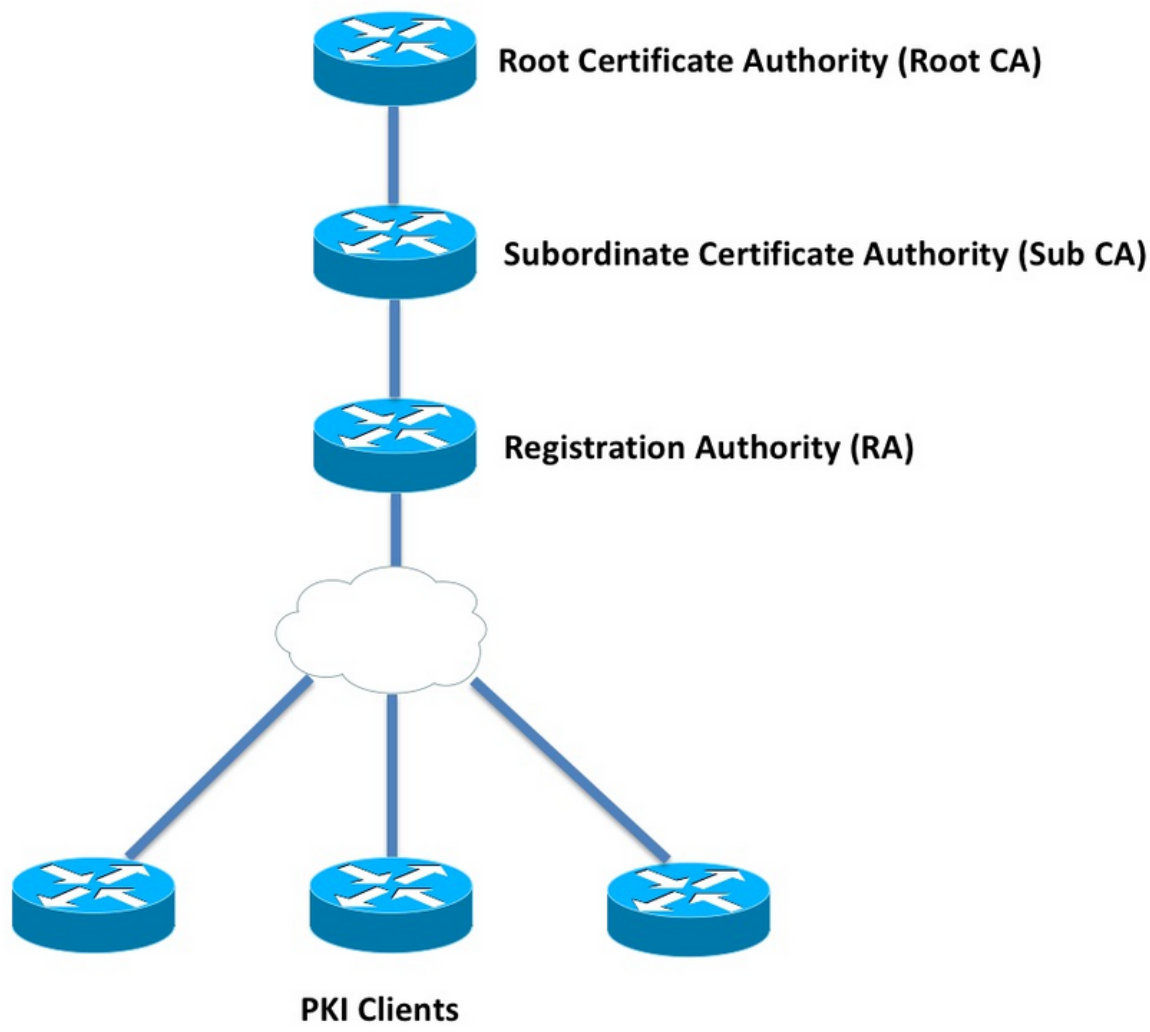
Guide de déploiement de PKI IOS : Conception initiale et déploiement

Contenu

Introduction

Ce document décrit des fonctionnalités de serveur et de client de PKI IOS en détail. Il adresse des considérations de conception et de déploiement d'initiale de PKI IOS.

Infrastructure de PKI



Autorité de certification

L'Autorité de certification (CA), également désigné sous le nom du serveur de PKI dans tout le

document, est une entité de confiance cette des Certificats de questions. Le PKI est basé sur la confiance, et les débuts de confiance-hiérarchie à l'autorité de certification de racine (Racine-CA). Puisque le Racine-CA est en haut de la hiérarchie, il a un certificat auto-signé.

Autorité de certification subalterne

Dans la Confiance-hiérarchie de PKI toute la racine ci-dessous d'autorités de certification sont connues en tant qu'autorités de certification subalternes (Sous-titre-CA). Évidemment, un certificat Sous-titre-CA est délivré par le CA, qui est un niveau ci-dessus.

Le PKI n'impose aucune limite au nombre de Sous-titre-CAS dans une hiérarchie donnée. Cependant, dans un déploiement en entreprise avec plus de 3 niveaux des autorités de certification il peut devenir difficile gérer.

Autorité d'enregistrement

Le PKI définit une autorité de certification spéciale connue sous le nom d'autorité d'enregistrement (RA), qui est responsable d'autoriser les clients de PKI de l'inscription à un Sous-titre-CA donné ou au Racine-CA. Le RA ne fournit pas des Certificats aux clients de PKI, au lieu de cela il décide quel PKI-client peut ou ne peut pas être émis un certificat par le Sous-titre-CA ou le Racine-CA.

Le rôle principal d'un RA est de débarquer la validation de base de demande de certificat client du CA, et protège le CA contre l'exposition directe aux clients. De cette façon, RA se tient entre les clients de PKI et le CA, de ce fait protégeant le CA contre n'importe quel genre d'attaques par déni de service.

Client de PKI

N'importe quel périphérique demandant pour un certificat basé sur une paire de clés entre le secteur public et le secteur privé résidente pour prouver son identité à d'autres périphériques est connu en tant que client de PKI.

Un client de PKI doit être capable de générer ou d'enregistrer une paire de clés entre le secteur public et le secteur privé telle que la RSA ou le DSA ou l'ECDSA.

Un certificat est une preuve d'identité et de validité d'une clé publique donnée, si la privé-clé correspondante existe sur le périphérique.

Serveur de PKI IOS

Évolution de caractéristique de serveur de PKI IOS du tableau 1.

Caractéristique	IOS [ISR-G1, ISR-G2]	IOS-XE [ASR1K, ISR4K]
Serveur IOS CA/PKI	12.3(4)T	XE 3.14.0/15.5(1)S
Renversement de certificat de serveur de PKI IOS	12.4(1)T	XE 3.14.0/15.5(1)S
PKI HA IOS	15.0(1)M	NA [la Redondance Inter-RP implicite est disponible]

RA IOS pour le tiers CA	15.1(3)T	XE 3.14.0/15.5(1)S

Avant d'obtenir dans la configuration du serveur de PKI, l'administrateur doit comprendre ces principaux concepts.

Source bien fondée de temps

Une des bases de l'infrastructure de PKI est temps. L'horloge système définit, qu'un certificat soit valide ou pas. Par conséquent, dans l'IOS, l'horloge doit être rendue bien fondée ou digne de confiance. Sans source bien fondée de temps, le serveur de PKI peut ne pas fonctionner comme prévu, et elle est fortement recommandée de rendre l'horloge sur l'IOS bien fondée suivre ces méthodes :

NTP (Network Time Protocol)

Synchroniser l'horloge système avec un Serveur de synchronisation est la seule manière vraie de rendre l'horloge système digne de confiance. Un routeur IOS peut être configuré en tant que client de NTP à un serveur réputé et stable de NTP dans le réseau :

```
configure terminal
ntp server <NTP Server IP address>
ntp source <source interface name>
ntp update-calendar

!! optional, if the NTP Server requires the clients to authenticate themselves
ntp authenticate
ntp authentication-key 1 md5 <key>

!! optionally an access-list can be configured to restrict time-updates from a specifc NTP
server
access-list 1 permit <NTP Server IP address>
ntp access-group peer 1
```

L'IOS peut également être configuré en tant que serveur de NTP, qui marquera l'horloge de système local comme bien fondée. Dans le déploiement à petite échelle de PKI, le serveur de PKI peut être configuré en tant que serveur de NTP pour ses clients de PKI :

```
configure terminal
ntp master <stratum-number>

!! optionally, NTP authentication can be enforced
ntp authenticate
ntp authentication-key 1 md5 <key-1>
ntp authentication-key 2 md5 <key-2>
ntp authentication-key 2 md5 <key-2>
ntp trusted-key 1 - 3

!! optinally, an access-list can be configured to restrict NTP clients
!! first allow the local router to synchronize with the local time-server
access-list 1 permit 127.127.7.1
ntp access-group peer 1

!! define an access-list to which the local time-server will serve time-synchronization services
access-list 2 permit <NTP-Client-IP>
```

```
ntp access-group serve-only 2
```

Horloge de repérage de matériel comme fait confiance

Dans l'IOS, l'horloge de matériel peut être marquée en tant qu'utilisation bien fondée :

```
config terminal
clock calendar-valid
```

Ceci peut être configuré avec le NTP, et la raison principale pour faire ceci est de maintenir l'horloge système bien fondée quand les routeurs rechargés, par exemple en raison d'une panne de courant, et des serveurs de NTP ne sont pas accessibles. À ce stade, les temporisateurs de PKI cesseront le fonctionnement, qui mène consécutivement pour délivrer un certificat le renouvellement/pannes inversées. **le clock calendar-valid** agit en tant que sauvegarde dans de telles situations.

