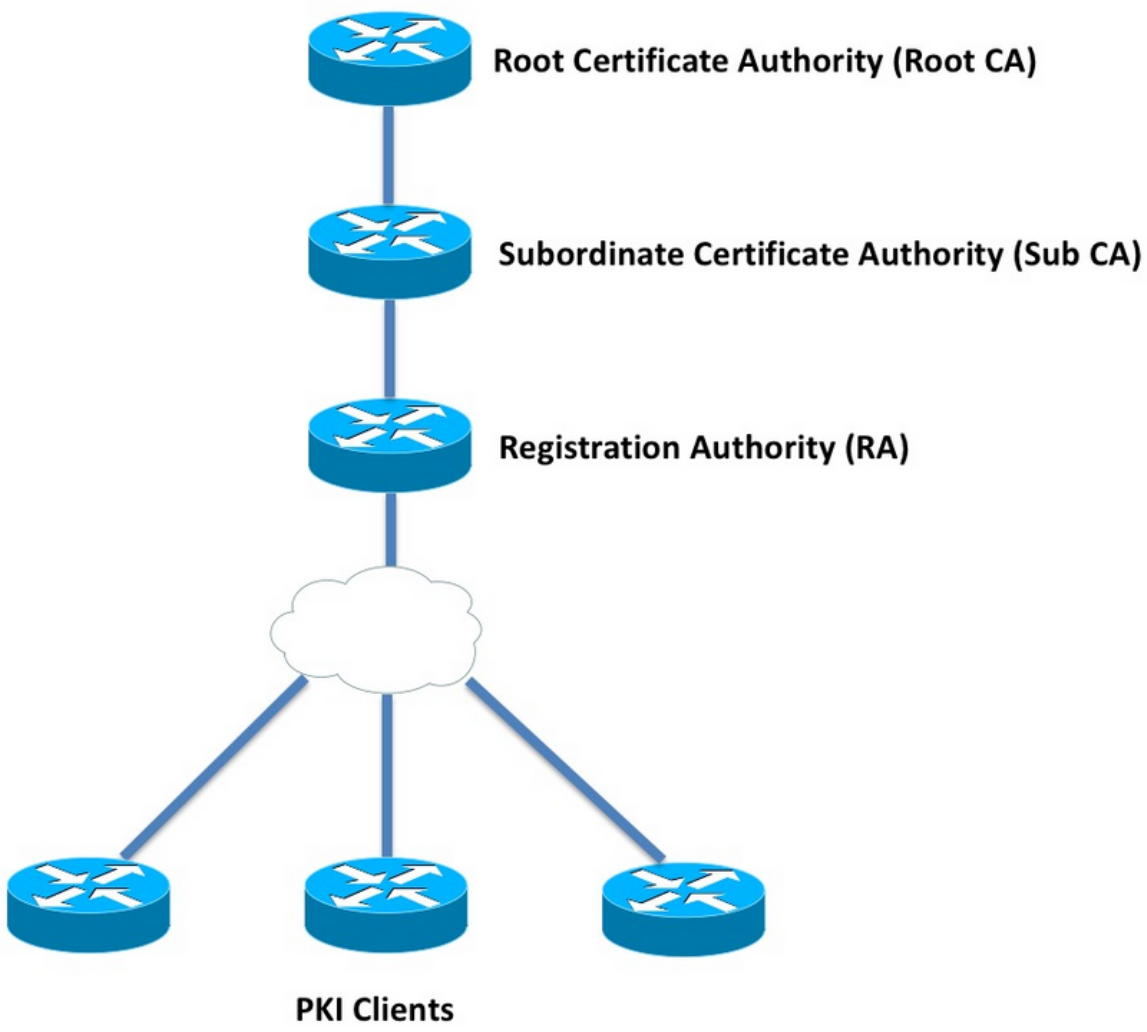
IOS PKI 導入ガイド：初期設計と展開』を参照してください。

内容

概要

このドキュメントは、IOS PKI サーバおよびクライアントの機能について詳しく説明します。 また、IOS PKI の初期設計と導入に関する考慮事項も扱います。

PKI インフラストラクチャ



認証局

認証局（CA）（このドキュメントでは PKI サーバとも呼んでいます）は、証明書を発行する信頼

できるエンティティです。PKI は信頼に基づいており、信頼階層はルート認証局（ルート CA）から始まります。 ルート CA は階層の最上部にあたるため、自己署名証明書があります。

下位認証局

PKI 信頼階層では、ルートより下の認証局はすべて、下位認証局（下位 CA）と呼ばれます。 明らかに、下位 CA 証明書は 1 つ上のレベルの CA によって発行されます。

PKI は、特定の階層での下位 CA の数に制限を課しません。ただし、認証局の層の数が 3 層より多くなるエンタープライズ展開では、管理が難しくなる可能性があります。

