PKI の設定

この章では、Cisco NX-OS での公開キー インフラストラクチャ（PKI）のサポートについて説明します。PKI を使用すると、ネットワーク上で通信を安全に行うためのデジタル証明書をデバイスが入手して使用できるようになり、セキュア シェル（SSH）の管理性と拡張性も向上します。

この章は、次の項で構成されています。

PKI の概要

ここでは、PKI について説明します。

CA とデジタル証明書

証明機関（CA）は証明書要求を管理して、ホスト、ネットワーク デバイス、ユーザなどの参加エンティティに証明書を発行します。CA は参加エンティティに対して集中型のキー管理を行います。

デジタル署名は、公開キー暗号法に基づいて、デバイスや個々のユーザをデジタル的に認証します。RSA 暗号化システムなどの公開キー暗号法では、各デバイスやユーザはキー ペアを持

ち、これには秘密キーと公開キーが含まれています。秘密キーは秘密裡に保管し、これを知っているのは所有するデバイスまたはユーザだけです。一方、公開キーは誰もが知っているものです。これらのキーの一方で暗号化されたものは、他方のキーで復号化できます。署名は、送信者の秘密キーを使用してデータを暗号化したときに作成されます。受信側は、送信側の公開キーを使用してメッセージを復号化することで、シグニチャを検証します。このプロセスは、受信者が送信者の公開キーのコピーを持っていて、これが本当に送信者のものであり、送信者を騙る他人のものではないことを高い確実性を持って知っていることを基盤としています。

デジタル証明書は、デジタル署名と送信者を結び付けるものです。デジタル証明書には、名前、シリアル番号、企業、部署または IP アドレスなど、ユーザまたはデバイスを特定する情報を含んでいます。また、エンティティの公開キーのコピーも含んでいます。証明書に署名する CA は、受信者が明示的に信頼する第三者機関であり、アイデンティティの正当性を立証し、デジタル証明書を作成します。

CA のシグニチャを検証するには、受信者は、CA の公開キーを認識している必要があります。一般的にはこのプロセスはアウトオブバンドか、インストール時に行われる操作によって処理されます。たとえば、通常の Web ブラウザでは、デフォルトで、複数の CA の公開キーが設定されています。

信頼モデル、トラストポイント、アイデンティティ CA

PKI の信頼モデルは、設定変更が可能な複数の信頼できる CA によって階層化されています。信頼できる CA のリストを使用して各参加デバイスを設定して、セキュリティ プロトコルの交換の際に入手したピアの証明書がローカルに信頼できる CA のいずれかで発行されていた場合には、これを認証できるようにすることができます。Cisco NX-OS ソフトウェアでは、信頼できる CA の自己署名ルート証明書（または下位 CA の証明書チェーン）をローカルに保存しています。信頼できる CA のルート証明書（または下位 CA の場合には全体のチェーン）を安全に入手するプロセスを、*CA* 認証と呼びます。

信頼できる CA について設定された情報をトラストポイントと呼び、CA 自体もトラストポイント *CA* と呼びます。この情報は、CA 証明書（下位 CA の場合は証明書チェーン）と証明書取消確認情報で構成されています。

Cisco NX-OS デバイスは、トラストポイントに登録して、アイデンティティ証明書を入手し、キーペアと関連付けることができます。このトラストポイントをアイデンティティ *CA* と呼びます。

CA証明書の階層

セキュアサービスの場合、通常は複数の信頼できるCAがあります。CAは通常、すべてのホストにバンドルとしてインストールされます。NX-OS PKIインフラストラクチャは、証明書チェーンのインポートをサポートします。ただし、現在のCLIでは、一度に1つのチェーンをインストールできます。インストールするCAチェーンが複数ある場合、この手順は面倒です。これには、複数の中間CAとルートCAを含むCAバンドルをダウンロードする機能が必要です。

トラストポイントインポートCLI

crypto CA trustpointコマンドは、CA証明書、CRL、アイデンティティ証明書、およびキーペアを名前付きラベルにバインドします。これらの各エンティティに対応するすべてのファイルは、NX-OS certstoreディレクトリ（/ isan / etc / certstore）に保存され、トラストポイントラベルでタグ付けされます。

CA証明書にアクセスするには、SSLアプリケーションは標準のNX-OS証明書ストアをポイントし、SSL初期化中にCAパスとして指定するだけです。CAがインストールされているトラストポイントラベルを認識する必要はありません。

クライアントがアイデンティティ証明書にバインドする必要がある場合は、トラストポイントラベルをバインディングポイントとして使用する必要があります。

import pkcsコマンドは、トラストポイントラベルの下にCA証明書をインストールするように拡張されています。CAバンドルをインストールするようにさらに拡張できます。importコマンド構造が変更され、pkcs7形式のCAバンドルファイルを提供するために使用されるpkcs7オプションが追加されました。提案された解決策は、CAバンドルを展開し、各CAチェーンを独自のラベルでインストールすることです。ラベルは、メイントラストポイントラベルにインデックスを追加することによって形成されます。

既存のトラストポイント設定は、内部で使用されます。新しい設定CLIを実装する必要はありません。クライアントアプリケーションからの変更は必要ありません。

一度インストールすると、バンドルへのすべてのCAチェーンの論理バインディングはありません。そのため、CAバンドルの置換または削除には、追加のロジックが必要になる場合があります。設定CLI、cabundle<bundle name> CAバンドルにトラストポイントをバインドするために提供できます。これは、バンドルの削除や変更、運用データの取得などに使用できます。

PKCS7 形式での CA 証明書バンドルのインポート

複数の独立した証明書チェーンで構成される CA 証明書バンドルのインポートをサポートするために、 'pkcs7' のオプションが crypto import コマンドに導入されました。

手順

	コマンドまたはアクション	目的
ステップ 1	**copy** *scheme*:// *server*/[*url* /]*filename* **bootflash:***filename* **例**： `switch# copy tftp:adminid.p7 bootflash:adminid.p7`	PKCS#7形式のファイルをリモートサーバからコピーします。 *scheme* 引数に対しては、**tftp:**、**ftp:**、**scp:**、または **sftp:** を入力できます。*server* 引数は、リモート サーバのアドレスまたは名前であり、*url* 引数はリモート サーバにあるソース ファイルへのパスです。

	コマンドまたはアクション	目的
		server、*url*、および *filename* の各引数は、大文字小文字を区別して入力します。

ステップ 2	**configure terminal**
	例：
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します

ステップ 3	**crypto ca import** <*baselabel*> **pksc7** <*uri0*>	コマンドには2つの入力引数があります。Ca バンドルファイルであるソースファイルは、<*uri0*>、入力ファイルは pkcs7 形式である必要があります。これは cabundle ファイルであることを示します。 複数の証明書チェーンが cabundle から抽出されます。このコマンドは、CA証明書チェーンが接続された複数のトラストポイントを生成します。<*baselabel*> 引数は、トラストポイント名のベースを形成する入力名を取ります。つまり、生成されるすべてのトラストポイントの名前は、ユーザの入力として指定されたベースラベル名から取得されます。

ステップ 4	**exit**
	例：
	`switch(config)# exit`
	`switch#`
		設定モードを終了します。

ステップ 5	（任意） **show crypto ca certificates** 例： `switch# show crypto ca certificates`	CA 証明書を表示します。
ステップ 6	（任意） **copy running-config startup-config** 例： `switch# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

CISCO-AV-PAIR パージング環境

Cisco NX-OSでは、CISCO-AV-PAIRの最初の属性として「shell：roles」が必須です。属性が後の段階にある場合は、考慮されません。NX-OSは、属性の到着に関係なく、この厳密な順序付け要件を緩和する必要があります。

たとえば、snmpv3属性は、次のように古いか新しいかに関係なく、引用符で標準化する必要があります。

```
cisco-av-pair=shell:roles="network-admin" snmpv3:auth=”SHA” priv=”AES-128”
```

snmpv3解析では値が厳密にチェックされないため、XXXSHAなどの値はSHAとして渡されます。RADIUS、TACACS、およびLDAPプロトコルでは、属性「shell：role」がサポートされています。ただし、「snmpv3」属性はLDAPでは使用できません。提案された変更はTACACSおよびRADIUSコードに組み込まれます。

（注） LDAPは「snmpv3」属性をサポートしていないため、この段階では変更は必要ありません。

現在、2番目のsnmpv3属性は、プロトコルに言及せずに許可されます。つまり、両方の属性の先頭に「snmpv3：」を付ける必要はありません。

After Shell属性は次のとおりです。

```
cisco-av-pair=shell:roles="network-admin" shell:priv-lvl=15 snmpv3:auth=”SHA”
priv=”AES-128”
```

[Before Shell Attributes]は次のとおりです。

```
cisco-av-pair= snmpv3:auth=”SHA” priv=”AES-128” shell:roles="network-admin"
shell:priv-lvl=15
```

「crypto ca import」CLI の DME 化

次の 2 つの CLI は DME サポートを提供します。

```
crypto ca import <trustpoint-label> pkcs12 bootflash:<file> <passphrase>
```

```
copy tftp://<ip>/<file-path>/<file-name> bootflash:<file-name> vrf management use-kstack
```

CLI

```
crypto ca
        import <trustpoint-label> pkcs12 bootflash:<file> <passphrase>
```

キーを復号化するために、トラストポイント `trustpoint-label` （`pkcs12` ファイル形式と `passphrase` を使用）のソース ファイルをインポートします。

最初に、ソースファイルをtftpの場所からブートフラッシュにコピーする必要があります。次のCLIを使用します。

```
copy tftp://10.10.1.1/test.txt bootflash:test.txt vrf management use-kstack
```

（注） DMEサポートは、「crypto ca import」と「copy tftp」の両方のCLIで必要です。 copy-tftpコマンドの宛先でサポートされる値は、`bootflash://` のみです。

DME 化の制限事項

「crypto ca import」および「copy tftp」アクション コマンドの DME 化には、次の制限があります。

1. Pkcs12 ファイル形式のみがサポートされます。Pkcs7 ファイル形式には、複数のトラストポイントが関連付けられています。その結果、pkcs7 ファイル形式は以降のリリースでサポートされる予定です。
2. Tftp コピーは bootflash: パスに対してのみ有効であるため、ユーザはスイッチにログインせずにファイルをインポートできます。
3. インポートおよび TFTP タスク管理オブジェクトの NX-API ポスト ペイロードは生成できません。
4. TFTP の複数のコピー タスクは並行してサポートされません。バックエンドはファイルのコピーに時間がかかります。

RSA のキー ペアとアイデンティティ証明書

アイデンティティ証明書を入手するには、1 つまたは複数の RSA キー ペアを作成し、各 RSA キー ペアと Cisco NX-OS デバイスが登録しようとしているトラストポイント CA を関連付けます。Cisco NX-OS デバイスは、CA ごとにアイデンティティを 1 つだけ必要とします。これは CA ごとに 1 つのキー ペアと 1 つのアイデンティティ証明書で構成されています。

Cisco NX-OS ソフトウェアでは、設定変更が可能なキーのサイズ（またはモジュラス）で RSA キー ペアを作成できます。デフォルトのキーのサイズは 512 です。また、RSA キー ペアのラベルも設定できます。デフォルトのキー ラベルは、デバイスの完全修飾ドメイン名（FQDN）です。

トラストポイント、RSA キー ペア、およびアイデンティティ証明書の関係を要約したものを次に示します。

- トラストポイントとは、Cisco NX-OS デバイスが、あらゆるアプリケーション（SSH など）のピア証明書用に信頼する特定の CA です。
- Cisco NX-OS デバイスでは、デバイス上に多くのトラストポイントを置くことができ、デバイス上のすべてのアプリケーションは、任意のトラストポイント CA によって発行されたピア証明書を信頼できます。
- トラストポイントは特定のアプリケーション用に限定されません。
- Cisco NX-OS デバイスは、トラストポイントに対応する CA に登録して、アイデンティティ証明書を入手します。デバイスは複数のトラストポイントに登録できます。これは、各トラストポイントから異なるアイデンティティ証明書を入手できることを意味します。アイデンティティ証明書は、発行する CA によって証明書に指定されている目的に応じてアプリケーションで使用します。証明書の目的は、証明書の拡張機能として証明書に保存されます。

- トラストポイントに登録するときには、証明を受ける RSA キー ペアを指定する必要があります。このキーペアは、登録要求を作成する前に作成されていて、トラストポイントに関連付けられている必要があります。トラストポイント、キーペア、およびアイデンティティ証明書との間のアソシエーション（関連付け）は、証明書、キーペア、またはトラストポイントが削除されて明示的になくなるまで有効です。
- アイデンティティ証明書のサブジェクト名は、Cisco NX-OS デバイスの完全修飾ドメイン名です。
- デバイス上には 1 つまたは複数の RSA キー ペアを作成でき、それぞれを 1 つまたは複数のトラストポイントに関連付けることができます。しかし、1 つのトラストポイントに関連付けられるキー ペアは 1 だけです。これは 1 つの CA からは 1 つのアイデンティティ証明書しか入手できないことを意味します。
- Cisco NX-OS デバイスが複数のアイデンティティ証明書を（それぞれ別の CA から）入手する場合は、アプリケーションがピアとのセキュリティ プロトコルの交換で使用する証明書は、アプリケーション固有のものになります。
- 1 つのアプリケーションに 1 つまたは複数のトラストポイントを指定する必要はありません。証明書の目的がアプリケーションの要件を満たしていれば、どのアプリケーションもあらゆるトラストポイントで発行されたあらゆる証明書を使用できます。
- あるトラストポイントから複数のアイデンティティ証明書を入手したり、あるトラストポイントに複数のキー ペアを関連付ける必要はありません。ある CA はあるアイデンティティ（または名前）を 1 回だけ証明し、同じ名前で複数の証明書を発行することはありません。ある CA から複数のアイデンティティ証明書を入手する必要があり、またその CA が同じ名前で複数の証明書の発行を許可している場合は、同じ CA 用の別のトラストポイントを定義して、別のキー ペアを関連付け、証明を受ける必要があります。

複数の信頼できる CA のサポート

Cisco NX-OS デバイスは、複数のトラストポイントを設定して、それぞれを別の CA に関連付けることにより、複数の CA を信頼できるようになります。信頼できる CA が複数あると、ピアに証明書を発行した特定の CA にデバイスを登録する必要がなくなります。代わりに、ピアが信頼する複数の信頼できる CA をデバイスに設定できます。すると、Cisco NX-OS デバイスは設定されている信頼できる CA を使用して、ピアから受信した証明書で、ピア デバイスの ID で定義されている CA から発行されたものではないものを検証できるようになります。

PKI の登録のサポート

登録とは、SSH などのアプリケーションに使用するデバイス用のアイデンティティ証明書を入手するプロセスです。これは、証明書を要求するデバイスと、認証局の間で生じます。

Cisco NX-OS デバイスでは、PKI 登録プロセスを実行する際に、次の手順を取ります。

- デバイスで RSA の秘密キーと公開キーのペアを作成します。
- 標準の形式で証明書要求を作成し、CA に送ります。

Note 要求が CA で受信されたとき、CA サーバでは CA アドミニストレータが登録要求を手動で承認しなくてはならない場合があります。

- 発行された証明書を CA から受け取ります。これは CA の秘密キーで署名されています。
- デバイスの不揮発性のストレージ領域（ブートフラッシュ）に証明書を書き込みます。

カットアンドペーストによる手動での登録

Cisco NX-OS ソフトウェアでは、手動でのカットアンドペーストによる証明書の取得と登録をサポートしています。カットアンドペーストによる登録とは、証明書要求をカットアンドペーストして、デバイスと CA 間で認証を行うことを意味します。

手動による登録プロセスでカットアンドペーストを使用するには、次の手順を実行する必要があります。

- 証明書登録要求を作成します。これは Cisco NX-OS デバイスで base64 でエンコードされたテキスト形式として表示されます。
- エンコードされた証明書要求のテキストを E メールまたは Web フォームにカットアンドペーストし、CA に送ります。
- 発行された証明書（base64 でエンコードされたテキスト形式）を CA から E メールまたは Web ブラウザによるダウンロードで受け取ります。
- 証明書のインポート機能を使用して、発行された証明書をデバイスにカットアンドペーストします。

複数の RSA キー ペアとアイデンティティ CA のサポート

複数のアイデンティティ CA を使用すると、デバイスが複数のトラストポイントに登録できるようになり、その結果、別々の CA から複数のアイデンティティ証明書が発行されます。この機能によって、Cisco NX-OS デバイスは複数のピアを持つ SSH およびアプリケーションに、これらのピアに対応する CA から発行された証明書を使用して参加できるようになります。

また複数の RSA キー ペアの機能を使用すると、登録している各 CA ごとの別々のキー ペアをデバイスで持てるようになります。これは、他の CA で指定されているキーの長さなどの要件と競合することなく、各 CA のポリシー要件に適合させることができます。デバイスでは複数の RSA キー ペアを作成して、各キー ペアを別々のトラストポイントに関連付けることができます。したがって、トラストポイントに登録するときには、関連付けられたキーペアを証明書要求の作成に使用します。

ピア証明書の検証

PKI では、Cisco NX-OS デバイスでのピア証明書の検証機能をサポートしています。Cisco NX-OS では、SSH などのアプリケーションのためのセキュリティ交換の際にピアから受け取った証明書を検証します。アプリケーションはピア証明書の正当性を検証します。Cisco NX-OS ソフトウェアでは、ピア証明書の検証の際に次の手順を実行します。

- ピア証明書がローカルの信頼できる CA のいずれかから発行されていることを確認します。
- ピア証明書が現在時刻において有効であること（期限切れでない）ことを確認します。
- ピア証明書が、発行した CA によって取り消されていないことを確認します。

取消確認については、Cisco NX-OS ソフトウェアでは証明書失効リスト（CRL）をサポートしています。トラストポイント CA ではこの方法を使用して、ピア証明書が取り消されていないことを確認できます。

証明書の取消確認

Cisco NX-OS ソフトウェアでは、CA 証明書の取消のステータスを確認できます。アプリケーションでは、指定した順序に従って取消確認メカニズムを使用できます。CRL、NDcPP:OCSP for Syslog、なし、またはこれらの方式の組み合わせを指定できます。

CRL のサポート

CA では証明書失効リスト（CRL）を管理して、有効期限前に取り消された証明書についての情報を提供します。CA では CRL をリポジトリで公開して、発行したすべての証明書の中にダウンロード用の公開 URL 情報を記載しています。ピア証明書を検証するクライアントは、発行した CA から最新の CRL を入手して、これを使用して証明書が取り消されていないかどうかを確認できます。クライアントは、自身の信頼できる CA のすべてまたは一部の CRL をローカルにキャッシュして、その CRL が期限切れになるまで必要に応じて使用することができます。

Cisco NX-OS ソフトウェアでは、先にダウンロードしたトラストポイントについての CRL を手動で設定して、これをデバイスのブートフラッシュ（cert-store）にキャッシュすることができます。ピア証明書の検証の際、Cisco NX-OS ソフトウェアは、CRL がすでにローカルにキャッシュされていて、取消確認でこの CRL を使用するよう設定されている場合にだけ、発行した CA からの CRL をチェックします。それ以外の場合、Cisco NX-OS ソフトウェアでは CRL チェックを実行せず、他の取消確認方式が設定されている場合を除き、証明書は取り消されていないと見なします。

NDcPP：syslog の OCSP

Online Certificate Status Protocol（OCSP）は、ピアがこの失効情報を取得し、それを検証して証明書失効ステータスを確認する必要がある場合に、証明書失効をチェックする方法です。この

方式では、クラウドを介してOCSPレスポンダに到達するピアの機能、または証明書失効情報を取得する証明書送信者のパフォーマンスによって、証明書失効ステータスが制限されます。

リモート syslog サーバが OCSP レスポンダ URL を持つ証明書を共有すると、クライアントはサーバ証明書を外部OCSP レスポンダ（CA）サーバに送信します。CA サーバはこの証明書を検証し、有効な証明書か失効した証明書かを確認します。この場合、クライアントは失効した証明書リストをローカルに保持する必要はありません。

証明書と対応するキー ペアのインポートとエクスポート

CA 認証と登録のプロセスの一環として、下位 CA 証明書（または証明書チェーン）とアイデンティティ証明書を標準の PEM（base64）形式でインポートできます。

トラストポイントでのアイデンティティ情報全体を、パスワードで保護される PKCS#12 標準形式でファイルにエクスポートできます。このファイルは、後で同じデバイス（システム クラッシュの後など）や交換したデバイスににインポートすることができます。PKCS#12 ファイル内の情報は、RSA キー ペア、アイデンティティ証明書、および CA 証明書（またはチェーン）で構成されています。

PKI の注意事項と制約事項

PKI に関する注意事項と制約事項は次のとおりです。

- Cisco NX-OS デバイスに設定できるキー ペアの最大数は 16 です。
- Cisco NX-OS デバイスで宣言できるトラスト ポイントの最大数は 16 です。
- Cisco NX-OS デバイスに設定できるアイデンティティ証明書の最大数は 16 です。
- CA 証明書チェーン内の証明書の最大数は 10 です。
- ある CA に対して認証できるトラストポイントの最大数は 10 です。
- 設定のロールバックでは PKI の設定はサポートしていません。
- Cisco NX-OS リリース 9.3 (5) 以降では、Cisco NX-OS ソフトウェアは NDcPP: OCSP for Syslog をサポートしています。

（注） Cisco IOS の CLI に慣れている場合、この機能の Cisco NX-OS コマンドは従来の Cisco IOS コマンドと異なる点があるため注意が必要です。

PKI のデフォルト設定

次の表に、PKI パラメータのデフォルト設定を示します。

***Table 1: PKI* パラメータのデフォルト値**

パラメータ	デフォルト
トラスト ポイント	なし
RSA キー ペア	なし
RSA キー ペアのラベル	デバイスの FQDN
RSA キー ペアのモジュール	512
RSA キー ペアのエクスポートの可否	イネーブル
取消確認方式	CRL

CA の設定とデジタル証明書

ここでは、Cisco NX-OS デバイス上で CA とデジタル証明書が相互に連携して動作するようにするために、実行が必要な作業について説明します。

ホスト名と IP ドメイン名の設定

デバイスのホスト名または IP ドメイン名をまだ設定していない場合は、設定する必要があります。これは、Cisco NX-OS ソフトウェアでは、アイデンティティ証明書のサブジェクトとして完全修飾ドメイン名（FQDN）を使用するためです。また、Cisco NX-OS ソフトウェアでは、キーの作成の際にラベルが指定されていないと、デバイスの FQDN をデフォルトのキー ラベルとして使用します。たとえば、DeviceA.example.com という名前の証明書は、DeviceA というデバイスのホスト名と example.com というデバイスの IP ドメイン名に基づいています。

Caution 証明書を作成した後にホスト名または IP ドメイン名を変更すると、証明書が無効になります。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**hostname** *hostname* **Example:** `switch(config)# hostname DeviceA`	デバイスのホスト名を設定します。
ステップ 3	**ip domain-name** *name* [**use-vrf** *vrf-name*] **Example:** `DeviceA(config)# ip domain-name example.com`	デバイスのIP ドメイン名を設定します。VRF 名が指定されていないと、このコマンドではデフォルトの VRF を使用します。

ステップ 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		コンフィギュレーション モードを終了します。

ステップ 5	(Optional) **show hosts** **Example:** `switch# show hosts`	IP ドメイン名を表示します。
ステップ 6	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

RSA キー ペアの生成

RSA キーペアは、アプリケーション向けのセキュリティプロトコルの交換時に、セキュリティペイロードの署名、暗号化、および復号化のために作成します。デバイスのための証明書を取得する前に、RSA キー ペアを作成する必要があります。

Cisco NX-OS リリース 9.3(3) 以降では、Cisco NX-OS デバイスをトラスト ポイント CA に関連付ける前に、明示的に RSA キー ペアを生成する必要があります。Cisco NX-OS リリース 9.3(3) よりも前では、使用できない場合、RSAキー ペアは自動生成されます。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します

ステップ 2	**crypto key generate rsa** [**label** *label-string*] [**exportable**] [**modulus** *size*] **Example:** `switch(config)# crypto key generate rsa exportable`	RSA キー ペアを生成します。デバイスに設定できるキー ペアの最大数は 16 です。 ラベル文字列には、大文字と小文字を区別して、最大 64 文字の英数字で値を指定します。デフォルトのラベル文字列は、ピリオド文字（.）で区切ったホスト名と FQDN です。 有効なモジュラスの値は 512、768、1024、1536、および 2048 です。デフォルトのモジュラスのサイズは 512 です。 **Note** 適切なキーのモジュラスを決定する際には、Cisco NX-OS デバイスと CA（登録を計画している対象）のセキュリティ ポリシーを考慮する必要があります。 デフォルトでは、キーペアはエクスポートできません。エクスポート可能なキーペアだけ、PKCS#12 形式でエクスポートできます。 **Caution** キー ペアのエクスポートの可否は変更できません。