Tout en configurant ceci, il est principal pour comprendre que l'horloge système sortira du sync si la batterie de système meurt, et le PKI commencera faire confiance à une horloge de -de-sync. Cependant, il est relativement plus sûr de configurer ceci, que n'ayant pas une source bien fondée de temps du tout.

> Remarque: la commande de **clock calendar-valid** a été ajoutée dans la version XE 3.10.0/15.3(3)S IOS-XE en avant.

Adresse Internet et nom de domaine

Il est recommandé pour configurer une adresse Internet et un domain-name sur le Cisco IOS en tant qu'une des premières étapes avant de configurer tous les services connexes de PKI. Le nom de hôte du routeur et le domain-name sont utilisés dans les scénarios suivants :

- Le nom par défaut de paire de clés RSA est dérivé en combinant l'adresse Internet et le domain-name
- En s'inscrivant pour un certificat, le subject-name par défaut se compose de l'attribut d'adresse Internet et d'un non structuré-nom, qui est adresse Internet et domain-name remontés.

Quant au serveur de PKI, l'adresse Internet et le domain-name ne sont pas utilisés :

- Le nom par défaut de paire de clés sera identique que celui du nom du serveur de PKI
- Le subject-name par défaut se compose de la NC, qui est identique que celle du nom du serveur de PKI.

La recommandation générale est de configurer une adresse Internet appropriée et un domain-name.

```
config terminal
hostname <string>
ip domain name <domain>
```

Serveur HTTP

Le serveur de PKI IOS est activé seulement si le serveur HTTP est activé. Il est important de noter que, si le serveur de PKI est dû désactivé au serveur HTTP étant désactivé, il peut continuer à accorder des demandes hors ligne [par l'intermédiaire du terminal]. La capacité de serveur HTTP est exigée pour traiter des demandes SCEP, et envoie des réponses SCEP.

Le serveur HTTP IOS est activé utilisant :

```
ip http server
```

Et le port de serveur HTTP par défaut peut être changé de 80 à n'importe quel numéro de port valide utilisant :

```
ip http port 8080
```

Max-connection de HTTP

Un des étranglements, tout en déployant l'IOS comme serveur de PKI utilisant SCEP est les connexions HTTP simultanées maximum et les connexions HTTP moyennes par minute. Actuellement, les connexions simultanées maximum sur un serveur HTTP IOS est limitées à 5 par défaut et peut être grimpées jusqu'à 16, qui est fortement recommandé dans un déploiement d'échelle moyenne :

```
ip http max-connections 16
```

Les installations cet IOS permettent les connexions HTTP simultanées maximum jusqu'à 1000 :

- IOS UniversalK9 avec le permis-positionnement uck9

Le CLI est automatiquement changé pour recevoir un argument numérique entre 1 à 1000

```
ip http max-connections 1000
```

Le serveur HTTP IOS permet 80 connexions par minute [580 dans le cas des releases IOS où des sessions simultanées maximum de HTTP peuvent être grimpées jusqu'à 1000] et quand cette limite est atteinte dans une minute, auditeur de HTTP IOS commence étrangler les connexions HTTP entrantes par arrêter l'auditeur pendant 15 secondes. Ceci mène aux demandes de connexion client étant dû relâché à la **limite de file d'attente de** connexion TCP **atteinte**. Plus d'informations sur ceci peuvent être trouvées ici

Paire de clés RSA

La paire de clés RSA pour la fonctionnalité de serveur de PKI sur l'IOS peut automatique-être générée ou manuellement générée.
Tout en configurant un serveur de PKI, l'IOS crée automatiquement un point de confiance par le même nom que le serveur de PKI afin d'enregistrer le certificat de serveur de PKI.

Générer manuellement la paire de clés RSA de serveur de PKI :

Étape 1. Créez une paire de clés RSA avec le même nom que celle du serveur de PKI :

```
crypto key generate rsa general-keys label <LABEL> modulus 2048
```

Étape 2. Avant d'activer le serveur de PKI, modifiez le point de confiance de serveur de PKI :

```
crypto pki trustpoint <PKI-SERVER-Name>
  rsakeypair <LABEL>
```

> Remarque: La valeur de module de paire de clés RSA mentionnée sous le point de confiance de serveur de PKI n'est pas prise en compte jusqu'au ver 15.4(3)M4 IOS, et c'est une mise en garde connue. Le module de clé par défaut est 1024 bits.

Automatique-générer la paire de clés RSA de serveur de PKI :

En activant le serveur de PKI, l'IOS génère automatiquement une paire de clés RSA avec le même nom que ce du serveur de PKI, et la taille principale de module est 1024 bits.

Commençant le ver 15.4(3)M5 IOS, cette configuration crée une paire de clés RSA avec <LABEL> car le nom et la puissance de la clé seront selon le module défini <MOD>.

```
crypto pki trustpoint <PKI-SERVER-Name>
rsakeypair <LABEL> <MOD>
```

Spoiler

Le serveur de PKI IOS CSCuu73408 devrait tenir compte de la taille de clé de non-par défaut pour le CERT inversé.

Le serveur de PKI IOS CSCuu73408 devrait tenir compte de la taille de clé de non-par défaut pour le CERT inversé.

Industriellement compatible en cours est d'utiliser un minimum de paire de clés RSA de 2048 bits.

Considération de temporisateur d'auto-rollover

Actuellement, le serveur de PKI IOS ne génère pas un certificat inversé par défaut, et il doit être explicitement activé sous le serveur de PKI utilisant la commande de **<days-before-expiry> d'auto-rollover**. Plus sur le renversement de certificat est expliqué dedans

Cette commande spécifie combien de jours avant que l'échéance de certificat du PKI Server/CA si l'IOS crée un certificat de CA inversé. Notez que le certificat de CA inversé est lancé une fois le certificat de CA actif en cours expire. La valeur par défaut est actuellement de 30 jours. Cette valeur devrait être placée à une valeur raisonnable selon la vie de certificat de CA, et ceci influence consécutivement la configuration de temporisateur d'auto-enroll sur le client de PKI.

> Remarque: Le temporisateur d'auto-rollover devrait toujours déclencher avant le temporisateur d'auto-enroll sur le client pendant le CA et le renversement de certificat client [connus sous le nom de]

Considérations CRL

L'infrastructure de PKI IOS prend en charge deux manières de distribuer CRL :

Éditez le CRL à un serveur HTTP

Le serveur de PKI IOS peut être configuré pour éditer le fichier CRL à un emplacement spécifique sur un serveur HTTP utilisant cette commande sous le serveur de PKI :

```
crypto pki server <PKI-SERVER-Name>
 database crl publish <URL>
```

Et le serveur de PKI peut être configuré pour inclure cet emplacement CRL dans tous les certificats client de PKI utilisant cette commande sous le serveur de PKI :

```
crypto pki server <PKI-SERVER-Name>
cdp-url <CRL file location>
```

Méthode SCEP GetCRL

Le serveur de PKI IOS enregistre automatiquement le fichier CRL à l'emplacement spécifique de base de données, qui par défaut est nvram, et est fortement recommandé de garder une copie sur un serveur SCP/FTP/TFTP utilisant cette commande sous le serveur de PKI :

```
crypto pki server <PKI-SERVER-Name>
database url <URL>
or
database crl <URL>
```

Par défaut, le serveur de PKI IOS n'inclut pas l'emplacement de CDP dans les certificats client de PKI. Si les clients de PKI IOS sont configurés pour exécuter le contrôle de révocation, mais le certificat étant validé n'a pas un CDP encastré dans lui, et le point de confiance de validation CA est configuré avec l'emplacement CA (utilisant le <CA-Serveur-IP ou le FQDN> de http://), des chutes IOS de nouveau à la méthode de GetCRL basée par SCEP par défaut.
SCEP GetCRL exécute la récupération CRL en exécutant le HTTP OBTIENT sur cet URL :

```
http:://<CA-Server-IP/FQDN>/cgi-bin/pkiclient.exe?operation=GetCRL
```

> Remarque: Dans IOS CLI, avant d'entrer **?** , presse clé-ordre **CTRL + V**.