登録局

PKI は、特定の下位 CA またはルート CA に登録するために PKI クライアントを認証する登録局（RA）と呼ばれる特別な認証局を定義します。RA は証明書を PKI クライアントに発行しませんが、代わりにその PKI クライアントが下位 CA またはルート CA によって証明書を発行されるかどうかを決定します。

RA の主な役割は、基本的なクライアント証明書要求の検証を CA が行わなくて済むようにし、CA がクライアントと直接通信しなくて済むように保護することにあります。このように、RA は PKI クライアントと CA の間に立って、あらゆる種類の Denial of Service（DoS）攻撃から CA を保護します。

PKI クライアント

他のデバイスにアイデンティティを証明するために、常駐の公開キーと秘密キーのペアに基づいて証明書を要求するデバイスは、PKI クライアントと呼ばれます。

PKI クライアントは、RSA、DSA、ECDSA のような公開キーと秘密キーのペアを生成するか、保管できる必要があります。

証明書は、対応する秘密キーがデバイスに存在する場合に、特定の公開キーのアイデンティティや有効性を証明するものです。

IOS PKI サーバ

表 1IOS PKI サーバ機能の進化

機能	IOS [ISR-G1、ISR-G2]	IOS-XE [ASR1K、ISR4K]
IOS CA/PKI サーバ	12.3(4)T	XE 3.14.0/15.5(1)S
IOS PKI サーバ証明書のロールオーバー	12.4(1)T	XE 3.14.0/15.5(1)S
IOS PKI HA	15.0(1)M	NA（暗示的な相互 RP 冗長性が利用可能）
サード パーティ CA の IOS RA	15.1(3)T	XE 3.14.0/15.5(1)S

PKI サーバの設定について考慮する前に、管理者は以下の中心概念について理解しておく必要があります。

正規の時刻源

PKI インフラストラクチャの基盤を成す構成要素の 1 つは、時間です。システム クロックは、証明書が有効であるかどうかを定義します。したがって、IOS ではクロックを正式な信頼できるものとして設定する必要があります。正規の時刻源がなければ、PKI サーバは期待どおりに機能しない可能性があります。それで、以下の方法を使用して、IOS 上のクロックを信頼できるものにすることを強くお勧めします。

NTP (ネットワーク タイム プロトコル)

タイム サーバとシステム クロックを同期する方法が、システム クロックを信頼できるものとする唯一の方法です。ネットワーク内のよく知られた安定している NTP サーバの NTP クライアントとして IOS ルータを設定できます。

```
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 を NTP サーバとして設定することもできます。このように設定すると、ローカル システム クロックは正規としてマークが付けられます。小規模の PKI 展開では、PKI サーバを PKI クライアントの NTP サーバとして設定することができます。

```
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
```

信頼できるものとしてのハードウェア クロックのマーク付け

IOS では、以下のコマンドを使用して、ハードウェア クロックに正規としてマークを付けることができます。

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

ハードウェア クロックは NTP と連動するように設定することができます。このように設定する主な理由は、停電などのためにルータがリロードされる場合や、NTP サーバに到達できない場合に、システム クロックを信頼できるものとして保持するためです。この段階では、PKI タイマーは機能を停止し、その結果として、証明書の更新/ロールオーバーは失敗してしまいます。**clock calendar-valid は、そのような状況での安全策として機能します。**

この設定を行う際に、システム バッテリが切れると、システム クロックは同期されなくなり、PKI は同期されていないクロックを信頼し始めることを理解しておく必要があります。しかし、正規の時刻源がまったくない状況と比べると、これを設定しておく方が安全です。

> **注：clock calendar-valid コマンドは IOS XE バージョン XE 3.10.0/15.3(3)S 以降で追加されました。**

ホスト名とドメイン名

PKI 関連のサービスを設定する前に、最初のステップの 1 つとして、Cisco IOS のホスト名とドメイン名を設定することをお勧めします。ルータのホスト名とドメイン名は、以下のシナリオで使用されます。

- デフォルトの RSA キー ペアの名前は、ホスト名とドメイン名を組み合わせたものになります。
- 証明書を登録すると、デフォルトのサブジェクト名は、ホスト名の属性と非構造化名（ホスト名とドメイン名を 1 つにした名前）で構成されます。

PKI サーバには、ホスト名とドメイン名は使用されません。

- デフォルトのキー ペアの名前は、PKI サーバ名の名前と同じです。
- デフォルトのサブジェクト名は、CN で構成されます。これは、PKI サーバ名と同じです。

一般的な推奨事項として、適切なホスト名とドメイン名を設定するようにお勧めします。

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

HTTP Server

IOS PKI サーバは、HTTP サーバが有効な場合にのみ、有効になります。HTTP サーバが無効になったため、PKI サーバが無効になる場合、オフライン要求を許可し続ける（端末経由）場合があることに注意してください。SCEP 要求を処理したり、SCEP 応答を送信したりするには、HTTP サーバ機能が必要になります。

次のコマンドを使用して、IOS HTTP サーバを有効にできます。

```
ip http server
```

次のコマンドを使用して、デフォルトの HTTP サーバ ポートを 80 から任意の有効なポート番号に変更できます。