ステップ 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		コンフィギュレーション モードを終了します。

ステップ 4	(Optional) **show crypto key mypubkey rsa** **Example:** `switch# show crypto key mypubkey rsa`	作成したキーを表示します。

	Command or Action	Purpose
ステップ 5	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

トラストポイント CA のアソシエーションの作成

Cisco NX-OS デバイスとトラスト ポイント CA を関連付ける必要があります。

Before you begin

RSA キー ペアを作成します。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**crypto ca trustpoint** *name*
	Example:
	`switch(config)# crypto ca trustpoint admin-ca`
	`switch(config-trustpoint)#`
		デバイスが信頼するトラストポイント CA を宣言し、トラストポイント コンフィギュレーション モードを開始します。
		Note デバイスに設定できるトラストポイントの最大数は 16 です。

ステップ 3	**cabundle** *baselabel* **Example:** `switch(config-trustpoint)# cabundle test`	特定のベースラベルの下にトラストポイントをグループ化します。また、設定されたベースラベルを持つCAバンドルからトラストポイントが生成されることを示します。
ステップ 4	**enrollment terminal** **Example:** `switch(config-trustpoint)# enrollment terminal`	手動でのカットアンドペーストによる証明書の登録をイネーブルにします。デフォルトではイネーブルになっています。

	Command or Action	Purpose
		Note Cisco NX-OS ソフトウェアでは、手動でのカットアンドペースト方式による証明書の登録だけをサポートしています。
ステップ 5	**rsakeypair** *label* **Example:** `switch(config-trustpoint)# rsakeypair SwitchA`	RSA キー ペアのラベルを指定して、このトラストポイントを登録用に関連付けます。 **Note** CA ごとに 1 つの RSA キー ペアだけを指定できます。

ステップ 6	**exit**
	Example:
	`switch(config-trustpoint)# exit`
	`switch(config)#`
		トラストポイント コンフィギュレーション モードを終了します。

ステップ 7	(Optional) **show crypto ca trustpoints** **Example:** `switch(config)# show crypto ca trustpoints`	トラストポイントの情報を表示します。
ステップ 8	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

Related Topics

RSA キー ペアの生成 （12 ページ）

CA の認証

CA が Cisco NX-OS デバイスに対して認証されると、CA を信頼するプロセスの設定が完了します。まず、PEM 形式の CA の自己署名証明書を入手し、Cisco NX-OS デバイスを CA に対して認証する必要があります。この証明書には、CA の公開キーが含まれています。この CA の証明書は自己署名（CA が自身の証明書に署名したもの）であるため、CA の公開キーは、CA アドミニストレータに連絡し、CA 証明書のフィンガープリントを比較して手動で認証する必要があります。

Note 認証する CA が他の CA の下位 CA である場合、認証する CA は自己署名 CA ではありません。その上位の CA がさらに別の CA の下位である場合もあります。最終的には自己署名 CA に到達します。このタイプの CA 証明書を、認証する CA の *CA* 証明書チェーンと呼びます。この場合は、CA 認証の際に、証明書チェーン内のすべての CA の CA 証明書の完全なリストを入力する必要があります。CA 証明書チェーン内の証明書の最大数は 10 です。

Before you begin

CA とのアソシエーションを作成します。

CA 証明書または CA 証明書チェーンを入手します。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します

ステップ 2	**crypto ca authenticate** *name* **pemfile** *uri0*
	
	Example:
	`switch(config)# crypto ca authenticate admin-ca`
	`input (cut & paste) CA certificate (chain) in PEM format;`
	`end the input with a line containing only END OF INPUT :`
	`-----BEGIN CERTIFICATE-----`
	`MIIC4jCCAoygAwIBAgIQBWDSiay0GZRPSRI1jK0ZejANBgkqhkiG9w0BAQUFADCB`
	`kDEgMB4GCSqGSIb3DQEJARYRYW1hbmRrZUBjaXNjby5jb20xCzAJBgNVBAYTAklO`
	`MRIwEAYDVQQIEwlLYXJuYXRha2ExEjAQBgNVBAcTCUJhbmdhbG9yZTEOMAwGA1UE`
	`ChMFQ2lzY28xEzARBgNVBAsTCm5ldHN0b3JhZ2UxEjAQBgNVBAMTCUFwYXJuYSBD`
	`QTAeFw0wNTA1MDMyMjQ2MzdaFw0wNzA1MDMyMjU1MTdaMIGQMSAwHgYJKoZIhvcN`
	`AQkBFhFhbWFuZGtlQGNpc2NvLmNvbTELMAkGA1UEBhMCSU4xEjAQBgNVBAgTCUth`
	`cm5hdGFrYTESMBAGA1UEBxMJQmFuZ2Fsb3JlMQ4wDAYDVQQKEwVDaXNjbzETMBEG`
	`A1UECxMKbmV0c3RvcmFnZTESMBAGA1UEAxMJQXBhcm5hIENBMFwwDQYJKoZIhvcN`
	`AQEBBQADSwAwSAJBAMW/7b3+DXJPANBsIHHzluNccNM87ypyzwuoSNZXOMpeRXXI`
	`OzyBAgiXT2ASFuUOwQ1iDM8rO/41jf8RxvYKvysCAwEAAaOBvzCBvDALBgNVHQ8E`
	`BAMCAcYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUJyjyRoMbrCNMRU2OyRhQ`
	`GgsWbHEwawYDVR0fBGQwYjAuoCygKoYoaHR0cDovL3NzZS0wOC9DZXJ0RW5yb2xs`
	`L0FwYXJuYSUyMENBLmNybDAwoC6gLIYqZmlsZTovL1xcc3NlLTA4XENlcnRFbnJv`
	`bGxcQXBhcm5hJTIwQ0EuY3JsMBAGCSsGAQQBgjcVAQQDAgEAMA0GCSqGSIb3DQEB`
	`BQUAA0EAHv6UQ+8nE399Tww+KaGr0g0NIJaqNgLh0AFcT0rEyuyt/WYGPzksF9Ea`
	`NBG7E0oN66zex0EOEfG1Vs6mXp1//w==`
	`-----END CERTIFICATE-----`
	`END OF INPUT`
	`Fingerprint(s): MD5 Fingerprint=65:84:9A:27:D5:71:03:33:9C:12:23:92:38:6F:78:12`
	`Do you accept this certificate? [yes/no]: yes`
		CA の証明書をカットアンドペーストするようプロンプトが表示されます。CA を宣言したときに使用した名前と同じ名前を使用します。
		
		また、CAチェーンを検証し、指定されたトラストポイントに直接接続します。
		
		ある CA に対して認証できるトラストポイントの最大数は 10 です。
		
		Note 下位 CA の認証の場合、Cisco NX-OS ソフトウェアでは、自己署名 CA に到達する CA 証明書の完全なチェーンが必要になります。これは証明書の検証や PKCS#12 形式でのエクスポートに CA チェーンが必要になるためです。

	Command or Action	Purpose

ステップ 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		コンフィギュレーション モードを終了します。

ステップ 4	(Optional) **show crypto ca trustpoints** **Example:** `switch# show crypto ca trustpoints`	トラストポイント CA の情報を表示します。
ステップ 5	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

Related Topics

トラストポイント CA のアソシエーションの作成 （14 ページ）

証明書取消確認方法の設定

クライアント（SSH ユーザなど）とのセキュリティ交換の際に、Cisco NX-OS デバイスは、クライアントから送られたピア証明書の検証を実行します。検証プロセスには、証明書の取消状況の確認が含まれます。

CA からダウンロードした CRL を確認するよう、デバイスに設定できます。CRL のダウンロードとローカルでの確認では、ネットワーク上にトラフィックは発生しません。しかし、証明書がダウンロードとダウンロードの中間で取り消され、デバイス側ではその取り消しに気付かない場合も考えられます。

Before you begin

CA を認証します。

CRL チェックを使用する場合は、CRL が設定済みであることを確認します。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

	Command or Action	Purpose

ステップ 2	**crypto ca trustpoint** *name*
	Example:
	`switch(config)# crypto ca trustpoint admin-ca`
	`switch(config-trustpoint)#`
		トラストポイントCAを指定し、トラストポイント コンフィギュレーション モードを開始します。

ステップ 3	**revocation-check** {**crl** [**none**] | **none**} **Example:** `switch(config-trustpoint)# revocation-check none`	証明書取消確認方法を設定します。デフォルトの方式は **crl**. です。 Cisco NX-OS ソフトウェアでは、指定した順序に従って証明書取消方式を使用します。

ステップ 4	**exit**
	Example:
	`switch(config-trustpoint)# exit`
	`switch(config)#`
		トラストポイントコンフィギュレーション モードを終了します。

ステップ 5	(Optional) **show crypto ca trustpoints** **Example:** `switch(config)# show crypto ca trustpoints`	トラストポイントCAの情報を表示します。
ステップ 6	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

Related Topics

CA の認証 （15 ページ）
CRL の設定 （24 ページ）

証明書要求の作成

使用する各デバイスの RSA キー ペア用に、対応するトラストポイント CA からアイデンティティ証明書を入手するために、要求を作成する必要があります。その後、表示された要求を CA 宛の E メールまたは Web サイトのフォームにカットアンドペーストします。

Before you begin

CA とのアソシエーションを作成します。

CA 証明書または CA 証明書チェーンを入手します。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**crypto ca enroll** *name*
	Example:
	`switch(config)# crypto ca enroll admin-ca`
	`Create the certificate request ..`
	`Create a challenge password. You will need to verbally provide this password to the CA Administrator in order to revoke your certificate. For security reasons your password will not be saved in the configuration.`
	`Please make a note of it.`
	`Password:nbv123`
	`The subject name in the certificate will be: DeviceA.cisco.com`
	`Include the switch serial number in the subject name? [yes/no]: no`
	`Include an IP address in the subject name [yes/no]: yes`
	`ip address:172.22.31.162`
	`The certificate request will be displayed...`
	`-----BEGIN CERTIFICATE REQUEST-----`
	`MIIBqzCCARQCAQAwHDEaMBgGA1UEAxMRVmVnYXMtMS5jaXNjby5jb20wgZ8wDQYJ`
	`KoZIhvcNAQEBBQADgY0AMIGJAoGBAL8YlIA12NC7jU1lDaSMqNIgI2kt8rl4lKY`
	`0JC1MnNy4qk8VeMXSiLJAJfIZWbkIDkTysrjuCXGvjb+wj0Fhv/y5lT9y`
	`P2NU8ornqShrvFZgC7ysNVPJMrKcgzhiVpj+rarqZxHtGJ9lXIqAVbVkSCzX8S`
	`VqHOvBxAgMBAAGgITzAVBgkqhkiG9w0BCQcxCBMGbmZ1MTIxMDYGCSqGSIb3DQEJ`
	`DjEpMCowJQYDVR0RAQH/BBswGYIRVmVnYXMtMS5jaXNjby5jb22HBKwWH6IwDQYJ`
	`KoZIhvcNAQEEBQADgYEAkI6OkFRGQo8rj0sDXZMSfIZhGR6JtDz3Gc99GlFWgt`
	`PfuNbWLE/pwGHayfQl2T3ecgNwel2dl5l33YEF2kkExiI6Jl88rIOjgIMjja8`
	`8a23bNpNsMrdklwAGfWkrVL8NUZEF3kdpjfrgPNZacJUS8ZqKMetkKytUkO=`
	`-----END CERTIFICATE REQUEST-----`
		認証したCAに対する証明書要求を作成します。
		Note チャレンジ パスワードを記憶しておいてください。このパスワードは設定と一緒に保存されません。証明書を取り消す必要がある場合には、このパスワードを入力する必要があります。

ステップ 3	**exit**
	Example:
	`switch(config-trustpoint)# exit`
	`switch(config)#`
		トラストポイント コンフィギュレーション モードを終了します。

ステップ 4	(Optional) **show crypto ca certificates** **Example:** `switch(config)# show crypto ca certificates`	CA 証明書を表示します。
ステップ 5	(Optional) **copy running-config startup-config** **Example:**	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

	Command or Action	Purpose
	`switch(config)# copy running-config startup-config`	

Related Topics

トラストポイント CA のアソシエーションの作成 （14 ページ）

アイデンティティ証明書のインストール

アイデンティティ証明書は、CA から E メールまたは Web ブラウザ経由で base64 でエンコードされたテキスト形式で受信できます。CA から入手したアイデンティティ証明書を、エンコードされたテキストをカットアンドペーストしてインストールする必要があります。

Before you begin

CA とのアソシエーションを作成します。

CA 証明書または CA 証明書チェーンを入手します。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**crypto ca import** *name* **certificate**
	
	Example:
	`switch(config)# crypto ca import admin-ca certificate`
	`input (cut & paste) certificate in PEM format:`
	`-----BEGIN CERTIFICATE-----`
	`MIIEADCCA6qgAwIBAgIKCjOOoQAAAAAAdDANBgkqhkiG9w0BAQUFADCBkDEgMB4G`
	`CSqGSIb3DQEJARYRYW1hbmRrZUBjaXNjby5jb20xCzAJBgNVBAYTAklOMRIwEAYD`
	`VQQIEwlLYXJuYXRha2ExEjAQBgNVBAcTCUJhbmdhbG9yZTEOMAwGA1UEChMFQ2lz`
	`Y28xEzARBgNVBAsTCm5ldHN0b3JhZ2UxEjAQBgNVBAMTCUFwYXJuYSBDQTAeFw0w`
	`NTExMTIwMzAyNDBaFw0wNjExMTIwMzEyNDBaMBwxGjAYBgNVBAMTEVZlZ2FzLTEu`
	`Y2lzY28uY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC/GNVACdjQu41C`
	`dQ1WkjKjSICdpLfK5eJSmNCQujGpzcuKsZPFXjF2UbiyeCYE8ylncWyw5E08rJ47`
	`glxr42/sI9IRIb/8udU/cj9jSSfKK56koa7xWYAu8rDfz8jMCnIM4W1aY/q2q4Gb`
	`x7RifdV06uFqFZEgs17/Elash9LxLwIDAQABo4ICEzCCAg8wJQYDVR0RAQH/BBsw`
	`GYIRVmVnYXMtMS5jaXNjby5jb22HBKwWH6IwHQYDVR0OBBYEFKCLi+2sspWEfgrR`
	`bhWmlVyo9jngMIHMBgNVHSMEgcQwgcGAFCco8kaDG6wjTEVNjskYUBoLFmxxoYGW`
	`pIGTMIGQMSAwHgYJKoZIhvcNAQkBFhFhbWFuZGtlQGNpc2NvLmNvbTELMAkGA1UE`
	`BhMCSU4xEjAQBgNVBAgTCUthcm5hdGFrYTESMBAGA1UEBxMJQmFuZ2Fsb3JlMQ4w`
	`DAYDVQQKEwVDaXNjbzETMBEGA1UECxMKbmV0c3RvcmFnZTESMBAGA1UEAxMJQXBh`
	`cm5hIENBghAFYNKJrLQZlE9JEiWMrRl6MGsGA1UdHwRkMGIwL6AtoCuGKWh0dHA6`
	`Ly9zc2UtMDgvQ2VydEVucm9sbC9BcGFybmElMjBDQS5jcmwwL6AtoCuGKWZpbGU6`
	`Ly9cXHNzZS0wOFxDZXJ0RW5yb2xsXEFwYXJuYSUyMENBLmNybDCBigYIKwYBBQUH`
	`AQEEfjB8MDsGCCsGAQUFBzAChi9odHRwOi8vc3NlLTA4L0NlcnRFbnJvbGwvc3Nl`
	`LTA4X0FwYXJuYSUyMENBLmNydDA9BggrBgEFBQcwAoYxZmlsZTovL1xcc3NlLTA4`
		admin-ca という名前の CA に対するアイデンティティ証明書をカットアンドペーストするよう、プロンプトが表示されます。
		
		デバイスに設定できるアイデンティティ証明書の最大数は 16 です。

	Command or Action	Purpose
	XENlcRBnJUGcc3NLIAMCEwXUuSUyMENEmNycDANBgkqhkiG9w0BAQUF AANBADcBGde7GNh9eOWBNm24U9ZSiDCcUZUUtgqnTkMPyejtsyflw E36cIZu4WsExREqxbTk8ycx7V5o= -----END CERTIFICATE-----	

ステップ 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		設定モードを終了します。

ステップ 4	(Optional) **show crypto ca certificates** **Example:** `switch# show crypto ca certificates`	CA 証明書を表示します。
ステップ 5	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

Related Topics

トラストポイント CA のアソシエーションの作成 （14 ページ）

トラストポイントの設定がリブート後も維持されていることの確認

トラストポイントの設定が、Cisco NX-OS デバイスのリブート後も維持されていることを確認できます。

トラストポイントの設定は、通常の Cisco NX-OS デバイスの設定であり、スタートアップ コンフィギュレーションに確実にコピーした場合にだけ、システムのリブート後も維持されます。トラスト ポイント設定をスタートアップ コンフィギュレーションにコピーしておけば、トラスト ポイントに関連する証明書、キー ペア、および CRL が自動的に保持されます。逆に、トラスト ポイントがスタートアップ コンフィギュレーションにコピーされていないと、証明書、キー ペア、および関連 CRL は保持されません。リブート後に、対応するトラスト ポイント設定が必要になるからです。設定した証明書、キー ペア、および CRL を確実に保持するために、必ず、実行コンフィギュレーションをスタートアップ コンフィギュレーションにコピーしてください。また、証明書またはキーペアを削除した後は実行コンフィギュレーションを保存して、削除が永続的に反映されるようにしてください。

トラストポイントに関連付けられた証明書と CRL は、そのトラストポイントがすでにスタートアップ コンフィギュレーションに保存されていれば、インポートした時点で（つまりスタートアップ コンフィギュレーションにコピーしなくても）維持されるようになります。

パスワードで保護したアイデンティティ証明書のバックアップを作成して、これを外部のサーバに保存することを推奨します。

Note コンフィギュレーションを外部サーバにコピーすると、証明書およびキー ペアも保存されます。

Related Topics

PKCS 12 形式でのアイデンティティ情報のエクスポート （22 ページ）

PKCS 12 形式でのアイデンティティ情報のエクスポート

アイデンティティ証明書を、トラストポイントの RSA キー ペアや CA 証明書（または下位 CA の場合はチェーン全体）と一緒に PKCS#12 ファイルにバックアップ目的でエクスポートすることができます。デバイスのシステム クラッシュからの復元の際や、スーパーバイザ モジュールの交換の際には、証明書や RSA キー ペアをインポートすることができます。

Note エクスポートの URL を指定するときに使用できるのは、bootflash:*filename* という形式だけです。

Before you begin

CA を認証します。

アイデンティティ証明書をインストールします。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します

ステップ 2	**crypto ca export** *name* **pkcs12** **bootflash:***filename password* **Example:** `switch(config)# crypto ca export admin-ca pkcs12 bootflash:adminid.p12 nbv123`	アイデンティティ証明書と、トラストポイント CA の対応するキー ペアと CA 証明書をエクスポートします。パスワードには、大文字と小文字を区別して、最大 128 文字の英数字で値を指定します。

ステップ 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		コンフィギュレーション モードを終了します。

	Command or Action	Purpose
ステップ 4	**copy bootflash:***filename scheme***://***server/* [*url /*]*filename* **Example:** `switch# copy bootflash:adminid.p12 tftp:adminid.p12`	PKCS#12 形式のファイルをリモート サーバにコピーします。 *scheme* 引数に対しては、**tftp:**、**ftp:**、**scp:**、または **sftp:** を入力できます。*server* 引数は、リモート サーバのアドレスまたは名前であり、*url* 引数はリモート サーバにあるソース ファイルへのパスです。 *server*、*url*、および *filename* の各引数は、大文字小文字を区別して入力します。

Related Topics

RSA キー ペアの生成 （12 ページ）
CA の認証 （15 ページ）
アイデンティティ証明書のインストール （20 ページ）

PKCS 12 形式でのアイデンティティ情報のインポート

デバイスのシステム クラッシュからの復元の際や、スーパーバイザ モジュールの交換の際には、証明書や RSA キー ペアをインポートすることができます。

Note インポートの URL を指定するときに使用できるのは、bbootflash:*filename* fという形式だけです。

Before you begin

CA 認証によってトラストポイントに関連付けられている RSA キー ペアがないこと、およびトラストポイントに関連付けられている CA がないことを確認して、トラストポイントが空であるようにします。

Procedure

	Command or Action	Purpose
ステップ 1	**copy** *scheme***://** *server/*[*url /*]*filename* **bootflash:***filename* **Example:** `switch# copy tftp:adminid.p12 bootflash:adminid.p12`	PKCS#12 形式のファイルをリモート サーバからコピーします。 *scheme* 引数に対しては、**tftp:**、**ftp:**、**scp:**、または **sftp:** を入力できます。*server* 引数は、リモート サーバのアドレスまたは名前であり、*url* 引数はリモー

	Command or Action	Purpose
		トサーバにあるソース ファイルへのパスです。 *server*、*url*、および *filename* の各引数は、大文字小文字を区別して入力します。

ステップ 2	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します

ステップ 3	**crypto ca import** *name* **pksc12 bootflash:***filename* **Example:** `switch(config)# crypto ca import admin-ca pkcs12 bootflash:adminid.p12 nbv123`	アイデンティティ証明書と、トラストポイントCAの対応するキーペアとCA証明書をインポートします。

ステップ 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		設定モードを終了します。

ステップ 5	(Optional) **show crypto ca certificates** **Example:** `switch# show crypto ca certificates`	CA 証明書を表示します。
ステップ 6	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

CRL の設定

トラストポイントからダウンロードしたCRLを手動で設定することができます。Cisco NX-OS ソフトウェアでは、CRLをデバイスのブートフラッシュ（cert-store）にキャッシュします。ピア証明書の検証の際、Cisco NX-OS ソフトウェアが発行したCAからのCRLをチェックするのは、CRL をデバイスにダウンロードしていて、この CRL を使用する証明書取消確認を設定している場合だけです。

Before you begin

証明書取消確認がイネーブルになっていることを確認します。

Procedure

	Command or Action	Purpose
ステップ 1	**copy** *scheme*:[//*server*/[*url* /]]*filename* **bootflash:***filename* **Example:** `switch# copy tftp:adminca.crl bootflash:adminca.crl`	リモート サーバから CRL をダウンロードします。 *scheme* 引数に対しては、**tftp:**、**ftp:**、**scp:**、または **sftp:** を入力できます。*server* 引数は、リモート サーバのアドレスまたは名前であり、*url* 引数はリモート サーバにあるソース ファイルへのパスです。 *server*、*url*、および *filename* の各引数は、大文字小文字を区別して入力します。
ステップ 2	**configure terminal** **Example:** `switch# configure terminal switch(config)#`	グローバル コンフィギュレーション モードを開始します
ステップ 3	**crypto ca crl request** *name* **bootflash:***filename* **Example:** `switch(config)# crypto ca crl request admin-ca bootflash:adminca.crl`	ファイルで指定されている CRL を設定するか、現在の CRL と置き換えます。
ステップ 4	**exit** **Example:** `switch(config)# exit switch#`	コンフィギュレーション モードを終了します。
ステップ 5	(Optional) **show crypto ca crl** *name* **Example:** `switch# show crypto ca crl admin-ca`	CA の CRL 情報を表示します。
ステップ 6	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

CA の設定からの証明書の削除

トラストポイントに設定されているアイデンティティ証明書やCA証明書を削除できます。最初にアイデンティティ証明書を削除し、その後でCA証明書を削除します。アイデンティティ

証明書を削除した後で、RSA キー ペアとトラストポイントの関連付けを解除できます。証明書の削除は、期限切れになった証明書や取り消された証明書、破損した（あるいは破損したと思われる）キー ペア、現在は信頼されていない CA を削除するために必要です。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**crypto ca trustpoint** *name*
	Example:
	`switch(config)# crypto ca trustpoint admin-ca`
	`switch(config-trustpoint)#`
		トラストポイント CA を指定し、トラストポイント コンフィギュレーション モードを開始します。

ステップ 3	**delete ca-certificate** **Example:** `switch(config-trustpoint)# delete ca-certificate`	CA 証明書または証明書チェーンを削除します。
ステップ 4	**delete certificate** [**force**] **Example:** `switch(config-trustpoint)# delete certificate`	アイデンティティ証明書を削除します。 削除しようとしているアイデンティティ証明書が証明書チェーン内の最後の証明書である場合や、デバイス内の唯一のアイデンティティ証明書である場合は、**force** オプションを使用する必要があります。この要件は、証明書チェーン内の最後の証明書や唯一のアイデンティティ証明書を誤って削除してしまい、アプリケーション（SSH など）で使用する証明書がなくなってしまうことを防ぐために設けられています。