Le serveur de PKI IOS peut également encastrer cet URL comme emplacement de CDP. L'avantage de faire ceci est double :

- Il s'assure que tous les clients du PKI NON-IOS basés par SCEP peuvent exécuter la récupération CRL.
- Sans CDP encastré, des messages de demande IOS SCEP GetCRL sont signés (utilisant un certificat auto-signé provisoire) comme définis dans l'ébauche SCEP. Cependant, des demandes de récupération CRL n'ont pas besoin d'être signées, et en encastrant l'URL de CDP pour la méthode de GetCRL, la signature des demandes CRL peut être évitée.

Vie de CRL

La vie CRL du serveur de PKI IOS peut être commandée utilisant cette commande sous le serveur de PKI :

```
crypto pki server <PKI-SERVER-Name>
 lifetime crl <0 - 360>
```

La valeur a lieu en quelques heures. Par défaut la vie du CRL est placée à 6 heures. Selon la façon dont fréquemment les Certificats sont retirés, la vie de accord CRL à une valeur optimale augmente la représentation de récupération CRL dans le réseau.

Considérations de base de données

Le serveur de PKI IOS utilise le nvram comme emplacement par défaut de base de données, et il est fortement recommandé d'utiliser un serveur de FTP ou TFTP ou SCP comme emplacement de base de données. Par défaut, le serveur de PKI IOS crée deux fichiers :

- <Server-Name>.ser – Ceci contient le dernier numéro de série émis par le CA dans l'hexa. Le fichier est dans le format de texte brut, et il contient ces informations :
  db_version = 1
  last_serial = 0x4

- <Server-Name>.crl – C'est le fichier CRL encodé par DER édité par le CA

Le serveur de PKI IOS stocke les informations dans la base de données à 3 niveaux configurables :

- Minimum – C'est le niveau par défaut, et à ce niveau aucun fichier n'est créé dans la base de données, et par conséquent aucune informations n'est disponible sur le serveur CA concernant les certificats client accordée dans le passé.
- Noms – À ce niveau le serveur de PKI IOS crée un fichier nommé <Serial-Number>.cnm pour chaque certificat client délivré, où le <Serial-Number> de nom se rapporte au numéro de série du certificat client délivré et ce fichier de cnm contient le subject-name et l'expiration date du certificat client.
- Complet – À ce niveau, le serveur de PKI IOS crée deux fichiers pour chaque certificat client délivré :
- <Serial-Number>.cnm
- <Serial-Number>.crt

ici, le fichier tube cathodique est le fichier de certificat client, qui est DER encodé.

Ces points sont importants :

- Avant de délivrer un certificat client, le serveur de PKI IOS se réfère à <Server-Name>.ser pour déterminer et dériver le numéro de série du certificat.
- Avec le positionnement de database level aux noms ou le besoin terminez-vous, <Serial-Number>.cnm et <Serial-Number>.crt d'être écrit à la base de données avant d'envoyer certificat accordé/délivré au client

- Avec le positionnement de database url aux noms ou terminez-vous, le database url doit avoir assez d'espace pour sauvegarder les fichiers. Par conséquent la recommandation est de configurer un serveur de fichiers externe [FTP ou TFTP ou SCP] comme database url.
- L'URL de base de données externe étant configuré, il est absolument nécessaire de s'assurer que le serveur de fichiers est accessible pendant le processus de concession de certificat, qui autrement marquerait le serveur CA comme handicapés. Et l'intervention manuelle est exigée pour apporter le dos de serveur CA en ligne.

Database archive

Tout en déployant un serveur de PKI, il est important de considérer les scénarios de panne et être préparé, devrait il y a une panne de hadrware. Il y a deux manières de réaliser ceci :

1. Redondance
   Dans ce cas, deux périphériques ou unités de traitement agissent en tant qu'Actif-standby pour fournir la Redondance.
   Le serveur de PKI IOS facilement disponible peut être réalisé utilisant deux routeurs ISR activés par HSRP [ISR G1 et ISR G2] comme expliqué dedans

   IOS XE a basé les systèmes [ISR4K et ASR1k] n'ont pas l'option de périphérique-Redondance disponible. Cependant, dans ASR1k Inter-RP la Redondance est disponible par défaut.
2. Archivage de la paire de clés et des fichiers de serveur CA
   L'IOS fournit une installation pour archiver la paire de clés de serveur de PKI et le certificat. L'archivage peut être fait utilisant deux types de fichiers :
   PEM - L'IOS crée les fichiers formatés par PEM pour enregistrer la clé publique RSA, clé privée chiffrée RSA, certificat de serveur CA. La paire de clés inversée et les Certificats sont automatiquement archivésPKCS12 - L'IOS crée un fichier PKCS12 simple contenant le certificat de serveur CA et la clé privée correspondante RSA chiffrés utilisant un mot de passe.

L'archivage de base de données peut être activé utilisant cette commande sous le serveur de PKI :

```
crypto pki server <PKI-SERVER-Name>
 database archive {pkcs12 | pem} password <password>
```

Il est également possible d'enregistrer les fichiers d'archivage à un serveur distinct, probablement utilisant un protocole sécurisé (SCP) utilisant la commande suivante sous le serveur de PKI :

```
crypto pki server <PKI-SERVER-Name>
 database url {p12 | pem} <URL>
```

De tous les fichiers dans la base de données excepté les fichiers d'archivage et. Le fichier de Ser, tous autres fichiers sont en texte clair et ne constituent aucune vraie menace si perdu, et par conséquent peuvent être enregistrés sur un serveur distinct sans encourir beaucoup de temps système tout en écrivant les fichiers, par exemple un serveur TFTP.

IOS comme Sous-titre-CA

Le serveur de PKI IOS par défaut prend le rôle d'une racine CA. Pour configurer un serveur subalterne de PKI (Sous-titre-CA), activez d'abord cette commande sous la section de configuration du serveur de PKI (avant d'activer le serveur de PKI) :

```
crypto pki server <Sub-PKI-SERVER-Name>
 mode sub-cs
```

Utilisant ceci configurez l'URL du Racine-Ca sous le point de confiance du serveur de PKI :

```
crypto pki trustpoint <Sub-PKI-SERVER-Name>
 enrollment url <Root-CA URL>
```

L'activation de ce serveur de PKI déclenche maintenant ces événements :

- Le point de confiance de serveur de PKI est authentifié afin d'installer le certificat Racine-CA.
- Après que le Racine-CA soit authentifié, l'IOS génère un CSR pour la contrainte Subalterne-CA [x509 de base contenant le CA : RECTIFIEZ l'indicateur] et l'envoyez au Racine-CA

Indépendamment du mode de concession configuré sur le Racine-CA, l'IOS met les demandes de certificat CA (ou RA) dans la file d'attente en suspens. Un administrateur doit manuellement accorder les Certificats CA.
Pour visualiser le certificat en attente demandez et le demande-id :

```
show crypto pki server <Server-Name> requests
```

Pour accorder la demande :

```
crypto pki server <Server-Name> grant <request-id>
```

- Utilisant ceci, l'exécution ultérieure de BALAYAGE SCEP (GetCertInitial) télécharge le certificat Sous-titre-CA et l'installe sur le routeur, qui active le serveur subalterne de PKI

IOS comme RA

Le serveur de PKI IOS peut être configuré comme autorité d'enregistrement à un subalterne ou à une racine donné CA. Pour configurer le serveur de PKI comme autorité d'enregistrement, activez d'abord cette commande sous la section de configuration du serveur de PKI (avant d'activer le serveur de PKI) :

```
crypto pki server <RA-SERVER-Name>
 mode ra
```

Après ceci, configurez l'URL du Ca sous le point de confiance du serveur de PKI. Ceci indique quel CA est protégé par le RA :

```
crypto pki trustpoint <RA-SERVER-Name>
 enrollment url <CA URL>
 subject-name CN=<Common Name>, OU=ioscs RA, OU=TAC, O=Cisco
```

Une autorité d'enregistrement ne délivre pas des Certificats, par conséquent la configuration d'**issuer-name** sous le RA n'est pas exigée, et n'est pas efficace même si elle est configurée. Le subject-name d'un RA est configuré sous le point de confiance de RA utilisant la commande de **subject-name**. Il est important de configurer **OU = RA d'ioscs** en tant qu'élément du subject-name afin de l'IOS CA pour identifier le RA IOS c.-à-d. pour identifier les demandes de certificat autorisé par le RA IOS.

