CHAPTER 8

証明書のプロビジョニング

Secure Sockets Layer（SSL）プロトコルにより、ネットワーク通信を保護し、送信前のデータを暗号化し、セキュリティを実現することができます。多くのアプリケーション サーバと Web サーバは、SSL 設定用の keystore の使用をサポートしています。スイッチと KMC との間での SSL 使用には、公開キー インフラストラクチャのプロビジョニングが必要です。

この章では、次の事項について説明します。

公開キー インフラストラクチャ証明書に関する情報

証明書は、サーバや企業などのエンティティを識別し、その ID を公開キーに関連付けるために使用される電子ドキュメントです。

認証局（CA）は、ID を確認して証明書を発行する機関です。CA が発行する証明書により、その証明書が特定するエンティティ名（サーバ名やデバイス名など）に、特定の公開キーがバインドされます。証明書で認証された公開キーだけが、証明書が特定するエンティティが所有している対の秘密キーと機能します。証明書により、偽装目的の疑似公開キーの使用を防ぐことができます。

SSL の前提条件

SSL を設定する前に、次の点に注意してください。

- キー、証明書、および証明書署名要求を生成するために、無料で使用できる OpenSSL アプリケーションなどのサードパーティ製ツールをインストールする必要があります。Windows 用の OpenSSL を次のリンクからダウンロードします。http://gnuwin32.sourceforge.net/packages/openssl.htm Windows にインストールした場合、openssl.exe はデフォルトでは c:\openssl\bin にあります。
- すべてのスイッチ、DCNM-SAN、および OpenSSL コマンドを実行しているシステムの時刻が同期していることを確認します。
- CA 証明書と KMC 証明書にそれぞれ異なる ID を指定します。
- JRE1.6 JAVA keytool のみが、Java Keystores（JKS）ファイルへの PKCS12 証明書のインポートをサポートします。

CLI を使用した SSL の設定

ここでは、SSL の設定に関する次の内容について説明します。

CA 証明書の作成

組織は事前に CA 証明書を所有している可能性があります。セキュリティ管理者に CA を要求する場合には、PEM フォーマットの CA 証明書を必要としており、SME 設定の一部として証明書に署名されている必要があることを伝えます。既存の CA がないかまたはそれを使用したくない場合は、OpenSSL コマンドを使用して新しい CA を作成できます。

このコマンドを使用して、認証局（CA）を作成します。このコマンドは、証明書（ID と公開キー）および秘密キーを作成します。秘密キーは必ず保護する必要があります。一般的な企業組織では、秘密キーは必ず事前に備えているはずです。

OpenSSL アプリケーションを使用して CA 証明書を作成します。365 日証明書に対しては、次のコマンドを入力します。

```
OpenSSL> req -x509 -days 365 -newkey rsa:2048 -out cacert.pem -outform PEM
```

このコマンドは、cacert.pem および privkey.pem という 2 つのファイルを、OpenSSL.exe があるディレクト内に作成します。cacert.pem ファイルは、証明書です。privkey.pem ファイルは、安全な場所に保存する必要があります。

トラストポイントの設定

この一連の手順は、DCNM-SAN サーバが管理するすべてのスイッチに対して実行する必要があります。必ず同じトラストポイント名をすべてのスイッチに使用します。

手順の詳細

トラストポイントを設定するには、次の手順を実行します。

ステップ 1　コンフィギュレーション モードを開始します。

```
switch# config t

Enter configuration commands, one per line.End with CNTL/Z.
```

ステップ 2　my_ca という名前のトラストポイントを作成します。

```
switch(config)# crypto ca trustpoint my_ca
```

ステップ 3　トラストポイント サブモードで、スイッチの RSA キーのペアを作成します。

```
switch(config-trustpoint)# rsakeypair my_ca_key 2048
```