ステップ 5	**exit**
	Example:
	`switch(config-trustpoint)# exit`
	`switch(config)#`
		トラストポイント コンフィギュレーション モードを終了します。

ステップ 6	(Optional) **show crypto ca certificates** [*name*] **Example:** `switch(config)# show crypto ca certificates admin-ca`	CA の証明書情報を表示します。

	Command or Action	Purpose
ステップ 7	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

Cisco NX-OSデバイスからの RSA キー ペアの削除

RSAキーペアが何らかの理由で破損し、現在は使用されてないと見られるときには、そのRSAキー ペアを Cisco NX-OS デバイスから削除することができます。

Note デバイスからRSAキーペアを削除した後、CAアドミニストレータに、そのCAにあるこのデバイスの証明書を取り消すよう依頼します。その証明書を最初に要求したときに作成したチャレンジ パスワードを入力する必要があります。

Procedure

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します

ステップ 2	**crypto key zeroize rsa** *label* **Example:** `switch(config)# crypto key zeroize rsa MyKey`	RSA キー ペアを削除します。

ステップ 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		コンフィギュレーション モードを終了します。

ステップ 4	(Optional) **show crypto key mypubkey rsa** **Example:** `switch# show crypto key mypubkey rsa`	RSA キー ペアの設定を表示します。
ステップ 5	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

Related Topics

証明書要求の作成 （18 ページ）

PKI の設定の確認

PKI 設定情報を表示するには、次のいずれかの作業を行います。

コマンド	目的
show crypto key mypubkey rsa	Cisco NX-OS デバイスで作成された RSA 公開キーの情報を表示します。
show crypto ca certificates	CA とアイデンティティ証明書についての情報を表示します。
show crypto ca crl	CA の CRL についての情報を表示します。
show crypto ca trustpoints	CA トラストポイントについての情報を表示します。

PKI の設定例

ここでは、Microsoft Windows Certificate サーバを使用して Cisco NX-OS デバイスで証明書と CRL を設定する作業の例について説明します。

Note デジタル証明書の作成には、どのようなタイプのサーバでも使用できます。Microsoft Windows Certificate サーバに限られることはありません。

Cisco NX-OS デバイスでの証明書の設定

Cisco NX-OS デバイスで証明書を設定するには、次の手順に従ってください。

Procedure

ステップ 1 デバイスの FQDN を設定します。

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# hostname Device-1
```

```
Device-1(config)#
```

ステップ 2 デバイスの DNS ドメイン名を設定します。

```
Device-1(config)# ip domain-name cisco.com
```

ステップ 3 トラストポイントを作成します。

```
Device-1(config)# crypto ca trustpoint myCA
Device-1(config-trustpoint)# exit
Device-1(config)# show crypto ca trustpoints
trustpoint: myCA; key:
revokation methods:  crl
```

ステップ 4 このデバイス用の RSA キー ペアを作成します。

```
Device-1(config)# crypto key generate rsa label myKey exportable modulus 1024
Device-1(config)# show crypto key mypubkey rsa
key label: myKey
key size: 1024
exportable: yes
```

ステップ 5 RSA キー ペアとトラストポイントを関連付けます。

```
Device-1(config)# crypto ca trustpoint myCA
Device-1(config-trustpoint)# rsakeypair myKey
Device-1(config-trustpoint)# exit
Device-1(config)# show crypto ca trustpoints
trustpoint: myCA; key: myKey
revokation methods:  crl
```

ステップ 6 Microsoft Certificate Service の Web インターフェイスから CA をダウンロードします。

ステップ 7 トラストポイントに登録する CA を認証します。

```
Device-1(config)# crypto ca authenticate myCA
input (cut & paste) CA certificate (chain) in PEM format;
end the input with a line containing only END OF INPUT :
-----BEGIN CERTIFICATE-----
MIIC4jCCAoygAwIBAgIQBWDSiay0GZRPSRIljK0ZejANBgkqhkiG9w0BAQUFADCB
kDEgMB4GCSqGSIb3DQEJARYRYW1hbmRrZUBjaXNjby5jb20xCzAJBgNVBAYTAklO
MRIwEAYDVQQIEwlLYXJuYXRha2ExEjAQBgNVBAcTCUJhbmdhbG9yZTEOMAwGA1UE
ChMFQ2lzY28xEzARBgNVBAsTCm5ldHN0b3JhZ2UxEjAQBgNVBAMTCUFwYXJuYSBD
QTAeFw0wNTA1MDMyMjQ2MzdaFw0wNzA1MDMyMjU1MTdaMIGQMSAwHgYJKoZIhvcN
AQkBFhFhbWFuZGtlQGNpc2NvLmNvbTELMAkGA1UEBhMCSU4xEjAQBgNVBAgTCUth
cm5hdGFrYTESMBAGA1UEBxMJQmFuZ2Fsb3JlMQ4wDAYDVQQKEwVDaXNjbzETMBEG
A1UECxMKbmV0c3RvcmFnZTESMBAGA1UEAxMJQXBhcm5hIENBMFwwDQYJKoZIhvcN
AQEBBQADSwAwSAJBAMW/7b3+DXJPANBsIHHzluNccNM87ypyzwuoSNZXOMpeRXXI
OzyBAgiXT2ASFuUOwQ1iDM8rO/41jf8RxvYKvysCAwEAAaOBvzCBvDALBgNVHQ8E
BAMCAcYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUJyjyRoMbrCNMRU2OyRhQ
GgsWbHEwawYDVR0fBGQwYjAuoCygKoYoaHR0cDovL3NzZS0wOC9DZXJ0RW5yb2xs
L0FwYXJuYSUyMENBLmNybDAwoC6gLIYqZmlsZTovL1xcc3NlLTA4XENlcnRFbnJv
bGxcQXBhcm5hJTIwQ0EuY3JsMBAGCSsGAQQBgjcVAQQDAgEAMA0GCSqGSIb3DQEB
BQUAA0EAHv6UQ+8nE399Tww+KaGr0g0NIJaqNgLh0AFcT0rEyuyt/WYGPzksF9Ea
NBG7E0oN66zex0EOEfG1Vs6mXp1//w==
-----END CERTIFICATE-----
END OF INPUT
Fingerprint(s): MD5 Fingerprint=65:84:9A:27:D5:71:03:33:9C:12:23:92:38:6F:78:12
Do you accept this certificate? [yes/no]:y
```

```
Device-1(config)# show crypto ca certificates
Trustpoint: myCA
CA certificate 0:
subject= /emailAddress=admin@yourcompany.com/C=IN/ST=Karnataka/
L=Bangalore/O=Yourcompany/OU=netstorage/CN=Aparna CA
issuer= /emailAddress=admin@yourcompany.com/C=IN/ST=Karnataka/
L=Bangalore/O=Yourcompany/OU=netstorage/CN=Aparna CA
serial=0560D289ACB419944F4912258CAD197A
notBefore=May  3 22:46:37 2005 GMT
notAfter=May  3 22:55:17 2007 GMT
MD5 Fingerprint=65:84:9A:27:D5:71:03:33:9C:12:23:92:38:6F:78:12
purposes: sslserver sslclient ike
```

ステップ 8 トラストポイントに登録するために使用する証明書要求を作成します。

```
Device-1(config)# crypto ca enroll myCA
 Create the certificate request ..
 Create a challenge password. You will need to verbally provide this
  password to the CA Administrator in order to revoke your certificate.
  For security reasons your password will not be saved in the configuration.
  Please make a note of it.
  Password: nbv123
 The subject name in the certificate will be: Device-1.cisco.com
 Include the switch serial number in the subject name? [yes/no]: no
 Include an IP address in the subject name [yes/no]: yes
ip address: 10.10.1.1
 The certificate request will be displayed...
-----BEGIN CERTIFICATE REQUEST-----
MIIBqzCCARQCAQAwHDEaMBgGA1UEAxMRVmVnYXMtMS5jaXNjby5jb20wgZ8wDQYJ
KoZIhvcNAQEBBQADgY0AMIGJAoGBAL8Y1UAJ2NC7jUJ1DVaSMqNIgJ2kt8rl4lKY
0JC6ManNy4qxk8VeMXZSiLJ4JgTzKWdxbLDkTTysnjuCXGvjb+wj0hEhv/y51T9y
P2NJJ8ornqShrvFZgC7ysN/PyMwKcgzhbVpj+rargZvHtGJ91XTq4WoVkSCzXv8S
VqyH0vEvAgMBAAGgTzAVBgkqhkiG9w0BCQcxCBMGbmJ2MTIzMDYGCSqGSIb3DQEJ
DjEpMCcwJQYDVR0RAQH/BBswGYIRVmVnYXMtMS5jaXNjby5jb22HBKwWH6IwDQYJ
KoZIhvcNAQEEBQADgYEAkT60KER6Qo8nj0sDXZVHSfJZh6K6JtDz3Gkd99GlFWgt
PftrNcWUE/pw6HayfQl2T3ecgNwel2d15133YBF2bktExiI6Ul88nTOjglXMjja8
8a23bNDpNsM8rklwA6hWkrVL8NUZEFJxqbjfngPNTZacJCUS6ZqKCMetbKytUx0=
-----END CERTIFICATE REQUEST-----
```

ステップ 9 Microsoft Certificate Service の Web インターフェイスからアイデンティティ証明書を要求します。

ステップ 10 アイデンティティ証明書をインポートします。

```
Device-1(config)# crypto ca import myCA certificate
input (cut & paste) certificate in PEM format:
-----BEGIN CERTIFICATE-----
MIIEADCCA6qgAwIBAgIKCjOOoQAAAAAAdDANBgkqhkiG9w0BAQUFADCBkDEgMB4G
CSqGSIb3DQEJARYRYW1hbmRrZUBjaXNjby5jb20xCzAJBgNVBAYTAklOMRIwEAYD
VQQIEwlLYXJuYXRha2ExEjAQBgNVBAcTCUJhbmdhbG9yZTEOMAwGA1UEChMFQ2lz
Y28xEzARBgNVBAsTCm5ldHN0b3JhZ2UxEjAQBgNVBAMTCUFwYXJuYSBDQTAeFw0w
NTExMTIwMzAyNDBaFw0wNjExMTIwMzEyNDBaMBwxGjAYBgNVBAMTEVZlZ2FzLTEu
Y2lzY28uY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC/GNVACdjQu41C
dQ1WkjKjSICdpLfK5eJSmNCQujGpzcuKsZPFXjF2UoiyeCYE8ylncWyw5E08rJ47
glxr42/sI9IRIb/8udU/cj9jSSfKK56koa7xWYAu8rDfz8jMCnIM4W1aY/q2q4Gb
x7RifdV06uFqFZEgs17/Elash9LxLwIDAQABo4ICEzCCAg8wJQYDVR0RAQH/BBsw
GYIRVmVnYXMtMS5jaXNjby5jb22HBKwWH6IwHQYDVR0OBBYEFKCLi+2sspWEfgrR
bhWmlVyo9jngMIHMBgNVHSMEgcQwgcGAFCco8kaDG6wjTEVNjskYUBoLFmxxoYGW
pIGTMIGQMSAwHgYJKoZIhvcNAQkBFhFhbWFuZGtlQGNpc2NvLmNvbTELMAkGA1UE
BhMCSU4xEjAQBgNVBAgTCUthcm5hdGFrYTESMBAGA1UEBxMJQmFuZ2Fsb3JlMQ4w
```

