PKI-Datenformate

Inhalt

[Einführung](#)
[Voraussetzungen](#)
[Anforderungen](#)
[Verwendete Komponenten](#)
[Konventionen](#)
[ASN.1 Notation](#)
[BER/CER/DER-Codierungen](#)
[DER-Hex-Dump](#)
[Base64-Codierung](#)
[PEM-Kodierung](#)
[X.509-Zertifikate und CRLs](#)
[PKCS-Standards](#)
[Zugehörige Informationen](#)

Einführung

In diesem Dokument werden die gängigsten PKI-Datenformate und -Codierungen beschrieben.

Voraussetzungen

Anforderungen

Cisco empfiehlt, über Kenntnisse in folgenden Bereichen zu verfügen:

- Public-Key-Verschlüsselung (grundlegende Konzepte).
- Public-Key-Infrastruktur (grundlegende Konzepte).

Verwendete Komponenten

Dieses Dokument ist nicht auf bestimmte Software- und Hardwareversionen beschränkt.

Die Informationen in diesem Dokument wurden von den Geräten in einer bestimmten Laborumgebung erstellt. Alle in diesem Dokument verwendeten Geräte haben mit einer leeren (Standard-)Konfiguration begonnen. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die potenziellen Auswirkungen eines Befehls verstehen.

Konventionen

Informationen zu Dokumentkonventionen finden Sie unter Cisco Technical Tips Conventions (Technische Tipps von Cisco zu Konventionen).

ASN.1 Notation

Abstract Syntax Notation One (ASN.1) ist eine formale Sprache für die Definition von Datentypen und Werten und für die Verwendung und Kombination dieser Datentypen und Werte in verschiedenen Datenstrukturen. Das Ziel des Standards besteht darin, die abstrakte Informationssyntax zu definieren, ohne die Art der Kodierung der Informationen für die Übertragung einzuschränken.

Das folgende Beispiel stammt aus der *RFC X.509*:

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

Siehe folgende Dokumente von den Standardisierungsstandorten der Internationalen Fernmeldeunion (ITU-T):

- X.680 ASN.1: Spezifikation der Basisnotation
- X.681 ASN.1: Datenobjektbeschreibung
- X.682 ASN.1: Einschränkungsspezifikation
- X.683 ASN.1: Parametrisierung von ASN.1-Spezifikationen

ITU-T Empfehlungssuche - Suchen Sie in **Rec. nach X.509. oder Standard** mit **Spracheinstellung** auf **ASN.1**.

BER/CER/DER-Codierungen

Die ITU-T hat eine Standardmethode zur Kodierung von Datenstrukturen definiert, die in ASN.1 in Binärdaten beschrieben sind. X.690 definiert grundlegende Kodierungsregeln (Basic Encoding Rules, BER) und ihre beiden Untersätze, Canonical Encoding Rules (CER) und Distinguished Encoding Rules (DER). Alle drei Werte basieren auf **Datenfeldern mit** typischen Werten, die in einer hierarchischen Struktur gepackt sind. Diese Daten werden aus **SEQUENCE**, **SET** und **CHOICEs** erstellt, wobei folgende Unterschiede bestehen:

- BER bietet mehrere Möglichkeiten, die gleichen Daten zu verschlüsseln, was für Krypto-Operationen nicht geeignet ist.
- CER bietet eine eindeutige Codierung und verwendet Daten mit unbestimmter Länge, in bestimmten Fällen mit einem End-of-Data-Marker.
- DER bietet eine eindeutige Codierung und verwendet in bestimmten Fällen explizite Längentags.

- Unter den drei ist der, der normalerweise bei PKI- und Krypto-Payloads vorkommt.

Beispiel: In DER wird der 20-Bit-Wert 1010 1011 1100 1101 1110 wie folgt codiert:

- **Tag**: `0 x 03` (Bitstring)
- **Länge**: `0 x 04` (Byte)
- **Wert**: `0 x 04 ABCDE0`
- **vollständige DER-Codierung**: `0x030404ABCDE0`

Der führende Wert `04` bedeutet, dass die letzten 4 Bit (gleich der folgenden `0`-Ziffer) des verschlüsselten Werts verworfen werden müssen, da der verschlüsselte Wert nicht auf einer Bytegrenze endet.