ステップ 4　トラストポイント サブモードを終了します。

```
switch(config-trustpoint)# exit
```

ステップ 5　手順 1 で作成した cacert.pem の内容をカット アンド ペーストして、トラストポイントの cacert.pem ファイルを認証します。

```
switch(config)# crypto ca authenticate my_ca

input (cut & paste) CA certificate (chain) in PEM format;
end the input with a line containing only END OF INPUT :
----BEGIN CERTIFICATE----
MIIDnjCCAwegAwIBAgIBADANBgkqhkiG9w0BAQQFADCBlzELMAkGA1UEBhMCVVMx
EzARBgNVBAgTCkNhbGlmb3JuaWExETAPBgNVBAcTCFNhbiBKb3NlMRowGAYDVQQK
ExFDaXNjbyBTeXN0ZW1zIEluYzEOMAwGA1UECxMFRGV2ZWwxETAPBgNVBAMTCG1h
bWFzc2V5MSEwHwYJKoZIhvcNAQkBFhJtYW1hc3NleUBjaXNjby5jb20wHhcNMDcx
MTIyMDgzNDM1WhcNMDgxMTIxMDgzNDM1WjCBlzELMAkGA1UEBhMCVVMxEzARBgNV
BAgTCkNhbGlmb3JuaWExETAPBgNVBAcTCFNhbiBKb3NlMRowGAYDVQQKExFDaXNj
byBTeXN0ZW1zIEluYzEOMAwGA1UECxMFRGV2ZWwxETAPBgNVBAMTCG1hbWFzc2V5
MSEwHwYJKoZIhvcNAQkBFhJtYW1hc3NleUBjaXNjby5jb20wgZ8wDQYJKoZIhvcN
AQEBBQADgY0AMIGJAoGBAMbZAv0+Ka/FS3/jwdaqItc8Ow3alpw9gyqEzA3uFLjN
tXSfHRu9OsrP5tliHHlJP+fezeAUuVfmMTPrOIxURcF2c7Yq1Ux5s4Ua3cMGf9BG
YBRbhO8Filt2mGDqY5u0mJY+eViR69MZk8Ouj+gRxQq83fB8MqJG39f1BedRcZLB
AgMBAAGjgfcwgfQwHQYDVR0OBBYEFGXsBg7f7FJcL/741j+M2dgI7rIyMIHEBgNV
HSMEgbwwgbmAFGXsBg7f7FJcL/741j+M2dgI7rIyoYGdpIGaMIGXMQswCQYDVQQG
EwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTERMA8GA1UEBxMIU2FuIEpvc2UxGjAY
BgNVBAoTEUNpc2NvIFN5c3RlbXMgSW5jMQ4wDAYDVQQLEwVEZXZlbDERMA8GA1UE
AxMIbWFtYXNzZXkxITAfBgkqhkiG9w0BCQEWEm1hbWFzc2V5QGNpc2NvLmNvbYIB
ADAMBgNVHRMEBTADAQH/MA0GCSqGSIb3DQEBBAUAA4GBAFmDucZlBZFJk09IihEm
5wd4oouxHsKPQroyG/CYShv1XXAyEGytxuCAITDzMq2IJiFbZt0kIiyuP9YRQLNR
z47G4IRJGp5J2HnOc2cdF8Mc0DDApdgndUiIX/lv7vuQfyxqX45oSncwQct3y38/
FPEbcRgZgnOgwcrqBzKV0Y3+
----END CERTIFICATE----
END OF INPUT
Fingerprint(s): MD5 Fingerprint=1E:18:10:69:7B:C1:CC:EA:82:08:67:FB:90:7D:58:EB

Do you accept this certificate? [yes/no]:yes
```

ステップ 6　手順 2 で作成したトラストポイントに登録するための証明書要求を生成します。この要求は、CA がスイッチの証明書に署名するために使用します。

```
switch(config)# crypto ca enroll my_ca

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: ips-vegas8.cisco.com
Include the switch serial number in the subject name? [yes/no]:no
Include an IP address in the subject name [yes/no]:no
The certificate request will be displayed...

----BEGIN CERTIFICATE REQUEST----
MIIBJTCB0AIBADAfMR0wGwYDVQQDExRpcHMtdmVnYXM4LmNpc2NvLmNvbTBcMA0G
CSqGSIb3DQEBAQUAA0sAMEgCQQCeAzv5w9d32YpPfYdNYoFjOW0yRVbYEe+mNHi8
b2VPOVZ6UOFdhIS1Im0/Xv1Bpcuy4TRktu7whNyyvvu3niVdAgMBAAGgTDAVBgkq
hkiG9w0BCQcxCBMGbmJ2MTIzMDMGCSqGSIb3DQEJDjEmMCQwIgYDVR0RAQH/BBgw
FoIUaXBzLXZlZ2FzOC5jaXNjby5jb20wDQYJKoZIhvcNAQEEBQADQQBzPcKE3Eje
TjODnPXNkz1WsU3oUdsuxOT/m1OSBZvhBfHICQZZpfS2ILqaQP16LiZCZydHWViN
Q+9LmHUZ4BDG
----END CERTIFICATE REQUEST----

switch(config)#
```