L'IOS peut agir en tant qu'autorité d'enregistrement au tiers CAs tel que Microsoft CA, et afin de rester compatible le RA IOS doit être activé utilisant cette commande sous la section de configuration du serveur de PKI (avant d'activer le serveur de PKI) :

```
mode ra transparent
```

En mode par défaut de RA, l'IOS signe les demandes [PKCS#10] de client utilisant le certificat de RA. Cette exécution indique le serveur de PKI IOS que la demande de certificat a été autorisé par

un RA.

Avec le mode transparent de RA, l'IOS en avant les demandes de client dans leur format d'origine sans introduire le certificat de RA, et c'est compatible avec Microsoft CA comme exemple réputé.

Client de PKI IOS

Un de l'entité de configuration la plus importante dans le client de PKI IOS est un point de confiance. Les paramètres de configuration de point de confiance sont expliqués en détail dans cette section.

Source bien fondée de temps

En tant que première, bien fondée source précisée de temps est une condition requise sur le client de PKI aussi bien. Le client de PKI IOS peut être configuré en tant qu'utilisation de client de NTP ces configuration :

```
configure terminal
ntp server <NTP Server IP address>
ntp source <source interface name>
ntp update-calendar

!! optional, if the NTP Server requires the clients to authenticate themselves
ntp authenticate
ntp authentication-key 1 md5 <key>

!! Optionally an access-list can be configured to restrict time-updates from a specifc NTP
server
access-list 1 permit <NTP Server IP address>
ntp access-group peer 1
```

Adresse Internet et nom de domaine

Une recommandation générale est de configurer une adresse Internet et un domain-name sur le routeur :

```
configure terminal
hostname <string>
ip domain name <domain>
```

Paire de clés RSA

Dans le client de PKI IOS, la paire de clés RSA pour une inscription indiquée de point de confiance peut être automatiquement générée ou manuellement générée.

Le procédé de génération automatique de clé RSA implique ce qui suit :

- L'IOS par défaut crée la paire de clés RSA de 512 bits
- Le nom automatiquement généré de paire de clés est hostname.domain-name, qui est l'adresse Internet de périphérique combinée avec le domain-name de périphérique
- La paire de clés automatique-générée n'est pas marquée en tant qu'exportable.

Le procédé de génération automatique de clé RSA implique ce qui suit :

- Sur option, une paire de clés RSA d'usage universel d'un point fort approprié peut être

manuellement générée utilisant :

- `crypto key generate rsa general-keys label <LABEL> modulus < MOD> [exportable]` Ici, ÉTIQUETTE - le nom de paire de clés RSA

  Modèle - Le module ou le point fort de clé RSA dans les bits entre 360 labourent 4096, qui sont traditionnellement 512, 1024, 2048 ou 4096.

  L'avantage de générer manuellement la paire de clés RSA est la capacité de marquer la paire de clés comme exportable, qui tient compte consécutivement pour que le certificat d'identité soit complètement exporté, qui peut alors être restauré sur un autre périphérique. Cependant, on devrait comprendre les implications en matière de sécurité de cette action.
- Une paire de clés RSA est liée à un point de confiance avant l'inscription utilisant cette commande

  `crypto pki trustpoint MGMT`

  `rsakeypair <LABEL> [<MOD> <MOD>]` Ici, si une paire de clés RSA nommée <LABEL> existe déjà, puis il est pris pendant l'inscription de point de confiance.

  Si une paire de clés RSA nommée <LABEL> n'existe pas, alors un de l'action suivante est exécuté pendant l'inscription :

  - Si aucun argument <MOD> n'est passé, alors 512 bits <LABEL> nommé par paire de clés est générés.

  - si un argument <MOD> est passé, alors une paire de clés d'usage universel de bits <MOD> nommée <LABEL> est générée

  - si deux arguments <MOD> sont passés, alors une paire de clés de signature de bits <MOD> et une paire de clés de cryptage de bits <MOD>, les deux <LABEL> Désignés sont générées

Point de confiance

Un point de confiance est un conteneur abstrait pour tenir un certificat dans l'IOS. Un point de confiance simple est capable d'enregistrer deux Certificats actifs à un moment donné :

- Un certificat de CA - Chargeant un certificat de CA dans un point de confiance donné est connu comme procédure d'authentification de point de confiance.
- Un certificat d'ID délivré par le CA - chargement ou importer un certificat d'ID dans un point de confiance donné est connu en tant que procédé d'inscription de point de confiance.

Une configuration de point de confiance est connue comme stratégie de confiance, et ceci définit cela :

- Quel certificat de CA est chargé dans le point de confiance ?
- Àquel CA le point de confiance s'inscrit-il ?
- Comment l'IOS s'inscrit-il le point de confiance ?
- Comment un certificat délivré par le CA donné [chargé dans le point de confiance] est validé ?

Des composants principaux d'un point de confiance sont expliqués ici.

Mode d'inscription

Un mode d'inscription de point de confiance, qui définit également l'authentication mode de point de confiance, peut être exécuté par l'intermédiaire de 3 méthodes principales :

1. Inscription terminale - méthode manuelle d'exécuter l'authentification et l'inscription de certificat de point de confiance utilisant la copie-pâte dans le terminal CLI.
2. Inscription SCEP - Authentification et inscription de point de confiance utilisant SCEP au-dessus de HTTP.
3. Profil d'inscription - Ici, des méthodes d'authentification et d'inscription sont définies séparément. Avec des méthodes de terminal et d'inscription SCEP, les profils d'inscription fournissent une option de spécifier des commandes HTTP/TFTP d'exécuter la récupération de fichier du serveur, qui est défini utilisant un URL d'authentification ou d'inscription sous le profil.

Interface et VRF de source

L'authentification et l'inscription de point de confiance au-dessus du HTTP (SCEP) ou du TFTP (profil d'inscription) emploie le système de fichiers IOS pour exécuter des exécutions E/S de fichier. Ces échanges de paquet peuvent être originaires d'une interface spécifique de source et d'un VRF.

En cas de configuration classique de point de confiance, cette fonctionnalité est activée utilisant des commandes secondaires d'**interface** et de **vrf de source** sous le point de confiance.

En cas de profils d'inscription, d'**interface** et d'**inscription de source | l'authentication url <http/tftp://Server-location > les** commandes de **<vrf-name> de vrf** fournissent la même fonctionnalité.

Exemple de configuration :

```
vrf definition MGMT
 rd 1:1
 address-family ipv4
 exit-address-family

crypto pki trustpoint MGMT
 source interface Ethernet0/0
 vrf MGMT
```

ou

```
crypto pki profile enrollment MGMT-Prof
 enrollment url http://10.1.1.1:80 vrf MGMT
 source-interface Ethernet0/0
crypto pki trustpoint MGMT
 enrollment profile MGMT-Prof
```

Inscription de certificat et renouvellement automatiques

Le client de PKI IOS peut être configuré pour exécuter l'inscription et le renouvellement automatiques utilisant cette commande sous la section de point de confiance de PKI :

```
crypto pki trustpoint MGMT
auto-enroll <percentage> <regenerate>
```

Ici, déclarer de commande **[régénérés] de <percentage> d'auto-enroll** que l'IOS devrait exécuter le renouvellement de certificat exactement à 80% de la vie du certificat valable.

Le régénéré de mot clé déclare que l'IOS devrait régénérer la paire de clés RSA connue sous le nom de paire de clés de shadow pendant chaque exécution de renouvellement de certificat.

C'est le comportement automatique d'inscription :

- **L'auto-enroll de** moment est configuré, si le point de confiance est authentifié, IOS exécutera une inscription automatique au serveur situé à l'URL mentionné en tant qu'élément de la commande **URL d'inscription** sous la section de point de confiance de PKI ou sous le profil d'inscription.
- Le moment où un point de confiance est inscrit avec un serveur de PKI ou un CA, un RENOUVELER ou un temporisateur de SHADOW est initialisé sur le client de PKI ont basé sur le pourcentage d'**auto-enroll du** certificat d'identité en cours ont installé sous le point de confiance. Ce temporisateur est visible sous la **crypto** commande de **temporisateur de PKI d'exposition**. Plus sur les fonctions de temporisateur *se rapportent*
- Le support de capacité de renouvellement provient le serveur de PKI. Plus sur ceci dedans Le client de PKI IOS exécute deux types de renouvellement :
  Renouvellement implicite : Si le serveur de PKI n'envoie pas le « renouvellement » comme capacité prise en charge, l'IOS exécute une première inscription au pourcentage défini d'auto-enroll. c.-à-d. l'IOS emploie un certificat auto-signé pour signer la demande de renouvellement.Renouvellement explicite : Quand le serveur de PKI prend en charge la caractéristique de renouvellement de certificat client de PKI, elle annonce le « renouvellement » comme capacité prise en charge. L'IOS prend en compte cette capacité pendant l'IOS de renouvellement de certificat c.-à-d. emploie le certificat d'identité actif en cours pour signer la demande de certificat de renouvellement.