```
DAYDVQQKEwVDaXNjbzETMBEGA1UECxMKbmV0c3RvcmFnZTESMBAGA1UEAxMJQXBh
cm5hIENBghAFYNKJrLQZlE9JEiWMrRl6MGsGA1UdHwRkMGIwLqAsoCqGKGh0dHA6
Ly9zc2UtMDgvQ2VydEVucm9sbC9BcGFybmElMjBDQS5jcmwwMKAuoCyGKmZpbGU6
Ly9cXHNzZS0wOFxDZXJ0RW5yb2xsXEFwYXJuYSUyMENBLmNybDCBigYIKwYBBQUH
AQEEfjB8MDsGCCsGAQUFBzAChi9odHRwOi8vc3NlLTA4L0NlcnRfbnJvbGwvc3Nl
LTA4X0FwYXJuYSUyMENBLmNydDA9BggrBgEFBQcwAoYxZmlsZTovL1xcc3NlLTA4
XENlcnRfbnJvbGxcc3NlLTA4X0FwYXJuYSUyMENBLmNydDANBgkqhkiG9w0BAQUF
AANBADbGBGsbe7GNLh9xeOTWBNbm24U69ZSuDDcOcUZUUTgrpnTqVpPyejtsyflw
E36cIZu4WsExREqxbTk8ycx7V5o=
-----END CERTIFICATE-----
Device-1(config)# exit
Device-1#
```

ステップ 11 証明書の設定を確認します。

ステップ 12 証明書の設定をスタートアップ コンフィギュレーションに保存します。

Related Topics

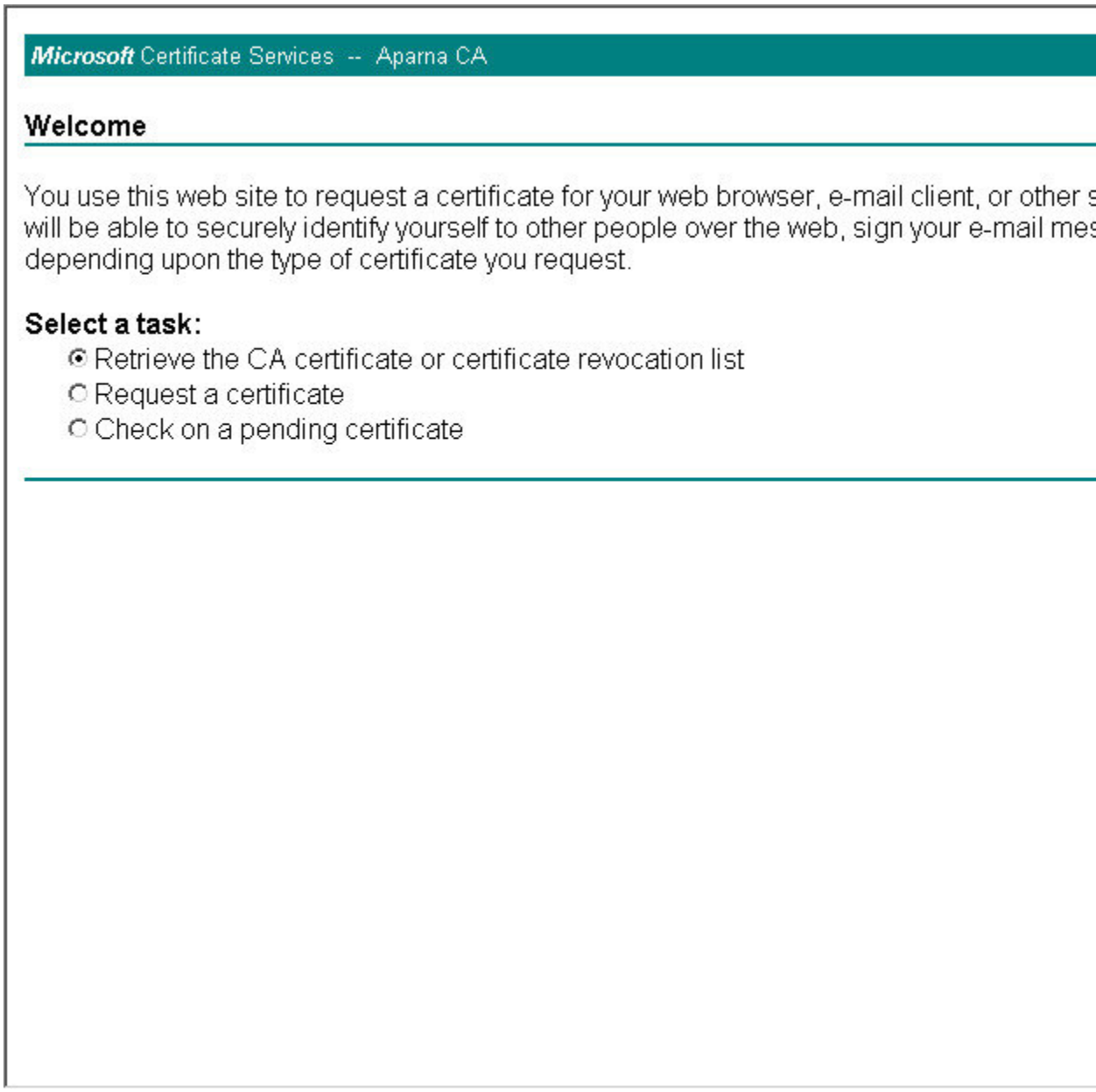
CA 証明書のダウンロード （31 ページ）
アイデンティティ証明書の要求 （37 ページ）

CA 証明書のダウンロード

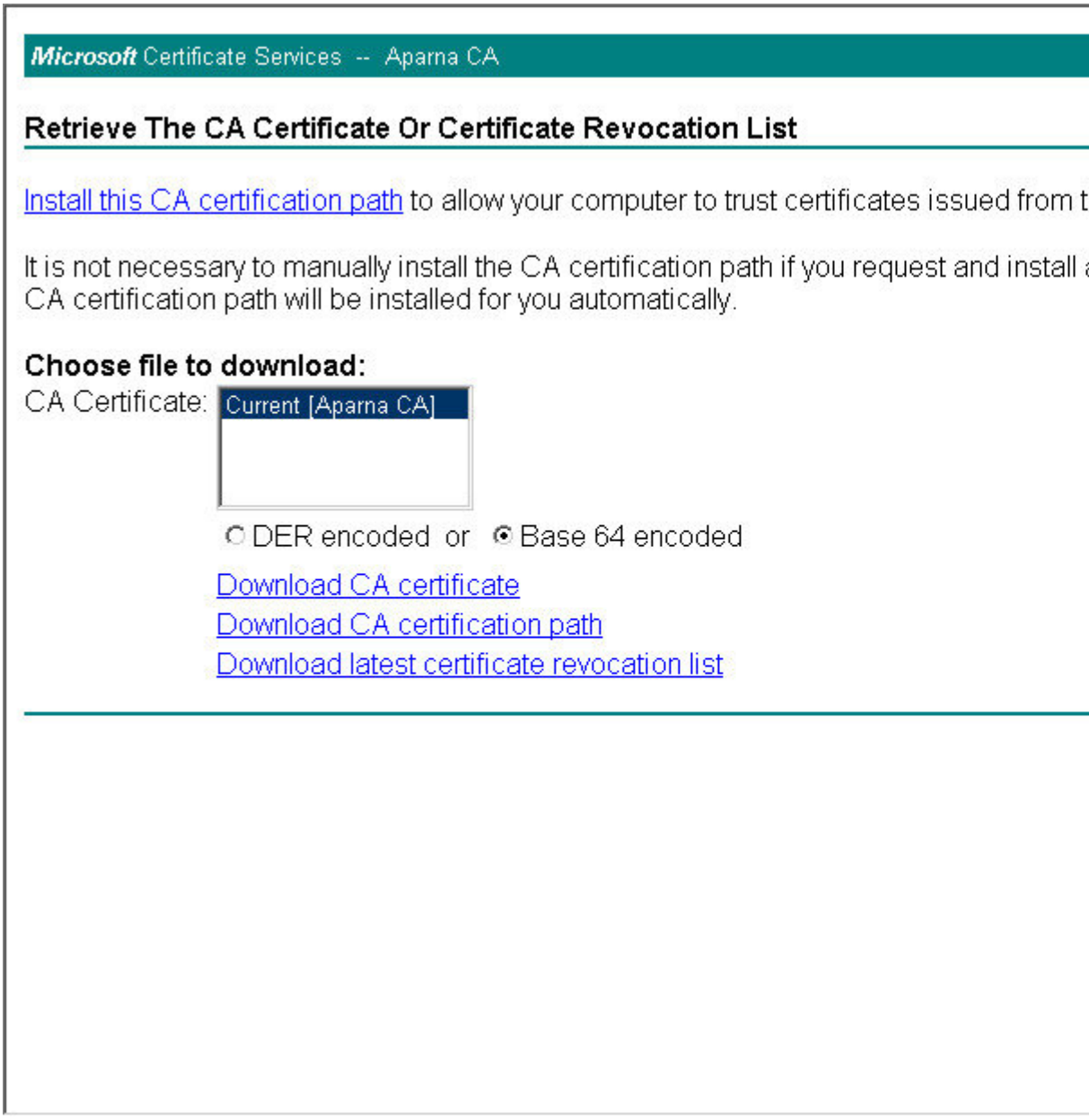
Microsoft Certificate Service の Web インターフェイスから CA 証明書をダウンロードする手順は、次のとおりです。

Procedure

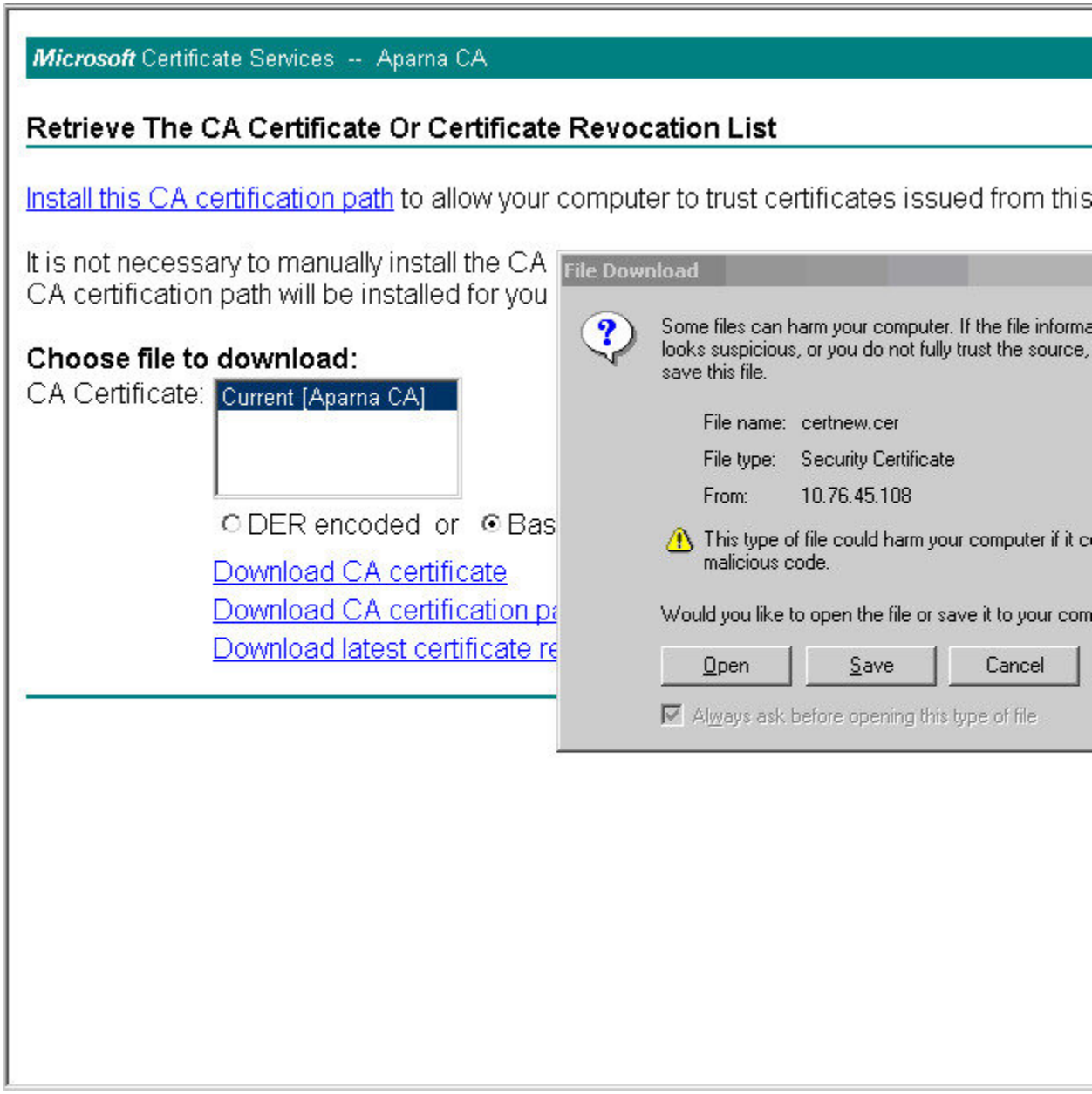
ステップ 1 Microsoft Certificate Services の Web インターフェイスから、[Retrieve the CA certificate or certificate revocation task] をクリックし、[Next] をクリックします。



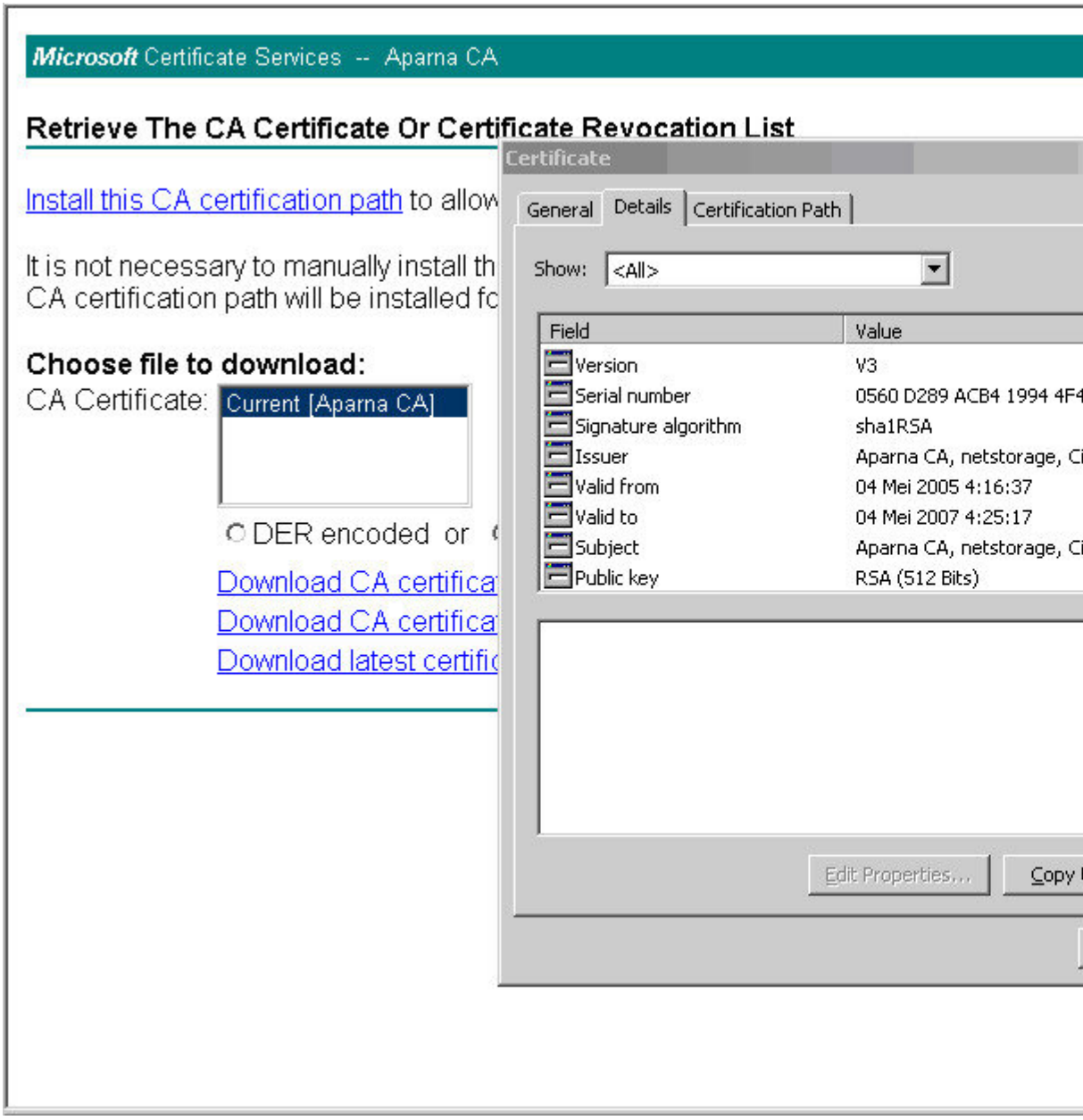
ステップ 2 表示されたリストから、ダウンロードする CA 証明書ファイルを選択します。[Base 64 encoded] をクリックし、[Download CA certificate] をクリックします。



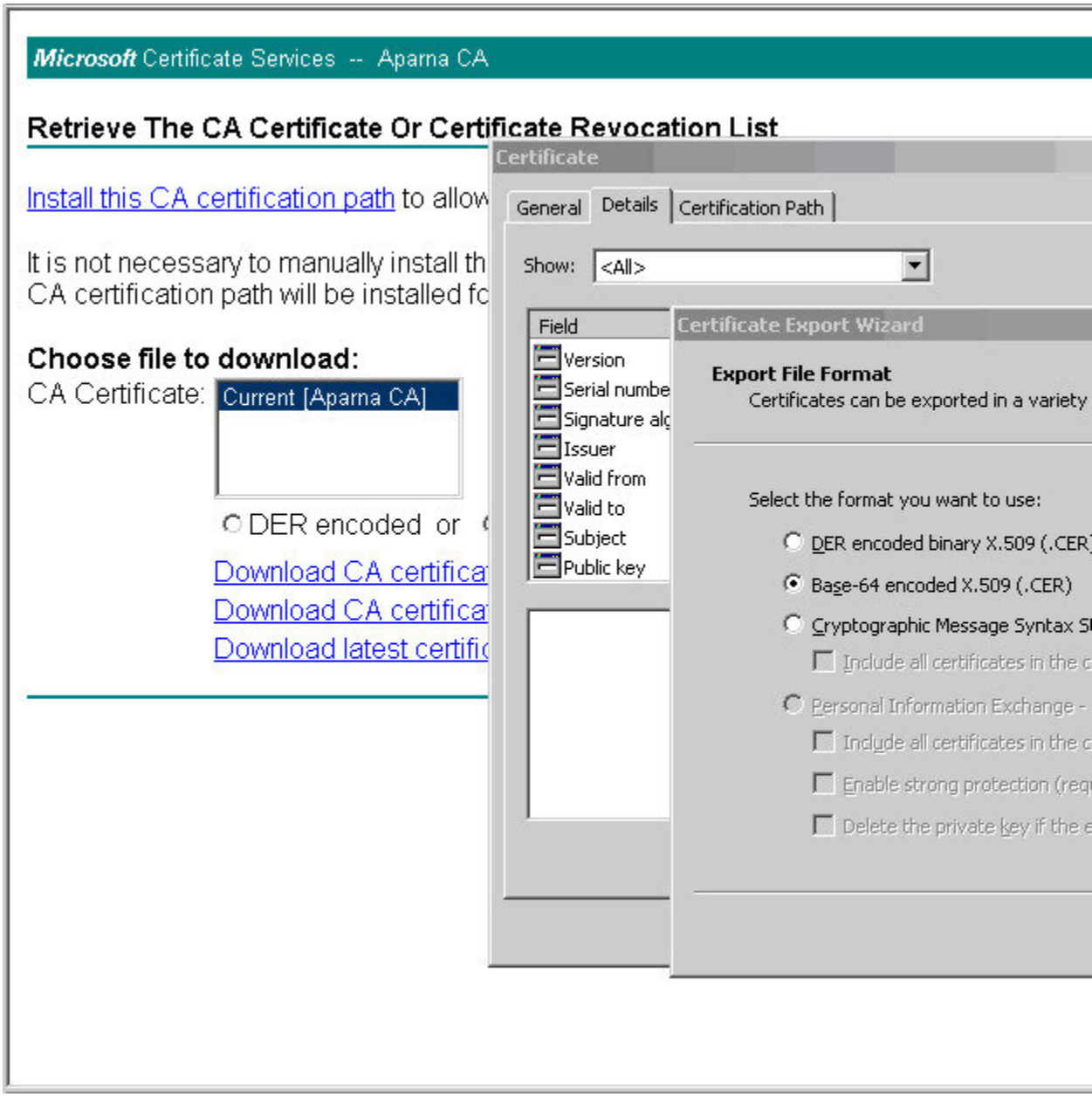
ステップ 3 [File Download] ダイアログボックスにある [Open] をクリックします。



ステップ 4 [Certificate] ダイアログボックスにある [Copy to File] をクリックし、[OK] をクリックします。



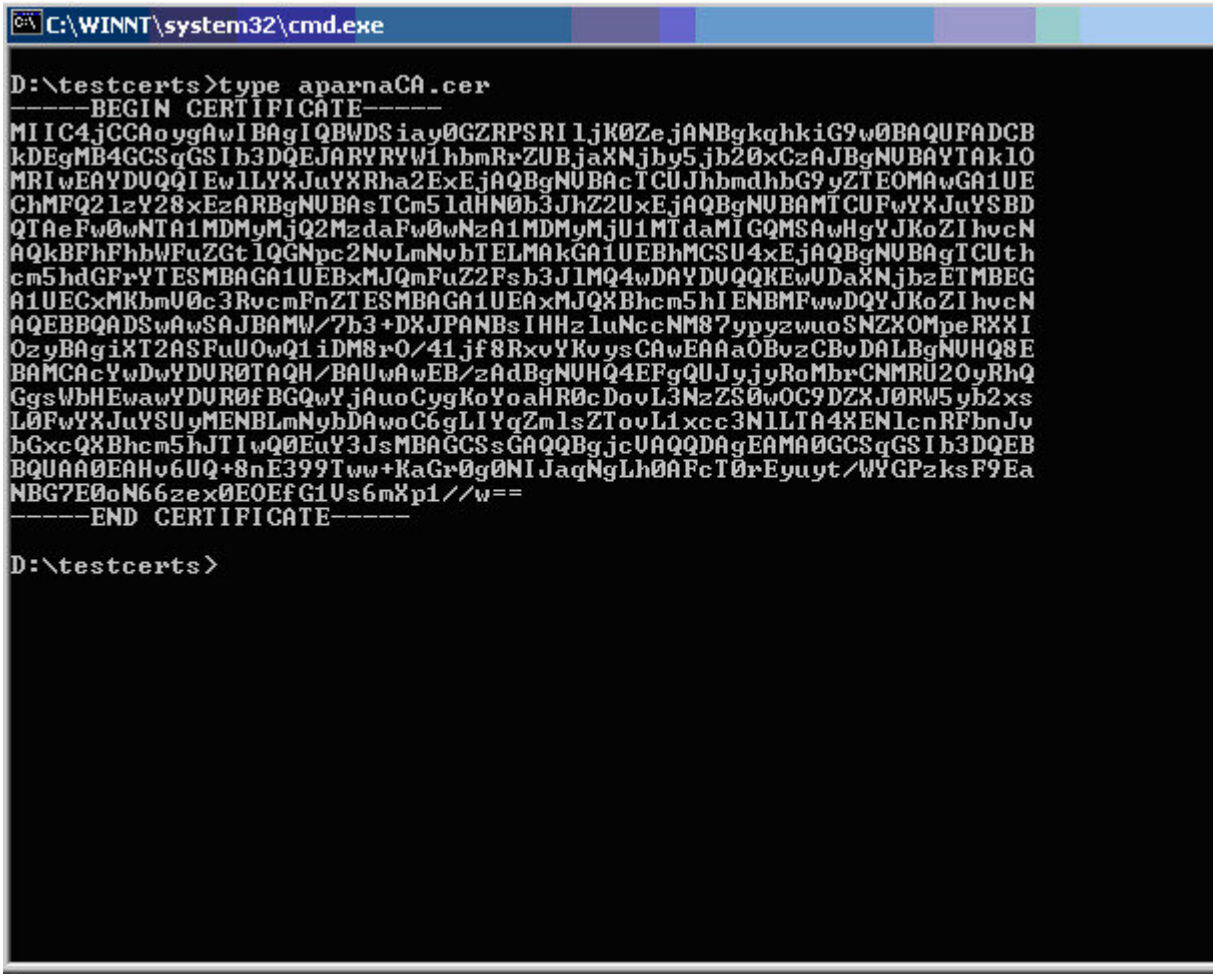
ステップ 5 [Certificate Export Wizard] ダイアログボックスから [Base-64 encoded X.509 (CER)] を選択し、[Next] をクリックします。



ステップ 6 [Certificate Export Wizard] ダイアログボックスにある [File name:] テキスト ボックスに保存するファイル名を入力し、[Next] をクリックします。

ステップ 7 [Certificate Export Wizard] ダイアログボックスで、[Finish] をクリックします。

ステップ 8 Microsoft Windows の type コマンドを入力して、Base-64（PEM）形式で保存されている CA 証明書を表示します。

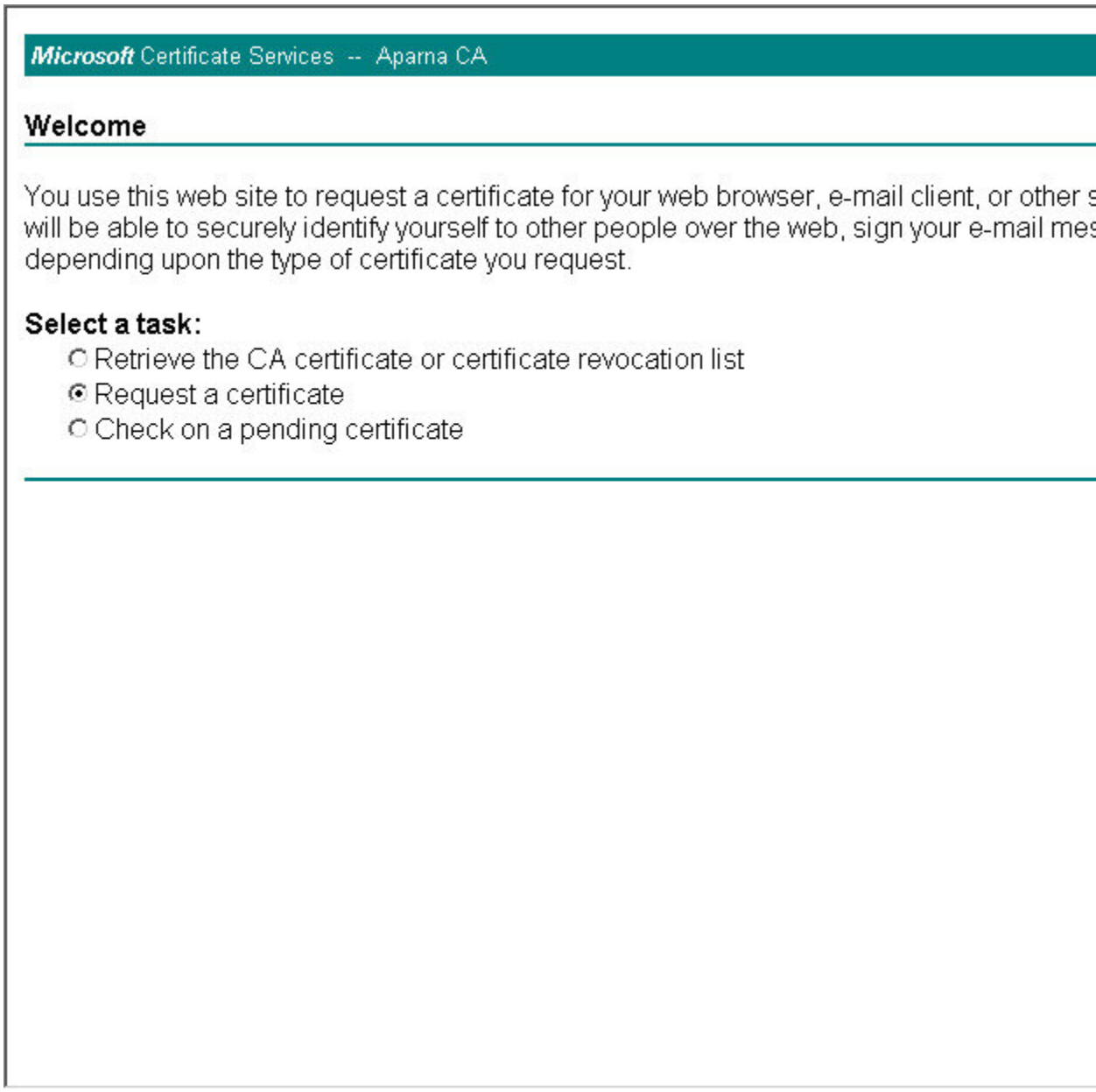


アイデンティティ証明書の要求

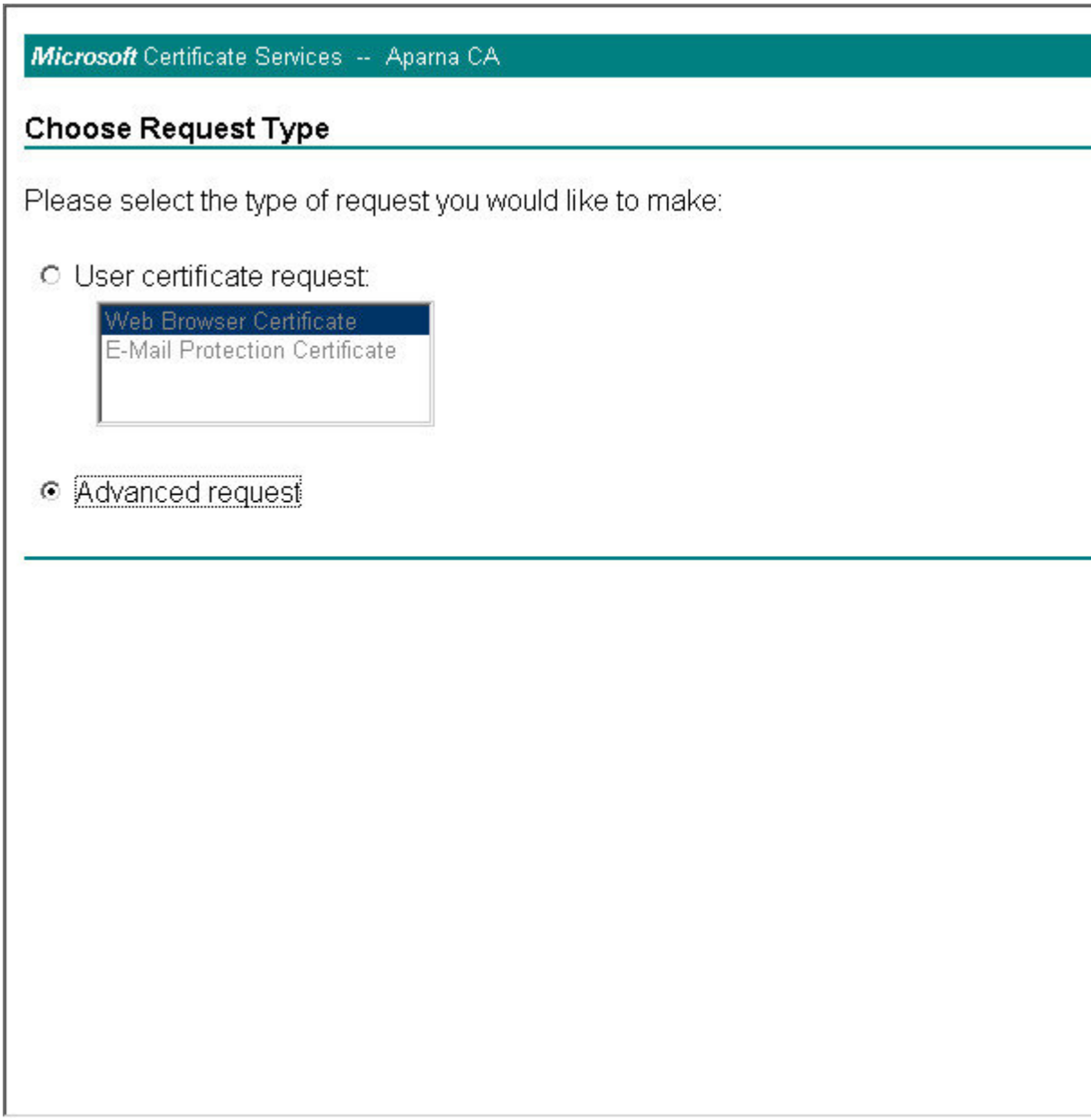
PKCS#12 証明書署名要求（CSR）を使用して Microsoft Certificate サーバにアイデンティティ証明書を要求するには、次の手順に従ってください。

Procedure

ステップ 1 Microsoft Certificate Services の Web インターフェイスから、**[証明書の要求（Request a certificate）]** をクリックし、**[次へ（Next）]** をクリックします。

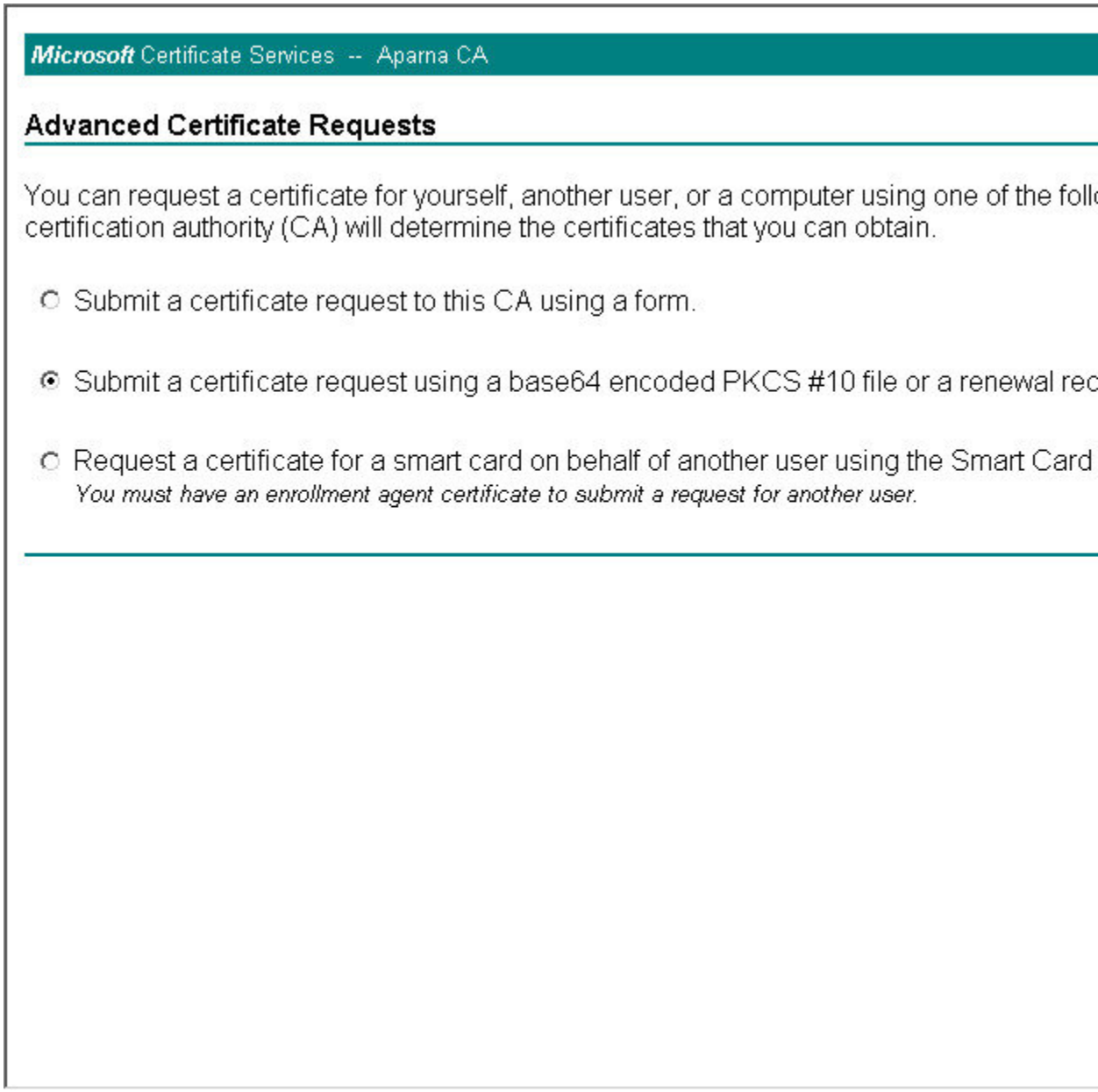


ステップ 2　**[詳細な要求（Advanced request）]** をクリックし、**[次へ（Next）]** をクリックします。



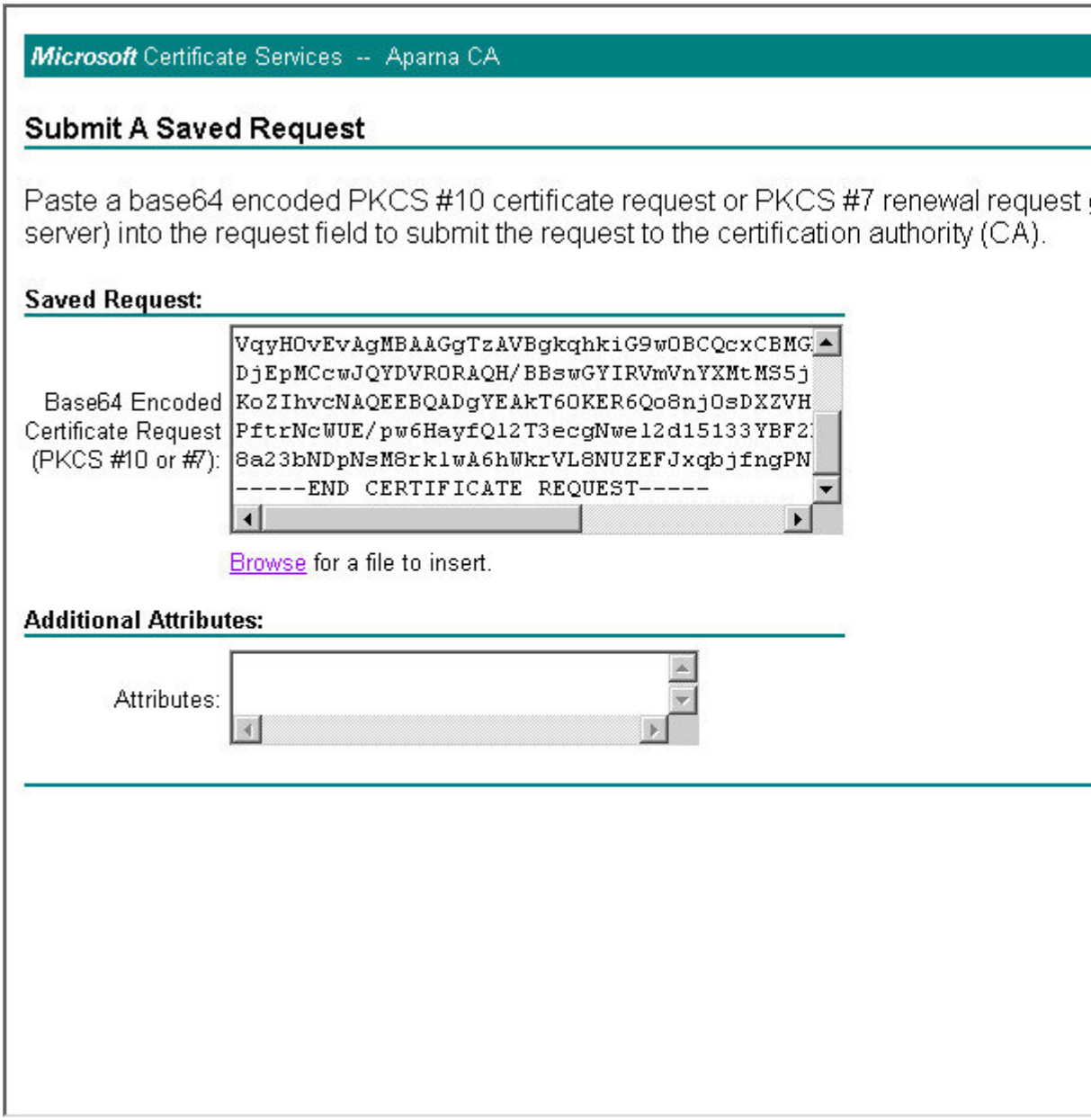