ステップ 7　OpenSSL.exe があるディレクトリ内に switch.csr というファイルを作成します。手順 6 で作成した証明書要求をカット アンド ペーストします。

ファイルの内容に、BEGIN CERTIFICATE REQUEST 行と END CERTIFICATE REQUEST 行を必ず組み込みます。

ステップ 8　次のコマンドを入力して、OpenSSL アプリケーションでスイッチ証明書要求を使用して証明書を生成します。

```
OpenSSL> x509 -req -days 365 -in switch.csr -CA cacert.pem -CAkey privkey.pem -set_serial
01 -out switch.pem
```

これが、CA によって署名されたスイッチのパブリック証明書になります。

（注）　セキュリティ管理者が CA を制御している場合、セキュリティ管理者に switch.csr ファイルを送信し、この手順の実行と switch.pem ファイルでの応答を依頼する必要があります。

ステップ 9　手順 8 で作成された switch.pem ファイルの内容をカット アンド ペーストして、スイッチ上に署名付き証明書をインポートします。

```
switch(config)# crypto ca import my_ca certificate
input (cut & paste) certificate in PEM format:
----BEGIN CERTIFICATE----
MIIB4jCCAUsCAQEwDQYJKoZIhvcNAQEEBQAwgZcxCzAJBgNVBAYTAlVTMRMwEQYD
VQQIEwpDYWxpZm9ybmlhMREwDwYDVQQHEwhTYW4gSm9zZTEaMBgGA1UEChMRQ2lz
Y28gU3lzdGVtcyBJbmMxDjAMBgNVBAsTBURldmVsMREwDwYDVQQDEwhtYW1hc3Nl
eTEhMB8GCSqGSIb3DQEJARYSbWFtYXNzZXlAY2lzY28uY29tMB4XDTA3MTIxNDAy
MzIzOVoXDTA4MTIxMzAyMzIzOVowHzEdMBsGA1UEAxMUaXBzLXZlZ2FzOC5jaXNj
by5jb20wXDANBgkqhkiG9w0BAQEFAANLADBIAkEAngM7+cPXd9mKT32HTWKBYzlt
MkVW2BHvpjR4vG9lTzlWelDhXYSEtSJtP179QaXLsuE0ZLbu8ITcsr77t54lXQID
AQABMA0GCSqGSIb3DQEBBAUAA4GBAKR3WAAF/9zMb2u9A42I2cB2G5lucSzndc4P
+O4sYZF5pBt7UpyAs1GKAqivGXVq2FJ2JetX78Fqy7jYCzanWm0tck0/G1dSfr/X
lCFXUuVed9de02yqxARSEx8mX4ifqzYHErHdbi+vDAaMzkUEvHWthOuUZ7fvpoNH
+xhRAuBo
----END CERTIFICATE----
```

これで、スイッチ上のトラストポイントの設定が完了しました。これには定義されたトラストポイント、認識される CA、公開キーと秘密キーのペア、およびスイッチを識別する CA 署名付き証明書が含まれます。署名付き証明書は、CA を認識するすべてのエンティティとの PKI 通信にも使用できます。ファブリック内のすべてのスイッチに対して、手順 1 ～ 9 を繰り返します。

トラストポイントの削除

この一連の手順は、暗号化された CA 署名済みトラストポイントを削除するために、すべてのスイッチに対して実行する必要があります。

手順の詳細

トラストポイントを削除するには、次の手順を実行します。

ステップ 1　コンフィギュレーション モードを開始します。

```
switch# config t

Enter configuration commands, one per line.End with CNTL/Z.
```

ステップ 2　トラストポイント モードに入ります。

```
switch(config)# crypto ca trustpoint my_ca
```

ステップ 3　トラストポイントに対応する証明書を削除します。

```
switch(config-trustpoint)# delete certificate force
```

