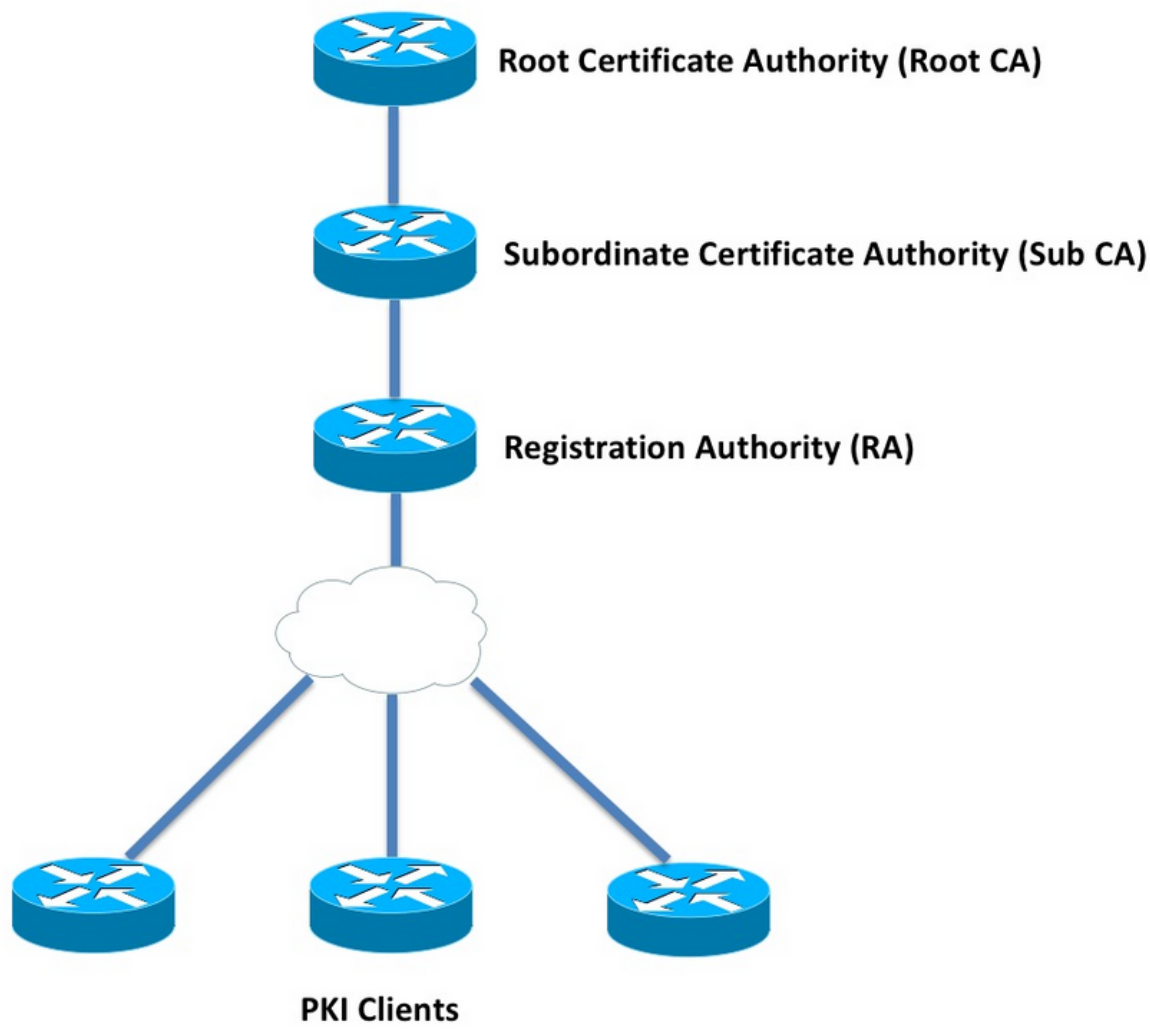
IOS PKI-Bereitstellungsleitfaden: Erstmaliges Design und erstmalige Bereitstellung

Inhalt

Einführung

In diesem Dokument werden IOS PKI Server- und Clientfunktionen detailliert beschrieben. Sie behandelt Überlegungen zum anfänglichen Design und zur Bereitstellung der IOS PKI.

PKI-Infrastruktur



Zertifizierungsstelle

Die Zertifizierungsstelle (Certificate Authority, CA), im gesamten Dokument auch als PKI-Server

bezeichnet, ist eine vertrauenswürdige Stelle, die Zertifikate ausgibt. PKI basiert auf Vertrauenswürdigkeit, und die Vertrauenshierarchie beginnt bei der Root Certificate Authority (Root-CA). Da die Root-CA an der Spitze der Hierarchie steht, verfügt sie über ein selbstsigniertes Zertifikat.

Nachrangige Zertifizierungsstelle

In der PKI-Vertrauenshierarchie werden alle Zertifikatsbehörden unterhalb des Stammes als untergeordnete Zertifizierungsstellen (Sub-CA) bezeichnet. Offensichtlich wird ein Zertifikat der Unterzertifizierungsstelle von der Zertifizierungsstelle ausgestellt, das eine Stufe über dem Wert liegt.

PKI legt keine Beschränkung für die Anzahl der Sub-CAs in einer bestimmten Hierarchie fest. In einem Unternehmen mit mehr als drei Zertifizierungsebenen können die Behörden jedoch schwer zu verwalten sein.

Registrierungsstelle

PKI definiert eine spezielle Zertifizierungsstelle, die so genannte Registrierungs Authority (RA), die für die Autorisierung der PKI-Clients für eine bestimmte Sub-CA oder Root-CA zuständig ist. RA gibt keine Zertifikate für PKI-Clients aus, sondern entscheidet, welche PKI-Client von der Sub-CA oder der Root-CA ausgestellt werden kann oder nicht.

Die Hauptrolle eines RA besteht darin, eine grundlegende Validierung von Client-Zertifikatsanforderungen von der Zertifizierungsstelle zu entlasten und die Zertifizierungsstelle vor direktem Kontakt mit den Clients zu schützen. Auf diese Weise steht RA zwischen den PKI-Clients und der CA und schützt so die CA vor jeglichen Denial-of-Service-Angriffen.

