PKI のデータ形式

内容

概要

このドキュメントでは、一般的な公開キー インフラストラクチャ（PKI）のデータ形式および工ンコーディングについて説明します。

前提条件

要件

次の項目に関する知識があることが推奨されます。

- 公開キー暗号化（基本概念）。
- 公開キー インフラストラクチャ（基本概念）。

使用するコンポーネント

このドキュメントの内容は、特定のソフトウェアやハードウェアのバージョンに限定されるものではありません。

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、初期（デフォルト）設定の状態から起動しています。対象のネットワークが実稼働中である場合には、どのようなコマンドについても、その潜在的

な影響について確実に理解しておく必要があります。

表記法

ドキュメント表記の詳細は、『シスコ テクニカル ティップスの表記法』を参照してください。

ASN.1 表記

抽象構文記法 1（ASN.1）は、データのタイプと値、およびさまざまなデータ構造でのその使用および組み合わせ方法を定義するための正式な言語です。規格の目的は、情報を送信用に符号化する方法を制限することなく、情報の抽象構文を定義することです。

次に示すのは、*X.509 RFC* から抜粋した例です。

```
   Version  ::=  INTEGER  {  v1(0), v2(1), v3(2)  }
CertificateSerialNumber ::= INTEGER
Validity ::= SEQUENCE {
notBefore Time,
notAfter Time }
Time ::= CHOICE {
utcTime UTCTime,
generalTime GeneralizedTime }
```

国際電気通信連合（ITU-T）規格のサイトにある、次のドキュメントを参照してください。

- X.680 ASN.1:Specification of basic notation
- X.681 ASN.1:Information object specification
- X.682 ASN.1:Constraint specification
- X.683 ASN.1:Parameterization of ASN.1 specifications

ITU-T recommendations search : **X.509 を [Rec.or standard] で、[Language] を [ASN.1]** に設定して検索してください。

BER/CER/DER 符号化

ITU-T では、ASN.1 に示すデータ構造をバイナリ データに符号化する標準的な方法を定義しています。X.690 は、基本符号化規則（BER）と 2 つのサブセット、標準符号化規則（CER）と識別符号化規則（DER）を定義しています。 この 3 つの規則はすべて、**SEQUENCE**、**SET**、および **CHOICE** で構築された階層構造の **type-length-value** データ フィールドに基づいており、次の相違点があります。

- BER は、同じデータを符号化する複数の方法を提供しており、暗号化操作には適していません。
- CER は、明確な符号化を提供し、使用するデータの長さは決まっていません。特定の場合に、end-of-data マーカーを使用します。
- DER は、明確な符号化を提供し、特定の場合に、長さが明示されたタグを使用します。
- この 3 つのうち、PKI および暗号化のペイロードを扱う場合に見られるのは、通常、DER です。

例：DER では、20 ビット値 1010 1011 1100 1101 1110 が、次のように符号化されます。

- **タグ**：0x03
- **length:**0x04
- **value**：0x04ABCDE**0**
- **完全なDERエンコード**：0x030404ABCDE0

先頭の **04　4　0**

ITU-T 規格のサイトにある、次のドキュメントを参照してください。

- X.690 ASN.1 encoding rules:Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)

次のドキュメントは、Wikipedia サイトで参照できます。

- Basic Encoding Rules
- Canonical Encoding Rules
- Distinguished Encoding Rules

DER の 16 進数ダンプ

Cisco IOS、適応型セキュリティ アプライアンス（ASA）、および他のデバイスでは、**show running-config コマンドで、DER の内容が 16 進数ダンプとして表示されます。** 出力は以下のとおりです。

```
crypto pki certificate chain root
certificate ca 01
30820213 3082017C A0030201 02020101 300D0609 2A864886 F70D0101 04050030
1D310C30 0A060355 040B1303 54414331 0D300B06 03550403 1304726F 6F74301E
170D3039 30373235 31313436 33325A17 0D313230 37323431 31343633 325A301D
...
```

このタイプの 16 進数ダンプは、さまざまな方法で変換して DER に戻すことができます。たとえば、次のように空白文字を削除し、**xxd プログラム**にパイプできます。

```
$ cat ca.hex | tr -d ' ' | xxd -r -p -c 32 | openssl x509 -inform der -text -noout
```

もう一つの簡単な方法は、次の Perl スクリプトを使用することです。

```
#!/usr/bin/perl
foreach (<>) {
s/[^a-fA-F0-9]//g;
print join("", pack("H*", $_));
}

$ perl hex2der.pl < hex-file.txt > der-file.der
```

また、以前にそれぞれ、拡張子 .hex が付いたファイルに手動でコピーされた証明書ダンプを bash コマンドラインから変換するコンパクトな方法は、次に示すとおりです。

```
for hex in *.hex; do
b="${hex%.hex}"
hex2der.pl < "$hex" > "$b".der
openssl x509 -inform der -in "$b".der > "$b".pem
openssl x509 -in "$b".pem -text -noout > "$b".txt
```

```
done
```
各ファイルは、次のようになります。

- **file.hex：元のファイル（16 進数だけを含む必要があります）。**
- **file.der：DER（バイナリ）形式の証明書。**
- **file.pem：PEM（Base64 + ヘッダー/フッター）形式の証明書。**
- **file.txt：ユーザ フレンドリで読み取り可能なバージョンの証明書。**