```
ip http port 8080
```

HTTP の最大接続数

SCEP を使用して PKI サーバとして IOS を展開する際にボトルネックとなるのは、1 分あたりの HTTP 最大同時接続数と HTTP 平均接続数です。
現在、IOS HTTP サーバでの最大同時接続数はデフォルトで 5 つまでに制限されていて、16 まで増やすことができます。中間の数で展開することを強くお勧めします。

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

次の IOS インストールでは、HTTP 最大同時接続数は 1000 まで可能です。

- UniversalK9 IOS with uck9 license-set

CLI は、1 ~ 1000 の間の数値引数を受け入れるように自動的に変更されます。

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

IOS HTTP サーバは 1 分間に 80 接続まで (最大 HTTP 同時セッションを 1000 まで増加できる IOS リリースの場合、580 まで) 可能です。1 分以内でこの制限に到達した場合、IOS HTTP リスナーは、15 秒間リスナーをシャットダウンすることによって、入力 HTTP 接続のスロットリングを開始します。これにより、TCP「connection queue limit reached」でクライアント接続要求はドロップされます。詳細については、ここを参照してください。

RSA キー ペア

IOS での PKI サーバ機能の RSA キー ペアは、自動で生成することも、手動で生成することもできます。
PKI サーバを設定する際、IOS は PKI サーバ証明書を保存するために、PKI サーバと同じ名前でトラストポイントを自動的に作成します。

PKI サーバの RSA キー ペアの手動生成：

ステップ 1：次のコマンドを使用して、PKI サーバと同じ名前で RSA キー ペアを作成します。

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

ステップ2:PKIサーバを有効にする前に、PKIサーバのトラストポイントを変更します。

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

> **注**：PKI サーバのトラストポイントに関して言及される RSA キー ペアのモジュラス値は、IOS バージョン 15.4(3)M4 以前では考慮されません。これは既知の問題です。デフォルトのキー係数は 1024 ビットです。

PKI サーバの RSA キー ペアの自動生成：

PKI サーバを有効にすると、IOS は PKI サーバと同じ名前で RSA キー ペアを自動的に生成します。キーのモジュラス サイズは 1024 ビットです。

IOS バージョン 15.4(3)M5 以降、この設定によって、名前として <LABEL> を使用して RSA キー ペアが作成されます。またキーの強度は定義された <MOD> モジュラスによって決定されます。

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

スポイラー

CSCuu73408 IOS PKI サーバはロールオーバー証明書に対してデフォルト以外のキー サイズを許可する必要があります。

CSCuu73408 IOS PKI サーバはロールオーバー証明書に対してデフォルト以外のキー サイズを許可する必要があります。

現在、業界標準として、最小で 2048 ビットの RSA キー ペアが使用されています。

自動ロールオーバー タイマーの考慮事項

現在、IOS PKI サーバは、ロールオーバー証明書をデフォルトでは生成しません。**auto-rollover <days-before-expiry> コマンドを使用して PKI サーバの下で明示的に有効にする必要があります**。証明書のロールオーバーの詳細については、次を参照してください。

このコマンドは、PKI サーバ/CA 証明書が期限切れになる何日前に、IOS がロールオーバー CA 証明書を作成するかを指定します。現在の CA 証明書が期限切れになると、ロールオーバー CA 証明書がアクティブになります。現在のデフォルト値は 30 日間です。この値は、CA 証明書の有効期間に応じて適切な値に設定する必要があります。また、この値は PKI クライアントの自動登録タイマーの設定にも影響します。

> **注**：CA およびクライアント証明書ロールオーバーの際、自動ロールオーバー タイマーは、必ずクライアントの自動登録タイマーよりも前にトリガーされる必要があります。

CRL の考慮事項

IOS PKI インフラストラクチャは、CRL を分散する 2 つの方法をサポートします。

HTTP サーバへの CRL の公開

PKI サーバで次のコマンドを使用することにより、HTTP サーバの特定の場所に CRL ファイルを公開するように IOS PKI サーバを設定することができます。

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

PKI サーバで次のコマンドを使用することにより、すべての PKI クライアント証明書にこの CRL の場所を組み込むように PKI サーバを設定することもできます。

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

SCEP GetCRL メソッド

IOS PKI サーバは特定のデータベースの場所（デフォルトでは nvram）に CRL ファイルを自動的に保存します。PKI サーバで次のコマンドを使用して、SCP/FTP/TFTP サーバにコピーを保持することを強くお勧めします。

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

デフォルトでは、IOS PKI サーバは PKI クライアント証明書に CDP の場所を組み込みません。IOS PKI クライアントが失効チェックを実行するように設定されているときに、検証される証明書に CDP が組み込まれていない場合、また CA の場所で CA トラストポイントを検証するように設定されている場合（http://<CA-Server-IP または FQDN> を使用）、IOS はデフォルトで GetCRL メソッドに基づいて SCEP にフォールバックします。
次の URL で HTTP GET を実行することによって、SCEP GetCRL は CRL の取得を実行します。

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