PKI-Client

Jedes Gerät, das ein Zertifikat anfordert, das auf einem öffentlichen/privaten Schlüsselpaar basiert, um seine Identität anderen Geräten nachzuweisen, wird als PKI-Client bezeichnet.

Ein PKI-Client muss in der Lage sein, ein öffentlich-privates Schlüsselpaar wie RSA oder DSA oder ECDSA zu generieren oder zu speichern.

Ein Zertifikat ist ein Nachweis für die Identität und Gültigkeit eines bestimmten öffentlichen Schlüssels, sofern der entsprechende private Schlüssel auf dem Gerät vorhanden ist.

IOS PKI-Server

Tabelle 1: IOS PKI Server Feature-Evolution

Funktion	IOS [ISR-G1, ISR-G2]	IOS-XE [ASR1K, ISR4K]
IOS CA/PKI-Server	12,3(4)T	XE 3.14.0 / 15.5(1)S
IOS PKI-ServerzertifikatRollover	12,4(1)T	XE 3.14.0 / 15.5(1)S
IOS PKI HA	15,0(1)M	NA [Implicit Inter-RP Redundancy is

		available]
IOS RA für CA von 3 Drittanbietern	15,1(3)T	XE 3.14.0 / 15.5(1)S

Bevor der Administrator in die PKI-Serverkonfiguration eintritt, muss er diese Kernkonzepte verstehen.

Ursprüngliche Zeitquelle

Eine der Grundlagen der PKI-Infrastruktur ist Time. Die Systemuhr definiert, ob ein Zertifikat gültig ist oder nicht. In IOS muss die Uhr daher autoritär oder vertrauenswürdig gemacht werden. Ohne eine autoritative Zeitquelle funktioniert der PKI-Server möglicherweise nicht wie erwartet. Es wird dringend empfohlen, die IOS-Uhr mithilfe der folgenden Methoden zu authentifizieren:

NTP (Network Time Protocol)

Das Synchronisieren der Systemuhr mit einem Time Server ist die einzige Möglichkeit, die Systemuhr vertrauenswürdig zu machen. Ein IOS-Router kann als NTP-Client für einen bekannten und stabilen NTP-Server im Netzwerk konfiguriert werden:

```
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
```

IOS kann auch als NTP-Server konfiguriert werden, der die lokale Systemuhr als autoritär kennzeichnet. In einer kleinen PKI-Bereitstellung kann der PKI-Server als NTP-Server für seine PKI-Clients konfiguriert werden:

```
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
```

Markierung der Hardware-Uhr als vertrauenswürdig

In IOS kann die Hardware-Uhr wie folgt als autorativ gekennzeichnet werden:

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

Dies kann zusammen mit dem NTP konfiguriert werden. Der Hauptgrund dafür ist, dass die Systemuhr beim Neuladen eines Routers, z. B. aufgrund eines Stromausfalls, autoritär bleibt und die NTP-Server nicht erreichbar sind. In dieser Phase werden PKI-Timer nicht mehr funktionieren, was wiederum zu Fehlern bei der Erneuerung oder beim Rollover von Zertifikaten führt. **clock kalender-gültig** dient in solchen Situationen als Schutz.

Bei der Konfiguration ist es wichtig zu verstehen, dass die Systemuhr bei einem Absturz des Systemakkus nicht mehr synchronisiert wird und PKI anfangen wird, einer außer Synchronisation gesetzten Uhr zu vertrauen. Es ist jedoch relativ sicherer, dies zu konfigurieren, als überhaupt keine maßgebliche Zeitquelle zu haben.

> **Hinweis**: In IOS-XE Version XE 3.10.0/15.3(3)S wurde der **kalenvalide** Befehl clock hinzugefügt.

Hostname und Domänenname

Es wird empfohlen, einen Hostnamen und einen Domänennamen in Cisco IOS als einen der ersten Schritte zu konfigurieren, bevor PKI-bezogene Dienste konfiguriert werden. Der Router-Hostname und der Domänenname werden in den folgenden Szenarien verwendet:

- Der Standard-RSA-Schlüsselpaarname wird durch Kombination von Hostname und Domänenname abgeleitet.
- Bei der Registrierung für ein Zertifikat besteht der Standard-Betreffname aus dem Hostnamen-Attribut und einem unstrukturierten Namen, d. h. dem Hostnamen und dem Domänennamen.

Wie bei PKI Server werden Hostname und Domänenname nicht verwendet:

- Der Standardname für Schlüsselpaare entspricht dem des PKI-Servernamen.
- Der Standard-Betreffname besteht aus CN, die mit dem Namen des PKI-Servers übereinstimmt.

Generell wird empfohlen, einen geeigneten Hostnamen und einen Domänennamen zu konfigurieren.

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

HTTP-Server

IOS PKI Server ist nur aktiviert, wenn HTTP Server aktiviert ist. Es ist wichtig zu beachten, dass, wenn der PKI-Server aufgrund der Deaktivierung des HTTP-Servers deaktiviert ist, weiterhin Offline-Anfragen [über Terminal] vergeben werden können. Zum Verarbeiten von SCEP-Anforderungen und Senden von SCEP-Antworten ist eine HTTP-Serverfunktion erforderlich.

IOS-HTTP-Server wird aktiviert mit:

```
ip http server
```

Der Standard-HTTP-Serverport kann mithilfe der folgenden Elemente von 80 in eine beliebige gültige Portnummer geändert werden:

```
ip http port 8080
```

HTTP Max. Verbindung

Einer der Engpässe bei der Bereitstellung von IOS als PKI-Server mithilfe von SCEP ist die maximale Anzahl gleichzeitiger HTTP-Verbindungen und durchschnittliche HTTP-Verbindungen pro Minute.
Derzeit ist die maximale Anzahl gleichzeitiger Verbindungen auf einem IOS HTTP-Server standardmäßig auf 5 begrenzt und kann auf 16 erhöht werden. Dies wird in einer mittelgroßen Bereitstellung dringend empfohlen:

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

Diese IOS-Installationen ermöglichen die maximale Anzahl gleichzeitiger HTTP-Verbindungen bis zu 1.000:

- UniversalK9 IOS mit uck9-Lizenzsatz

Die CLI wird automatisch so geändert, dass ein numerisches Argument zwischen 1 und 1000 akzeptiert wird.

