Formati dati PKI

Sommario

Introduzione

In questo documento vengono descritti i formati di dati e le codifiche più comuni di Public Key Infrastructure (PKI).

Prerequisiti

Requisiti

Cisco raccomanda la conoscenza dei seguenti argomenti:

- crittografia a chiave pubblica (concetti di base).
- infrastruttura a chiave pubblica (concetti di base).

Componenti usati

Il documento può essere consultato per tutte le versioni software o hardware.

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata ripristinata ai valori predefiniti. Se la rete è operativa, valutare attentamente eventuali conseguenze derivanti dall'uso dei comandi.

Convenzioni

Per ulteriori informazioni sulle convenzioni usate, consultare il documento Cisco sulle convenzioni nei suggerimenti tecnici.

Notazione ASN.1

Abstract Syntax Notation One (ASN.1) è un linguaggio formale per la definizione dei tipi di dati e dei valori e per il modo in cui tali tipi di dati e valori vengono utilizzati e combinati in varie strutture di dati. L'obiettivo dello standard è quello di definire la sintassi astratta delle informazioni senza vincolare il modo in cui le informazioni sono codificate per la trasmissione.

Di seguito è riportato un esempio estratto dall'*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 }
```

Per ulteriori informazioni, fare riferimento ai seguenti documenti pubblicati sui siti standard dell'Unione internazionale delle telecomunicazioni (ITU-T):

- X.680 ASN.1: Specificazione della notazione di base
- X.681 ASN.1: Specifica oggetto informazioni
- X.682 ASN.1: Specifica vincolo
- X.683 ASN.1: Parametrizzazione delle specifiche ASN.1

Ricerca suggerimenti ITU-T - Ricerca di **X.509** in **Rec. o standard** con **Lingua** impostata su **ASN.1**.

Codifiche BER/CER/DER

L'ITU-T ha definito un metodo standard per codificare le strutture di dati descritte in ASN.1 in dati binari. X.690 definisce le regole di codifica di base (BER, Basic Encoding Rules) e i relativi sottoinsiemi, ovvero le regole di codifica canonica (CER, Canonical Encoding Rules) e le regole di codifica differenziata (DER, Distinguished Encoding Rules). Tutti e tre i valori sono basati su campi dati **tipo-lunghezza-valore** compressi in una struttura gerarchica, creata da **SEQUENCE**, **SET** e **CHOICE**, con le seguenti differenze:

- La tecnologia BER offre diversi metodi di codifica degli stessi dati, che non sono adatti per le operazioni di crittografia.
- La tecnologia CER fornisce una codifica inequivocabile e utilizza dati di lunghezza indefinita, con un indicatore di fine dati in casi specifici.
- DER fornisce una codifica inequivocabile e utilizza tag di lunghezza esplicita in casi specifici.
- Tra i tre, DER è quello che si incontra di solito quando si tratta di PKI e payload di crittografia.

Esempio: In DER, il valore a 20 bit 1010 1011 100 1101 1110 è codificato come:

- **tag**: `0x03` (stringa di bit)
- **lunghezza:** `0x04` (byte)
- **valore:** `0x04ABCDE0`
- **codifica DER completa**: `0x030404ABCDE0`

Il valore iniziale `04` indica che gli ultimi 4 bit (uguale alla cifra `0` finale) del valore codificato devono essere eliminati perché il valore codificato non termina su un limite di byte.

Fare riferimento a questi documenti dal sito degli standard TU-T:

- Regole di codifica X.690 ASN.1: Specifica delle regole di codifica di base (BER), delle regole di codifica canonica (CER) e delle regole di codifica distinte (DER)

Dal sito di Wikipedia, fare riferimento a questi documenti:

- Regole di codifica di base
- Regole di codifica canonica
- Regole di codifica distinte

Dump esadecimale DER

Cisco IOS, Adaptive Security Appliance (ASA) e altri dispositivi visualizzano il contenuto DER come un **dump esadecimale** con il comando **show running-config**. Questo è l'output:

```
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
...
```

Questo tipo di dump esadecimale può essere riconvertito in DER in vari modi. Ad esempio, è possibile rimuovere gli spazi e reindirizzarli al **programma xxd**:

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

Un altro modo facile è usare questo script Perl :

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

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

Inoltre, un modo compatto per convertire **i dump di certificati**, ognuno precedentemente copiato manualmente in un file con estensione **.hex**, da una riga di comando **bash** come mostrato di seguito:

```
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
```
Ogni file genera:

- **file.hex -** Il file originale (deve contenere solo cifre esadecimali).
- **file.der -** Certificato in formato DER (binario).
- **file.pem -** Certificato in formato PEM (Base64 + intestazione/piè di pagina).
- **file.txt -** Versione del certificato facile da leggere.

Codifica Base64

La codifica Base64 rappresenta i dati binari con solo 64 caratteri stampabili (`A-Za-z0-9+/`) in modo simile a **uencode**. Nella conversione da binary a Base64, ogni blocco a 6 bit dei dati originali viene codificato in un carattere ASCII stampabile a 8 bit con una tabella di conversione. Pertanto, le dimensioni dei dati dopo la codifica sono aumentate del 33% (i dati moltiplicati per 8 diviso per 6 bit, equivalgono a 1,333).

Un buffer a 24 bit viene utilizzato per la conversione di tre (3) gruppi di otto (8) bit in quattro (4) gruppi di sei (6) bit. Pertanto, potrebbero essere necessari uno (1) o due (2) byte di spaziatura interna alla fine del flusso di dati di input. La spaziatura interna è indicata alla fine dei dati con codifica Base64 da un segno di uguale (`=`) per ogni gruppo di otto (8) bit di spaziatura interna aggiunti all'input durante la codifica.

Fare riferimento a questo esempio da Wikipedia.

Fare riferimento alle informazioni più recenti nella RFC 4648: Codifiche dati Base16, Base32 e Base64.

Codifica PEM

Privacy Enhanced Mail (PEM) è uno standard PKI completo di Internet Engineering Task Force (IETF) per lo scambio di messaggi protetti. Non è più ampiamente utilizzato in quanto tale, ma la sua sintassi di incapsulamento è stata ampiamente presa in prestito per formattare e scambiare dati con codifica Base64 relativi alla PKI.

La RFC 1421 PEM, sezione 4.4: Il meccanismo di incapsulamento definisce i messaggi PEM come delimitati dai limiti di incapsulamento (EB), basati sulla RFC 934, nel seguente formato:

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

Base64-encoded data
...
-----END PRIVACY-ENHANCED MESSAGE-----
```
In pratica oggi, quando vengono distribuiti dati in formato PEM, viene utilizzato questo formato limite:

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

type può essere associato ad altre chiavi o certificati, ad esempio:

- `CHIAVE PRIVATA RSA`
- `CHIAVE PRIVATA CRITTOGRAFATA`
- `CERTIFICATO`
- `RICHIESTA CERTIFICATO`
- `CRL X509`

Nota: Sebbene le RFC non rendano obbligatoria questa impostazione, il numero di trattini iniziali e finali (-) negli EB è significativo e dovrebbe sempre essere cinque (5). In caso contrario, alcune applicazioni, ad esempio OpenSSL, interrompono l'input. D'altra parte, altre applicazioni, come Cisco IOS, non richiedono per nulla gli EB.

Per ulteriori informazioni, fare riferimento a queste RFC più recenti:

- RFC 1421: PEM parte I: Procedure di crittografia e autenticazione dei messaggi
- RFC 1422: PEM parte II: Gestione delle chiavi basata su certificati
- RFC 1423: PEM parte III: Algoritmi, modalità e identificatori
- RFC 1424: PEM parte IV: Certificazione chiave e servizi correlati

Certificati X.509 e CRL

X.509 è un sottoinsieme di X.500, una specifica ITU estesa sull'interconnessione dei sistemi aperti. Si occupa specificamente di certificati e chiavi pubbliche ed è stato adattato come standard Internet dall'IETF. X.509 fornisce una struttura e una sintassi, espressa nella RFC con notazione ASN.1, per memorizzare le informazioni sui certificati e gli elenchi di revoche di certificati.

In una PKI X.509, una CA emette un certificato che associa una chiave pubblica, ad esempio: una chiave RSA (Rivest-Shamir-Adleman) o DSA (Digital Signature Algorithm) per un particolare nome distinto (DN) o per un nome alternativo, ad esempio un indirizzo di posta elettronica o un nome di dominio completo (FQDN). Il DN segue la struttura negli standard X.500. Di seguito è riportato un esempio:

```
CN=nome-comune,OU=unità-
organizzativa,O=organizzazione,L=ubicazione,C=paese
```

A causa della definizione ASN.1, i dati X.509 possono essere codificati in DER per essere scambiati in formato binario e, facoltativamente, convertiti in Base64/PEM per mezzi di comunicazione basati su testo, come copia-incolla su un terminale.

- Fare riferimento al documento sugli standard ITU-T X.509 Open Systems Interconnection - The Directory: Framework di certificati di attributo e chiave pubblica.
- Per ulteriori informazioni, fare riferimento alla RFC 5280: Profilo X.509 Certificate and Certificate Revocation List (CRL) per ulteriori informazioni.

Standard PKCS

Gli standard PKCS (Public-Key Cryptography Standards) sono specifiche dei laboratori RSA che si sono in parte evoluti in standard di settore. Le persone incontrate più spesso trattano questi

argomenti; tuttavia, non tutte le soluzioni gestiscono i formati dei dati.

PKCS#1 (RFC 3347**)** - Copre gli aspetti dell'implementazione della crittografia basata su RSA (primitive di crittografia, schemi di crittografia/firma, sintassi ASN.1).

PKCS#5 (RFC 2898**)** - Copre la derivazione delle chiavi basata su password.

PKCS#7 (RFC 2315**)** e **S/MIME** RFC 3852 **-** Definisce una sintassi del messaggio per trasmettere dati firmati e crittografati e certificati correlati. Spesso utilizzato semplicemente come contenitore per i certificati X.509.

PKCS#8: definisce la sintassi di un messaggio per il trasporto di chiavi RSA non crittografate o non crittografate.

PKCS#9 (RFC 2985**)** - Definisce ulteriori classi oggetto e attributi di identità.

PKCS#10 (RFC 2986**)** - Definisce la sintassi di un messaggio per le richieste di firma di certificato (CSR). Un CSR viene inviato da un'entità a una CA e contiene le informazioni che devono essere firmate dalla CA, ad esempio le informazioni sulla chiave pubblica, l'identità e gli attributi aggiuntivi.

PKCS#12: definisce un contenitore per il packaging dei dati PKI correlati (in genere, **coppia di chiavi entità + certificato entità + certificati CA radice e intermedi**) all'interno di un singolo file. Si tratta di un'evoluzione del formato PFX (Personal Information Exchange) di Microsoft.

Fare riferimento alle seguenti risorse:

- Articolo di Wikipedia su PKCS
- Pagina RSA Labs su PKCS

Informazioni correlate

- **Documentazione e supporto tecnico – Cisco Systems**