructeur de chemin trouve que la CRL pour vérifier l'état de révocation de E est produite par C2 ; un certificat avec le nom de sujet "C", mais avec une clé différente de celle qui a été utilisée pour signer E. C2 est un signataire de CRL. Un constructeur de chemin de certification sans restriction peut alors construire un chemin comme le suivant pour le certificat C2 de signataire de CRL : X->Y->Z->C2.

Si un chemin tel que celui-ci est permis, rien ne peut être conclu sur l'état de révocation de E car C2 est une CA différente de C.

Heureusement, empêcher ce problème de sécurité n'est pas difficile et la solution rend aussi très efficace la construction de chemins de certification de signataire de CRL. Dans le cas où le certificat de signataire de CRL est identique au certificat d'autorité de certification, le chemin de certification de l'autorité de certification devrait être utilisé pour vérifier la CRL ; aucune construction de chemin supplémentaire n'est requise. Si le certificat de signataire de CRL n'est pas identique au certificat d'autorité de certification, un second chemin devrait être construit pour le certificat de signataire de CRL exactement de la même façon que pour tout certificat, mais avec les directives supplémentaires suivantes :

1. Ancre de confiance : Le chemin de certification du signataire de CRL devrait commencer par la même ancre de confiance que le chemin de certification de l'autorité de certification. Tout certificat d'ancre de confiance avec un DN sujet qui correspond à celui de l'ancre de confiance de l'autorité de certification devrait être considéré comme acceptable bien que de priorité inférieure à celui d'une clé publique et DN sujet qui correspondent. Bien que des clés publiques d'ancre de confiance différentes soient acceptables au début du chemin de certification du signataire de CRL et du chemin de certification de l'autorité de certification, les deux clés doivent être de confiance pour le consommateur d'assertions selon les recommandations du paragraphe 8.1.

2. Correspondance de nom de CA : Les DN sujets pour tous les certificats de CA dans les deux chemins de certification devraient correspondre un à un (en ignorant les certificats auto produits) pour toute la longueur du plus court des deux chemins.

3. Longueur de chemin de certification de signataire de CRL : La longueur du chemin de certification de signataire de CRL (en ignorant les certificats auto produits) devrait être égale ou inférieure à la longueur du chemin de certification de l'autorité de certification plus (+) un. Cela permet à une certaine autorité de certification de produire un certificat à un signataire de CRL délégué/subordonné. Cette dernière configuration représente la longueur maximum de chemin de certification pour un certificat de signataire de CRL.

Le raisonnement derrière la première directive est directement apparent. Y manquer ainsi qu'à la seconde directive, ferait que toute CA de confiance pourrait produire des CRL pour toute autre CA, même si les PKI sont sans relation d'aucune sorte. Par exemple, une société pourrait révoquer des certificats produits par une autre société si le consommateur d'assertions fait confiance à l'ancre de confiance des deux sociétés. Les deux directives empêchent aussi des vérifications de CRL erronées car l'unicité mondiale des noms n'est pas garantie.

La seconde directive empêche les chemins de certification en itinérance comme celui décrit précédemment dans l'exemple de chemin de certification de signataire de CRL pour A->B->C->E. Il est particulièrement important que le "ignorant les certificats auto produits" soit mis en œuvre correctement. Les certificats auto produits déjouent la comparaison de nom afin de permettre le retournement de clé. L'algorithme de construction de chemin peut être optimisé pour ne considérer que les certificats avec le DN sujet acceptable pour le point considéré dans le chemin de certification de signataire de CRL lors de la construction le chemin.

La troisième et dernière directive assure que la CRL utilisée est celle prévue. Sans une restriction sur la longueur du chemin de certification du signataire de CRL, le chemin pourrait divaguer sans contrôle dans un autre domaine et satisfaire encore les deux premières directives. Par exemple, en utilisant encore le chemin A->B->C->E, l'autorité de certification C, et un signataire de CRL C2, un chemin de certification de signataire de CRL tel que le suivant pourrait satisfaire aux deux premières directives : A->B->C->D->X->Y->CAfélon->C2

Dans l'exemple précédent, l'ancre de confiance est identique pour les deux chemins et la confrontation de noms un à un réussit pour A->B->C. Cependant, accepter un tel chemin a des conséquences de sécurité évidentes, de sorte que la troisième directive est utilisée pour empêcher cette situation. En appliquant les secondes et troisièmes directives au chemin de certification ci-dessus, le constructeur de chemin pourrait avoir immédiatement détecté que ce chemin n'est pas acceptable (avant de le construire) en examinant le DN de producteur dans C2. Étant données les directives de longueur et de nom, le constructeur de chemin pourrait détecter que "CAfélon" n'est pas dans l'ensemble de noms possibles en le comparant à l'ensemble des DN de producteurs possibles de signataire de CRL, spécifiquement, A, B, ou C.