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

Der IOS-HTTP-Server ermöglicht 80 Verbindungen pro Minute [580 bei IOS-Versionen, bei denen die Anzahl der gleichzeitigen HTTP-Sitzungen auf 1000 erhöht werden kann]. Wenn dieses Limit innerhalb einer Minute erreicht wird, beginnt der IOS-HTTP-Listener, die eingehenden HTTP-Verbindungen zu drosseln, indem er den Listener für 15 Sekunden herunterfährt. Dies führt dazu, dass Client-Verbindungsanforderungen aufgrund des **erreichten Grenzwerts** der TCP-**Verbindungswarteschlange** verworfen werden. Weitere Informationen hierzu finden Sie hier

RSA-Schlüsselpaar

RSA-Schlüsselpaar für PKI-Server-Funktionen in IOS können automatisch generiert oder manuell generiert werden.
Beim Konfigurieren eines PKI-Servers erstellt IOS automatisch einen Trustpoint mit demselben Namen wie der PKI-Server, um das PKI-Server-Zertifikat zu speichern.

Manuelles Generieren des RSA-Schlüsselpaars des PKI-Servers:

Schritt 1: Erstellen Sie ein RSA-Schlüsselpaar mit demselben Namen wie der PKI-Server:

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

Schritt 2: Ändern Sie vor der Aktivierung des PKI-Servers den Trustpoint des PKI-Servers:

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

> **Hinweis**: Der unter dem PKI-Server-Trustpoint erwähnte Moduluswert für RSA-Schlüsselpaare wird erst berücksichtigt, wenn IOS ver 15.4(3)M4 ausgeführt wurde. Dies ist eine bekannte Einschränkung. Der Standard-Schlüsselmodulus ist 1024 Bit.

Automatische Generierung von PKI-Server-RSA-Schlüsselpaaren:

Bei der Aktivierung des PKI-Servers generiert IOS automatisch ein RSA-Schlüsselpaar mit dem gleichen Namen wie der PKI-Server. Die Modulusgröße des Schlüsselmoduls beträgt 1024 Bit.

Ab IOS Version 15.4(3)M5 erstellt diese Konfiguration ein RSA-Schlüsselpaar mit **<LABEL>**, da der Name und die Schlüsselstärke gemäß dem definierten **<MOD>**-Modul lautet.

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

Spoiler

Der CSCuu73408 IOS PKI-Server sollte eine nicht standardmäßige Schlüsselgröße für die Rollover-Zertifizierung zulassen.

Der CSCuu73408 IOS PKI-Server sollte eine nicht standardmäßige Schlüsselgröße für die Rollover-Zertifizierung zulassen.

Der aktuelle Industriestandard sieht vor, mindestens 2048 Bit RSA-Schlüsselpaar zu verwenden.

Berücksichtigung des Auto-Rollover-Timers

Derzeit generiert der IOS PKI Server standardmäßig kein Rollover-Zertifikat und muss explizit unter dem PKI-Server mithilfe des Befehls **auto-rollover <days-before-expare>** aktiviert werden. Weitere Informationen zum Certificate Rollover finden Sie unter

Mit diesem Befehl wird festgelegt, wie viele Tage vor Ablauf des PKI-Servers/CA-Zertifikats das IOS ein Rollover-CA-Zertifikat erstellen soll. Beachten Sie, dass das Rollover-CA-Zertifikat aktiviert wird, sobald das aktuelle aktive CA-Zertifikat abläuft. Der Standardwert ist derzeit 30 Tage. Dieser Wert sollte abhängig von der Lebensdauer des Zertifizierungsstellenzertifikats auf einen angemessenen Wert festgelegt werden. Dies wirkt sich wiederum auf die Konfiguration des Timers für die automatische Registrierung auf dem PKI-Client aus.

> **Hinweis**: Der automatische Rollover-Timer sollte immer vor dem automatischen Anmeldungs-Timer auf dem Client während des Rollover von CA- und Client-Zertifikaten ausgelöst werden [bekannt als].

CRL-Überlegungen

Die IOS-PKI-Infrastruktur unterstützt zwei Möglichkeiten zur Verteilung von CRL:

Veröffentlichen des CRL auf einem HTTP-Server

Der IOS-PKI-Server kann so konfiguriert werden, dass die CRL-Datei an einem bestimmten Speicherort auf einem HTTP-Server mithilfe dieses Befehls unter dem PKI-Server veröffentlicht wird:

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

Der PKI-Server kann so konfiguriert werden, dass dieser CRL-Standort mit diesem Befehl unter dem PKI-Server in alle PKI-Clientzertifikate eingebettet wird:

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

SCEP GetCRL-Methode

IOS PKI Server speichert die CRL-Datei automatisch am bestimmten Datenbankspeicherort, der standardmäßig nvram ist. Es wird dringend empfohlen, eine Kopie auf einem SCP/FTP/TFTP-Server unter Verwendung dieses Befehls unter dem PKI-Server zu speichern:

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

Standardmäßig bettet der IOS PKI-Server den CDP-Speicherort nicht in die PKI-Clientzertifikate ein. Wenn die IOS PKI-Clients für die Durchführung einer Widerrufsprüfung konfiguriert sind, aber das zu validierende Zertifikat kein CDP enthält und der validierende CA-Vertrauenspunkt mit dem CA-Standort konfiguriert ist (unter Verwendung von http://<CA-Server-IP oder FQDN>), wird IOS standardmäßig auf die SCEP-basierte GetCRL-Methode zurückgesetzt.
SCEP GetCRL führt einen CRL-Abruf durch, indem HTTP GET unter folgender URL ausgeführt

wird:

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

Hinweis: Drücken Sie in der IOS-CLI vor der Eingabe **?** die Tastenfolge **Strg + V**.

IOS PKI Server kann diese URL auch als CDP-Speicherort einbetten. Dies hat zwei Vorteile:

- Sie stellt sicher, dass alle nicht IOS SCEP-basierten PKI-Clients einen CRL-Abruf durchführen können.
- Ohne eingebettetes CDP werden IOS SCEP GetCRL-Anforderungsnachrichten signiert (mithilfe eines temporären, selbstsignierten Zertifikats), wie im SCEP-Entwurf definiert. CRL-Abrufanforderungen müssen jedoch nicht signiert werden. Durch Einbetten der CDP-URL für die GetCRL-Methode kann das Signieren der CRL-Anforderungen vermieden werden.