ステップ 3　**[Base64 エンコード済み PKCS#10 を使用する証明書要求または base64 エンコード済み PKCS#7 ファイルを使用する更新要求を送信する（Submit a certificate request using a base64 encoded PKCS#10 file or a renewal request using a base64 encoded PKCS#7 file）]** をクリックし、**[次へ**

（**Next**）] をクリックします。

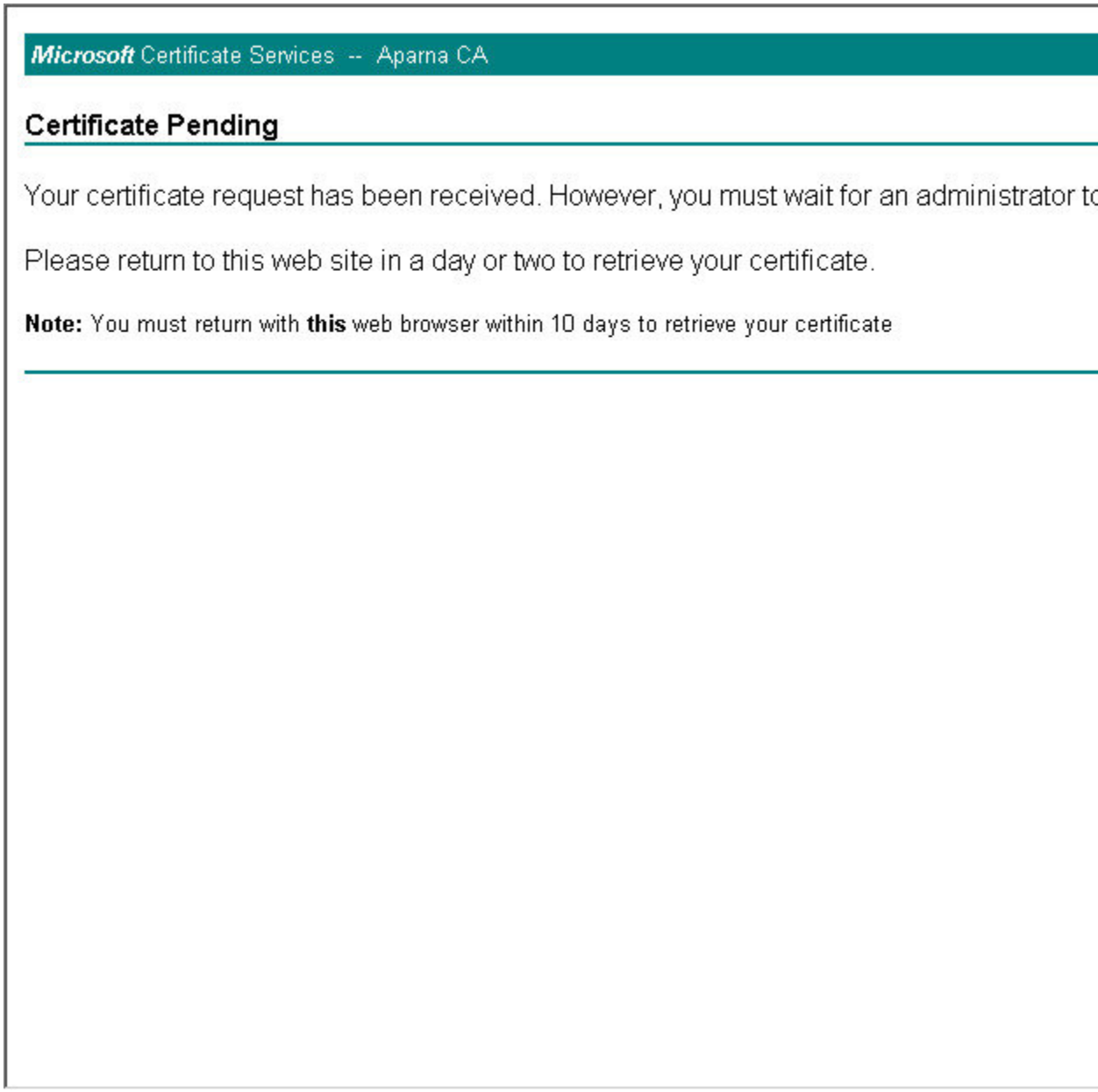


ステップ 4 [保存済みの要求（Saved Request）] テキスト ボックスに、base64 の PKCS#10 証明書要求をペーストし、**[次へ（Next）]** をクリックします。証明書要求が Cisco NX-OS デバイスのコンソール

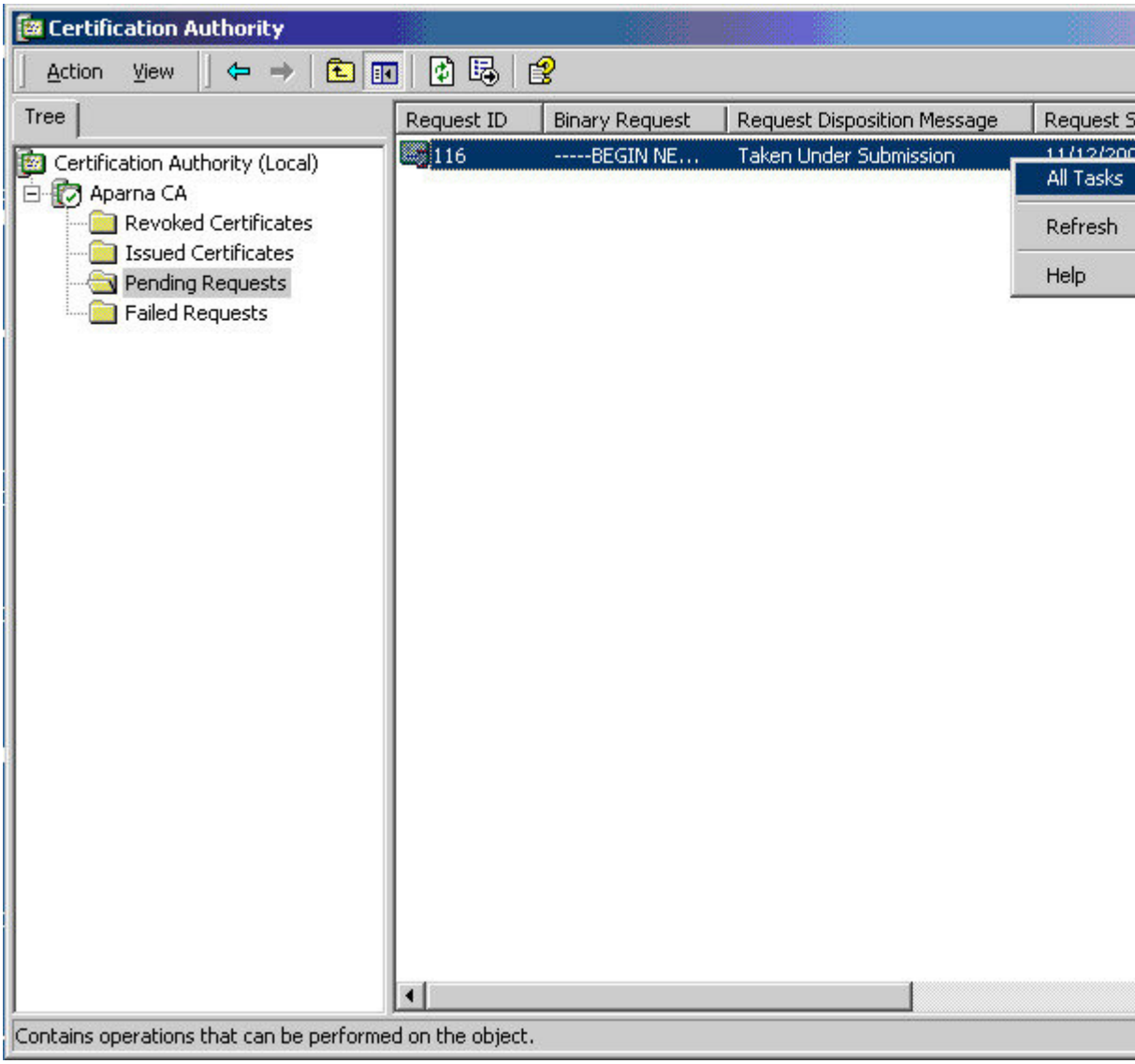
からコピーされます。


Microsoft Certificate Services -- Aparna CA
Submit A Saved Request
Paste a base64 encoded PKCS #10 certificate request or PKCS #7 renewal request
server) into the request field to submit the request to the certification authority (CA).
Saved Request:
Base64 Encoded
Certificate Request
(PKCS #10 or #7):
VqyHOvEvAgMBAAGgTzAVBgkqhkiG9wOBCQcxCBMG
DjEpMCcwJQYDVRORAQH/BBswGYIRVmVnYXMtMS5j
KoZIhvcNAQEEBQADgYEAkT6OKER6Qo8njOsDXZVH
PftrNcWUE/pw6HayfQl2T3ecgNwel2d15133YBF2
8a23bNDpNsM8rklwA6hWkrVL8NUZEFJxqbjfngPN
-----END CERTIFICATE REQUEST-----
Browse for a file to insert.
Additional Attributes:
Attributes:

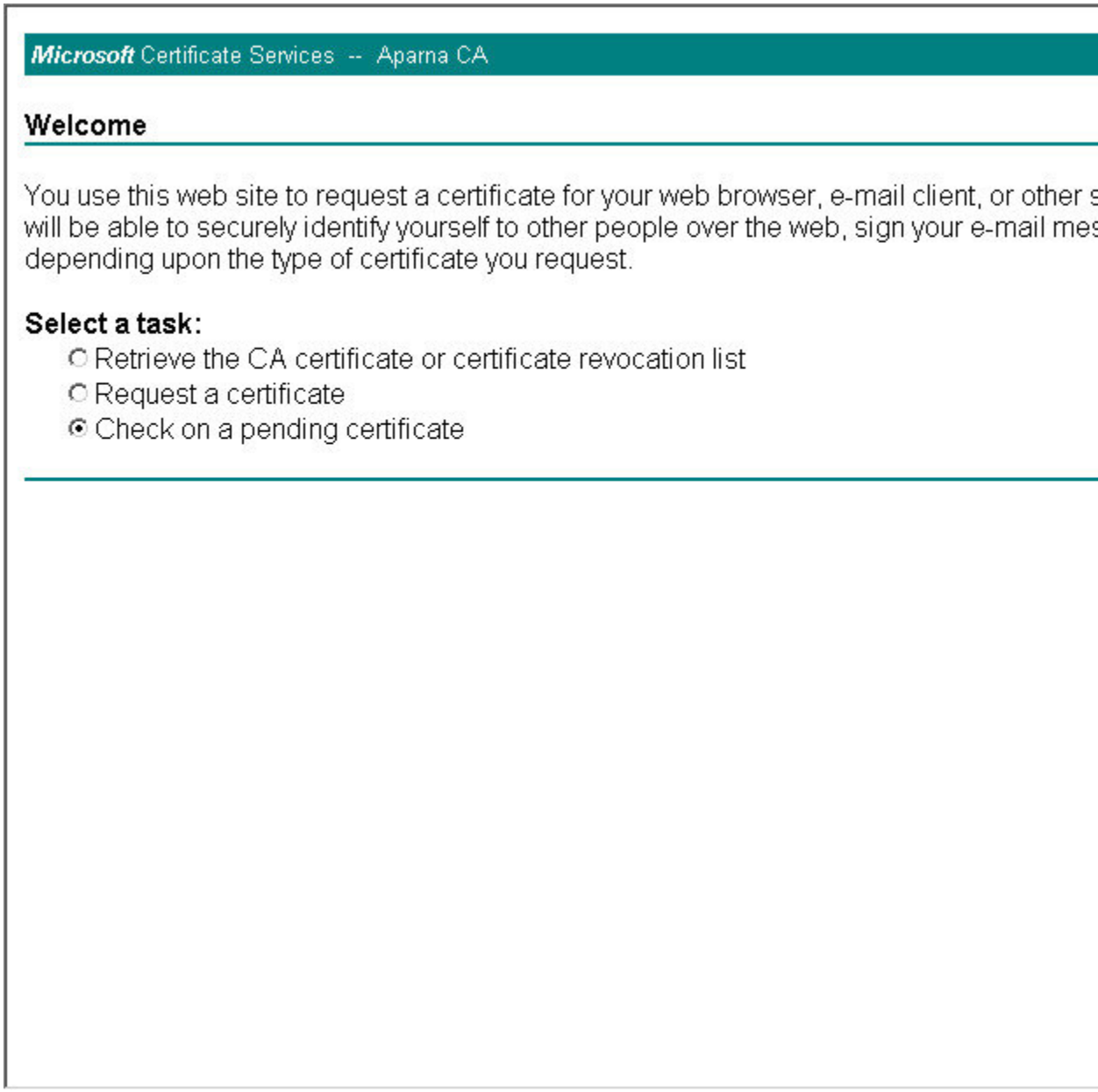
ステップ 5 CA アドミニストレータから証明書が発行されるまで、1 ～ 2 日間待ちます。



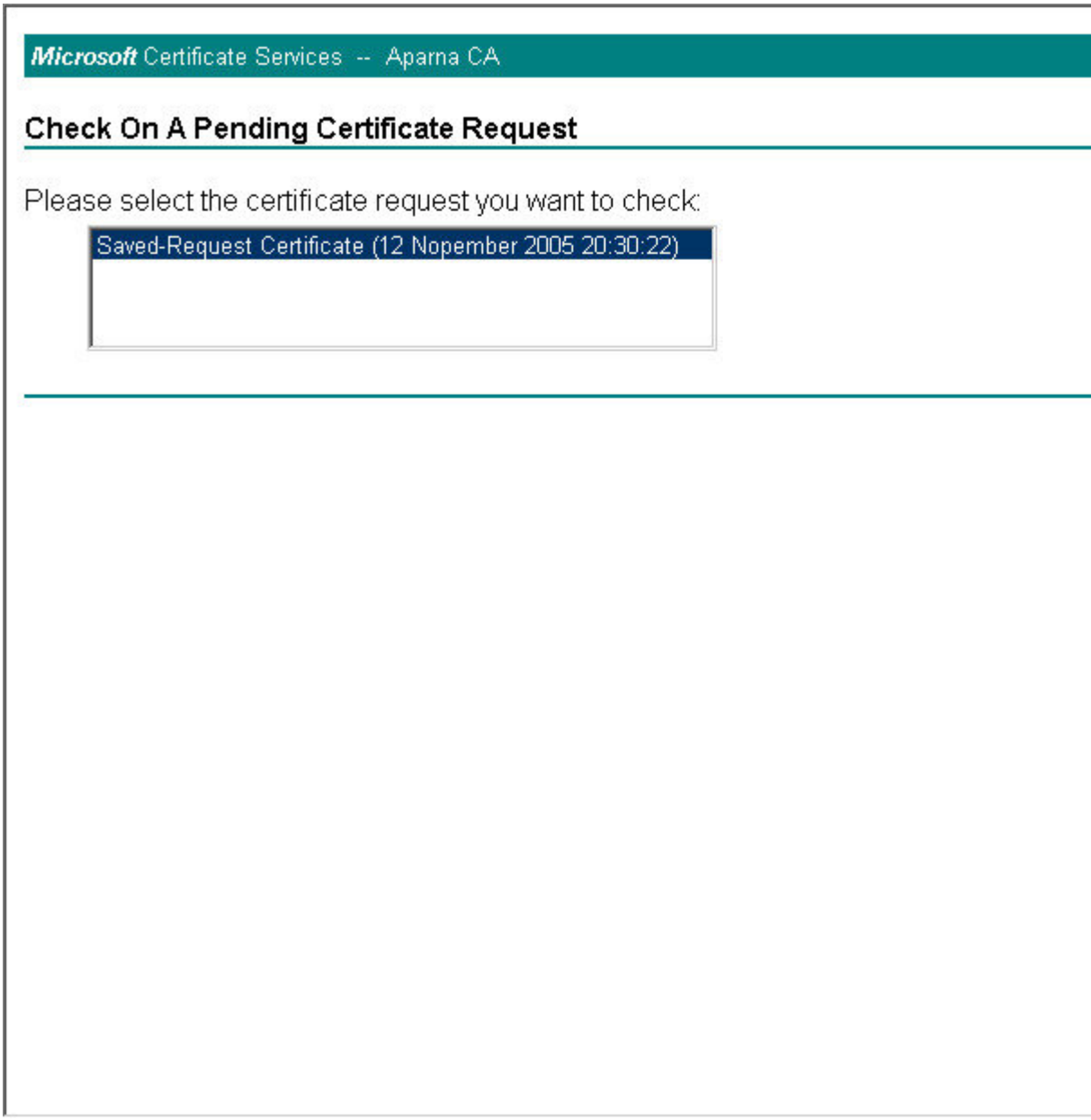
ステップ 6　CA アドミニストレータが証明書要求を承認するのを確認します。



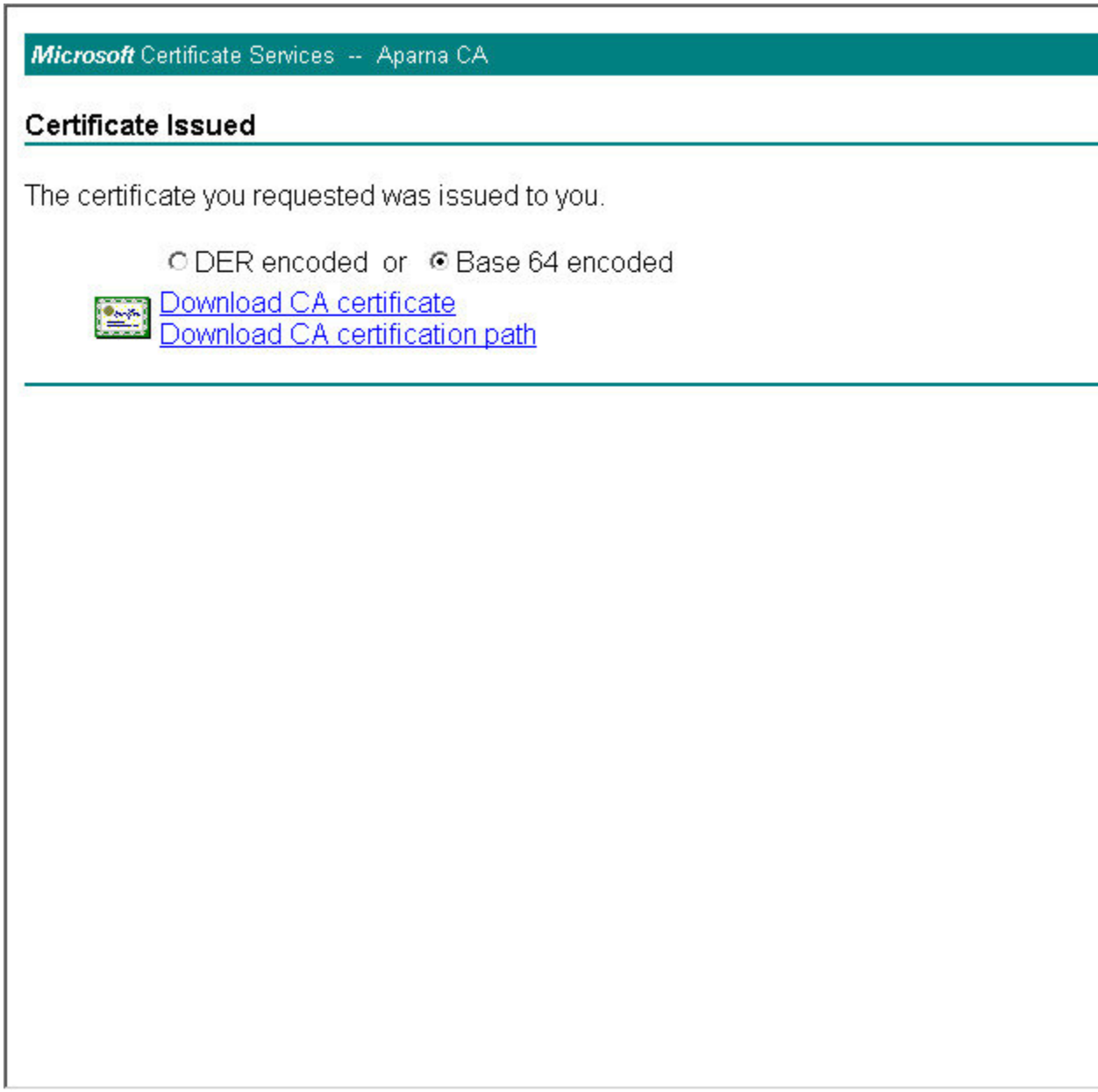
ステップ 7 Microsoft Certificate Services の Web インターフェイスから、**[保留中の証明書をチェックする（Check on a pending certificate）]** をクリックし、**[次へ（Next）]** をクリックします。



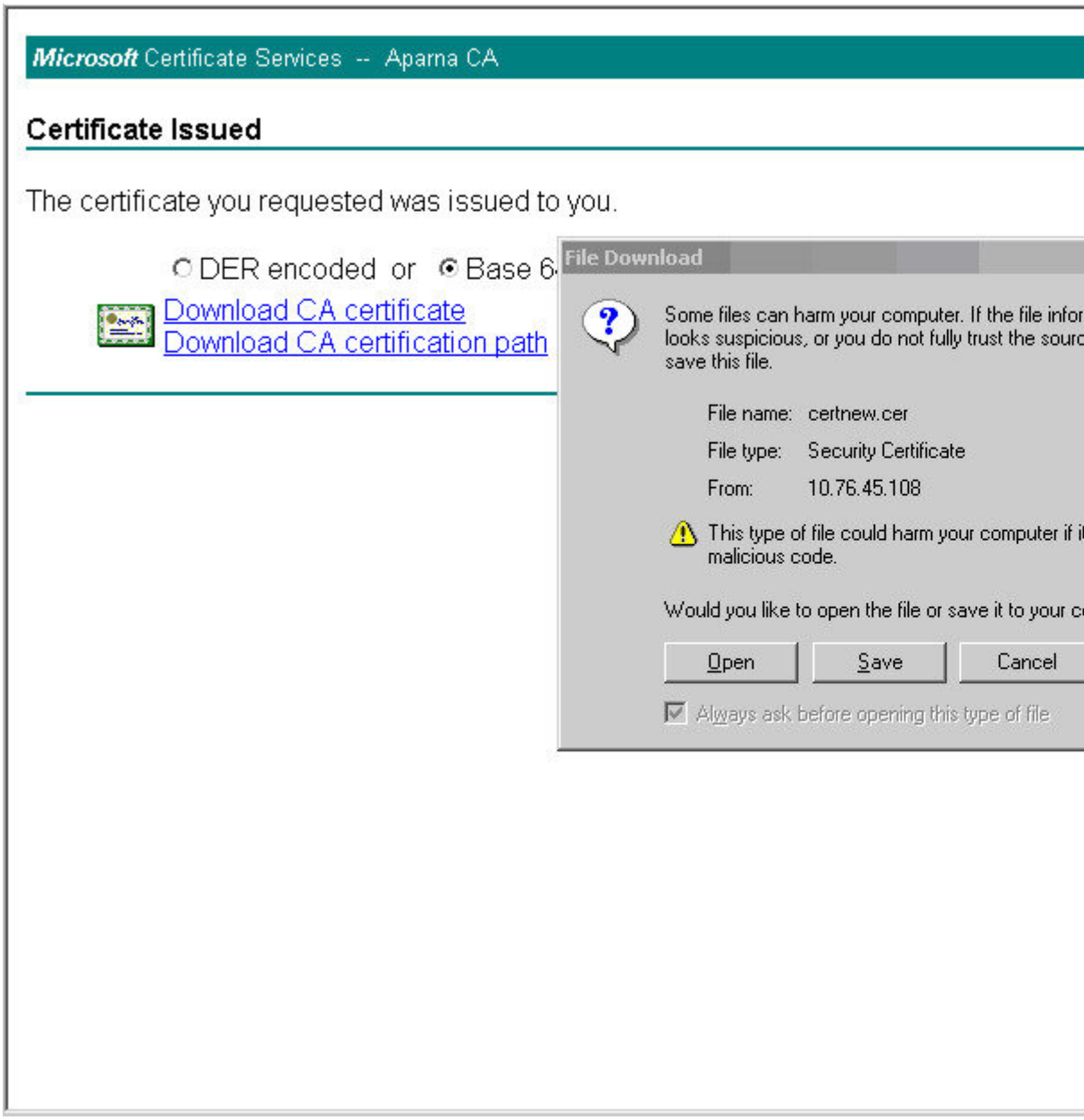
ステップ 8 チェックする証明書要求を選択して、**[次へ（Next）]** をクリックします。



ステップ 9　**[Base 64 エンコード済み（Base 64 encoded）]** をクリックして、**[CA 証明書のダウンロード（Download CA certificate）]** をクリックします。

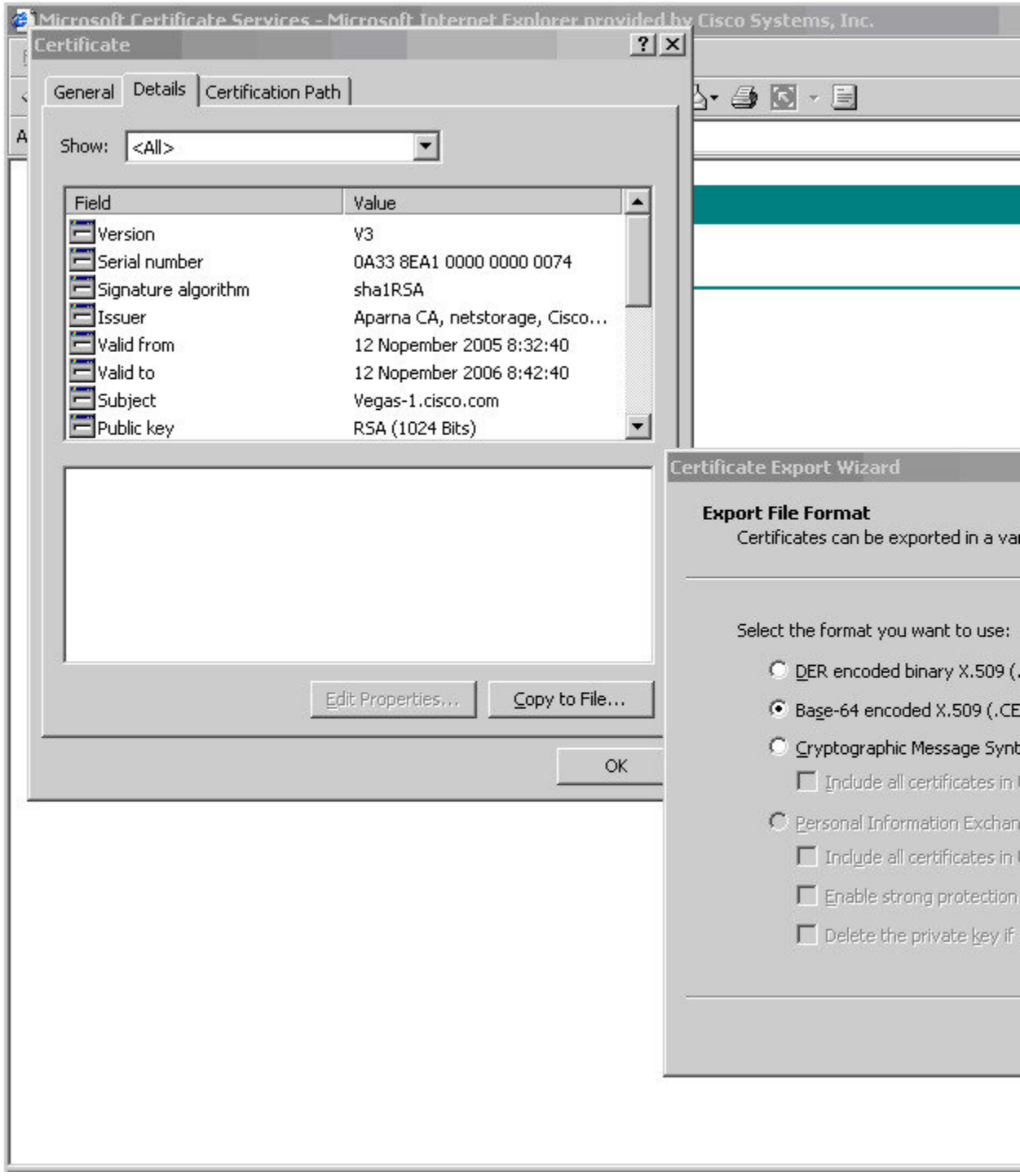


ステップ 10 [ファイルのダウンロード（File Download）] ダイアログ ボックスで、**[開く（Open）]** をクリックします。



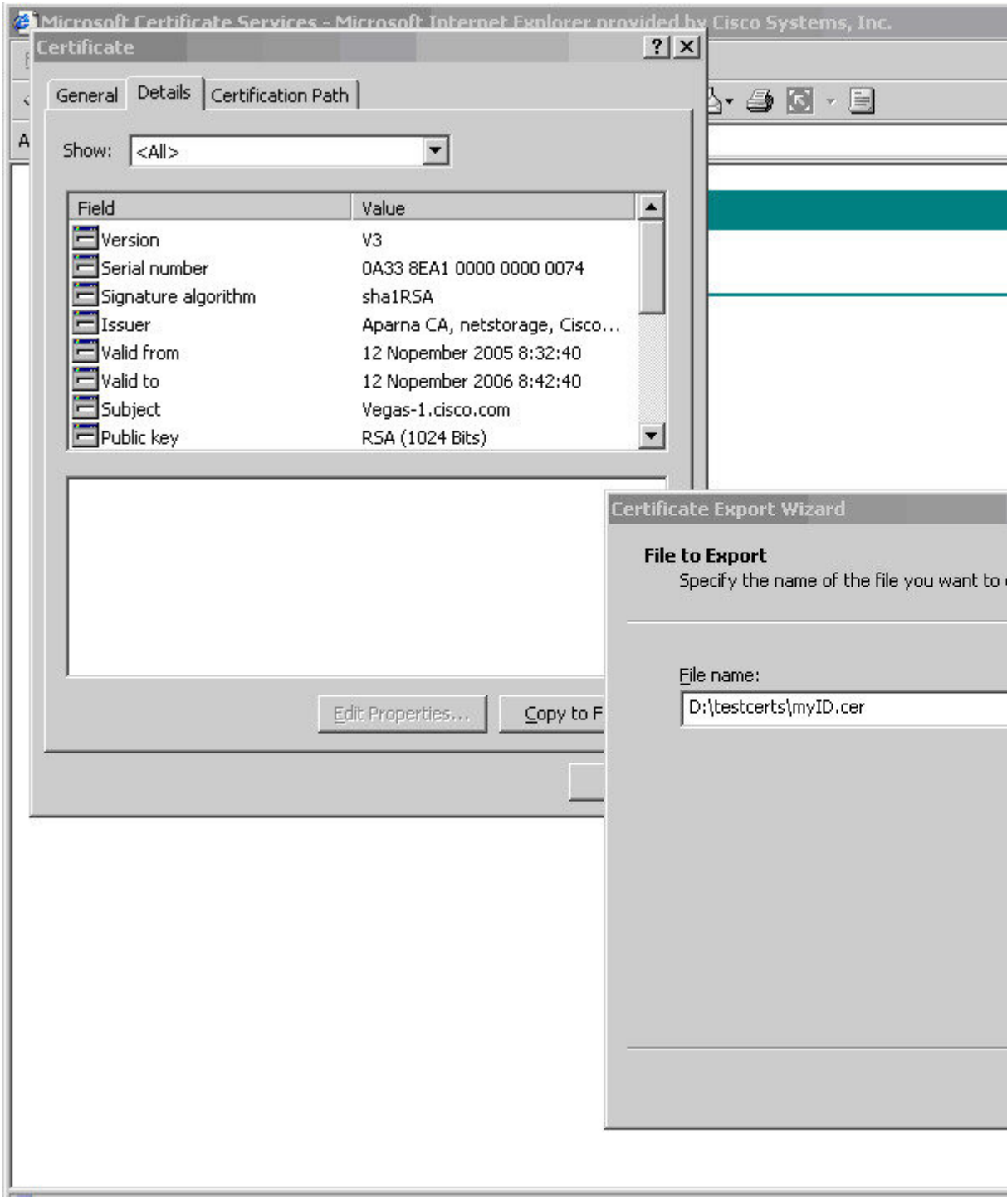
ステップ 11 [Certificate] ボックスで、**[Details]** タブをクリックし、**[Copy to File...]** をクリックします。. [証明書のエクスポート ダイアログ（Certificate Export Dialog）] ボックスで、**[Base-64 エンコード済み X.509 (.CER)（Base-64 encoded X.509 (.CER)）]** をクリックし、**[次へ（Next）]** をクリッ

クします。

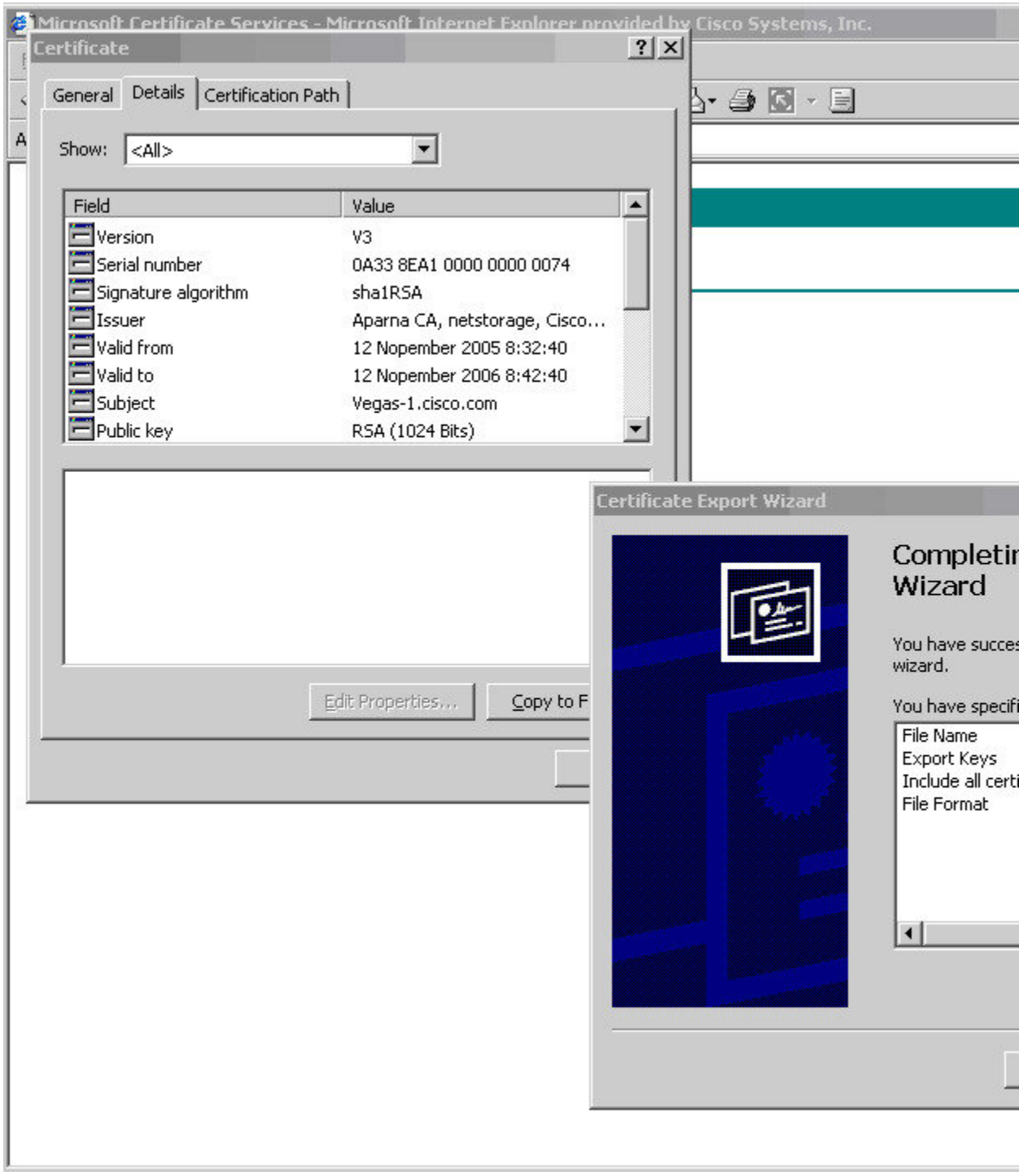


ステップ 12 [証明書エクスポート ウィザード（Certificate Export Wizard）] ダイアログ ボックスにある [ファイル名：（File name:）] テキスト ボックスに保存するファイル名を入力し、**[次へ（Next）]**を

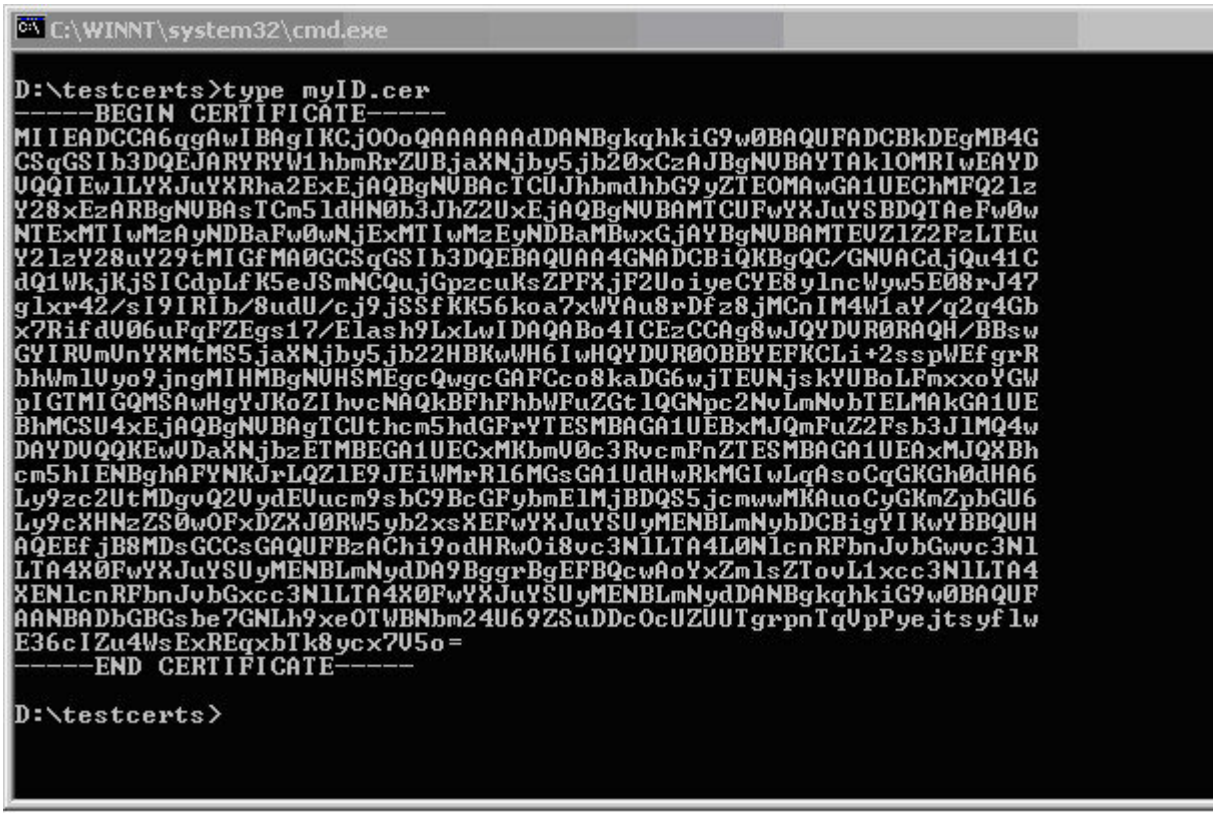
クリックします。