Le soin devrait être pris tout en configurant le pourcentage d'auto-enroll. Sur n'importe quel client indiqué de PKI dans le déploiement, si une condition surgit où le certificat d'identité expire en même temps que le certificat de CA émettant, puis la valeur d'auto-enroll devrait toujours déclencher [l'exécution de renouvellement de shadow] après que le CA ait créé le certificat inversé. *Référez-vous à la* section de **dépendances de temporisateur de PKI** dedans

Revocation-check de certificat

Un point de confiance authentifié de PKI c.-à-d. un point de confiance de PKI contenant un certificat de CA est capable d'exécuter la validation de certificat pendant une négociation d'IKE ou SSL, où le certificat de pair est soumis à une validation complète de certificat. Une des méthodes de validation est de vérifier l'état de révocation de certificat de pair utilisant une des deux méthodes suivantes :

- Liste des révocations de certificat (CRL) - C'est un fichier contenant les numéros de série des Certificats retirés par un CA donné. Ce fichier est signé utilisant le certificat de CA émettant. La méthode CRL implique de télécharger le fichier CRL utilisant le HTTP ou le LDAP.

- L'état en ligne Protocol (OCSP) de certificat - IOS établit la voie de transmission avec une entité appelée comme responder OCSP, qui est un serveur indiqué par le CA émettant. Un client tel que l'IOS envoie une demande contenant le numéro de série du certificat est validé. Le responder OCSP répond avec l'état de révocation du numéro de série donné. La voie de transmission pourrait être établie utilisant n'importe quels application/protocole de transport pris en charge, qui est habituellement HTTP.

Le contrôle de révocation peut être défini utilisant ces derniers commandent sous la section de point de confiance de PKI :

```
crypto pki trustpoint MGMT
 revocation-check crl ocsp none
```
Par défaut, un point de confiance est configuré pour exécuter le contrôle de révocation utilisant le crl.

Les méthodes peuvent être commandées à nouveau, et le contrôle d'état de révocation est exécuté dans la commande définie. La méthode « aucun » saute la revocation-check.

Cache CRL

Le CRL étant basé la revocation-check, chaque validation de certificat peut déclencher un téléchargement de fichier frais CRL. Et car le fichier CRL obtient plus grand ou si le point de distribution CRL (CDP) est plus loin parti, télécharger le fichier pendant chaque processus de validation entrave la représentation de la personne à charge de protocole sur la validation de certificat. Par conséquent, la mise en cache CRL est exécutée pour améliorer la représentation, et la mise en cache le CRL prend en compte la validité CRL.

La validité CRL est définie utilisant deux paramètres de temps : **LastUpdate**, qui est la dernière fois le CRL a été édité par le CA émettant, et **NextUpdate**, qui est le temps est le futur où une nouvelle version de fichier CRL est éditée par le CA émettant.

L'IOS cache chaque CRL téléchargé pour tant que le CRL est valide. Cependant, sous certaines circonstances telles que le CDP n'étant pas accessible temporairement, il peut être nécessaire de retenir le CRL dans le cache pendant une longue période. Dans l'IOS un CRL caché peut retenu pour tant que pendant 24 heures après que la validité CRL expire, et ceci peut être configuré utilisant cette commande sous la section de point de confiance de PKI :

```
crypto pki trustpoint MGMT
 crl cache extend <0 - 1440>
!! here the value is in minutes
```
Sous certaines circonstances telles qu'un CA émettant retirant des Certificats au cours de la période de validité CRL, l'IOS peut confiugured pour supprimer le cache plus fréquemment. En supprimant le CRL pr3maturément, l'IOS est forcé pour télécharger le CRL plus fréquemment pour maintenir le cache CRL à jour. Cette option de configuration est disponible sous la section de point de confiance de PKI :

```
crypto pki trustpoint MGMT
 crl cache delete-after <1-43200>
!! here the value is in minutes
```
Et en conclusion, l'IOS peut être configuré pour ne pas cacher le fichier CRL utilisant cette commande sous la section de point de confiance de PKI :

```
crypto pki trustpoint MGMT
 crl cache none
```

Configuration recommandée

Un déploiement typique CA avec le Racine-CA et une configuration Sous-titre-CA est en tant que ci-dessous. L'exemple inclut également une configuration Sous-titre-CA protégée par un RA.

Avec la paire de clés RSA de 2048 bits d'un bout de l'affaire à l'autre, cet exemple recommande une installation où :
Le Racine-CA a une vie de 8 ans

Le Sous-titre-CA a une vie de 3 ans
Des certificats client sont délivrés pendant une année, qui sont configurés pour demander pour un renouvellement de certificat, automatiquement.

RACINE CA - Configuration

```
crypto pki server ROOTCA
database level complete
database archive pkcs12 password p12password
issuer-name CN=RootCA,OU=TAC,O=Cisco
lifetime crl 120
lifetime certificate 1095
lifetime ca-certificate 2920
grant auto rollover ca-cert
auto-rollover 85
database url ftp:://10.1.1.1/CA/ROOT/
database url crl ftp:://10.1.1.1/CA/ROOT/
database url crl publish ftp:://10.1.1.1/WWW/CRL/ROOT/
cdp-url http://10.1.1.1/WWW/CRL/ROOT/ROOTCA.crl
```

SUBCA sans RA - Configuration

```
crypto pki server SUBCA
database level complete
database archive pkcs12 password p12password
issuer-name CN=SubCA,OU=TAC,O=Cisco
lifetime crl 12
lifetime certificate 365
grant auto SUBCA
auto-rollover 85
database url ftp:://10.1.1.1/CA/SUB/
database url crl ftp:://10.1.1.1/CA/SUB/
database url crl publish ftp:://10.1.1.1/WWW/CRL/SUB/
cdp-url http://10.1.1.1/WWW/CRL/SUB/SUBCA.crl
mode sub-cs


crypto pki trustpoint SUBCA
revocation-check crl
rsakeypair SUBCA 2048
enrollment url http://172.16.1.1
```

SUBCA avec du RA - Configuration

```
crypto pki server SUBCA
database level complete
database archive pkcs12 password p12password
issuer-name CN=SubCA,OU=TAC,O=Cisco
lifetime crl 12
lifetime certificate 365
grant ra-auto
grant auto rollover ra-cert
auto-rollover 85
 database url ftp:://10.1.1.1/CA/SUB/
database url crl ftp:://10.1.1.1/CA/SUB/
database url crl publish ftp:://10.1.1.1/WWW/CRL/SUB/
cdp-url http://10.1.1.1/WWW/CRL/SUB/SUBCA.crl
mode sub-cs

crypto pki trustpoint SUBCA
revocation-check crl
rsakeypair SUBCA 2048
```

```
enrollment url http://172.16.1.1
```

RA pour SUBCA - Configuration

```
crypto pki server RA-FOR-SUBCA
database level complete
database archive pkcs12 password p12password
mode ra
grant auto RA-FOR-SUBCA
auto-rollover 85
database url ftp:://10.1.1.1/CA/RA4SUB/


crypto pki trustpoint RA-FOR-SUBCA
enrollment url http://172.16.1.2:80
password ChallengePW123
subject-name CN=RA,OU=ioscs RA,OU=TAC,O=Cisco
revocation-check crl
rsakeypair RA 2048
```

Inscription de certificat

Inscription manuelle

L'Inscription manuelle comporte la génération hors ligne CSR sur le client de PKI, qui est manuellement copié plus de sur l'administrateur CA signe manuellement la demande, qui est alors importée dans le client.