注：IOS CLIで？と入力する前に、**Ctrl + Vキー**シーケンスを押します。

IOS PKI サーバは、CDP の場所としてこの URL を組み込むこともできます。これを行うことには、二重の利点があります。

- PKI クライアントに基づくすべての非 IOS SCEP で、CRL の取得を確実に実行できます。
- 組み込まれた CDP なしに、IOS SCEP GetCRL 要求メッセージは、SCEP ドラフトで定義されているように署名されます（一時的な自己署名証明書を使用します）。ただし、CRL 取得要求は署名する必要がありません。GetCRL メソッドの CDP URL を組み込むことで、CRL 要求の署名を回避できます。

CRL の有効期間

PKI サーバで次のコマンドを使用することによって、IOS PKI サーバの CRL 有効期間を制御することができます。

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

値は時間単位です。デフォルトでは、CRL の有効期間は 6 時間に設定されています。どれほどの頻度で証明書を無効にするかに応じて CRL の有効期間を最適な値に調整することにより、ネットワークにおける CRL の取得に関するパフォーマンスを向上させることができます。

データベースの考慮事項

IOS PKI サーバはデフォルトのデータベースの場所として nvram を使用します。データベースの場所として FTP、TFTP、または SCP サーバを使用することを強くお勧めします。デフォルトでは、IOS PKI サーバは 2 つのファイルを作成します。

- <Server-Name>.ser:16進数でCAによって発行された最後のシリアル番号が含まれます。ファイルはプレーンテキスト形式で、以下の情報が含まれます。
  db_version = 1
  last_serial = 0x4

- <Server-Name>.crl：これは CA によって公開される DER エンコード CRL ファイルです。

IOS PKI サーバは、3 段階の設定可能レベルでデータベースに情報を保存します。

- Minimum：これはデフォルトのレベルで、このレベルでは、データベースに作成されるファイルはありません。そのため、過去に許可されたクライアント証明書に関する情報は CA サーバでは取得できません。
- Names：このレベルでは、IOS PKI サーバは <Serial-Number>.cnm という名前のファイルを作成します。ここで、名前の <Serial-Number> は、発行されたクライアント証明書のシリアル番号を参照します。また、この cnm ファイルにはクライアント証明書のサブジェクト名と有効期日が含まれます。
- 完了：このレベルでは、IOS PKIサーバは発行されたクライアント証明書ごとに2つのファイルを作成します。
- <Serial-Number>.cnm
- <Serial-Number>.crt

この crt ファイルは、DER でエンコードされたクライアント証明書ファイルです。

以下の点が重要になります。

- クライアント証明書を発行する前に、IOS PKI サーバは <Server-Name>.ser を参照して、証明書のシリアル番号を判断し、名前に付けます。
- データベース レベルを Names または Complete に設定した場合、許可/発行された証明書をクライアントに送信する前に、<Serial-Number>.cnm および <Serial-Number>.crt をデータベースに書き込む必要があります。

- データベース URL を Names または Complete に設定した場合、データベース URL には、ファイルを保存するための十分なスペースが必要になります。そのため、データベース URL として外部ファイル サーバ（FTP、TFTP、または SCP）を設定することをお勧めします。
- 外部データベース URL を設定する場合、証明書の許可プロセス中にファイル サーバに必ず到達できるようにする必要があります。そうしないと、CA サーバには無効のマークが付けられます。CA サーバをオンラインに戻すには、手動による介入が必要になります。

データベースのアーカイブ

PKI サーバを導入している間、障害シナリオについて考慮し、万が一ハードウェア障害が発生した場合に備えをしておくことは重要です。これを実現するには、次の 2 通りの方法があります。

1. 冗長性
   この場合、2 つのデバイスまたは処理装置はそれぞれアクティブ/スタンバイとして機能し、冗長性を提供します。
   2 台の HSRP 対応 ISR ルータ（ISR G1 および ISR G2）を使用して、IOS PKI サーバの高可用性を実現することができます。

   IOS-XE ベース システム（ISR4K および ASR1k）には、使用できるデバイス冗長性オプションはありません。ただし、ASR1k では、相互 RP 冗長性をデフォルトで使用できます。
2. CA サーバ キー ペアとファイルのアーカイブ
   IOS では、PKI サーバ キー ペアと証明書をアーカイブする機能が提供されます。アーカイブは、2 種類のファイルを使用して実行できます。
   PEM：IOS は PEM 形式のファイルを作成し、RSA 公開キー、暗号化 RSA 秘密キー、CA サーバ証明書を保存します。ロールオーバー キー ペアと証明書は、自動的にアーカイブされますPKCS12：IOS は、CA サーバ証明書と、パスワードを使用して暗号化される対応する RSA 秘密キーを格納する単一の PKCS12 ファイルを作成します。

PKI サーバで次のコマンドを使用して、データベースのアーカイブを有効にできます。

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

PKI サーバで次のコマンドを使用し、別個のサーバにアーカイブ ファイルを保存することもできます（可能であればセキュアなプロトコル（SCP）を使用）。

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

アーカイブされたファイルと.Serファイルを除くデータベース内のすべてのファイルの中で、他のすべてのファイルはクリアテキストであり、失われた場合は実質的な脅威を与えません。したがって、TFTPサーバなどのファイルの書き込み中に大きなオーバーヘッドが発生発生ない。

下位 CA としての IOS

IOS PKI サーバはデフォルトで、ルート CA の役割を果たします。下位の PKI サーバ（下位 CA）を設定するには、まず PKI サーバ設定セクションで次のコマンドを有効にします（PKI サーバを有効にする前）。