Microsoft Certificate Services - Microsoft Internet Explorer provided by Cisco Systems, Inc.
Certificate
General
Details
Certification Path
Show:
<All>
Field
Value
Version
V3
Serial number
0A33 8EA1 0000 0000 0074
Signature algorithm
sha1RSA
Issuer
Aparna CA, netstorage, Cisco...
Valid from
12 Nopember 2005 8:32:40
Valid to
12 Nopember 2006 8:42:40
Subject
Vegas-1.cisco.com
Public key
RSA (1024 Bits)
Edit Properties...
Copy to F
Certificate Export Wizard
File to Export
Specify the name of the file you want to
File name:
D:\testcerts\myID.cer

ステップ 13 [完了（Finish）] をクリックします。



ステップ 14 Microsoft Windows の **type** コマンドを入力して、アイデンティティ証明書を Base-64 でエンコードされた形式で表示します。



Related Topics

証明書要求の作成 （18 ページ）
Cisco NX-OS デバイスでの証明書の設定 （28 ページ）

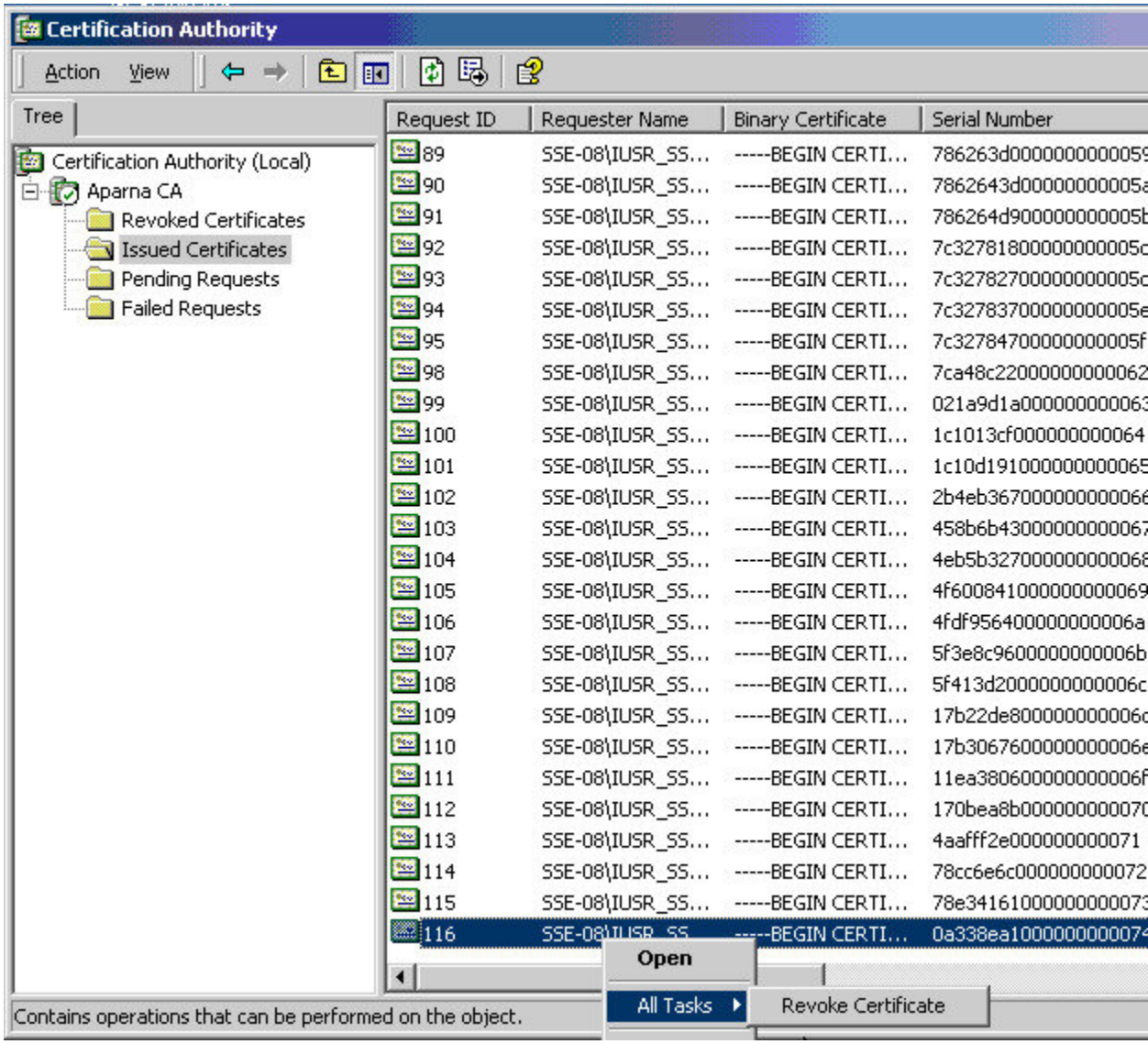
証明書の取り消し

Microsoft CA 管理者プログラムを使用して証明書を取り消す手順は、次のとおりです。

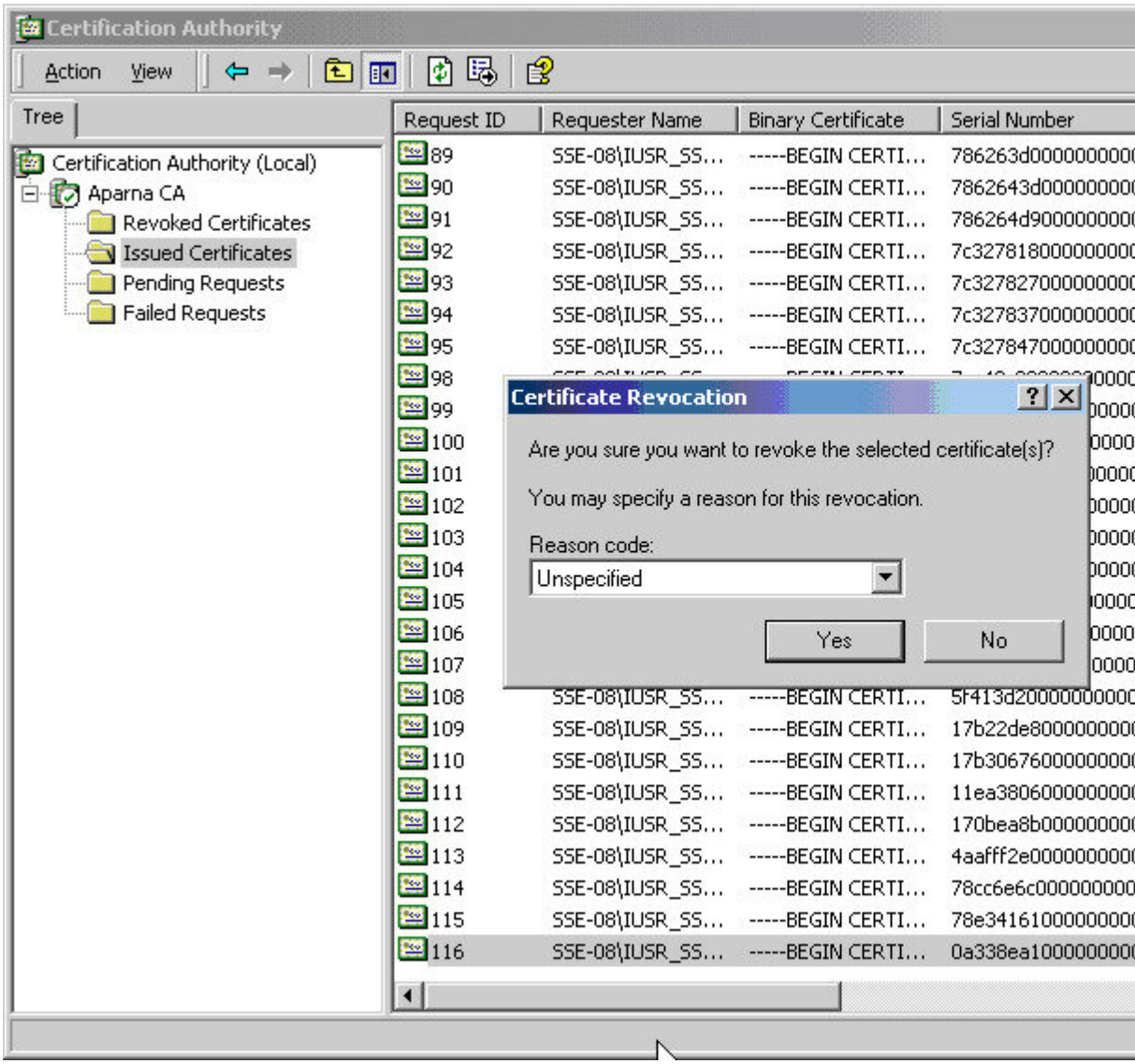
Procedure

ステップ 1 [Certification Authority] ツリーから、[Issued Certificates] フォルダをクリックします。リストから、取り消す証明書を右クリックします。

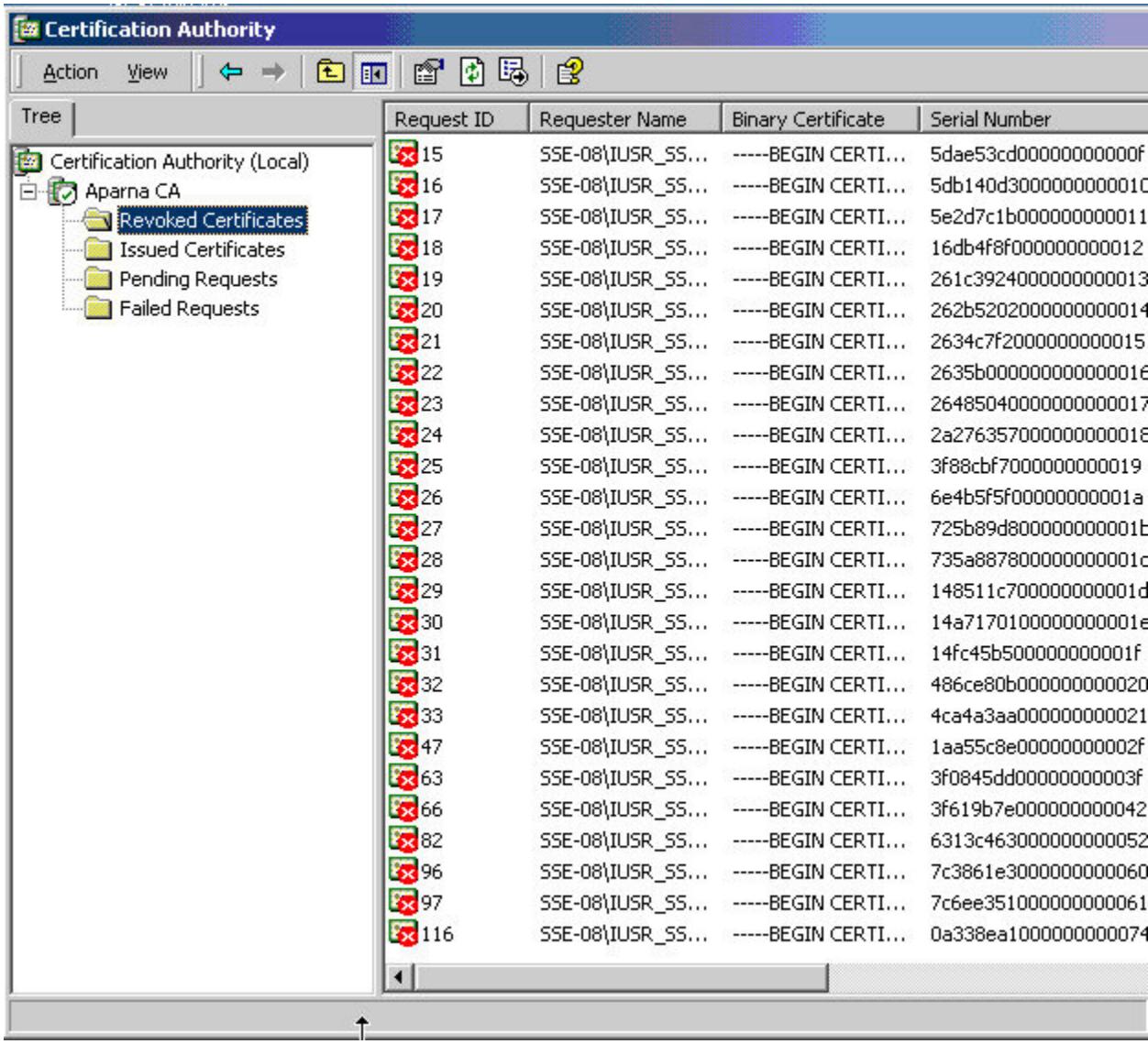
ステップ 2 [All Tasks] > [Revoke Certificate] の順に選択します。



ステップ 3 [Reason code] ドロップダウン リストから取り消しの理由を選択し、[Yes] をクリックします。



ステップ 4 [Revoked Certificates] フォルダをクリックして、証明書の取り消しを表示および確認します。

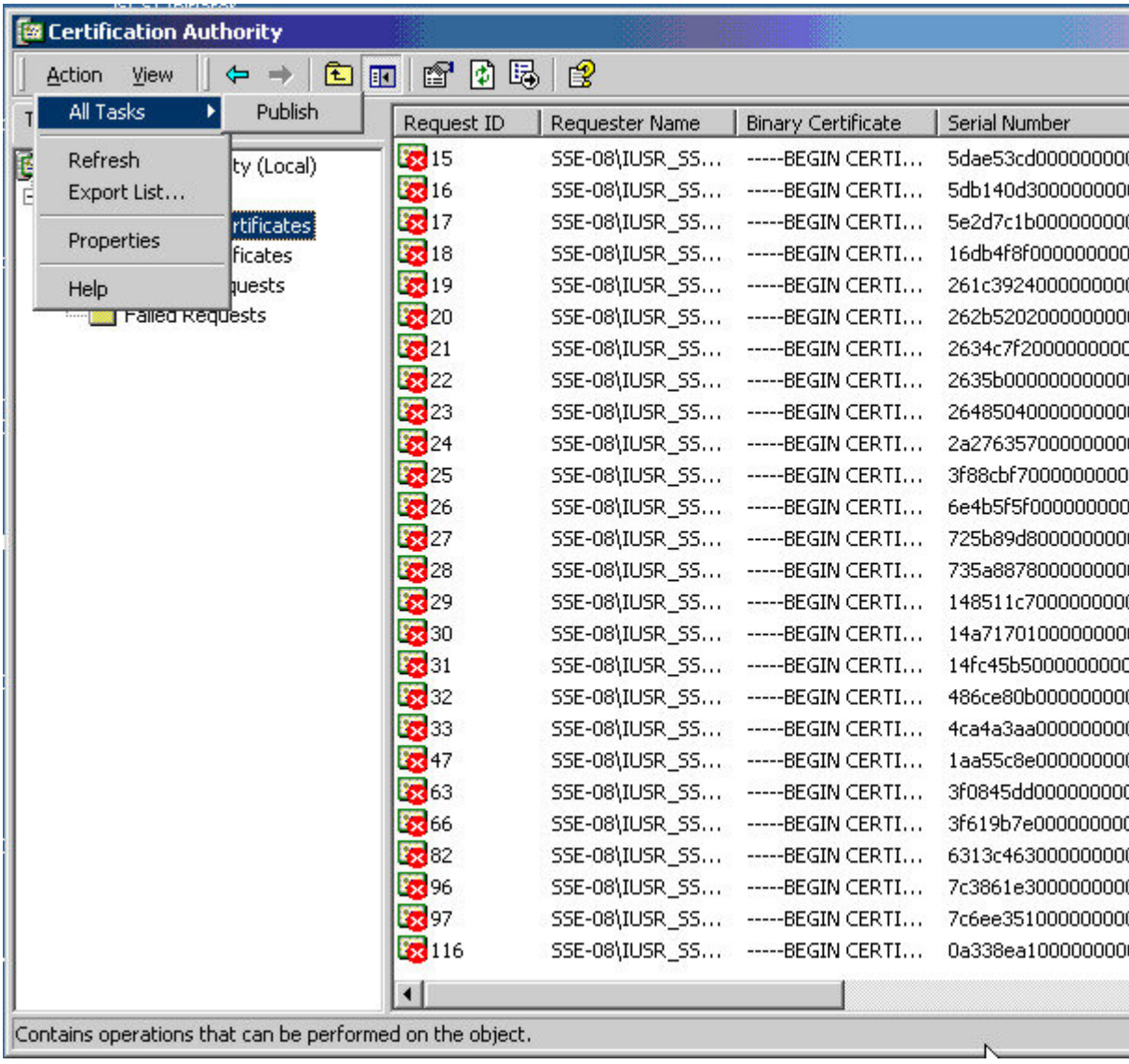


CRL の作成と公開

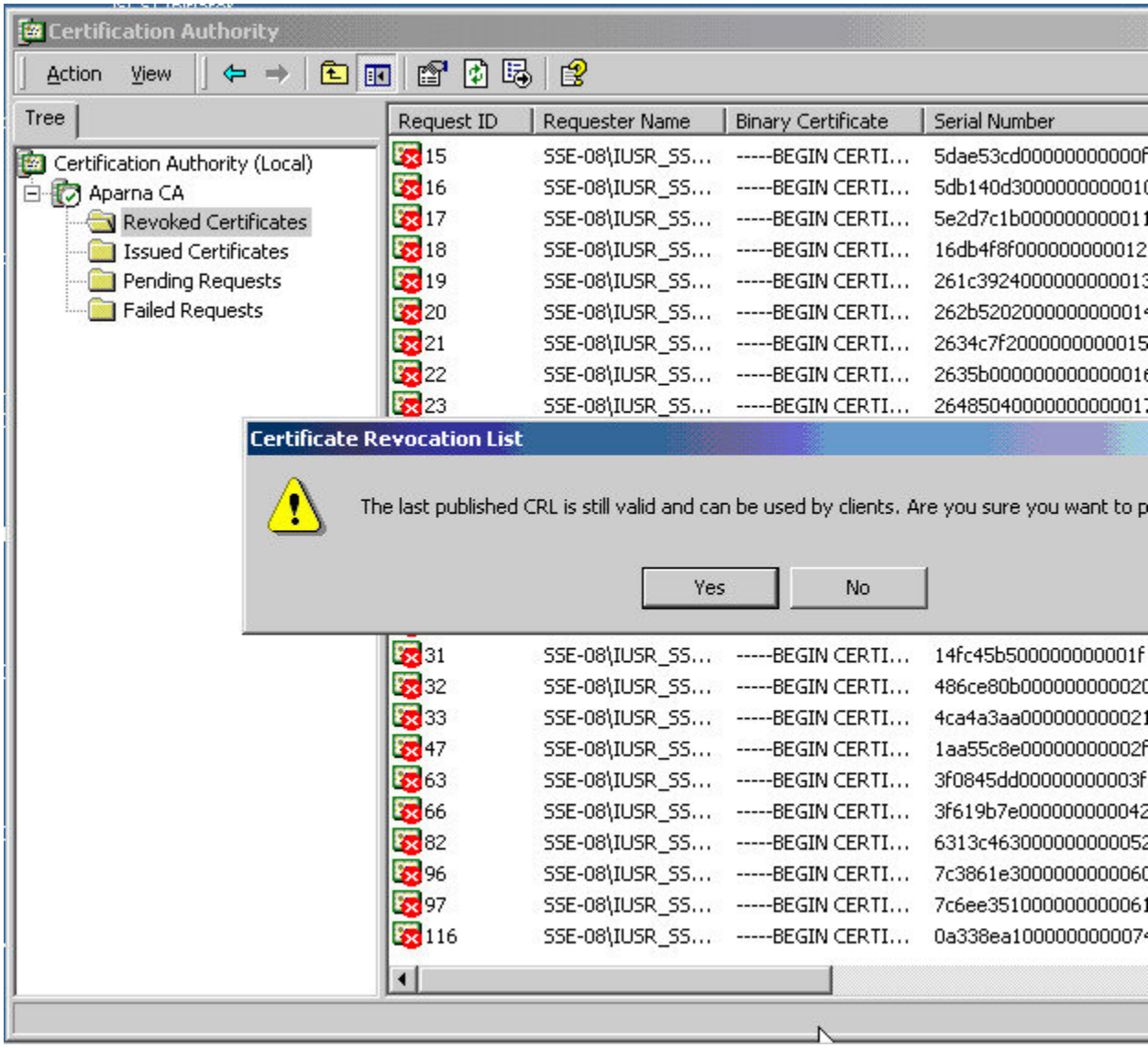
Microsoft CA 管理者プログラムを使用して CRL を作成および公開する手順は、次のとおりです。

Procedure

ステップ 1 [Certification Authority] の画面から、[Action] > [All Tasks] > [Publish] の順に選択します。



ステップ 2 [Certificate Revocation List] ダイアログボックスで、[Yes] をクリックして最新の CRL を公開します。

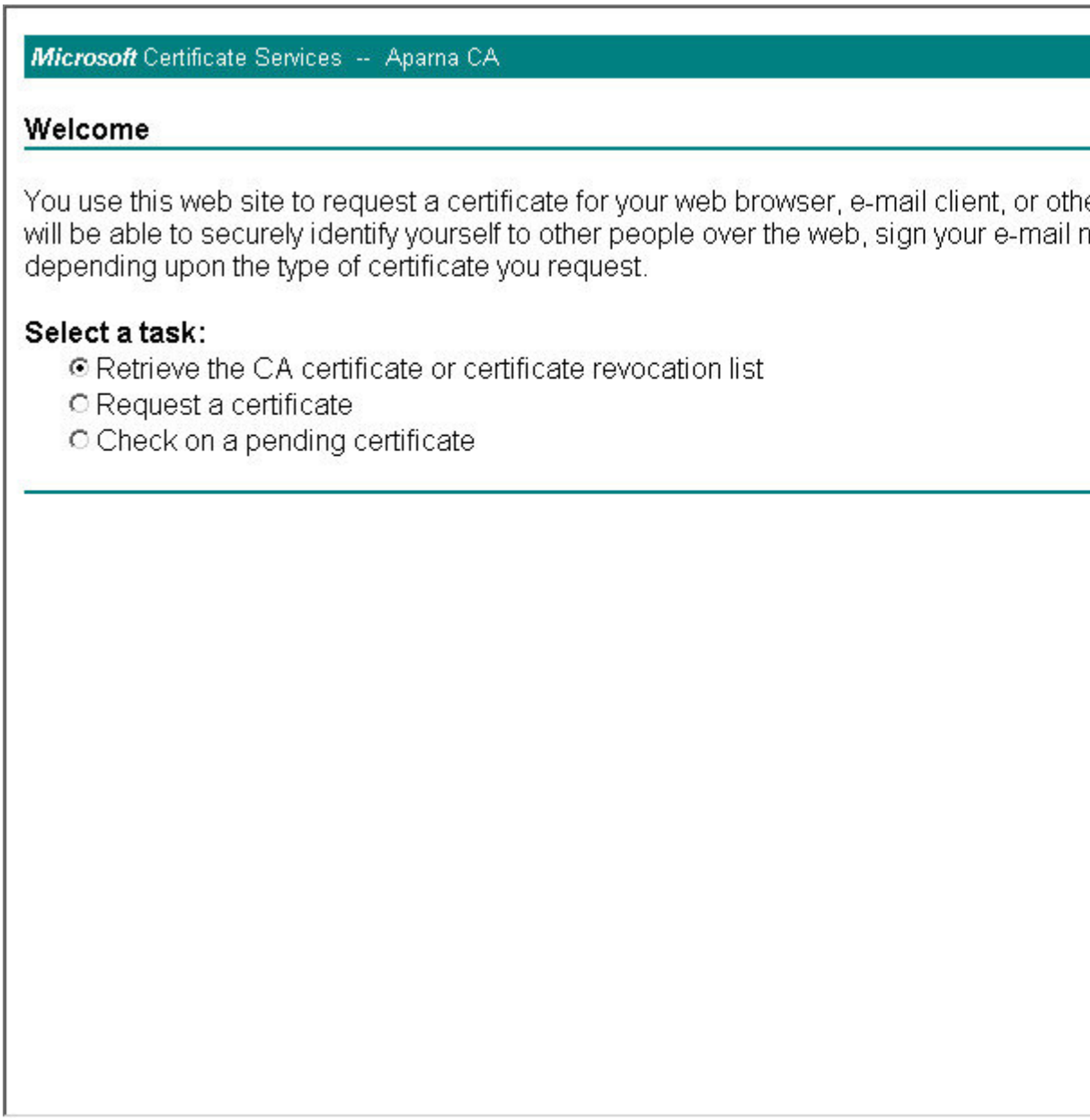


CRL のダウンロード

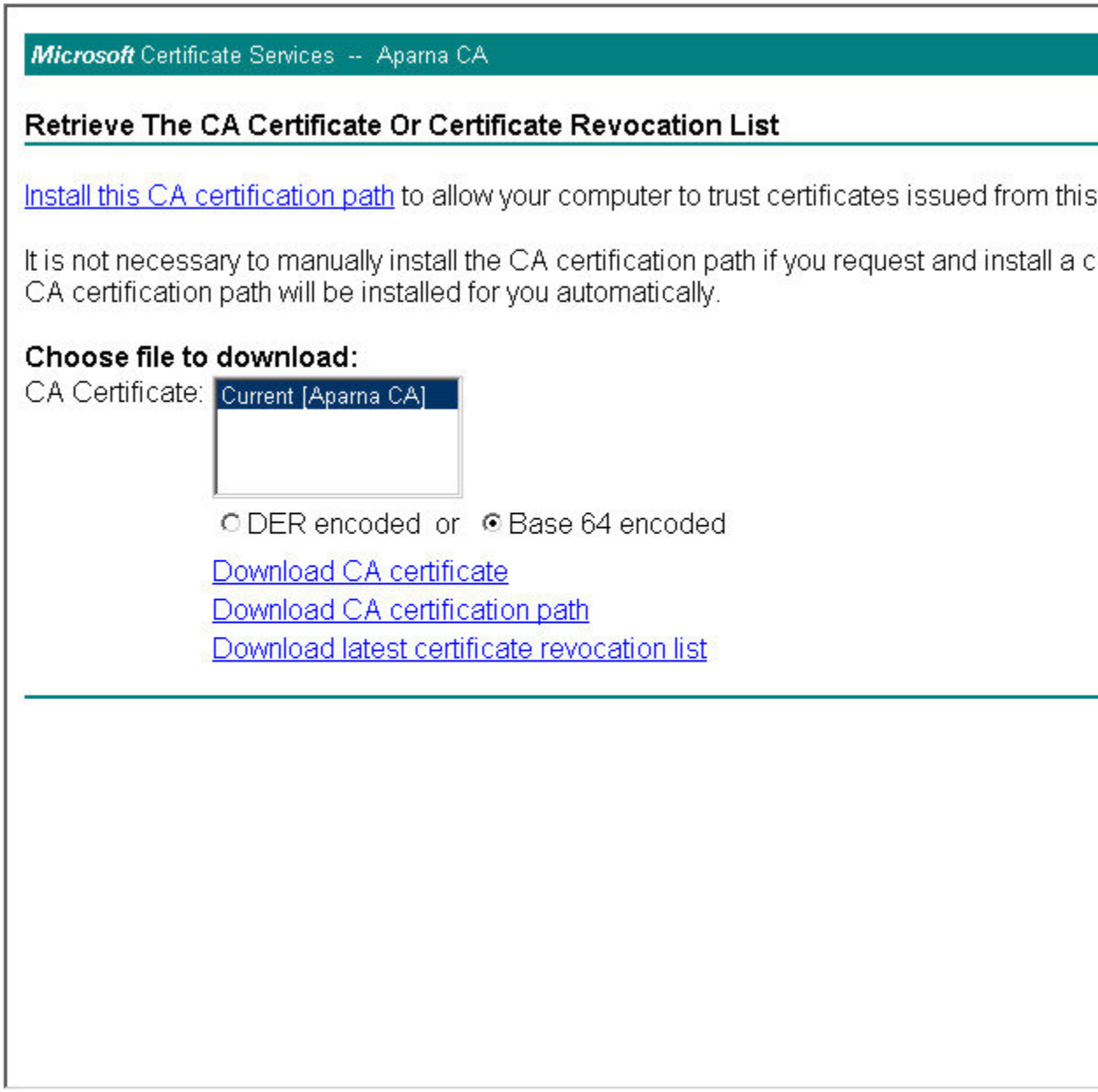
Microsoft 社の CA の Web サイトから CRL をダウンロードする手順は、次のとおりです。

Procedure

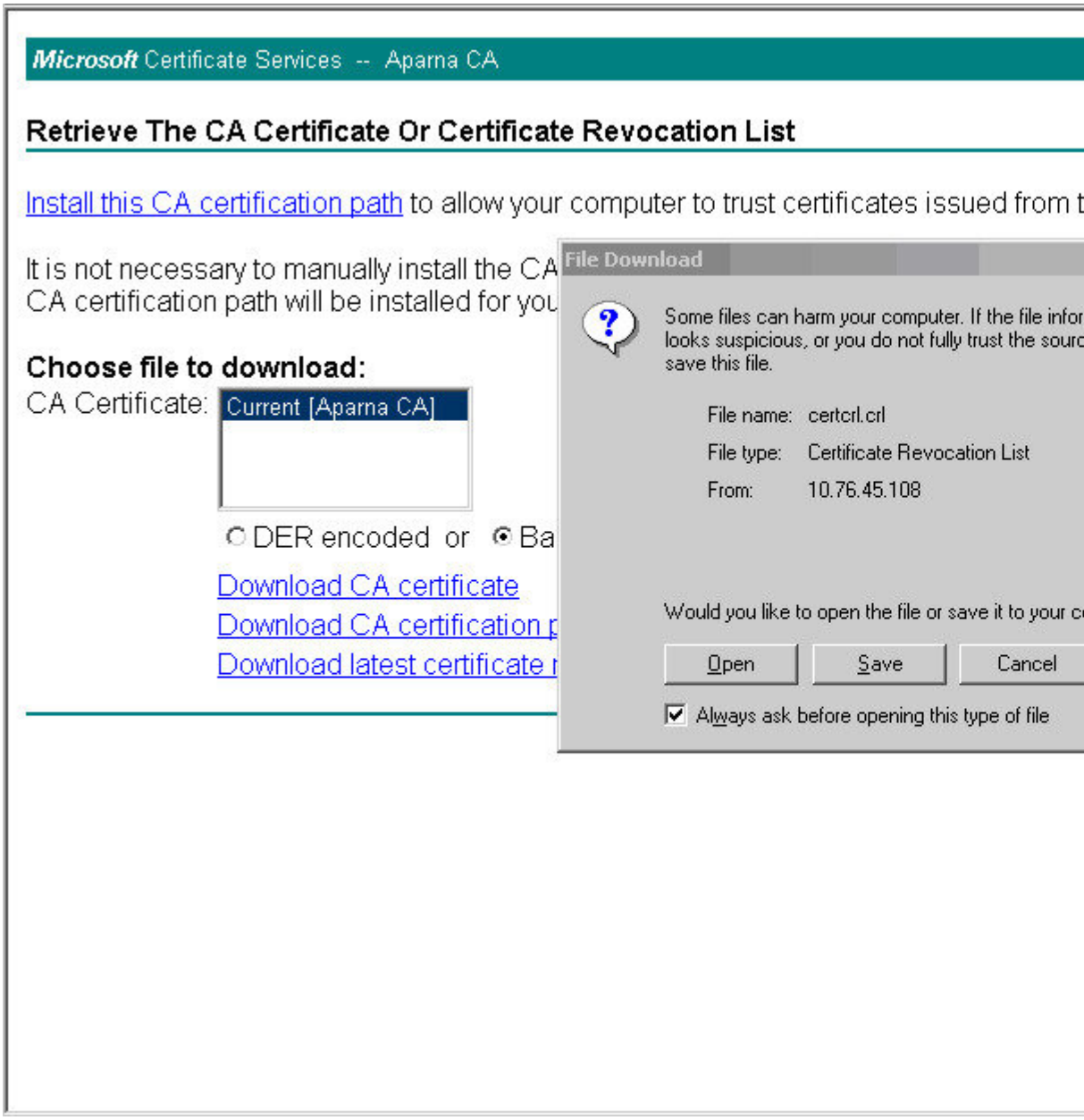
ステップ 1 Microsoft Certificate Services の Web インターフェイスから、[Retrieve the CA certificate or certificate revocation list] をクリックし、[Next] をクリックします。



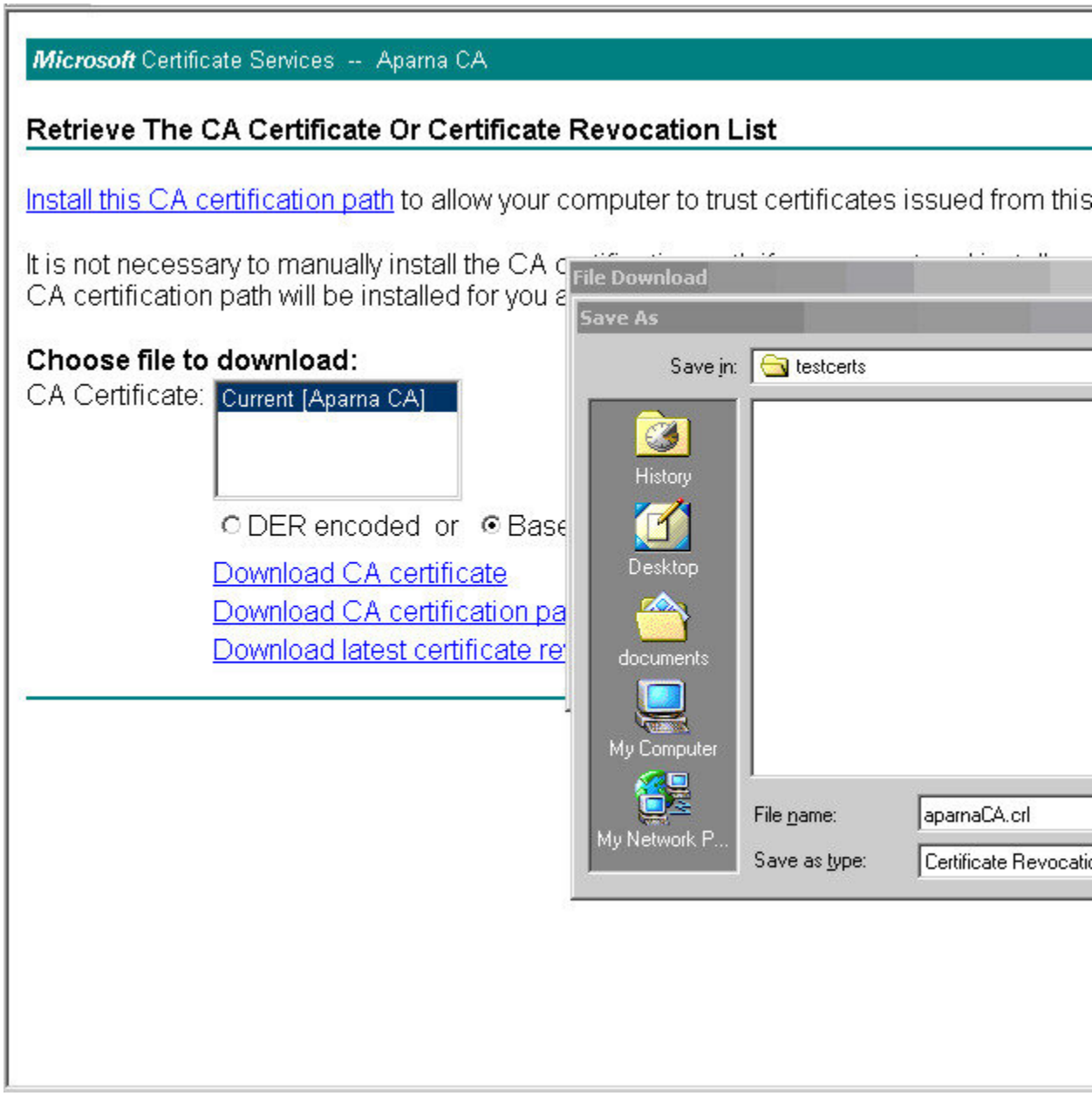
ステップ 2 [Download latest certificate revocation list] をクリックします。



ステップ 3 [File Download] ダイアログボックスで、[Save] をクリックします。



ステップ 4 [Save As] ダイアログボックスで、保存するファイル名を入力して、[Save] をクリックします。



ステップ 5 Microsoft Windows の type コマンドを入力して、CRL を表示します。

Related Topics

証明書取消確認方法の設定 （17 ページ）

CRL のインポート

CRL を CA に対応するトラストポイントにインポートする手順は、次のとおりです。

Procedure

ステップ 1 CRL ファイルを Cisco NX-OS デバイスのブートフラッシュにコピーします。