Client de PKI

Configuration de client de PKI :

```
crypto pki trustpoint MGMT
enrollment terminal
serial-number
ip-address none
password ChallengePW123
subject-name CN=PKI-Client,OU=MGMT,OU=TAC,O=Cisco
revocation-check crl
rsakeypair PKI-Key
```

Étape 1. Authentifiez d'abord le point de confiance (ceci peut également être exécuté après l'étape 2).

```
crypto pki authenticate MGMT
!! paste the CA, in this case the SUBCA, certificate in pem format and enter “quit” at the end
in a line by itself] PKI-Client-1(config)# crypto pki authenticate MGMT

Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIDODCCAiCgAwIBAgIBAjANBgkqhkiG9w0BAQUFADAvMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ8wDQYDVQQDEwZSb290Q0EwHhcNMTUxMDE4MjA0MjI3
WhcNMTgxMDE3MjA0MjI3WjAuMQ4wDAYDVQQKEwVDaXNjbzEMMAoGA1UECxMDVEFD
MQ4wDAYDVQQDEwVTdWJDQTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AJ7hKmBfDo/GOQAEYY/1ptpg28DejUE0ZlDorDkADP2vKfRI0kalSnOs2PIe01ip
```

```
7pHFurFVUx/p8teMCkmvnbrSBfyUrWo9YfQeGOELb4d3dSW4jGakm6M8lNRkO7HP
s+IVVTuJSeUZxov6DPa92Y/6HLayX15Iq8ZL+KwmA9oS5NeTi1tBbrcc3Hq8W2Ay
879nDDOqDOsQMQqKtc7E/IA7SBjowImra6FUxzgJ5ye5MymRfRYAH+c4qZJxwHTc
/tSmjiOJlM7X5dtehU/XPEEEbs78peXO9FyzAbhOtCRBVTnhc8WWijq84xu8Oej7
LbXGBKIHSP0uDe32CV0noEUCAwEAAaNgMF4wDwYDVR0TAQH/BAUwAwEB/zALBgNV
HQ8EBAMCAYYwHwYDVR0jBBgwFoAU+oNBdIj9mjpieQ2Z7v79JhKnL68wHQYDVR0O
BBYEFFOv8xtHRojMdJ65oQ2PFBeD5oHiMA0GCSqGSIb3DQEBBQUAA4IBAQAZ/W3P
Wqs4vuQ2jCnVE0v1PVQe/VNS54P/fprQRelceawiBCHA3D0SRgHqUWJUIqBLv4sD
QBegmyTmS76C8YC/jN7VbI30hf6R4qP7CWu8Ef9sWPRC/+Oy6e8AinrK+sVd2dp/
LLDMVoBhS2bQFLWiyRvC9FgyczXRdF+rhKTKeEVXGs7C/Yk/9z+/00rVmSGZAS+v
aPpZWjoC3459t51t8Y3iE6GtjBvmyxBwWt01/5gCu6Mszi7X/kXdmqgNfT5bBBnv
yjWE2ZS8NsH4hwDZpmDJqx4qhrH6bw3iUm+pK9fceZ/HTYasxtcr4NUvvxwXc60y
Wrtlpq3g2XfG+qFB
-----END CERTIFICATE-----
quit
Trustpoint 'MGMT' is a subordinate CA and holds a non self signed cert
Certificate has the following attributes:
      Fingerprint MD5: DBE6AFAC 9E1C3697 01C5466B 78E0DFE3
     Fingerprint SHA1: EAD41B32 BB37BC11 6E0FBC13 41701BFE 200DC46E

% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

Étape 2. Générez la demande de signature de certificat et prenez le CSR au CA et obtenez le certificat accordé :

```
PKI-Client-1(config)# crypto pki enroll MGMT
% Start certificate enrollment ..

% The subject name in the certificate will include: CN=PKI-Client,OU=MGMT,OU=TAC,O=Cisco
% The subject name in the certificate will include: PKI-Client-1.cisco.com
% The serial number in the certificate will be: 104Certificate Request follows:

MIIC2zCCAcMCAQAwdTEOMAwGA1UEChMFQ2lzY28xDDAKBgNVBAsTA1RBQzENMAsG
A1UECxMETUdNVDETMBEGA1UEAxMKUEtJLUNsaWVudDExMAoGA1UEBRMDMTA0MCMG
CSqGSIb3DQEJAhYWUEtJLUNsaWVudC0xLmNpc2NvLmNvbTCCASIwDQYJKoZIhvcN
AQEBBQADggEPADCCAQoCggEBANwa7g+DJxG57sMg020w1Fdv9+mIZ6R4livbt7vo
AbW8jpzQlMv4lV3r6ulTJumhBvV7xI+1ZijXP0EqqQZLNboYv37UTJgm83DGO57I
8RTn9DfDQpHiqvhtNuC5S3SCC/hvCxFXnfNXqC3dkfuVkVWoJiLZY87R6j44jUq0
tTL5d8t6lz2L0BeekzKJlOs73gONx0VgQyI/WjDiEwL0xF4DNHURaYyOxBWJc7/B
psDCf7376mb7XXz0LB++E8SvvM/Li6+yQzYv1Lagr0b8C4uE+tCDxG5OniNDiS82
JXsVd43vKRFW85W2ssrElgkuWAvS017XlwK+UDX21dtFdfUCAwEAAaAhMB8GCSqG
SIb3DQEJDjESMBAwDgYDVR0PAQH/BAQDAgWgMA0GCSqGSIb3DQEBBQUAA4IBAQA+
UqkqUZZar9TdmB8I7AHku5m79l42o8cuhwOccehxE6jmzh9P+Ttb9Me7l7L8Y2iR
yYyJHsL7m6tjK2+Gllg7RJdoxG8l8aMZS1ruXOBqFBrmo7OSz1nfXpiTyh88jyca
Hw/8G8uaYuQbZIj53BWmQGRpm7J//ktn0D4W3Euh9HttMuYYX7BOct05BLqqiCCw
n+kKHZxzGXy7JSZpUlDtvPPnnuqWK7iVoy3vtV6GoFOrxRoo05QVFehS0/m4NFQI
mXA0eTEgujSaQi4iWte/UxruO/3p/eHr67MtZXLRl0YDFgaQd7vD7fCsDx5pquKV
jNEUT6FNHdsnqrAKqodO

---End - This line not part of the certificate request---

Redisplay enrollment request? [yes/no]: no
```

Étape 3. Importez maintenant le certificat accordé par l'intermédiaire du terminal :

```
PKI-Client-1(config)# crypto pki import MGMT certificate

Enter the base 64 encoded certificate.
```

```
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIDcDCCAligAwIBAgIBAzANBgkqhkiG9w0BAQQFADAuMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ4wDAYDVQQDEwVTdWJDQTAeFw0xNTEwMTkyMDM1MDZa
Fw0xNjEwMTgyMDM1MDZaMHUxDjAMBgNVBAoTBUNpc2NvMQwwCgYDVQQLEwNUQUMx
DTALBgNVBAsTBE1HTVQxEzARBgNVBAMTClBLSS1DbGllbnQxMTAKBgNVBAUTAzEw
NDAjBgkqhkiG9w0BCQIWFlBLSS1DbGllbnQtMS5jaXNjby5jb20wggEiMA0GCSqG
SIb3DQEBAQUAA4IBDwAwggEKAoIBAQDcGu4PgycRue7DINNtMNRXb/fpiGekeJYr
27e76AG1vI6c0JTL+JVd6+rpUybpoQb1e8SPtWYo1z9BKqkGSzW6GL9+1EyYJvNw
xjueyPEU5/Q3w0KR4qr4bTbguUt0ggv4bwsRV53zV6gt3ZH7lZFVqCYi2WPO0eo+
OI1KtLUy+XfLepc9i9AXnpMyiZTrO94DjcdFYEMiP1ow4hMC9MReAzR1EWmMjsQV
iXO/wabAwn+9++pm+1189CwfvhPEr7zPy4uvskM2L9S2oK9G/AuLhPrQg8RuTp4j
Q4kvNiV7FXeN7ykRVvOVtrLKxJYJLlgL0tNe15cCvlA19tXbRXX1AgMBAAGjUjBQ
MA4GA1UdDwEB/wQEAwIFoDAfBgNVHSMEGDAWgBRTr/MbR0aIzHSeuaENjxQXg+aB
4jAdBgNVHQ4EFgQUK+9/lrlL+TyYxvsgxzPwwrhmS5UwDQYJKoZIhvcNAQEEBQAD
ggEBAIrLrzFLnm9z7ula1uRh03r6dSCFy9XkOk6ZaHfksbENoDmkcgIwKoAsSF9E
rQmA9W5qXVU7PEsqOmcu8zEv7uuiqM4D4nDP69HsyToPjxVcoG7PSyKJYnXRgkVa
IYyMaSaRKWlhb2uWj3XPLzS0/ZBOGAG9rMBVzaqLfLAZgnQUVJvwsNofe+ASojk9
mCRsEHD8WVuAzcnwYKXx3j3x/T7jbB3ibPfbYKQqlS12XFHhJoK+HfSA2fyZBFLF
syN/B2Ow0bvc71YlYOQuYwz3XOMIHD6vARTO4f0ZIQti2dy1kHc+5lIdhLsn/bA5
yUo7WxnAE8LOoYIf9iU9q0mqkMU=
-----END CERTIFICATE-----
quit
% Router Certificate successfully imported
```

Serveur de PKI

Étape 1. Exportez d'abord le certificat de CA émettant du CA, qui est dans ce cas certificat SUBCA. Ceci est importé pendant l'étape 1 ci-dessus sur le client de PKI, c.-à-d. authentification de point de confiance.