```
crypto pki server <Sub-PKI-SERVER-Name>
```

```
 mode sub-cs
```
次のコマンドを使用して、PKI サーバのトラストポイントでルート CA の URL を設定します。

```
crypto pki trustpoint <Sub-PKI-SERVER-Name>
 enrollment url <Root-CA URL>
```
この PKI サーバを有効にすると、以下のイベントがトリガーされます。

- ルート CA 証明書をインストールするために、PKI サーバのトラストポイントが認証されます。
- ルートCAが認証されると、IOSは下位CA(x509)のCSRを生成し（CA:TRUEフラグを含む）、ルートCAに送信します

ルート CA で設定された許可モードに関係なく、IOS は保留キューに CA（または RA）証明書要求を配置します。管理者は手動で CA 証明書を許可する必要があります。
保留中の証明書要求と要求 ID を表示するには、次のコマンドを実行します。

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

要求を許可するには、次のコマンドを実行します。

```
crypto pki server <Server-Name> grant <request-id>
```
- このコマンドを使用すると、後続の SCEP POLL（GetCertInitial）操作によって、下位 CA 証明書がダウンロードされ、下位 PKI サーバが有効にされるルータにインストールされます。

RA としての IOS

IOS PKI サーバは、特定の下位 CA またはルート CA の登録局として設定できます。PKI サーバを登録局として設定するには、まず PKI サーバ設定セクションで次のコマンドを有効にします（PKI サーバを有効にする前）。

```
crypto pki server <RA-SERVER-Name>
 mode ra
```
次のコマンドを使用して、PKI サーバのトラストポイントで CA の URL を設定します。これは、どの CA が RA によって保護されるかを指定します。

```
crypto pki trustpoint <RA-SERVER-Name>
 enrollment url <CA URL>
 subject-name CN=<Common Name>, OU=ioscs RA, OU=TAC, O=Cisco
```
登録局は証明書を発行しないため、RA の下に **issuer-name 設定する必要はなく、設定したとしても、効力はありません。**RA のサブジェクト名は、**subject-name コマンドを使用して RA トラストポイントで設定されます。**IOS CA が IOS RA を識別できるようにするため（つまり、IOS RA によって承認された証明書要求を識別する）、**OU = ioscs RA をサブジェクト名の一部として設定することが重要です。**

IOS は Microsoft CA などのサードパーティ製 CA の登録局として機能することもできます。互換性を維持するため、PKI サーバ設定セクションで次のコマンドを使用して、IOS RA を有効にする必要があります（PKI サーバを有効にする前に）。

```
mode ra transparent
```
デフォルトの RA モードでは、IOS は RA 証明書を使用してクライアント要求（PKCS#10）に署名します。この操作は、証明書要求が RA によって承認される IOS PKI サーバを指定します。

トランスペアレント RA モードでは、IOS はクライアント要求を、RA 証明書を導入せずに元の形式で転送します。これは、有名な例として Microsoft CA と互換性があります。

IOS PKI クライアント

IOS PKI クライアントにおける最も重要な設定エンティティの 1 つは、トラストポイントです。トラストポイントの設定パラメータについて、このセクションで詳しく説明します。

正規の時刻源

前にも指摘したように、正規の時刻源は PKI クライアントの要件でもあります。以下の設定を使用して、IOS PKI クライアントを、NTP クライアントとして設定できます。

```
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
```
ホスト名とドメイン名

一般的に、ルータでホスト名とドメイン名を設定することが推奨されています。

```
configure terminal
hostname <string>
ip domain name <domain>
```
RSA キー ペア

IOS PKI クライアントでは、特定のトラストポイントを登録するための RSA キー ペアを自動で生成することも、手動で生成することもできます。

RSA キーの手動生成プロセスは、以下のように行われます。

- IOS はデフォルトで 512 ビット RSA キー ペアを作成します。
- 自動で生成されるキー ペアの名前は hostname.domain-name のように、デバイスのホスト名とデバイスのドメイン名を組み合わせた形になります。
- 自動生成されたキー ペアは、エクスポート可能としてマークされていません。

RSA キーの手動生成プロセスは、以下のように行われます。

- オプションで、以下のコマンドを使用して、適切な強度の汎用 RSA キー ペアを手動で生成できます。
-
```
crypto key generate rsa general-keys label <LABEL> modulus < MOD> [exportable]
```
ここで、LABEL：RSA キー ペアの名前は、次のとおりです。
MOD - RSA キーのモジュラス値または強度。範囲は 360 ~ 4096 ビット。512、1024、2048、または 4096 がよく使用されています。
RSA キー ペアを手動で生成する利点は、キー ペアをエクスポート可能としてマークできることにあります。これにより、ID 証明書のために完全にエクスポートすることができ、別のデバイスで復元することができます。ただし、このアクションはセキュリティに影響を与える可能性があることを理解しておく必要があります。
- 登録する前に、次のコマンドを使用して、RSA キー ペアをトラストポイントにリンクします。