Lebensdauer von CRL

Die CRL-Lebensdauer des IOS PKI-Servers kann mithilfe dieses Befehls unter dem PKI-Server gesteuert werden:

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

Der Wert ist in Stunden angegeben. Standardmäßig ist die Lebensdauer des CRL auf 6 Stunden festgelegt. Je nachdem, wie häufig die Zertifikate widerrufen werden, erhöht die Anpassung der CRL-Lebensdauer auf einen optimalen Wert die CRL-Abrufleistung im Netzwerk.

Datenbanküberlegungen

Der IOS PKI-Server verwendet nvram als standardmäßigen Datenbankspeicherort. Es wird dringend empfohlen, einen FTP-, TFTP- oder SCP-Server als Datenbankspeicherort zu verwenden. Standardmäßig erstellt IOS PKI Server zwei Dateien:

- <Server-Name>.ser - Enthält die letzte Seriennummer, die von der CA in Hex ausgegeben wird. Die Datei hat ein Textformat und enthält folgende Informationen:
  db_version = 1
  last_serial = 0x4
- <Servername>.crl - Dies ist die DER-codierte CRL-Datei, die von der CA veröffentlicht wird.

Der IOS PKI Server speichert Informationen in der Datenbank auf drei konfigurierbaren Ebenen:

- Minimum - Dies ist die Standardstufe, und auf dieser Ebene wird keine Datei in der Datenbank

erstellt, und daher sind auf dem CA-Server keine Informationen zu den in der Vergangenheit erteilten Client-Zertifikaten verfügbar.
- Namen: Auf dieser Ebene erstellt der IOS-PKI-Server für jedes ausgegebene Client-Zertifikat eine Datei mit dem Namen <Serial-Number>.cnm, wobei der Name <Serial-Number> auf die Seriennummer des ausgestellten Client-Zertifikats verweist. Diese cnm-Datei enthält den Betreffnamen und das Ablaufdatum des Client-Zertifikats.
- Complete (Abgeschlossen) - Auf dieser Ebene erstellt der IOS PKI Server zwei Dateien für jedes ausgestellte Client-Zertifikat:
- <Seriennummer>.cnm
- <Seriennummer>.crt

Hier ist die Crt-Datei die Client-Zertifikatsdatei, die DER-kodiert ist.

Diese Punkte sind wichtig:

- Vor der Ausgabe eines Clientzertifikats bezieht sich der IOS PKI Server auf <Server-Name>.ser, um die Seriennummer des Zertifikats zu ermitteln und abzuleiten.
- Wenn die Datenbankebene auf "Namen" oder "Complete" festgelegt ist, müssen <Serial-Number>.cnm und <Serial-Number>.crt in die Datenbank geschrieben werden, bevor das erteilte/ausgestellte Zertifikat an den Client gesendet wird.
- Wenn die Datenbank-URL auf Names (Namen) oder Complete (Vollständig) festgelegt ist, muss die Datenbank-URL genügend Speicherplatz zum Speichern der Dateien haben. Daher wird empfohlen, einen externen Dateiserver [FTP, TFTP oder SCP] als Datenbank-URL zu konfigurieren.
- Wenn die URL für eine externe Datenbank konfiguriert ist, muss unbedingt sichergestellt werden, dass der Dateiserver während des Zertifikatsgenehmigungsprozesses erreichbar ist. Andernfalls wird der CA-Server als deaktiviert gekennzeichnet. Außerdem ist ein manueller Eingriff erforderlich, um den CA-Server wieder online zu stellen.

Datenbankarchiv

Bei der Bereitstellung eines PKI-Servers ist es wichtig, Fehlerszenarien zu berücksichtigen und bei einem Hardware-Fehler vorzubereiten. Dafür gibt es zwei Möglichkeiten:

1. Redundanz
   In diesem Fall fungieren zwei Geräte oder Verarbeitungseinheiten als Aktiv/Standby, um Redundanz zu gewährleisten.
   Die IOS PKI-Serververfügbarkeit kann mit zwei HSRP-fähigen ISR-Routern [ISR G1 und ISR G2] erreicht werden, wie in erläutert.

   IOS XE-basierte Systeme [ISR4K und ASR1k] verfügen nicht über eine Option zur Geräteredundanz. Im ASR1k-Inter-RP ist jedoch standardmäßig Redundanz verfügbar.
2. Archivieren von CA Server-Schlüsselpaaren und -Dateien
   IOS bietet die Möglichkeit, das PKI-Server-Schlüsselpaar und das Zertifikat zu archivieren. Die Archivierung kann mit zwei Dateitypen durchgeführt werden:
   PEM - IOS erstellt PEM-formatierte Dateien zum Speichern des öffentlichen RSA-Schlüssels, des verschlüsselten privaten RSA-Schlüssels und des Zertifizierungsstellen-Serverzertifikats. Schlüsselpaar und Zertifikate des Rollover-Schlüssels werden automatisch archiviertPKCS12 - IOS erstellt eine einzelne PKCS12-Datei, die das CA Server-Zertifikat und den

entsprechenden RSA Private Key enthält und mit einem Kennwort verschlüsselt wird.
Die Datenbankarchivierung kann mit diesem Befehl unter dem PKI-Server aktiviert werden:

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

Es ist auch möglich, die archivierten Dateien auf einem separaten Server zu speichern, möglicherweise mithilfe eines sicheren Protokolls (SCP), das den folgenden Befehl unter dem PKI-Server verwendet:

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

Von allen Dateien in der Datenbank, mit Ausnahme der archivierten Dateien und der Ser-Datei, sind alle anderen Dateien im Klartext und stellen bei Verlust keine echte Bedrohung dar. Sie können daher auf einem separaten Server gespeichert werden, ohne beim Schreiben der Dateien einen hohen Overhead zu verursachen, z. B. auf einem TFTP-Server.