```
SUBCA(config)# crypto pki export SUBCA pem terminal
% CA certificate: !! Root-CA certificate
-----BEGIN CERTIFICATE-----
MIIDPDCCAiSgAwIBAgIBATANBgkqhkiG9w0BAQQFADAvMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ8wDQYDVQQDEwZSb290Q0EwHhcNMTUxMDE4MjAwOTIx
WhcNMjMxMDE2MjAwOTIxWjAvMQ4wDAYDVQQKEwVDaXNjbzEMMAoGA1UECxMDVEFD
MQ8wDQYDVQQDEwZSb290Q0EwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQCajfMy8gU3ZXQfKgP/wYKLB0cuywzYcDaSoNVlEvUZOWgUltCGP4CiCXyw0U0U
Zmy0rusibMV7mtkTX5muaPC0XfT98rswPiZV0qvEYpHF2YodPOUoqR3FeKj/tDbI
IikcLrfj87aeMJjCrWD888wfTN9Hw9x2QVDoSxLbzTLticXdXxwS5wxlM16GspmT
WL4fg1JRWgjRqMmOcpf716Or88XJ2N2HeWxxVFIwYQf3thHR6DgTdCgJ1uqjVE6q
1LQ1g8k81mvuCZX0uLZiTMJ69xo+Ot/RpeeE2RShxK5rh56ObQq4MT4lbIPKqIxU
lbKzWdh10NiYwjgTNwTs9GGvAgMBAAGjYzBhMA8GA1UdEwEB/wQFMAMBAf8wDgYD
VR0PAQH/BAQDAgGGMB8GA1UdIwQYMBaAFPqDQXSI/Zo6YnkNme7+/SYSpy+vMB0G
A1UdDgQWBBT6g0F0iP2aOmJ5DZnu/v0mEqcvrzANBgkqhkiG9w0BAQQFAAOCAQEA
VKwqI9vpmoRh9QoOJGtOA3qEgV4eCfXdMuYxmmo0sdaBYBfQm2RhZeQ1X90vVBso
G4Wx6cJVSXCtkqZTm1IoMtya+gdhLbKqZmxc+I5/js88SrbrBIm4zj+sOoySV9kW
THEEmZjdTCWXo2wNCr23gGdnb4RqZ0FTOfoZO/2Xnpcbvhz2/K7wlDRJ5k1wrsRW
RRwsQEh4LYMFIg0aBs4gmRLZ8ytwrvvrhQTVrAA/MeomUEPhcIYESg1AlWxoCYZU
0iqKfDa9+4wEJ+PMGDhM2UV0fuP0rWitKWxecSVbo54z3VHYwwCbz2jCs8XGE61S
+XlxCZKFVdlVaMmuaZTdFg==
-----END CERTIFICATE-----

% General Purpose Certificate: !! SUBCA certificate
-----BEGIN CERTIFICATE-----
MIIDODCCAiCgAwIBAgIBAjANBgkqhkiG9w0BAQUFADAvMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ8wDQYDVQQDEwZSb290Q0EwHhcNMTUxMDE4MjA0MjI3
WhcNMTgxMDE3MjA0MjI3WjAuMQ4wDAYDVQQKEwVDaXNjbzEMMAoGA1UECxMDVEFD
MQ4wDAYDVQQDEwVTdWJDQTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
```

```
AJ7hKmBfDo/GOQAEYY/1ptpg28DejUE0ZlDorDkADP2vKfRI0kalSnOs2PIe01ip
7pHFurFVUx/p8teMCkmvnbrSBfyUrWo9YfQeGOELb4d3dSW4jGakm6M8lNRkO7HP
s+IVVTuJSeUZxov6DPa92Y/6HLayX15Iq8ZL+KwmA9oS5NeTi1tBbrcc3Hq8W2Ay
879nDDOqDOsQMQqKtc7E/IA7SBjowImra6FUxzgJ5ye5MymRfRYAH+c4qZJxwHTc
/tSmjiOJlM7X5dtehU/XPEEEbs78peXO9FyzAbhOtCRBVTnhc8WWijq84xu8Oej7
LbXGBKIHSP0uDe32CV0noEUCAwEAAaNgMF4wDwYDVR0TAQH/BAUwAwEB/zALBgNV
HQ8EBAMCAYYwHwYDVR0jBBgwFoAU+oNBdIj9mjpieQ2Z7v79JhKnL68wHQYDVR0O
BBYEFFOv8xtHRojMdJ65oQ2PFBeD5oHiMA0GCSqGSIb3DQEBBQUAA4IBAQAZ/W3P
Wqs4vuQ2jCnVE0v1PVQe/VNS54P/fprQRelceawiBCHA3D0SRgHqUWJUIqBLv4sD
QBegmyTmS76C8YC/jN7VbI30hf6R4qP7CWu8Ef9sWPRC/+Oy6e8AinrK+sVd2dp/
LLDMVoBhS2bQFLWiyRvC9FgyczXRdF+rhKTKeEVXGs7C/Yk/9z+/00rVmSGZAS+v
aPpZWjoC3459t51t8Y3iE6GtjBvmyxBwWt01/5gCu6Mszi7X/kXdmqgNfT5bBBnv
yjWE2ZS8NsH4hwDZpmDJqx4qhrH6bw3iUm+pK9fceZ/HTYasxtcr4NUvvxwXc60y
Wrtlpq3g2XfG+qFB
-----END CERTIFICATE-----
```

Étape 2. Après Step-2 sur le PKI-client, prenez le CSR du client et fournissez-le pour se connecter le SUBCA utilisant cette commande :

```
crypto pki server SUBCA request pkcs10 terminal pem
```

Cette commande suggère que le SUBCA reçoive une demande de signature de certificat du terminal, et une fois accordé, les données de certificat sont imprimées dans le format PEM.

```
SUBCA# crypto pki server SUBCA request pkcs10 terminal pem
PKCS10 request in base64 or pem

% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
MIIC2zCCAcMCAQAwdTEOMAwGA1UEChMFQ2lzY28xDDAKBgNVBAsTA1RBQzENMAsG
A1UECxMETUdNVDETMBEGA1UEAxMKUEtJLUNsaWVudDExMAoGA1UEBRMDMTA0MCMG
CSqGSIb3DQEJAhYWUEtJLUNsaWVudC0xLmNpc2NvLmNvbTCCASIwDQYJKoZIhvcN
AQEBBQADggEPADCCAQoCggEBANwa7g+DJxG57sMg020w1Fdv9+mIZ6R4livbt7vo
AbW8jpzQlMv4lV3r6ulTJumhBvV7xI+1ZijXP0EqqQZLNboYv37UTJgm83DGO57I
8RTn9DfDQpHiqvhtNuC5S3SCC/hvCxFXnfNXqC3dkfuVkVWoJiLZY87R6j44jUq0
tTL5d8t6lz2L0BeekzKJlOs73gONx0VgQyI/WjDiEwL0xF4DNHURaYyOxBWJc7/B
psDCf7376mb7XXz0LB++E8SvvM/Li6+yQzYv1Lagr0b8C4uE+tCDxG5OniNDiS82
JXsVd43vKRFW85W2ssrElgkuWAvS017XlwK+UDX21dtFdfUCAwEAAaAhMB8GCSqG
SIb3DQEJDjESMBAwDgYDVR0PAQH/BAQDAgWgMA0GCSqGSIb3DQEBBQUAA4IBAQA+
UqkqUZZar9TdmB8I7AHku5m79l42o8cuhwOccehxE6jmzh9P+Ttb9Me7l7L8Y2iR
yYyJHsL7m6tjK2+Gllg7RJdoxG8l8aMZS1ruXOBqFBrmo7OSz1nfXpiTyh88jyca
Hw/8G8uaYuQbZIj53BWmQGRpm7J//ktn0D4W3Euh9HttMuYYX7BOct05BLqqiCCw
n+kKHZxzGXy7JSZpUlDtvPPnnuqWK7iVoy3vtV6GoFOrxRoo05QVFehS0/m4NFQI
mXA0eTEgujSaQi4iWte/UxruO/3p/eHr67MtZXLRl0YDFgaQd7vD7fCsDx5pquKV
jNEUT6FNHdsnqrAKqodO
quit
% Enrollment request pending, reqId=1
```

Si le CA est en mode d'automatique-concession, le certificat accordé est affiché dans le format PEM ci-dessus. Quand le CA est en mode de octroi manuel, la demande de certificat est marquée en tant qu'**en suspens**, est assignée une valeur d'id et alignée dans la file d'attente de demandes d'inscription.