```
Device-1# copy tftp:apranaCA.crl bootflash:aparnaCA.crl
```

ステップ 2 CRL を設定します。

```
Device-1# configure terminal
Device-1(config)# crypto ca crl request myCA bootflash:aparnaCA.crl
Device-1(config)#
```

ステップ 3 CRL の内容を表示します。

```
Device-1(config)# show crypto ca crl myCA
Trustpoint: myCA
CRL:
Certificate Revocation List (CRL):
        Version 2 (0x1)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: /emailAddress=admin@yourcompany.com/C=IN/ST=Karnatak
Yourcompany/OU=netstorage/CN=Aparna CA
        Last Update: Nov 12 04:36:04 2005 GMT
        Next Update: Nov 19 16:56:04 2005 GMT
        CRL extensions:
            X509v3 Authority Key Identifier:
            keyid:27:28:F2:46:83:1B:AC:23:4C:45:4D:8E:C9:18:50:1
            1.3.6.1.4.1.311.21.1:
                ...
Revoked Certificates:
    Serial Number: 611B09A1000000000002
        Revocation Date: Aug 16 21:52:19 2005 GMT
Serial Number: 4CDE464E000000000003
        Revocation Date: Aug 16 21:52:29 2005 GMT
    Serial Number: 4CFC2B42000000000004
        Revocation Date: Aug 16 21:52:41 2005 GMT
    Serial Number: 6C699EC2000000000005
        Revocation Date: Aug 16 21:52:52 2005 GMT
    Serial Number: 6CCF7DDC000000000006
        Revocation Date: Jun  8 00:12:04 2005 GMT
    Serial Number: 70CC4FFF000000000007
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 4D9B1116000000000008
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 52A802300000000000009
        Revocation Date: Jun 27 23:47:06 2005 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code:
            CA Compromise
Serial Number: 5349AD4600000000000A
        Revocation Date: Jun 27 23:47:22 2005 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code:
            CA Compromise
Serial Number: 53BD173C00000000000B
        Revocation Date: Jul  4 18:04:01 2005 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code:
            Certificate Hold
Serial Number: 591E7ACE00000000000C
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 5D3FD52E00000000000D
        Revocation Date: Jun 29 22:07:25 2005 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code:
            Key Compromise
Serial Number: 5DAB771300000000000E
        Revocation Date: Jul 14 00:33:56 2005 GMT
    Serial Number: 5DAE53CD00000000000F
```