Informationen hierzu finden Sie auf der TU-T-Standardseite:

- X.690 ASN.1-Kodierungsregeln: Spezifikation von Basic Encoding Rules (BER), Canonical Encoding Rules (CER) und Distinguished Encoding Rules (DER)

Weitere Informationen finden Sie auf der Wikipedia-Website:

- Grundlegende Kodierungsregeln
- Kanonische Kodierungsregeln
- Unterscheidete Kodierungsregeln

DER-Hex-Dump

Cisco IOS, Adaptive Security Appliance (ASA) und andere Geräte zeigen DEN-Inhalt als **Hex-Dump** mit dem Befehl **show running-config an**. Die Ausgabe lautet:

```
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
...
```

Diese Art von Hexadezimaldump kann auf verschiedene Weise zurück in DER konvertiert werden. Sie können z. B. Leerzeichen entfernen und zum **Programm xxd** leiten:

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

Eine weitere einfache Möglichkeit ist, dieses Perl-Skript zu verwenden:

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

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

Darüber hinaus ist eine kompakte Methode zum Konvertieren von **Zertifikatsabbildern** verfügbar, die zuvor manuell in eine Datei mit der Erweiterung **.hex** kopiert wurden, und zwar aus einer **Bash-**Befehlszeile, wie hier gezeigt:

```
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
```

Jede Datei bietet folgende Vorteile:

- **file.hex -** Die ursprüngliche Datei (darf nur Hexadezimalziffern enthalten).
- **file.der -** Zertifikat im DER-Format (binär).
- **file.pem -** Zertifikat im PEM-Format (Base64 + Header/Footer).
- **file.txt -** Benutzerfreundliche, lesbare Version des Zertifikats.

Base64-Codierung

Die Base64-Codierung stellt die Binärdaten mit nur 64 druckbaren Zeichen (`A-Za-z0-9+/`) dar, die dem **Uuencode** ähneln. Bei der Konvertierung von binären Daten in Base64 wird jeder 6-Bit-Block der ursprünglichen Daten in ein 8-Bit-druckbares ASCII-Zeichen mit einer Übersetzungstabelle kodiert. Daher ist die Größe der Daten nach der Kodierung um 33 Prozent gestiegen (Datenzeiten 8 geteilt durch 6 Bit, entspricht 1,333).

Ein 24-Bit-Puffer wird für die Übersetzung von drei (3) Gruppen von acht (8) Bit in vier (4) Gruppen von sechs (6) Bit verwendet. Daher kann am Ende des Datenstroms für die Eingabe ein (1) oder zwei (2) Byte Padding erforderlich sein. Das Padding wird am Ende der Base64-kodierten Daten durch ein Gleichheitszeichen (`=`) für jede Gruppe von acht (8) Padding-Bits angezeigt, die während der Kodierung zur Eingabe hinzugefügt werden.

Siehe dieses Beispiel von Wikipedia.

Lesen Sie die neuesten Informationen in RFC 4648: Die Datencodierungen Base16, Base32 und Base64.

PEM-Kodierung

Privacy Enhanced Mail (PEM) ist ein vollständiger IETF-PKI-Standard (Internet Engineering Task Force) zum Austausch sicherer Nachrichten. Sie wird nicht mehr als solche verwendet, aber ihre Kapselungssyntax wurde weithin ausgeliehen, um Base64-kodierte PKI-Daten zu formatieren und auszutauschen.

PEM RFC 1421, Abschnitt 4.4: Kapselungsmechanismus definiert PEM-Meldungen als durch EBs (Encapsulation Boundaries) abgegrenzt, die auf RFC 934 basieren, mit folgendem Format:

