Formatos de datos PKI

Contenido

Introducción

Este documento describe los formatos de datos y codificaciones más comunes de Infraestructura de clave pública (PKI).

Prerequisites

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

- criptografía de clave pública (conceptos básicos).
- infraestructura de clave pública (conceptos básicos).

Componentes Utilizados

Este documento no tiene restricciones específicas en cuanto a versiones de software y de hardware.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Convenciones

Consulte Convenciones de Consejos Técnicos de Cisco para obtener información sobre las convenciones sobre documentos.

Notación ASN.1

Resumen La Notación de Sintaxis Uno (ASN.1) es un lenguaje formal para la definición de tipos y valores de datos, y cómo se utilizan y combinan esos tipos y valores de datos en diversas estructuras de datos. El objetivo del estándar es definir la sintaxis abstracta de la información sin limitar la forma en que se codifica la información para su transmisión.

Este es un ejemplo extraído del *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 }
```

Consulte estos documentos de los sitios web de estándares de la Unión Internacional de Telecomunicaciones (ITU-T):

- X.680 ASN.1: Especificación de la notación básica
- X.681 ASN.1: Especificación de objeto de información
- X.682 ASN.1: Especificación de restricción
- X.683 ASN.1: Parametrización de las especificaciones de ASN.1

Búsqueda de recomendaciones ITU-T - Buscar **X.509** en **Rec. o estándar** con **Language** establecido en **ASN.1**.

Codificaciones BER/CER/DER

La ITU-T ha definido una forma estándar de codificar estructuras de datos descritas en ASN.1 en datos binarios. X.690 define las reglas básicas de codificación (BER) y sus dos subconjuntos, las reglas canónicas de codificación (CER) y las reglas de codificación distinguidas (DER). Los tres se basan en campos de datos **type-length-value** empaquetados en una estructura jerárquica, que se genera a partir de **SECUENCIAS**, **SET** y **CHOICE**, con estas diferencias:

- BER proporciona varias formas de codificar los mismos datos, lo que no es adecuado para las operaciones de cifrado.
- La RCE proporciona una codificación inequívoca y utiliza datos de longitud indefinida, con un marcador de fin de datos en casos específicos.
- DER proporciona una codificación inequívoca y utiliza etiquetas de longitud explícitas en casos específicos.
- Entre los tres, DER es el que suele encontrarse cuando se trata de PKI y cargas útiles de

cifrado.

Ejemplo: En DER, el valor de 20 bits 1010 1011 1100 1101 11110 se codifica como :

- **etiqueta**: `0x03` (cadena de bits)
- **longitud:** `0x04` (bytes)
- **valor:** 0x`04` `ABCDE` `0`
- **codificación DER completa**: `0x030404ABCDE0`

El `04` inicial significa que los últimos 4 bits (es igual al `0` dígito final) del valor codificado deben descartarse porque el valor codificado no termina en un límite de bytes.

Consulte estos documentos desde el sitio de estándares de TU-T:

- Reglas de codificación ASN.1 X.690: Especificación de reglas básicas de codificación (BER), reglas canónicas de codificación (CER) y reglas de codificación distinguidas (DER)

Desde el sitio de Wikipedia, consulte estos documentos:

- Normas básicas de codificación
- Reglas de codificación canónica
- Reglas de codificación distinguidas

DER hex volcado

Cisco IOS, Adaptive Security Appliance (ASA) y otros dispositivos muestran el contenido DER como un **volcado hexadecimal** con el comando **show running-config**. Aquí se muestra el resultado:

```
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
...
```

Este tipo de volcado hexadecimal se puede volver a convertir en DER de varias maneras. Por ejemplo, puede quitar los caracteres de espacio y canalizarlos al **programa xxd**:

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

Otra forma fácil es usar este guión Perl :

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

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

Además, una forma compacta de convertir **vaciados de cert**, cada uno previamente copiado manualmente a un archivo con extensión **.hex**, desde una línea de comandos **bash** como se muestra aquí:

```
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
```

Cada archivo da como resultado:

- **file.hex -** El archivo original (sólo debe contener dígitos hexadecimales).
- **file.der -** Certificado en formato DER (binario).
- **file.pem -** Certificado en formato PEM (Base64 + encabezado/pie de página).
- **file.txt:** Versión fácil de usar y legible del certificado.

Codificación Base64

La codificación Base64 representa los datos binarios con sólo 64 caracteres imprimibles (`A-Za-z0-9+/`) de forma similar a **uencode**. En la conversión de binario a Base64, cada bloque de 6 bits de los datos originales se codifica en un carácter ASCII imprimible de 8 bits con una tabla de traducción. Por lo tanto, el tamaño de los datos después de la codificación ha aumentado en un 33% (los datos se dividen en 8 por 6 bits, es igual a 1,333).

Se utiliza un búfer de 24 bits para la traducción de tres (3) grupos de ocho (8) bits en cuatro (4) grupos de seis (6) bits. Por lo tanto, al final de la secuencia de datos de entrada podría ser necesario un (1) o dos (2) bytes de relleno. El relleno se indica al final de los datos codificados en Base64, por un signo igual (`=`) para cada grupo de ocho (8) bits de relleno agregados a la entrada durante la codificación.

Refiérase a este ejemplo de Wikipedia.

Consulte la información más reciente en RFC 4648: Las codificaciones de datos Base16, Base32 y Base64.

Codificación PEM

Privacy Enhanced Mail (PEM) es un estándar PKI completo de Internet Engineering Task Force (IETF) para intercambiar mensajes seguros. Ya no se utiliza ampliamente como tal, pero su sintaxis de encapsulación se ha prestado ampliamente para dar formato e intercambiar datos relacionados con PKI codificados en Base64.

PEM RFC 1421, sección 4.4: Mecanismo de encapsulación, define los mensajes PEM como delimitados por Límites de encapsulación (EB), que se basan en RFC 934, con este formato:

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

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

En la práctica, cuando se distribuyen datos con formato PEM, se utiliza este formato de límite:

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

type puede estar con otras claves o certificados como:

- `CLAVE PRIVADA RSA`
- `CLAVE PRIVADA CIFRADA`
- `CERTIFICADO`
- `Solicitud de certificado`
- `X509 CRL`

Nota: Aunque los RFCs no hacen esto obligatorio, el número de guiones iniciales y finales (-) en los EBs es significativo y siempre debe ser cinco (5). De lo contrario, algunas aplicaciones, como OpenSSL, se bloquean en la entrada. Por otra parte, otras aplicaciones, como Cisco IOS, no requieren EB en absoluto.

Consulte estos RFC más recientes para obtener más información:

- RFC 1421: PEM parte I: Encriptación de mensajes y procedimientos de autenticación
- RFC 1422: PEM parte II: Administración de claves basada en certificados
- RFC 1423: PEM parte III: Algoritmos, modos e identificadores
- RFC 1424: PEM parte IV: Certificación clave y servicios relacionados

Certificados X.509 y CRL

X.509 es un subconjunto de X.500, que es una especificación ITU extendida sobre la interconexión de sistemas abiertos. Se ocupa específicamente de los certificados y las claves públicas y ha sido adaptada como estándar de Internet por el IETF. X.509 proporciona una estructura y sintaxis, expresadas en el RFC con Notación ASN.1, para almacenar la información del certificado y las listas de revocación de certificados.

En una PKI X.509, una CA emite un certificado que enlaza una clave pública, por ejemplo: una clave Rivest-Shamir-Adleman (RSA) o Digital Signature Algorithm (DSA) para un nombre distinguido (DN) concreto o para un nombre alternativo, como una dirección de correo electrónico o un nombre de dominio completo (FQDN). El DN sigue la estructura en los estándares X.500. Aquí tiene un ejemplo:

```
CN=nombre común,OU=unidad de
organización,O=organización,L=ubicación,C=país
```

Debido a la definición ASN.1, los datos X.509 se pueden codificar en DER para ser intercambiados en formato binario, y opcionalmente, se pueden convertir en Base64/PEM para medios de comunicación basados en texto, como copiar-pegar en un terminal.

- Consulte este documento de estándares ITU-T X.509 Open Systems Interconnection - The Directory: Marcos de certificado de clave pública y atributo.
- Refiérase a RFC 5280: Perfil de Lista de Certificados y Revocación de Certificados (CRL) X.509 para obtener más información.

Estándares PKCS

Los estándares de criptografía de clave pública (PKCS) son especificaciones de RSA Labs que han evolucionado en parte hacia estándares del sector. Los que se encuentran con más frecuencia se ocupan de estos temas; sin embargo, no todos ellos se ocupan de los formatos de datos.

PKCS#1 (RFC 3347): cubre los aspectos de implementación de la criptografía basada en RSA (primitivas criptográficas, esquemas de cifrado/firma, sintaxis ASN.1).

PKCS#5 (RFC 2898): cubre la derivación de clave basada en contraseña.

PKCS#7 (RFC 2315) y **S/MIME RFC 3852 : define una sintaxis de mensaje para transmitir datos firmados y cifrados y certificados relacionados.** A menudo se utiliza simplemente como contenedor para los certificados X.509.

PKCS#8 - Define una sintaxis de mensaje para transportar texto sin formato o pares RSA cifrados.

PKCS#9 (RFC 2985): define clases de objeto adicionales y atributos de identidad.

PKCS#10 (RFC 2986): define una sintaxis de mensaje para las solicitudes de firma de certificados (CSR). Una entidad envía un CSR a una CA y contiene la información que debe firmar la CA, como información de clave pública, identidad y atributos adicionales.

PKCS#12 - Define un contenedor para los datos de PKI relacionados con el empaquetado (normalmente, **par de claves de entidad + certificado de entidad + certificados raíz y de CA intermedia**) dentro de un solo archivo. Se trata de una evolución del formato de Intercambio de información personal (PFX) de Microsoft.

Consulte estos recursos:

- Artículo de Wikipedia sobre PKCS
- Página de RSA Labs en PKCS

Información Relacionada

- **Soporte Técnico y Documentación - Cisco Systems**