```
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 5DB140D3000000000010
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 5E2D7C1B000000000011
        Revocation Date: Jul  6 21:12:10 2005 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code:
            Cessation Of Operation
Serial Number: 16DB4F8F000000000012
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 261C3924000000000013
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 262B5202000000000014
        Revocation Date: Jul 14 00:33:10 2005 GMT
    Serial Number: 2634C7F2000000000015
        Revocation Date: Jul 14 00:32:45 2005 GMT
    Serial Number: 2635B000000000000016
        Revocation Date: Jul 14 00:31:51 2005 GMT
    Serial Number: 26485040000000000017
        Revocation Date: Jul 14 00:32:25 2005 GMT
    Serial Number: 2A276357000000000018
Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 3F88CBF7000000000019
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 6E4B5F5F00000000001A
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 725B89D800000000001B
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 735A887800000000001C
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 148511C700000000001D
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 14A71701000000000001E
        Revocation Date: Aug 16 21:53:15 2005 GMT
    Serial Number: 14FC45B500000000001F
        Revocation Date: Aug 17 18:30:42 2005 GMT
    Serial Number: 486CE80B000000000020
        Revocation Date: Aug 17 18:30:43 2005 GMT
    Serial Number: 4CA4A3AA000000000021
        Revocation Date: Aug 17 18:30:43 2005 GMT
    Serial Number: 1AA55C8E00000000002F
        Revocation Date: Sep  5 17:07:06 2005 GMT
    Serial Number: 3F0845DD00000000003F
        Revocation Date: Sep  8 20:24:32 2005 GMT
    Serial Number: 3F619B7E000000000042
        Revocation Date: Sep  8 21:40:48 2005 GMT
    Serial Number: 6313C463000000000052
        Revocation Date: Sep 19 17:37:18 2005 GMT
    Serial Number: 7C3861E3000000000060
        Revocation Date: Sep 20 17:52:56 2005 GMT
    Serial Number: 7C6EE351000000000061
        Revocation Date: Sep 20 18:52:30 2005 GMT
    Serial Number: 0A338EA1000000000074   <-- Revoked identity certificate
        Revocation Date: Nov 12 04:34:42 2005 GMT
    Signature Algorithm: sha1WithRSAEncryption
        0b:cb:dd:43:0a:b8:62:1e:80:95:06:6f:4d:ab:0c:d8:8e:32:
        44:8e:a7:94:97:af:02:b9:a6:9c:14:fd:eb:90:cf:18:c9:96:
        29:bb:57:37:d9:1f:d5:bd:4e:9a:4b:18:2b:00:2f:d2:6e:c1:
        1a:9f:1a:49:b7:9c:58:24:d7:72
```

Note 取り消されたデバイスのアイデンティティ証明書（シリアル番号は 0A338EA1000000000074）が最後に表示されています。

PKI に関する追加情報

ここでは、PKI の実装に関する追加情報について説明します。

PKI の関連資料

関連項目	マニュアル タイトル
Cisco NX-OS のライセンス	*Cisco NX-OS* ライセンス ガイド
VRF コンフィギュレーション	『*Cisco Nexus 9000* シリーズ *NX-OS* ユニキャスト ルーティング設定ガイド』

PKI の標準規格

標準	タイトル
この機能でサポートされる新規の標準または変更された標準はありません。また、既存の標準のサポートは変更されていません。	—

Resource Public Key Infrastructure（RPKI）

RPKI は、BGP（インターネット）プレフィックスを認証済みの送信元 AS 番号にマッピングする情報を含む、グローバルに配布されたデータベースです。BGP パスの送信元 AS を検証するために、BGP を実行しているルータは、RPKI に接続できます。

RPKI-Cache-to-Router 接続は多対多にすることができ、1 つの RPKI キャッシュは複数のルーターに origin-AS 検証データを提供でき、1 つのルーターは複数の RPKI キャッシュに接続できます。ルーターは RPKI キャッシュに接続して情報をダウンロードし、BGP がインターネット ルーティング テーブルの発信元 AS 番号を検証するために使用できる特別な RPKI データベースを構築します。

RPKI データベースは、BGP が接続するさまざまな RPKI キャッシュから集約された Route-Origin-Attestation（ROA）オブジェクトのセットです。ROA オブジェクトは、BGP プレフィックス ブロックと、そのブロックの発信を許可された AS 番号との間のマッピングを提供します。

RPKI 構成

RPKI 構成は次のように分類されます。

- RPKI キャッシュに接続するためのコマンド。
- 受信プレフィックスに RPKI 検証状態をマークするためのコマンド。
- BGP ベストパス計算で RPKI 検証状態を使用するためのコマンド。
- route-map を使用して特定の検証状態を持つプレフィックスを削除または操作するためのコマンド。

RPKI キャッシュに接続するためのコマンド

RPKI キャッシュ構成は、router-bgp サブモードの新しい rpki-cache サブモードで行います。これは、デフォルトの VRF での BGP ピアの構成に似ています。サブモードに入るには、「rpki cache <IP address>」コマンドを使用します。サブモードに入ると、RPKI キャッシュのさまざまなパラメータを構成できます。

```
router bgp 100
 rpki cache 147.28.0.11
    description        A description to identify the cache
    shutdown           Shutdown the cache
    transport tcp port Transport port on which cache is listening
    vrf                Vrf in which RPKI cache is reachable
    refresh-interval   Specify periodic wait time between cache poll attempts
    retry-interval     Specify wait time before retrying failed serial or reset query
    expiry-interval    Specify how long to use current data while unable to perform
successful query
```

（注） トランスポート TCP ポートが明示的に構成されていない限り、RPKI-RTR ポート 323 で RPKI キャッシュへの接続を試みる必要があります。

明示的に設定されていない限り、すべての間隔は、データ PDU の末尾の RPKI キャッシュによって提案されたとおりに決定されます。

受信プレフィックスを RPKI 検証状態でマークするためのコマンド

RPKI プレフィックス検証処理の動作を制御するためのノブがあります。これらのノブは、アドレス ファミリ レベルで構成できます。

- **origin-as validate**：アドレス ファミリ レベルで構成すると、ROA データベースに対する eBGP パス検証が有効になります。デフォルトでは無効になっています。

（注） このコマンドは、iBGP パスには関係ありません。iBGP パスは、ROA データベースに対して検証されません。iBGP パスでパス検証状態をマークする唯一の方法は、BGP プレフィックス発信元検証状態拡張コミュニティを受信することであり、コマンドを構成せずにデフォルトで実行されます。

- **origin-as validate signal ibgp**：アドレス ファミリ レベルで構成すると、BGP プレフィックス発信元検証状態拡張コミュニティを介した検証状態の iBGP シグナリングが有効になります。

BGP 最適パス計算で RPKI 検証状態を使用するためのコマンド

RPKI プレフィックス検証処理の動作を制御するためのノブがあります。これらのノブは、アドレス ファミリ レベルで構成できます。

- **bestpath origin-as use-validity**：アドレス ファミリ レベルで構成することで、BGP ベストパス処理でのパスのプリファレンスに影響する BGP パスの有効性状態を有効にします。デフォルトでは無効になっています。
- **bestpath origin-as allow invalid**：アドレス ファミリ レベルで構成することで、すべての「無効な」パスが BGP 最適パス計算のために考慮されるようにします（best-path origin-as 検証が設定されている場合、そのようなパスはどれも最適パス候補ではありません）。デフォルトでは無効になっています。

route-mapを使用して特定の検証状態を持つプレフィックスを削除または操作するためのコマンド

以下は、ルートマップを使用して特定の検証状態を持つプレフィックスを削除または操作するためのコマンドです。

```
route-map sample1 permit 10
  match rpki {not-found | invalid | valid}
```

この match 句は、インバウンド ルートマップにのみ関連します。

iBGP で学習されたパスの場合、更新の入力 BGP プレフィックス発信元検証状態拡張コミュニティが、このルートマップ句と比較されます。

eBGP 学習パスの場合、ROA データベース ルックアップによって取得された検証状態が、このルートマップ句と比較されます。

検証状態が無効であるとマークされたプレフィックスは、BGP での最適パスの計算に考慮されないため、無効になりますが、管理者は、システム メモリを節約するために、そのようなプレフィックスを完全に削除するように決定する場合があります。この目的には、次のインバウンド ルート マップが推奨されます。

```
route-map sample deny 10
match rpki invalid
route-map sample permit 20
```

RPKI Show コマンド

次に、この機能のために追加された 4 つの主要な show コマンドを示します。

- **show bgp rpki summary** ：構成されている RPKI キャッシュの数、構成されているグローバル ノブ、および RPKI データベース サイズを含む RPKI 統計の概要を表示します。

```
jsonrpc 2.0
result
body
TABLE_RPKISUM
ROW_RPKISUM
address 11.0.0.1
trans TCP
port 3323
state NONE
time 10:04:05.590102
ipv4roa 0
ipv6roa 0
address 2.2.2.2
trans TCP
port 323
state NONE
time 23:17:14.282530
ipv4roa 0
ipv6roa 0
address 3.3.3.3
trans TCP
port 323
state NONE
time 23:17:14.282654
ipv4roa 0
ipv6roa 0
address 10.194.61.8
trans TCP
port 3323
state ESTAB
time 23:17:18.484155
ipv4roa 286065
ipv6roa 67186
address 10.48.32.134
trans TCP
port 3323
state ESTAB
time 23:17:18.710239
ipv4roa 286084
ipv6roa 67186
id 1
```

- **show bgp rpki table {ipv4|ipv6} {IP アドレス/マスク長}**：現在の RPKI ROA データベースを表示します。オプションを指定しなかった場合、コマンドは IPv4 ROA データベースを表示します。IPv6 オプション（show bgp rpki table ipv6）を指定すると、このコマンドは IPv6 ROA データベースを表示します。（接続の問題などにより）一時的にダウンしているキャッシュから受信した ROA は（*）で表示されます。キャッシュ セッションがその

キャッシュのパージ時間内に確立されない場合、これらの ROA は RPKI データベースから削除されます。

```
show bgp rpki cache 10.194.61.8
jsonrpc 2.0
result
body
TABLE_RPKICACHE
ROW_RPKICACHE
cacheaddr 10.194.61.8
trans TCP
port 3323
state ESTAB
retry 1
socket 86
serial 16511
nonce 6B20
pstate DATA_END
refreshtime 657
resptime 30
purgetime 60
addipv4roa 293096
addipv6roa 71307
delipv4roa 7008
delipv6roa 4117
id 1
```

table show コマンドの後に ROA プレフィックス ブロックが指定されている場合（たとえば、show bgp rpki table 67.21.36.0/24 max 24）、その特定の ROA エントリが詳細に表示されます（ROA が存在する場合）。

（注） 1 つの ROA（IP アドレス/最小-最大）は、複数のオリジン AS を持つことができ、複数のキャッシュからソースを取得できます。

```
RPKI ROA entry for 67.21.36.0/24-24
  Origin-AS: 3970 from 147.28.0.11
* Origin-AS: 3970 from 198.180.150.1

* Source cache is down / Entry pending removal
```

- **show bgp rpki cache {IP address}** ：以下に示すように、「show bgp summary」など、構成されているすべてのキャッシュとそのパラメータの要約リストを表示します。

```
show bgp ipv4 unicast origin-as validity-state invalid

jsonrpc 2.0
result
body
TABLE_vrf
ROW_vrf
vrf-name-out default
TABLE_afi
ROW_afi
afi 1
TABLE_safi
ROW_safi
```

```
safi 1
af-name IPv4 Unicast
table-version 429
router-id 51.51.51.51
TABLE_rd
ROW_rd
TABLE_prefix
ROW_prefix
ipprefix 52.1.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best none
type external
origin_as_validity_code I
statuscode *
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
bestcode
ipprefix 52.2.2.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best none
type external
origin_as_validity_code I
statuscode *
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
bestcode
ipprefix 52.3.3.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best none
type external
origin_as_validity_code I
statuscode *
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
bestcode
ipprefix 52.4.4.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best none
type external
origin_as_validity_code I
```

```
statuscode *
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
bestcode
ipprefix 200.8.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.17.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.18.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.19.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.21.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.25.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.27.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.32.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.35.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.37.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.40.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.42.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.43.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.44.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.45.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.46.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.48.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.49.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.59.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.60.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.63.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.72.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.73.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.74.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.76.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.80.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.82.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.83.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.84.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.85.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.86.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.87.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.93.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.94.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.104.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.105.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.106.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.107.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.109.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.110.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.111.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.117.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.118.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.119.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.120.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.121.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.122.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.124.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.125.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.127.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.129.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.130.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.131.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.132.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.133.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.134.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.135.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.137.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.139.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.145.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.160.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.169.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.197.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
statuscode *
bestcode >
typecode e
ipnexthop 50.51.50.50
weight 0
aspath 14000
origin ?
metric 0
ipprefix 200.219.1.0/24
TABLE_path
ROW_path
pathnr 0
status valid
best bestpath
type external
origin_as_validity_code I
```

```
    statuscode *
    bestcode >
    typecode e
    ipnexthop 50.51.50.50
    weight 0
    aspath 14000
    origin ?
    metric 0
    ipprefix 200.224.1.0/24
    TABLE_path
    ROW_path
    pathnr 0
    status valid
    best bestpath
    type external
    origin_as_validity_code I
    statuscode *
    bestcode >
    typecode e
    ipnexthop 50.51.50.50
    weight 0
    aspath 14000
    origin ?
    metric 0
    ipprefix 200.236.1.0/24
    TABLE_path
    ROW_path
    pathnr 0
    status valid
    best bestpath
    type external
    origin_as_validity_code I
    statuscode *
    bestcode >
    typecode e
    ipnexthop 50.51.50.50
    weight 0
    aspath 14000
    origin ?
    metric 0
    ipprefix 200.237.1.0/24
    TABLE_path
    ROW_path
    pathnr 0
    status valid
    best bestpath
    type external
    origin_as_validity_code I
    statuscode *
    bestcode >
    typecode e
    ipnexthop 50.51.50.50
    weight 0
    aspath 14000
    origin ?
    metric 0
    id 1
```

前のコマンドでキャッシュIPアドレスが指定されている場合、以下に示すように、そのキャッシュの詳細情報が表示されます。

```
RPKI Cache 10.0.0.2 pref 1
  Transport: SSH port 22
  Cache State: ESTAB
```

```
  Conn attempts: 1
  Last reset
    Timest: Thu May  5 19:06:15 2011
    Reason: protocol error
 TCP information
  FD: 10
 RPKI-RTR protocol information
  Serial number: 100
  Cache nonce: 0xFF
  Protocol State: DATA_END
  Protocol exchange
  Total bytes RX: 528
    Total bytes TX: 0
    ROAs announced:    132 IPv4      45 IPv6
    ROAs withdrawn:      0 IPv4       0 IPv6
    Error Reports :      0 sent       0 rcvd
  Last protocol error
    Reason: response timeout
    Detail: response timeout while in SERIAL_QUERY_SENT state
```

show bgp {ipv4 unicast|ipv6 unicast} origin-as validation-state {valid|invalid|unknown}：「show bgp」コマンドは、path->validation_stateに基づいてBGPテーブル出力をフィルタリングする新しいオプションで拡張されました。

```
show bgp rpki table ipv4 52.1.1.0/24
jsonrpc 2.0
result
id 1
```

このコマンドで有効性状態（有効、無効、または不明）を指定すると、BGPテーブルのフィルタとして機能します。その有効性状態に一致するBGPパスのみが表示されます。

```
Network            Next Hop            Metric LocPrf Weight Path
*  1.1.1.1/32         20.0.0.2                              0 400 i
*  2.2.2.2/32         20.0.0.2                              0 400 i
*  3.3.3.3/32         20.0.0.2                              0 400 i
*  4.4.4.4/32         20.0.0.2                              0 400 i
*  5.5.5.5/32         20.0.0.2                              0 400 i
```

RPKI Clear コマンド

以下はRPKI Clear コマンドです。

- **clear bgp rpki cache *** - このコマンドは、構成されているすべてのRPKIキャッシュのトランスポート セッションをリセットし、すべてのキャッシュから受信したすべてのIPv4およびIPv6 ROAのRPKIデータベースを即座に消去します。

RPKI Debug および Event History コマンド

以下は、RPKI Debug および Event History コマンドです。

- **debug bgp rpki** - このコマンドは、プレフィックス検証を除くすべてのRPKI関連操作のデバッグをオンにします。 これには、RPKIキャッシュ接続、RPKIキャッシュのプロトコ

ル ステート マシン、ROA の挿入や削除などの RPKI データベース イベントなどのデバッグ イベントが含まれます。

- **sh bgp event-history rpki** - このコマンドは、RPKI に関する高レベルの情報をダンプします。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。