IOS als Sub-CA

IOS PKI Server übernimmt standardmäßig die Rolle einer Root CA. Um einen untergeordneten PKI-Server (Sub-CA) zu konfigurieren, aktivieren Sie diesen Befehl zunächst im Konfigurationsabschnitt des PKI-Servers (bevor Sie den PKI-Server aktivieren):

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

Konfigurieren Sie auf diese Weise die URL der Root-CA unter dem Trustpoint des PKI-Servers:

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

Durch die Aktivierung dieses PKI-Servers werden jetzt folgende Ereignisse ausgelöst:

- Der PKI-Server-Trustpoint wird authentifiziert, um das Root-CA-Zertifikat zu installieren.
- Nachdem die Root-CA authentifiziert wurde, generiert IOS eine CSR für die untergeordnete CA [x509-Basiseinschränkung mit CA:TRUE-Flag] und sendet sie an die Root-CA.

Unabhängig vom für die Root-CA konfigurierten Gewährleistungsmodus setzt IOS die Zertifikatsanforderungen der CA (oder RA) in die ausstehende Warteschlange. Ein Administrator muss die Zertifizierungsstellenzertifikate manuell zuweisen.
So zeigen Sie die ausstehende Zertifikatsanforderung und die Anfrage-ID an:

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

So gewähren Sie die Anforderung:

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

- Auf diese Weise lädt der nachfolgende SCEP POLL-Vorgang (GetCertInitial) das SubCA-Zertifikat herunter und installiert es auf dem Router, wodurch der untergeordnete PKI-Server aktiviert wird.

IOS als RA

Der IOs-PKI-Server kann für eine bestimmte untergeordnete oder Stammzertifizierungsstelle als Registrierungsstelle konfiguriert werden. Um den PKI-Server als Registrierungsstelle zu konfigurieren, aktivieren Sie diesen Befehl zunächst im Konfigurationsabschnitt des PKI-Servers (bevor Sie den PKI-Server aktivieren):

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

Konfigurieren Sie anschließend die CA-URL unter dem Trustpoint des PKI-Servers. Dies zeigt an, welche CA durch das RA geschützt ist:

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

Eine Registrierungsstelle stellt keine Zertifikate aus, daher ist die Konfiguration des **Emittenten-Namens** im Rahmen der RA nicht erforderlich und auch bei einer Konfiguration nicht wirksam. Der Betreffname eines RA wird mithilfe des Befehls **subject-name** unter dem RA Trustpoint konfiguriert. Es ist wichtig, **OU = ioscs RA** als Teil des Betreffnamens zu konfigurieren, damit die IOS-CA die IOS-RA identifizieren kann, d. h. um die von der IOS-RA autorisierten Zertifikatsanforderungen zu identifizieren.

IOS kann als Registrierungsstelle für CAs von Drittanbietern wie Microsoft CA fungieren. Um Kompatibilität zu gewährleisten, muss IOS RA mithilfe dieses Befehls im Konfigurationsabschnitt des PKI-Servers aktiviert werden (bevor der PKI-Server aktiviert wird):

```
mode ra transparent
```

Im Standard-RA-Modus signiert IOS die Clientanforderungen [PKCS#10] mit dem RA-Zertifikat. Dieser Vorgang weist den IOS PKI-Server darauf hin, dass die Zertifikatsanforderung von einem RA autorisiert wurde.

Im transparenten RA-Modus leitet IOS Client-Anfragen in ihrem ursprünglichen Format weiter, ohne das RA-Zertifikat einzuführen. Dies ist ein bekanntes Beispiel für die Kompatibilität mit Microsoft CA.

IOS PKI-Client

Eine der wichtigsten Konfigurationskomponenten im IOS PKI-Client ist ein Trustpoint. Die Konfigurationsparameter für Trustpoints werden in diesem Abschnitt ausführlich erläutert.

Ursprüngliche Zeitquelle

Wie bereits erwähnt, ist eine autoritative Zeitquelle auch auf dem PKI-Client erforderlich. Der IOS-PKI-Client kann mithilfe der folgenden Konfiguration als NTP-Client konfiguriert werden:

```
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
```

Hostname und Domänenname

Eine allgemeine Empfehlung besteht darin, einen Hostnamen und einen Domänennamen auf dem Router zu konfigurieren:

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

RSA-Schlüsselpaar

Im IOS-PKI-Client kann das RSA-Schlüsselpaar für eine bestimmte Vertrauenspunktregistrierung entweder automatisch generiert oder manuell generiert werden.

Die automatische RSA-Schlüsselgenerierung umfasst Folgendes:

- IOS erstellt standardmäßig 512-Bit-RSA-Schlüsselpaar
- Der automatisch generierte Schlüsselpaarname lautet hostname.domain-name, d. h. der Geräte-Hostname in Verbindung mit dem Gerätenamen.
- Das automatisch generierte Schlüsselpaar ist nicht als exportfähig markiert.

Die automatische RSA-Schlüsselgenerierung umfasst Folgendes:

- Optional kann ein RSA-Schlüsselpaar einer geeigneten Stärke manuell erstellt werden, indem:
- `crypto key generate rsa general-keys label <LABEL> modulus < MOD> [exportable]`
  Hier LABEL - der Name des RSA-Schlüsselpaars
  MOD - RSA-Schlüsselmodul oder Stärke in Bits zwischen 360 und 4096, das traditionell 512, 1024, 2048 oder 4096 ist.
  Der Vorteil der manuellen Generierung des RSA-Schlüsselpaars besteht in der Möglichkeit, das Schlüsselpaar als exportfähig zu kennzeichnen. Dadurch kann das Identitätszertifikat vollständig exportiert und dann auf einem anderen Gerät wiederhergestellt werden. Man sollte jedoch die Sicherheitsauswirkungen dieser Aktion verstehen.
- Ein RSA-Schlüsselpaar ist vor der Registrierung mit diesem Befehl mit einem Vertrauenspunkt verknüpft.

```
crypto pki trustpoint MGMT
rsakeypair <LABEL> [<MOD> <MOD>]
```

Wenn hier bereits ein RSA-Schlüsselpaar mit dem Namen **<**LABEL**>** vorhanden ist, wird es bei der Anmeldung für Trustpoint übernommen.