Des considérations similaires devraient être faites lors de la construction du chemin pour le certificat de répondeur OCSP lorsque le CA est le signataire de réponse OCSP ou que la CA a délégué la signature de la réponse OCSP à une autre entité.

9. Remerciements

Les auteurs reconnaissent leur dette à l'égard de David Lemire pour ses efforts dans la rédaction de l'ouvrage "Gérer l'interopérabilité dans une infrastructure de clé publique non hiérarchique" *(Managing Interoperability innon.n-Hierarchical Public Key Infrastructures)* auquel les paragraphes introductifs du présent ouvrage ont beaucoup emprunté.

Le présent document a aussi beaucoup bénéficié de la relecture et des corrections techniques fournies par le Dr. Santosh Chokhani, Carl Wallace, Denis Pinkas, Steve Hanna, Alice Sturgeon, Russ Housley, et Tim Polk.

10. Références normatives

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

11. Références pour information

[MINHPKIS] Hesse, P., and D. Lemire, "Managing Interoperability in non-Hierarchical Public Key Infrastructures", Conference Proceedings of the Internet Society Network and Distributed System Security Symposium, février 2002.

[RFC1777] W. Yeong, T. Howes, S. Kille, "Protocole léger d'accès de répertoire", mars 1995. *(Obsolète, voir* RFC3494) *(Historique)*

[RFC2560] M. Myers, R. Ankney, A. Malpani, S. Galperin et C. Adams, "Protocole d'état de certificat en ligne d'infrastructure de clé publique X.509 pour l'Internet - OCSP", juin 1999. *(P.S.) (Remplacée par* RFC6960)

[RFC2587] S. Boeyen, T. Howes, P. Richard, "Schéma LDAPv2 d'infrastructure de clé publique X.509 pour l'Internet", juin 1999. (*Obsolète, voir* RFC4523) *(P.S.)*

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

[RFC3377] J. Hodges, R. Morgan, "Protocole léger d'accès à un répertoire (v3) : Spécification technique", septembre 2002. *Obsolète, voir* RFC4510) *(P.S.)*

[RFC3820] S. Tuecke et autres, "Profil de certificat de mandataire d'infrastructure de clé publique X.509 (PKI) pour l'Internet", juin 2004. *(P.S.)*

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

[RFC4387] P. Gutmann, éd., "Protocoles de fonctionnement d'infrastructure de clé publique X.509 pour l'Internet : Accès aux mémoires de certificats via HTTP", février 2006. *(P.S.)*

[X.501] Recommandation UIT-T X.501, "Technologies de l'information - Interconnexion des systèmes ouverts - : Modèles de répertoire", 1993.

[X.509] Recommandation UIT-T X.509 (2000 E), "Technologies de l'information - Interconnexion des systèmes ouverts – L'annuaire : cadre d'authentification ", mars 2000.

Adresse des auteurs

Matt Cooper
Orion Security Solutions, Inc.
1489 Chain Bridge Rd, Ste. 300
McLean, VA 22101, USA
téléphone : +1-703-917-0060
mél : mcooper@orionsec.com

Yuriy Dzambasow
A&N Associates, Inc.
999 Corporate Blvd Ste. 100
Linthicum, MD 21090, USA
téléphone : +1-410-859-5449 x107
mél : yuriy@anassoc.com

Peter Hesse
Gemini Security Solutions, Inc.
4451 Brookfield Corporate Dr. Ste. 200
Chantilly, VA 20151, USA
téléphone : +1-703-378-5808 x105
mél : pmhesse@geminisecurity.com

Susan Joseph
Van Dyke Technologies
6716 Alexander Bell Drive
Columbia, MD 21046
mél : susan.joseph@vdtg.com

Richard Nicholas
BAE Systems Information Technology
141 National Business Parkway, Ste. 210
Annapolis Junction, MD 20701, USA
téléphone : +1-301-939-2722
mél : richard.nicholas@it.baesystems.com

Déclaration complète de droits de reproduction

Propriété intellectuelle

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

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

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

Remerciement

Le financement de la fonction d'édition des RFC est fourni par la Administrative Support Activity (IASA) de l'IETF.