```
crypto pki trustpoint MGMT
rsakeypair <LABEL> [<MOD> <MOD>]
```
ここで、<LABEL> という名前の RSA キー ペアがすでに存在する場合、トラストポイントの登録時にピックアップされます。
<LABEL> という名前の RSA キー ペアがなければ、登録時に次のいずれかのアクションが実行されます。
：<MOD> 引数が渡されない場合、<LABEL> という名前の 512 ビットのキー ペアが生成されます。
：<MOD> 引数が 1 つ渡される場合、<LABEL> という名前の <MOD> ビットの汎用キー ペアが生成されます。
：<MOD> 引数が 2 つ渡される場合、<MOD> ビットの署名キー ペア 1 つと <MOD> ビットの暗号キー ペア 1 つ（両方とも <LABEL> という名前）が生成されます。

トラストポイント

トラストポイントは、IOS で証明書を保持する概念的なコンテナです。1 つのトラストポイントに、以下の 2 つのアクティブな証明書をいつでも保存できます。

- CA 証明書：CA 証明書を特定のトラストポイントにロードする操作は、トラストポイント認証プロセスと呼ばれます。
- CA によって発行された ID 証明書：ID 証明書を特定のトラストポイントにロードする、またはインポートする操作は、トラストポイント登録プロセスと呼ばれます。

トラストポイント設定は、信頼ポリシーと呼ばれ、以下を定義します。

- トラストポイントにロードする CA 証明書
- トラストポイントが登録する CA
- IOS がトラストポイントを登録する方法
- 特定の CA によって発行される（トラストポイントにロードされる）証明書の検証方法

トラストポイントの主なコンポーネントを次に示します。

登録モード

トラストポイントの登録モード（トラストポイント認証モードも定義します）は、主に 3 つの方

法で実行されます。

1. 端末登録：CLI 端末でのコピー アンド ペーストを使用してトラストポイントの認証および証明書の登録を手動で行う方法。
2. SCEP 登録：HTTP での SCEP を使用したトラストポイントの認証および登録。
3. 登録プロファイル：この方法では、認証方法と登録方法が個別に定義されます。端末登録および SCEP 登録の方法とともに、登録プロファイルは、サーバからのファイル検索を実行するための HTTP/TFTP コマンドを指定するオプションを提供します。これは、プロファイルで認証または登録用の URL を使用して定義されます。

送信元インターフェイスと VRF

HTTP (SCEP) または TFTP (登録プロファイル) でのトラストポイントの認証と登録では、ファイル I/O 操作を実行するために、IOS ファイル システムが使用されます。これらのパケット交換は、特定の送信元インターフェイスおよび VRF から提供されます。

標準的なトラストポイント設定の場合、この機能は、トラストポイントの下で **source interface および vrf サブコマンドを使用して有効にします。**

登録プロファイルでは、**source interface と enrollment | authentication url <http/tftp://Server-location> vrf <vrf-name>コマンド**は、同じ機能を提供します。

設定例：

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

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

または