Wenn kein RSA-Schlüsselpaar mit dem Namen <LABEL> vorhanden ist, wird während der Registrierung eine der folgenden Aktionen ausgeführt:

- Wenn kein <MOD>-Argument übergeben wird, wird ein 512-Bit-Schlüsselpaar mit dem Namen <LABEL> generiert.
- Wenn ein <MOD>-Argument übergeben wird, wird ein <MOD>-Bit-Schlüsselpaar für allgemeine Zwecke mit dem Namen <LABEL> generiert.
- Wenn zwei <MOD>-Argumente übergeben werden, wird ein <MOD>-Bit-Signaturschlüsselpaar und ein <MOD>-Bit-Verschlüsselungsschlüsselpaar generiert, beide mit dem Namen <LABEL>

Trustpoint

Ein Trustpoint ist ein abstrakter Container, in dem ein Zertifikat in IOS gespeichert wird. Ein einzelner Trustpoint kann jeweils zwei aktive Zertifikate speichern:

- Zertifizierungsstellenzertifikat - Das Laden eines Zertifizierungsstellenzertifikats in einen bestimmten Vertrauenspunkt wird als Trustpoint-Authentifizierungsprozess bezeichnet.
- Ein von der Zertifizierungsstelle ausgestelltes ID-Zertifikat - Laden oder Importieren eines ID-Zertifikats in einen bestimmten Vertrauenspunkt wird als Anmeldeprozess für Trustpoint bezeichnet.

Eine TrustPoint-Konfiguration wird als Vertrauensrichtlinie bezeichnet. Dies definiert Folgendes:

- Welches Zertifizierungsstellenzertifikat wird in den Vertrauenspunkt geladen?
- Für welche CA ist die Trustpoint-Anmeldung erforderlich?
- Wie registriert das IOS den Trustpoint?
- Wie wird ein von der jeweiligen Zertifizierungsstelle ausgestelltes [in den Trustpoint geladenes] Zertifikat validiert?

Die Hauptkomponenten eines Trustpoints werden hier erläutert.

Registrierungsmodus

Ein TrustPoint-Registrierungs-Modus, der auch den Trustpoint-Authentifizierungsmodus definiert, kann mithilfe von drei Hauptmethoden durchgeführt werden:

1. Terminalregistrierung: Manuelle Methode zur Durchführung der Trustpoint-Authentifizierung und Zertifikatregistrierung mithilfe von Copy-Paste im CLI-Terminal.
2. SCEP-Anmeldung - Trustpoint-Authentifizierung und -Registrierung mit SCEP über HTTP.
3. Registrierungsprofil - Hier werden Authentifizierungs- und Registrierungsmethoden separat definiert. Neben den Terminal- und SCEP-Anmeldemethoden bieten die Registrierungsprofile eine Option zum Angeben von HTTP/TFTP-Befehlen zum Abrufen von Dateien vom Server, die mithilfe einer Authentifizierungs- oder Anmeldungs-URL unter dem Profil definiert wird.

Quellschnittstelle und VRF

Die Trustpoint-Authentifizierung und -Registrierung über HTTP (SCEP) oder TFTP (Enrollment Profile) verwendet das IOS-Dateisystem, um Datei-i/o-Vorgänge auszuführen. Der Austausch dieser Pakete kann von einer bestimmten Quellschnittstelle und einer VRF-Instanz erfolgen.

Bei einer klassischen TrustPoint-Konfiguration wird diese Funktion mithilfe der **Quellschnittstelle** und der **VRF-**Unterbefehle unter dem Trustpoint aktiviert.

Bei Registrierungsprofilen, **Quellschnittstelle** und **Registrierung | authentication url <tftp://Server-location> vrf <vrf-name>**-Befehle verfügen über die gleiche Funktionalität.

Beispielkonfiguration:

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

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

oder

```
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
```

Automatische Registrierung und Verlängerung von Zertifikaten

Der IOS PKI-Client kann so konfiguriert werden, dass er die automatische Registrierung und Verlängerung mithilfe dieses Befehls im Abschnitt "PKI Trustpoint" durchführt:

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

Hier gibt der Befehl **auto-enroll <percentage> [regenerate]** an, dass das IOS eine Zertifikatsverlängerung mit genau 80 % der Lebensdauer des aktuellen Zertifikats durchführen soll.

Das Schlüsselwort **regenerate** gibt an, dass IOS das RSA-Schlüsselpaar, das Schattenschlüsselpaar, bei jedem Verlängerungsvorgang für Zertifikate neu generieren soll.

Dies ist das automatische Anmeldeverhalten:

- Wenn **die automatische Anmeldung** konfiguriert ist und der Vertrauenspunkt authentifiziert wird, führt IOS eine automatische Registrierung für den Server durch, der sich unter der URL befindet, die als Teil des Befehls **enrollment url** im PKI-Trustpoint-Abschnitt oder im Registrierungsprofil erwähnt wird.
- Wenn ein Trustpoint bei einem PKI-Server oder einer CA angemeldet ist, wird auf dem PKI-Client ein RENEW- oder ein SHADOW-Timer initialisiert, der auf dem unter dem Trustpoint installierten **automatischen** Registrierungsprozentsatz des aktuellen Identitätszertifikats basiert. Dieser Timer wird unter **show crypto pki timer** command angezeigt. Weitere Informationen zu den Timer-Funktionen *finden Sie unter*
- Die Unterstützung für die Verlängerungsfunktion wird vom PKI-Server bereitgestellt. Mehr dazu in

Der IOS-PKI-Client führt zwei Verlängerungstypen aus:
Implizite Verlängerung: Wenn der PKI-Server keine "Verlängerung" als unterstützte Funktion sendet, führt IOS eine Erstregistrierung mit dem festgelegten Prozentsatz für die automatische Registrierung durch. Das heißt, IOS verwendet ein selbstsigniertes Zertifikat, um die Verlängerungsanfrage zu unterzeichnen.Explizite Verlängerung: Wenn der PKI-Server die PKI-Clientzertifikatverlängerungsfunktion unterstützt, kündigt er "Renewal" als unterstützte Funktion an. IOS berücksichtigt diese Funktion bei der Zertifikatsverlängerung, d. h. IOS verwendet das aktuelle aktive Identitätszertifikat, um die Zertifikatsverlängerung zu unterzeichnen.