```
SUBCA#show crypto pki server SUBCA requests
Enrollment Request Database:

Router certificates requests:
ReqID  State      Fingerprint                      SubjectName
--------------------------------------------------------------------
1      pending    7710276982EA176324393D863C9E350E serialNumber=104+hostname=PKI-Client-
1.cisco.com,cn=PKI-Client,ou=MGMT,ou=TAC,o=Cisco
```

Étape 3. Accordez manuellement cette demande utilisant cette commande :

```
SUBCA# crypto pki server SUBCA grant 1
% Granted certificate:
-----BEGIN CERTIFICATE-----
MIIDcDCCAligAwIBAgIBAzANBgkqhkiG9w0BAQQFADAuMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ4wDAYDVQQDEwVTdWJDQTAeFw0xNTEwMTkyMDM1MDZa
Fw0xNjEwMTgyMDM1MDZaMHUxDjAMBgNVBAoTBUNpc2NvMQwwCgYDVQQLEwNUQUMx
DTALBgNVBAsTBE1HTVQxEzARBgNVBAMTClBLSS1DbGllbnQxMTAKBgNVBAUTAzEw
NDAjBgkqhkiG9w0BCQIWFlBLSS1DbGllbnQtMS5jaXNjby5jb20wggEiMA0GCSqG
SIb3DQEBAQUAA4IBDwAwggEKAoIBAQDcGu4PgycRue7DINNtMNRXb/fpiGekeJYr
27e76AG1vI6c0JTL+JVd6+rpUybpoQb1e8SPtWYo1z9BKqkGSzW6GL9+1EyYJvNw
xjueyPEU5/Q3w0KR4qr4bTbguUt0ggv4bwsRV53zV6gt3ZH7lZFVqCYi2WPO0eo+
OI1KtLUy+XfLepc9i9AXnpMyiZTrO94DjcdFYEMiP1ow4hMC9MReAzR1EWmMjsQV
iXO/wabAwn+9++pm+1189CwfvhPEr7zPy4uvskM2L9S2oK9G/AuLhPrQg8RuTp4j
Q4kvNiV7FXeN7ykRVvOVtrLKxJYJLlgL0tNe15cCvlA19tXbRXX1AgMBAAGjUjBQ
MA4GA1UdDwEB/wQEAwIFoDAfBgNVHSMEGDAWgBRTr/MbR0aIzHSeuaENjxQXg+aB
4jAdBgNVHQ4EFgQUK+9/lrlL+TyYxvsgxzPwwrhmS5UwDQYJKoZIhvcNAQEEBQAD
ggEBAIrLrzFLnm9z7ula1uRh03r6dSCFy9XkOk6ZaHfksbENoDmkcgIwKoAsSF9E
rQmA9W5qXVU7PEsqOmcu8zEv7uuiqM4D4nDP69HsyToPjxVcoG7PSyKJYnXRgkVa
IYyMaSaRKWlhb2uWj3XPLzS0/ZBOGAG9rMBVzaqLfLAZgnQUVJvwsNofe+ASojk9
mCRsEHD8WVuAzcnwYKXx3j3x/T7jbB3ibPfbYKQqlS12XFHhJoK+HfSA2fyZBFLF
syN/B2Ow0bvc71YlYOQuYwz3XOMIHD6vARTO4f0ZIQti2dy1kHc+5lIdhLsn/bA5
yUo7WxnAE8LOoYIf9iU9q0mqkMU=
-----END CERTIFICATE-----
```

Remarque: L'Inscription manuelle d'un Sous-titre-CA à un Racine-CA n'est pas possible.

Remarque: Un CA dans un état handicapé dû a désactivé le serveur HTTP peut manuellement accorder les demandes de certificat.

Inscription utilisant SCEP

La configuration de client de PKI est :

```
crypto pki trustpoint MGMT
enrollment url http://172.16.1.2:80
 serial-number
 ip-address none
 password 7 110A1016141D5A5E57
 subject-name CN=PKI-Client,OU=MGMT,OU=TAC,O=Cisco
 revocation-check crl
 rsakeypair PKI-Key 2048
```

La configuration du serveur de PKI est :

```
SUBCA# show run all | section pki server
crypto pki server SUBCA
 database level complete
 database archive pkcs12 password 7 01100F175804575D72
 issuer-name CN=SubCA,OU=TAC,O=Cisco
 lifetime crl 12
 lifetime certificate 365
 lifetime ca-certificate 1095
 lifetime enrollment-request 168
 mode sub-cs
 auto-rollover 85
 database url ftp:://10.1.1.1/CA/SUB/
database url crl ftp:://10.1.1.1/CA/SUB/
```

```
database url crl publish ftp:://10.1.1.1/WWW/CRL/SUB/
```

Le mode par défaut de l'octroi de demande de certificat est manuel :

```
SUBCA# show crypto pki server
Certificate Server SUBCA:
   Status: enabled
   State: enabled
   Server's configuration is locked  (enter "shut" to unlock it)
   Issuer name: CN=SubCA,OU=TAC,O=Cisco
   CA cert fingerprint: DBE6AFAC 9E1C3697 01C5466B 78E0DFE3
   Server configured in subordinate server mode
   Upper CA cert fingerprint: CD0DE4C7 955EFD60 296B7204 41FB6EF6
    Granting mode is: manual
   Last certificate issued serial number (hex): 4
   CA certificate expiration timer: 21:42:27 CET Oct 17 2018
   CRL NextUpdate timer: 09:42:37 CET Oct 20 2015
   Current primary storage dir: unix:/SUB/
   Current storage dir for .crl files: unix:/SUB/
   Database Level: Complete - all issued certs written as <serialnum>.cer
   Auto-Rollover configured, overlap period 85 days
   Autorollover timer: 21:42:27 CET Jul 24 2018
```

Concession manuelle

Étape 1. Client de PKI : Dans un premier temps, qui est obligatoire, authentifiez le point de confiance sur le client de PKI :

```
PKI-Client-1(config)# crypto pki authenticate MGMT
Trustpoint 'MGMT' is a subordinate CA and holds a non self signed cert
Certificate has the following attributes:
      Fingerprint MD5: DBE6AFAC 9E1C3697 01C5466B 78E0DFE3
     Fingerprint SHA1: EAD41B32 BB37BC11 6E0FBC13 41701BFE 200DC46E

% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
```

Étape 2. PKI-client : Après l'authentification de point de confiance, le client de PKI peut être inscrit pour un certificat.

> Remarque: Si l'auto-enroll est configuré, le client exécutera automatiquement l'inscription.

```
config terminal
crypto pki enroll MGMT
```

Dans les coulisses, ces événements ont lieu :

- L'IOS recherche une paire de clés RSA nommée PKI-Clé. S'il existe, il est pris pour demander un certificat d'identité. Sinon, l'IOS crée 2048 un bit PKI-Clé nommée par paire de clés, et puis l'utilise pour demander un certificat d'identité.
- L'IOS crée une demande de signature de certificat dans le format PKCS10.
- L'IOS chiffre alors ce CSR utilisant une clé symétrique aléatoire. La clé symétrique aléatoire est chiffrée utilisant la clé publique du destinataire, qui est le SUBCA (la clé publique de SUBCA est due disponible à l'authentification de point de confiance). Le CSR chiffré, la clé symétrique aléatoire chiffrée et les informations réceptives est remonté dans des données

enveloppées par PKCS#7.

- Ces données enveloppées par PKCS#7 sont signées utilisant un certificat auto-signé provisoire pendant l'inscription initiale. Le PKCS#7 a enveloppé des données, le certificat de signature utilisé par le client et la signature du client sont remontées dans un paquet de données signé par PKCS#7. C'est base64 encodé, et puis URL encodé. La goutte en résultant des données est envoyée en tant qu'argument de « message » dans l'URI de HTTP envoyé au CA :

```
GET /cgi-bin/pkiclient.exe?operation=PKIOperation&message=MII…. HTTP/1.0
```

Étape 3. Pki-server :

Quand le serveur de PKI IOS reçoit la demande, elle vérifie ces derniers :

1. Vérifie si la base de données de demande d'inscription contient une demande de certificat avec la même Id de transaction associée avec la nouvelle demande.

> Remarque: Une Id de transaction est des informations parasites de MD5 de la clé publique, pour laquelle un certificat d'identité est demandé par le client.

2. Vérifie si la base de données de demande d'inscription contient une demande de certificat avec le même mot de passe de défi que celui envoyé par le client.

> Remarque: Si (1) renvoie vrai ou (1) et (2) ensemble vrai de retour, alors un serveur CA est capable de rejeter la demande en raison de la demande en double d'identité. Cependant, en pareil cas le serveur de PKI IOS remplace la demande plus ancienne par la demande plus nouvelle.

Étape 4. Pki-server :

Accordez manuellement les demandes sur le serveur de PKI :

Pour visualiser la demande :

```
show crypto pki server SUBCA requests
```

Pour accorder une demande spécifique ou toutes les demandes :

```
crypto pki server SUBCA grant <id|all>
```

Étape 5. PKI-client :

En attendant, un client de PKI met en marche un temporisateur de BALAYAGE. Ici, l'IOS exécute GetCertInitial à intervalles réguliers jusqu'à ce que SCEP CertRep = AIT ACCORDÉ avec le certificat accordé soit reçu par le client.

Une fois le certificat accordé est reçu, IOS l'installe automatiquement.