```
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
```

自動証明書登録と更新

PKI トラストポイント セクションで次のコマンドを使用することにより、自動登録および更新を実行するように IOS PKI クライアントを設定できます。

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

ここで、auto-enroll <percentage> [regenerate] コマンドは、IOS が現在の証明書の有効期間のちょうど 80 % で証明書の更新を実行する必要があることを示しています。

キーワード **regenerate は、毎回の証明書更新操作中に IOS がシャドウ キー ペアと呼ばれる** RSA

キー ペアを再生成する必要があることを示しています。

以下は、自動登録の動作です。

- **auto-enroll が設定されると、トラストポイントが認証されていれば、IOS は PKI トラストポイント セクションの下、または登録プロファイルの下で enrollment url コマンドの一部として指定された URL にあるサーバへの自動登録を実行します。**
- トラストポイントが PKI サーバまたは CA に登録されると、トラストポイントの下にインストールされた現在の ID 証明書の **auto-enroll パーセンテージに基づいて、RENEW または SHADOWCA タイマーが PKI クライアントにインストールされます。**このタイマーは、**show crypto pki timer コマンドで表示されます。**タイマーの詳細については、*次を参照してください。*
- 更新機能は、PKI サーバによってサポートされます。詳細については、次を参照してください。
  IOS PKI クライアントは、次の 2 種類の更新を実行します。
  暗黙的な更新：PKI サーバがサポート対象の機能として「更新」を送信しない場合、IOS は定義された auto-enroll パーセンテージで最初の登録を実行します。つまり、IOS は自己署名証明書を使用して、更新要求に署名します。明示的な更新：PKI サーバが PKI クライアント証明書の更新機能をサポートする場合、サポート対象の機能として「更新」をアドバタイズします。IOS は証明書の更新時にこの機能を考慮に入れます。つまり IOS は現在のアクティブな ID 証明書を使用して、更新証明書要求に署名します。

auto-enroll のパーセンテージを設定する際は、細心の注意を払う必要があります。展開内の特定の PKI クライアントで、発行側 CA 証明書と同時に ID 証明書が期限切れになる状況が発生した場合、CA がロールオーバー証明書を作成した後に、auto-enroll 値によって常に [shadow] 更新操作がトリガーされる必要があります。*セクション*「**PKI タイマーの依存関係**」を参照してください。

証明書失効チェック

認証された PKI トラストポイント（つまり、CA 証明書を含む PKI トラストポイント）は、IKE または SSL ネゴシエーション中に（ここで、ピア証明書が完全な証明書検証を受けます）、証明書の検証を行うことができます。検証方法の 1 つは、次の 2 つの方法のいずれかを使用して、ピア証明書の失効ステータスを確認する方法です。

- 証明書失効リスト（CRL）：これは、特定の CA によって無効にされた証明書のシリアル番号が記載されたファイルです。このファイルは、発行側 CA 証明書を使用して署名されます。CRL を使用する方法では、HTTP または LDAP を使用して CRL ファイルがダウンロードされます。

- Online Certificate Status Protocol（OCSP）：IOS は OCSP 応答と呼ばれるエンティティ（発行側 CA によって指定されたサーバ）との通信チャネルを確立します。IOS などのクライアントは、検証される証明書のシリアル番号を含む要求を送信します。OCSP 応答は、特定のシリアル番号の失効ステータスで対応します。通信チャネルは、HTTP でサポートされるアプリケーション/転送プロトコルを使用して確立することができます。通常、HTTP が使用されます。

PKI トラストポイント セクションで次のコマンドを使用して、失効チェックを定義できます。

```
crypto pki trustpoint MGMT
 revocation-check crl ocsp none
```
デフォルトでは、トラストポイントは CRL を使用して失効チェックを実行するように設定されています。

このメソッドの順序は変更できます。失効ステータス チェックは、定義された順序で実行されます。メソッド「none」は、失効チェックをバイパスします。

CRL キャッシュ

CRL ベースの失効チェックでは、すべての証明書検証で新しい CRL ファイルのダウンロードがトリガーされます。また CRL ファイルが大きくなる、または CRL 分散ポイント（CDP）が離れていく場合、すべての検証プロセスでのファイルのダウンロードが、証明書検証に依存するプロトコルのパフォーマンスを低下させます。そのため、パフォーマンスを向上させるために、CRL のキャッシングが行われます。CRL のキャッシングでは、CRL の有効性が考慮されます。

CRL の有効性は、次の 2 つの時間パラメータを使用して定義されます。**LastUpdate** は、発行側 CA によって公開された最後の時間です。**NextUpdate** は、発行側 CA によって CRL ファイルの新しいバージョンが発行される将来の時間です。

IOS は、CRL が有効である限り、ダウンロードしたすべての CRL をキャッシュします。ただし、CDP に一時的に到達できないなど特定の状況下では、キャッシュの CRL を長期間にわたって維持することが必要になる場合があります。IOS では、キャッシュされた CRL を CRL の有効期限が切れてから 24 時間維持することができます。PKI トラストポイントのセクションで、次のコマンドを使用して、この時間を設定することができます。

```
crypto pki trustpoint MGMT
 crl cache extend <0 - 1440>
!! here the value is in minutes
```
発行側 CA が CRL の有効期限内に証明書を取り消すような特定の状況下では、キャッシュをより頻繁に削除するように IOS を設定できます。時期を早めて CRL を削除することによって、CRL キャッシュを最新の状態に保つため、より頻繁に CRL をダウンロードするように IOS に強制することができます。PKI トラストポイントのセクションで、次の設定オプションを使用できます。

```
crypto pki trustpoint MGMT
 crl cache delete-after <1-43200>
!! here the value is in minutes
```
最後に、PKI トラストポイントのセクションで次のコマンドを使用することにより、CRL ファイルをキャッシュしないように IOS を設定できます。

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

推奨設定

ルート CA および下位 CA を指定する一般的な CA の導入は、次のようになります。この例では、RA によって保護される下位 CA 設定が含まれています。

全体で 2048 ビットの RSA キー ペアを使用する場合、この例では以下のような設定をお勧めし

ます。
ルート CA に 8 年の有効期間を設定
下位 CA に 3 年の有効期間を設定
クライアント証明書は 1 年間の有効期間で発行されます（証明書を自動的に要求するように設定します）。

ルート CA：設定