Bei der Konfiguration des Prozentsatzes für die automatische Anmeldung ist darauf zu achten. Wenn bei einem bestimmten PKI-Client in der Bereitstellung eine Bedingung auftritt, bei der das Identitätszertifikat gleichzeitig mit dem ausstellenden Zertifizierungsstellenzertifikat abläuft, sollte der Wert für die automatische Registrierung immer den [Schatten] Verlängerungsvorgang auslösen, nachdem die Zertifizierungsstelle das Rollover-Zertifikat erstellt hat. *Weitere Informationen finden Sie* im Abschnitt **Abhängigkeiten** des PKI-Timers

Zertifikat-Widerrufsüberprüfung

Ein authentifizierter PKI-Trustpoint, d. h. ein PKI-Trustpoint, der ein CA-Zertifikat enthält, kann während einer IKE- oder SSL-Aushandlung eine Zertifikatsvalidierung durchführen, bei der das Peer-Zertifikat einer gründlichen Zertifikatsvalidierung unterzogen wird. Eine der Validierungsmethoden besteht darin, den Widerrufsstatus von Peer-Zertifikaten mithilfe einer der folgenden beiden Methoden zu überprüfen:

- Certificate Revocation List (CRL) - Dies ist eine Datei mit den Seriennummern der Zertifikate, die von einer bestimmten Zertifizierungsstelle widerrufen werden. Diese Datei wird mit dem Zertifikat der ausstellenden Zertifizierungsstelle signiert. Bei der CRL-Methode wird die CRL-Datei mithilfe von HTTP oder LDAP heruntergeladen.

- Online Certificate Status Protocol (OCSP) - IOS stellt einen Kommunikationskanal mit einer als OCSP Responder bezeichneten Einheit her, die von der ausstellenden Zertifizierungsstelle als Server bezeichnet wird. Ein Client wie IOS sendet eine Anfrage mit der Seriennummer des Zertifikats, die validiert wird. Der OCSP Responder antwortet mit dem Widerrufsstatus der angegebenen Seriennummer. Der Kommunikationskanal kann mit jedem unterstützten Anwendungs-/Transportprotokoll eingerichtet werden, das normalerweise HTTP ist.

Die Revocation Check kann mithilfe des folgenden Befehls im Abschnitt PKI Trustpoint definiert werden:

```
crypto pki trustpoint MGMT
 revocation-check crl ocsp none
```

Standardmäßig wird ein Trustpoint so konfiguriert, dass er eine Widerrufsprüfung mithilfe von crl durchführt.

Die Methoden können erneut bestellt werden, und die Überprüfung des Widerrufsstatus wird in der definierten Reihenfolge durchgeführt. Die Methode "none" umgeht die Widerrufsprüfung.

CRL-Cache

Bei einer CRL-basierten Widerrufsprüfung kann jede Zertifikatsvalidierung einen neuen Download der CRL-Datei auslösen. Wenn die CRL-Datei größer wird oder der CRL Distribution Point (CDP) weiter entfernt ist, wird durch das Herunterladen der Datei während jedes Validierungsprozesses die Leistung des Protokolls je nach Zertifikatsvalidierung beeinträchtigt. Daher wird das CRL-Caching durchgeführt, um die Leistung zu verbessern, und das Caching der CRL berücksichtigt die CRL-Validität.

Die CRL-Validität wird mithilfe von zwei Zeitparametern definiert: **LastUpdate**, das ist das letzte Mal, dass die CRL von der herausgebenden CA veröffentlicht wurde, und **NextUpdate**, das die Zukunft ist, wenn eine neue Version der CRL-Datei von der herausgebenden CA veröffentlicht wird.

IOS speichert alle heruntergeladenen CRL, solange die CRL gültig ist. Unter bestimmten Umständen, z. B. wenn das CDP nicht zeitweilig erreichbar ist, kann es erforderlich sein, das CRL für einen längeren Zeitraum im Cache zu belassen. In IOS kann ein zwischengespeichertes CRL 24 Stunden nach Ablauf der CRL-Gültigkeit beibehalten werden. Dies kann mithilfe dieses Befehls im Abschnitt zu PKI-Vertrauenspunkten konfiguriert werden:

```
crypto pki trustpoint MGMT
 crl cache extend <0 - 1440>
!! here the value is in minutes
```

Unter bestimmten Umständen, wie z. B. der Ausstellung von Zertifikaten zur Widerrufung einer Zertifizierungsstelle innerhalb der Gültigkeitsdauer der CRL, kann IOS so konfiguriert werden, dass der Cache häufiger gelöscht wird. Durch vorzeitiges Löschen der CRL-Datei ist IOS gezwungen, die CRL häufiger herunterzuladen, um den CRL-Cache auf dem neuesten Stand zu halten. Diese Konfigurationsoption ist im PKI-Trustpoint-Abschnitt verfügbar:

```
crypto pki trustpoint MGMT
 crl cache delete-after <1-43200>
!! here the value is in minutes
```

Schließlich kann IOS so konfiguriert werden, dass die CRL-Datei nicht mithilfe dieses Befehls im Abschnitt "PKI Trustpoint" zwischengespeichert wird:

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

Empfohlene Konfiguration

Nachfolgend finden Sie eine typische CA-Bereitstellung mit Root-CA und einer Sub-CA-Konfiguration. Das Beispiel enthält auch eine Sub-CA-Konfiguration, die durch ein RA geschützt ist.

Bei einem RSA-Schlüsselpaar mit 2048 Bit wird in diesem Beispiel eine Konfiguration empfohlen, bei der Folgendes gilt:
Root-CA hat eine Lebensdauer von 8 Jahren
Sub-CA hat eine Lebensdauer von 3 Jahren
Client-Zertifikate werden für ein Jahr ausgestellt, die automatisch für die Anforderung einer Zertifikatserneuerung konfiguriert werden.

ROOT CA - Konfiguration

```
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 ohne RA - Konfiguration

```
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 mit RA - Konfiguration

```
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 für SUBCA - Konfiguration

```
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
```

Zertifikatsregistrierung

Manuelle Registrierung

Die manuelle Registrierung umfasst die Offline-CSR-Generierung auf dem PKI-Client, die manuell in die CA kopiert wird. Der Administrator signiert die Anforderung manuell, die dann in den Client importiert wird.

PKI-Client

PKI-Client-Konfiguration:

```
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
```

Schritt 1: Authentifizieren Sie zunächst den Trustpoint (dies kann auch nach Schritt 2 durchgeführt werden).