Base64 符号化

Base64 符号化は、**uuencode** と同様に、64 個の印刷可能な文字（`A-Za-z0-9+/`）だけのバイナリ データを表示します。バイナリから Base64 への変換では、元のデータのすべての 6 ビット ブロックが変換テーブルで、8 ビットの印刷可能な ASCII 文字に符号化されます。したがって、符号化後のデータのサイズは 33% 増加します（データのビット数の 8 をかけて 6 で割ると 1.333 になるため）。

8 ビットの 4 グループを 6 ビットの 3 グループに変換するために 24 ビット バッファが使用されます。そのため、入力データ ストリームの最後に、1 または 2 バイトのパディングが必要となる場合があります。パディングは、Base64 符号化データの末尾で、各符号化中に入力に追加された 8 個のパディング ビットのグループごとに 1 個の等号（=

Wikipedia の例を参照してください。

最新の情報については、RFC 4648:The Base16, Base32, and Base64 Data Encodings を参照してください。

PEM 符号化

Privacy Enhanced Mail（PEM）は、安全なメッセージを交換するためのインターネット技術特別調査委員会（IETF）PKI 標準一式です。これ自体は、広く使用されてはいませんが、Base64 符号化 PKI 関連データの書式設定と交換のために、そのカプセル化構文は広く借用されています。

PEM の RFC 1421、項 4.4「Encapsulation Mechanism」（カプセル化のメカニズム）では、RFC 934 に基づく Encapsulation Boundaries（EB）で区切られていると PEM メッセージを定義しています。その形式は、次のとおりです。

```
-----BEGIN PRIVACY-ENHANCED MESSAGE-----
Header: value
Header: value
...

Base64-encoded data
...
-----END PRIVACY-ENHANCED MESSAGE-----
```
現在、実際には、PEM 形式のデータの配布時に、次の境界形式が使用されます。

```
-----BEGIN type-----
...
-----END type-----
```

type には、次のような他のキーまたは証明書を使用できます。

- `RSA PRIVATE KEY`
- `ENCRYPTED PRIVATE KEY`
- `CERTIFICATE`
- `CERTIFICATE REQUEST`
- `X509 CRL`

注：RFC では、これは必須ではありませんが、EB 内の先頭または末尾のダッシュ（-）の数は重要で、常に 5 である必要があります。 それ以外の場合は、OpenSSL などの一部のアプリケーションで入力に障害が発生します。一方、Cisco IOS などの他のアプリケーションでは、EB をまたく必要としません。

詳細については、次の最新の RFC を参照してください。

- RFC 1421 :PEM Part I:Message Encryption and Authentication Procedures
- RFC 1422 :PEM Part II:Certificate-Based Key Management
- RFC 1423 :PEM Part III:Algorithms, Modes, and Identifiers
- RFC 1424 :PEM Part IV:Key Certification and Related Services

X.509 証明書と CRL

X.509 は、Open Systems Interconnection に関する拡張 ITU 仕様である X.500 のサブセットです。これは、特に証明書と公開キーに関するもので、IETF によってインターネット標準として翻案されています。X.509 は、ASN.1 表記で RFC に表現された構造と構文を提供し、証明書情報と証明書失効リストを保存します。

X.509 PKI では、CA が、Rivest-Shamir-Adleman（RSA）キーまたはデジタル署名アルゴリズム（DSA）キーなどの公開キーを、特定の識別名（DN）、または電子メール アドレスや完全修飾ドメイン名（FQDN）などの代替名にバインドする証明書を発行します。 DN は、X.500 標準の構造に従います。以下が一例です。

```
CN=common-name,OU=organizational-
unit,O=organization,L=location,C=country
```

ASN.1 定義により、X.509 データは DER に符号化され、バイナリ形式で交換されます。また、端末でのコピー/貼り付けのようなテキスト ベースの通信手段として、Base64/PEM に変換することも選択できます。

- ITU-T 標準ドキュメント、X.509 Open Systems Interconnection - The Directory:Public-key and attribute certificate frameworks を参照してください。
- 詳細については、RFC 5280:X.509 Certificate and Certificate Revocation List (CRL) Profile を参照してください。

PKCS 規格

Public Key Cryptography Standards（PKCS）は、RSA ラボでの仕様であり、その一部は業界標準へと進化しました。最も頻繁に目にするものは、次のトピックを扱っています。ただし、データ形式を扱っていないものもあります。

PKCS#1（RFC 3347）：RSA ベースの暗号化の実装（暗号プリミティブ、暗号化/署名方式、ASN.1 構文）について説明します。

PKCS#5（RFC 2898）：パスワード ベースのキーの導出について説明します。

PKCS#7（RFC 2315）および **S/MIME RFC 3852：**署名され、暗号化されたデータと関連証明書を送信するためのメッセージ構文を定義します。多くの場合、単に X.509 証明書のコンテナとして使用されます。

PKCS#8：クリア テキストまたは暗号化された RSA キーペアを伝送するためのメッセージ構文を定義します。

PKCS#9（RFC 2985）：その他のオブジェクト クラスおよび ID 属性を定義します。

PKCS#10（RFC 2986）：証明書署名要求（CSR）のためのメッセージ構文を定義します。CSR は、エンティティによって CA に送信され、公開キー情報、ID、追加属性など、CA が署名する情報を含んでいます。

PKCS#12：関連するPKIデータ(通常は**entity keypair + entity cert + rootおよびintermediate CA certificates**)を1つのファイル内にパッケージングするためのコンテナを定義します。これは、Microsoft の個人情報交換（PFX）形式が進化したものです。

次のリソースを参照してください。

- PKCS に関する Wikipedia 記事
- PKCS に関する RSA ラボのページ

関連情報

- テクニカル サポートとドキュメント – Cisco Systems