```
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
```

RA なしの SUBCA：設定

```
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
```

RA ありの SUBCA：設定

```
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
```

SUBCA 用の RA：設定

```
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
```

証明書登録

手動登録

手動登録では、PKIクライアントでオフラインのCSRが生成され、CAに手動でコピーされます。管理者が手動で要求に署名し、クライアントにインポートします。

PKI クライアント

PKI クライアント設定：

```
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
```

ステップ1：最初にトラストポイントを認証します（これは、ステップ2の後でも実行できます）。

```
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
```

ステップ 2：証明書署名要求を生成し、CSR を CA に取り込み、許可された証明書を取得します。

```
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
```

ステップ3：次に、ターミナル経由で許可された証明書をインポートします。

```
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 サーバ

ステップ1：最初にCAからIssuing CA certificateをエクスポートします。この場合はSUBCA証明書です。これは上記の PKI クライアントのステップ 1（つまりトラストポイントの認証）でインポートされます。

```
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-----
```

ステップ2:PKIクライアントのステップ2の後、クライアントからCSRを取得し、次のコマンドを使用してSUBCAに署名するために提供します。

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

次のコマンドで、SUBCA が端末からの証明書署名要求を受け入れることを提案します。許可すると、証明書データは PEM 形式で印刷されます。

```
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
```

CA が自動許可モードの場合、許可された証明書は上記のように PEM 形式で表示されます。CA が手動許可モードの場合、証明書要求は **pending** としてマーク付けされ、ID 値が割り当てられ、登録要求キューに入れられます。

```
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
```

ステップ 3：次のコマンドを使用して、この要求を手動で許可します。

```
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-----
```

注：ルート CA への下位 CA の手動登録はできません。

注：HTTP サーバが無効であるために無効な状態にある CA は証明書要求を手動で許可できます。

SCEP を使用した登録

PKI クライアントの設定は以下のとおりです。

```
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 サーバの設定は以下のとおりです。

```
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/
```

証明書要求の許可のデフォルト モードは手動です。

```
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
```

手動許可

ステップ 1：PKI クライアント:最初の必須のステップとして、PKI クライアントでトラストポイントを認証します。

```
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.
```

ステップ 2：PKI クライアント:トラストポイントの認証後に、証明書のために PKI クライアントを登録できます。

注：自動登録を設定した場合、クライアントは自動的に登録を実行します。

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

バックグラウンドで、以下のイベントが発生します。

- IOS は「PKI-Key」という名前の RSA キー ペアを探します。キーが存在する場合、ID 証明書を要求するためにそのキーをピックアップします。存在しない場合、IOS は「PKI-Key」という名前で 2048 ビット キー ペアを作成し、ID 証明書を要求するためにこのキーを使用します。

- IOS は PKCS10 形式で証明書署名要求を作成します。

- IOS は、ランダム対称キーを使用して、この CSR を暗号化します。ランダム対称キーは、受信者（SUBCA）の公開キーを使用して暗号化されます（SUBCA の公開キーはトラストポイントの認証のために使用できます）。 暗号化された CSR、暗号化されたランダム対称キー、および受信者情報は、PKCS#7 エンベロープ データにまとめられます。

- この PKCS#7 エンベロープ データは、最初の登録時に一時的な自己署名証明書を使用して署名されます。PKCS#7 エンベロープ データ、クライアントによって使用される署名証明書、およびクライアントの署名は、PKCS#7 署名済みデータ パケットにまとめられます。これは base64 でエンコードされ、その後 URL でエンコードされます。結果として生成されるデータは、CA に送信される HTTP URI の「メッセージ」引数として送信されます。

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

ステップ 3：PKI サーバ：

IOS PKI サーバが要求を受信すると、以下をチェックします。

1.登録要求データベースに、新しい要求に関連付けられた同じトランザクションIDを持つ証明書要求が含まれているかどうかを確認します。

注：トランザクション ID は、公開キーの MD5 ハッシュで、このために ID 証明書がクライアントによって要求されます。

2.登録要求データベースに、クライアントから送信されたものと同じチャレンジパスワードの証明書要求が含まれているかどうかを確認します。

注：（1）で true が返されるか、（1）と（2）の両方で true が返される場合、CA サーバは、ID 要求が重複しているという理由で要求を拒否することができます。ただし、このような場合、IOS PKI サーバは、新しい要求で古い要求を置き換えます。

ステップ 4：PKI サーバ：

PKI サーバで要求を手動で許可します。

要求を表示するには、次のコマンドを実行します。

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

特定の要求または要求すべてを許可するには、次のコマンドを実行します。

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

ステップ 5：PKI クライアント:

その一方で、PKI クライアントは POLL タイマーを開始します。ここで、IOS は、クライアントが許可された証明書とともに SCEP CertRep = GRANTED を受け取るまで、一定の間隔で GetCertInitial を実行します。

一度許可された証明書を受け取ると、IOS はそれを自動的にインストールします。