```
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
```

Schritt 2: Erstellen Sie eine Zertifikatssignierungsanfrage, und reichen Sie die CSR-Anfrage an die Zertifizierungsstelle ein, um das erteilte Zertifikat zu erhalten:

```
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
```

Schritt 3: Importieren Sie jetzt das erteilte Zertifikat über 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
```

PKI-Server

Schritt 1: Exportieren Sie zunächst das Zertifikat der ausstellenden Zertifizierungsstelle von der Zertifizierungsstelle, die in diesem Fall das Zertifikat SUBCA ist. Dieser wird während Schritt 1 oben auf den PKI-Client importiert, d. h. die Trustpoint-Authentifizierung.

```
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-----
```

Schritt 2: Nach Schritt 2 auf dem PKI-Client müssen Sie den CSR vom Client übernehmen und diesen für die Signierung auf dem SUBCA mit dem folgenden Befehl bereitstellen:

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

Dieser Befehl legt nahe, dass die SUBCA eine Zertifikatssignierungsanfrage vom Terminal akzeptiert und die Zertifikatsdaten nach der Erteilung im PEM-Format gedruckt werden.

```
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
```

Wenn sich die Zertifizierungsstelle im Modus für die automatische Vergabe befindet, wird das erteilte Zertifikat oben im PEM-Format angezeigt. Wenn sich die Zertifizierungsstelle im manuellen Zuweisungsmodus befindet, wird die Zertifikatsanforderung als **ausstehend** markiert, ein ID-Wert zugewiesen und in die Warteschlange für Registrierungsanfragen gestellt.

```
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
```

Schritt 3: Erteilen Sie diese Anforderung manuell mit dem folgenden Befehl:

```
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-----
```

Hinweis: Die manuelle Registrierung einer Sub-CA bei einer Root-CA ist nicht möglich.

Hinweis: Eine CA in einem deaktivierten Zustand aufgrund eines deaktivierten HTTP-Servers kann die Zertifikatsanforderungen manuell zuweisen.

Registrierung mit SCEP

PKI-Client-Konfiguration:

```
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
```

PKI-Serverkonfiguration lautet:

```
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/
```

Der Standardmodus für die Zertifikatsanforderungszuweisung ist manuell:

```
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
```

Manueller Zuschuss

Schritt 1: PKI-Client: Als ersten Schritt, der obligatorisch ist, authentifizieren Sie den Vertrauenspunkt auf dem PKI-Client:

```
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.
```

Schritt 2: PKI-Client: Nach der Trustpoint-Authentifizierung kann der PKI-Client für ein Zertifikat registriert werden.

> **Hinweis**: Wenn die automatische Anmeldung konfiguriert ist, führt der Client die Registrierung automatisch durch.

```
config terminal
```

```
crypto pki enroll MGMT
```

Hinter den Kulissen finden diese Veranstaltungen statt:

- IOS sucht nach einem RSA-Schlüsselpaar mit dem Namen PKI-Key. Wenn es vorhanden ist, wird es abgeholt, um ein Identitätszertifikat anzufordern. Andernfalls erstellt IOS ein 2048-Bit-Schlüsselpaar mit dem Namen PKI-Key und fordert dann ein Identitätszertifikat an.

- IOS erstellt eine Zertifikatssignierungsanfrage im PKCS10-Format.

- IOS verschlüsselt diesen CSR dann mithilfe eines zufälligen symmetrischen Schlüssels. Der zufällige symmetrische Schlüssel wird mit dem öffentlichen Schlüssel des Empfängers verschlüsselt, der der SUBCA ist (der öffentliche Schlüssel des SUBCA ist aufgrund der Trustpoint-Authentifizierung verfügbar). Der verschlüsselte CSR, der verschlüsselte, zufällige symmetrische Schlüssel und die Empfängerinformationen werden in umhüllten PKCS#7-Daten zusammengefasst.

- Diese PKCS#7-umhüllten Daten werden während der Erstregistrierung mit einem temporären, selbstsignierten Zertifikat signiert. Die PKCS#7-umhüllten Daten, das vom Client verwendete Signaturzertifikat und die Signatur des Clients werden in einem PKCS#7-signierten Datenpaket zusammengestellt. Dies ist Base64-kodiert und dann URL-kodiert. Der resultierende Datenblock wird als "Message"-Argument in HTTP URI gesendet, die an die CA gesendet wird:

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

Schritt 3: PKI-Server:

Wenn der IOS PKI-Server die Anforderung empfängt, prüft er folgende Punkte:

1. Überprüft, ob die Datenbank für die Registrierungsanforderung eine Zertifikatsanforderung mit derselben Transaktions-ID enthält, die der neuen Anforderung zugeordnet ist.

> **Hinweis**: Eine Transaktions-ID ist ein MD5-Hash des öffentlichen Schlüssels, für den vom Client ein Identitätszertifikat angefordert wird.

2. Überprüft, ob die Datenbank für die Registrierungsanfrage eine Zertifikatsanforderung mit demselben herausfordernden Kennwort enthält wie das vom Client gesendete.

> **Hinweis**: Wenn (1) true oder beide (1) und (2) zusammen true zurückgibt, kann ein CA-Server die Anforderung aufgrund einer doppelten Identitätsanforderung ablehnen. In einem solchen Fall ersetzt jedoch der IOS PKI-Server die ältere Anforderung durch die neuere Anforderung.

Schritt 4: PKI-Server:

Gewähren Sie die Anforderungen auf dem PKI-Server manuell:

So zeigen Sie die Anforderung an:

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

So erteilen Sie einen bestimmten Antrag oder alle Anträge:

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

Schritt 5: PKI-Client:

In der Zwischenzeit startet ein PKI-Client einen POLL-Timer. Hier führt IOS GetCertInitial in regelmäßigen Abständen durch, bis SCEP CertRep = GRANTED zusammen mit dem erteilten Zertifikat vom Client empfangen wird.

Nach Erhalt des erteilten Zertifikats wird es vom IOS automatisch installiert.