ステップ 4　トラストポイント サブモードでスイッチの RSA キー ペアを削除します。

```
switch(config-trustpoint)# no rsakeypair my_ca_key
```

ステップ 5　トラストポイントに対応する CA 証明書を削除します。

```
switch(config-trustpoint)# delete ca-certificate
```

ステップ 6　トラストポイント サブモードを終了します。

```
switch(config-trustpoint)# exit
```

ステップ 7　設定されているトラストポイントを削除します。

```
switch(config)# no crypto ca trustpoint my_ca
```

KMC 証明書の生成

手順の詳細

KMC 証明書を生成するには、次の手順に従います。OpenSSL アプリケーションで次のコマンドを入力して、KMC 証明書を生成します。

ステップ 1　KCM サーバの秘密キーを作成します。

```
OpenSSL> genrsa -out sme_kmc_server.key 2048
```

ステップ 2　手順 1 で作成した秘密キーを使用して、証明書署名要求を作成します。

```
OpenSSL> req -new -key sme_kmc_server.key -out sme_kmc_server.csr -config openssl.conf
```

ステップ 3　証明書と秘密キーを使用して、KMC サーバの署名付き証明書を作成します。

```
OpenSSL> x509 -req -days 365 -in sme_kmc_server.csr -CA cacert.pem -CAkey privkey.pem
          -CAcreateserial -out sme_kmc_server.cert
```

（注）　セキュリティ管理者が CA を制御している場合、セキュリティ管理者に sme_kmc_server.csr ファイルを送信し、この手順の実行と sme_kmc_server.cert での応答を依頼する必要があります。

ステップ 4　署名付き KMC 証明書を pkcs12 フォーマットにエクスポートします。

```
OpenSSL> pkcs12 -export -in sme_kmc_server.cert -inkey sme_kmc_server.key -out
          sme_kmc_server.p12
```

ステップ 5　この PKCS12 キーストアを Java キーストアに、JAVA keytool（JRE 1.6）を使用してインポートします。

"<JAVA_HOME>\bin\keytool" -importkeystore -srckeystore sme_kmc_server.p12 -srcstoretype PKCS12 -destkeystore sme_kmc_server.jks -deststoretype JKS

（注）　パスワードはプロパティ ファイル内で更新する必要があるため、覚えておいてください。

ステップ 6　CA 証明書を Java キーストアに、JAVA keytool（JRE 1.6）を使用してインポートします。

"<JAVA_HOME>\bin\keytool" -importcert -file cacert.pem -keystore sme_kmc_trust.jks -storetype JKS

ステップ 7　キーストア ファイルを <*install path*>dcm\fm\conf\cert ディレクトリ内に配置します。

ステップ 8　DCNM-SAN Web クライアントのキー マネージャ設定にある、KMC SSL の設定を変更します。

ステップ 9　DCNM-SAN サーバを再起動します。

（注）　さらに、手順 5 と 6 で作成した Java キーストアを使用する代わりに、sme_kmc_server.p12 を KMC 証明書として、および cacert.pem を KMC トラスト証明書として使用することもできます。

（注）　クラスタが SSL ON オプションで稼働している場合、すべての DCNM のアップグレードにキーストア ファイルの配置が必要です。DCNM のアップグレードでは、キーストア ファイルは保持されません。

SSL の機能履歴

表 8-1 に、この機能のリリース履歴を示します。

表 8-1　SSL の機能履歴

機能名	リリース	機能情報
ソフトウェアの変更	5.2(1)	Release 5.2(1) では、Fabric Manager は DCNM for SAN（DCNM-SAN）という名前に変更されました。
	4.1(1c)	Release 4.1(1b) 以降、MDS SAN-OS ソフトウェアは MDS NX-OS ソフトウェアに名前が変更されました。旧リリース名は変更されておらず、参照はすべて維持されています。
自己署名証明書の生成およびインストール	4.1(1c)	Release 4.1(1c) 以降、KMC の場合の SSL 設定は、Fabric Manager Server からは独立しています。
Secure Sockets Layer（SSL）の導入	3.3(1c)	SME ウィザードで SME に対して SSL を設定し、SSL の設定を編集する方法を説明します。