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

Base64-encoded data
...
-----END PRIVACY-ENHANCED MESSAGE-----
```

In der Praxis wird dieses Grenzformat heute bei der Verteilung von PEM-formatierten Daten verwendet:

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

Typ kann mit anderen Schlüsseln oder Zertifikaten sein, z. B.:

- RSA PRIVATE KEY
- VERSCHLÜSSELTER PRIVATER SCHLÜSSEL
- ZERTIFIKAT
- ZERTIFIKATANFORDERUNG
- X509 CRL

Hinweis: Obwohl die RFCs dies nicht zwingend vorschreiben, ist die Anzahl der in den EBs vorkommenden und nachgestellten Bindestriche (-) signifikant und sollte immer fünf (5) betragen. Andernfalls wird die Eingabe durch einige Anwendungen wie OpenSSL blockiert. Andere Anwendungen, wie Cisco IOS, benötigen dagegen überhaupt keine EBs.

Weitere Informationen finden Sie in den folgenden aktuellen RFCs:

- RFC 1421: PEM Teil I: Nachrichtenverschlüsselungs- und Authentifizierungsverfahren
- RFC 1422: PEM Teil II: Zertifikatsbasierte Schlüsselverwaltung
- RFC 1423: PEM Teil III: Algorithmen, Modi und Identifikatoren
- RFC 1424: PEM Teil IV: Key-Zertifizierung und zugehörige Services

X.509-Zertifikate und CRLs

X.509 ist eine Teilmenge von X.500, einer erweiterten ITU-Spezifikation für Open Systems Interconnection. Es befasst sich speziell mit Zertifikaten und öffentlichen Schlüsseln und wurde von der IETF als Internetstandard angepasst. X.509 stellt eine Struktur und Syntax bereit, die in der RFC mit ASN.1 Notation ausgedrückt ist, um Zertifikatsinformationen und Zertifikatswiderrufungslisten zu speichern.

In einer X.509-PKI gibt eine CA ein Zertifikat aus, das einen öffentlichen Schlüssel bindet, z. B.: einen Rivest-Shamir-Adleman (RSA)- oder DSA-Schlüssel (Digital Signature Algorithm) für einen bestimmten DN oder einen alternativen Namen wie eine E-Mail-Adresse oder einen vollqualifizierten Domänennamen (FQDN). Der DN entspricht der Struktur der X.500-Standards. Hier ein Beispiel:

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

Aufgrund der ASN.1-Definition können X.509-Daten in DER codiert werden, um in binärer Form ausgetauscht und optional für textbasierte Kommunikationsmittel, wie z. B. Copy-Paste auf einem Terminal, in Base64/PEM konvertiert zu werden.

- Weitere Informationen finden Sie im ITU-T-Standarddokument X.509 Open Systems Interconnection - The Directory: Public-Key- und Attributzertifikat-Frameworks.
- Siehe RFC 5280: X.509 Certificate and Certificate Revocation List (CRL) Profile für weitere Informationen.

PKCS-Standards

Die Public-Key Cryptography Standards (PKCS) sind Spezifikationen von RSA Labs, die sich teilweise zu Branchenstandards entwickelt haben. Behandeln Sie die am häufigsten aufgetretenen Themen. Allerdings befassen sich nicht alle mit Datenformaten.

PKCS#1 (RFC 3347**)** - Behandelt die Implementierungsaspekte der RSA-basierten Verschlüsselung (Verschlüsselungsprimitive, Verschlüsselungs-/Signaturschemata, ASN.1-Syntax).

PKCS#5 (RFC 2898**)** - Gilt für die Kennwortbasierte Schlüsselableitung.

PKCS#7 (RFC 2315**)** und **S/MIME** RFC 3852 **-** Definiert eine Nachrichtensyntax zum Übertragen signierter und verschlüsselter Daten und damit verbundener Zertifikate. Wird häufig einfach als Container für X.509-Zertifikate verwendet.

PKCS#8: Definiert eine Nachrichtensyntax zum Transport von Klartext oder verschlüsselten RSA-Schlüsselpaaren.

PKCS#9 (RFC 2985**)** - Definiert zusätzliche Objektklassen und Identitätsattribute.

PKCS#10 (RFC 2986**)** - Definiert eine Nachrichtensyntax für Zertifikatssignierungsanforderungen (Certificate Signing Requests, CSRs). Eine CSR-Anfrage wird von einer Stelle an eine CA gesendet und enthält die von der CA zu signierenden Informationen, z. B. Informationen zu öffentlichen Schlüsseln, Identität und zusätzliche Attribute.

PKCS#12 - Definiert einen Container zum Verpacken verwandter PKI-Daten (i. d. R. **Entitätsschlüsselair** + **Entitätszertifikat** + **Stamm- und Zwischenzertifikate**) in einer einzigen Datei. Es handelt sich um eine Weiterentwicklung des PFX-Formats (Personal Information Exchange) von Microsoft.

Weitere Informationen finden Sie unter:

- Wikipedia-Artikel zu PKCS
- RSA-Labs-Seite auf PKCS

Zugehörige Informationen

- **Technischer Support und Dokumentation - Cisco Systems**