Formatos de dados PKI

Contents

[Introduction](#)
[Prerequisites](#)
[Requirements](#)
[Componentes Utilizados](#)
[Conventions](#)
[Notação ASN.1](#)
[Codificações BER/CER/DER](#)
[Despejo sexual de DER](#)
[Codificação Base64](#)
[Codificação PEM](#)
[Certificados X.509 e CRLs](#)
[PPKCS Padrões](#)
[Informações Relacionadas](#)

Introduction

Este documento descreve os formatos de dados e as codificações mais comuns de infraestrutura de chave pública (PKI).

Prerequisites

Requirements

A Cisco recomenda que você tenha conhecimento destes tópicos:

- criptografia de chave pública (conceitos básicos).
- infraestrutura de chave pública (conceitos básicos).

Componentes Utilizados

Este documento não se restringe a versões de software e hardware específicas.

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.

Conventions

Consulte as Convenções de Dicas Técnicas da Cisco para obter informações sobre convenções de documentos.

Notação ASN.1

A notação de sintaxe abstrata um (ASN.1) é uma linguagem formal para a definição de tipos de dados e valores, e como esses tipos de dados e valores são usados e combinados em várias estruturas de dados. O objetivo do padrão é definir a sintaxe abstrata das informações sem restringir a forma como as informações são codificadas para transmissão.

Aqui está um exemplo extraído do *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 estes documentos dos sites de padrões da União Internacional de Telecomunicações (ITU-T):

- X.680 ASN.1: Especificação da notação básica
- X.681 ASN.1: Especificação de objeto de informações
- X.682 ASN.1: Especificação de restrição
- X.683 ASN.1: parametrização das especificações ASN.1

Pesquisa de recomendações ITU-T - Pesquisa por **X.509** em **Rec. ou padrão** com **Idioma** definido como **ASN.1**.

Codificações BER/CER/DER

A ITU-T definiu uma maneira padrão de codificar estruturas de dados descritas em ASN.1 em dados binários. O X.690 define BER (Basic Encoding Rules, regras básicas de codificação) e seus dois subconjuntos, CER (Canonical Encoding Rules, regras canônicas de codificação) e DER (Distinguished Encoding Rules, regras de codificação distintas). Todos os três são baseados em campos de dados **de tamanho de tipo** compactados em uma estrutura hierárquica, que é criada a partir de **SEQUÊNCIAS**, **SETs** e **CHOICEs**, com estas diferenças:

- O BER oferece várias maneiras de codificar os mesmos dados, o que não é adequado para operações de criptografia.
- A CER fornece codificação inequívoca e utiliza dados de comprimento indefinidos, com um marcador de fim de dados em casos específicos.
- DER fornece codificação inequívoca e usa tags de comprimento explícitas em casos específicos.
- Entre os três, DER é o normalmente encontrado ao lidar com cargas úteis de PKI e

criptografia.

Exemplo: Em DER, o valor de 20 bits 1010 1011 1100 1101 1110 é codificado como:

- **tag**: `0x03` (bitstring)
- **comprimento:** `0x04` (bytes)
- **valor:** `0x04 ABCDE0`
- **codificação DER completa**: `0x030404ABCDE0`

O `04` inicial significa que os últimos 4 bits (igual ao `0` dígito final) do valor codificado devem ser descartados porque o valor codificado não termina em um limite de byte.

Consulte estes documentos do site de padrões TU-T:

- Regras de codificação X.690 ASN.1: Especificação de BER (Basic Encoding Rules), CER (Canonical Encoding Rules) e DER (Distinguished Encoding Rules)

No site da Wikipédia, consulte estes documentos:

- Regras básicas de codificação
- Regras de codificação canônica
- Distinguished Encoding Rules

Despejo sexual de DER

O Cisco IOS, o Adaptive Security Appliance (ASA) e outros dispositivos exibem o conteúdo DER como um **dump hexadecimal** com o comando **show running-config**. Esta é a saída:

```
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
...
```

Esse tipo de despejo hexadecimal pode ser convertido de volta ao DER de várias maneiras. Por exemplo, você pode remover os caracteres de espaço e canalizá-los para o **programa xxd**:

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

Outra maneira fácil é usar este 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
```

Além disso, uma forma compacta de converter **dumps de certificado**, cada um copiado anteriormente manualmente para um arquivo com extensão **.hex**, de uma linha de comando **bash** como mostrado aqui:

```
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 arquivo resulta em:

- **file.hex -** O arquivo original (deve conter apenas dígitos hexadecimais).
- **file.der -** Certificado no formato DER (binário).
- **file.pem -** Certificado no formato PEM (Base64 + cabeçalho/rodapé).
- **file.txt -** Versão legível e fácil de ler do certificado.

Codificação Base64

A codificação Base64 representa os dados binários com apenas 64 caracteres imprimíveis (`A-Za-z0-9+/`), de forma semelhante à **codificação**. Na conversão de binário para Base64, cada bloco de 6 bits dos dados originais é codificado em um caractere ASCII imprimível de 8 bits com uma tabela de conversão. Portanto, o tamanho dos dados após a codificação aumentou 33% (tempos de dados 8 divididos por 6 bits, é igual a 1,333).

Um buffer de 24 bits é usado para a conversão de três (3) grupos de oito (8) bits em quatro (4) grupos de seis (6) bits. Portanto, um (1) ou dois (2) bytes de preenchimento podem ser necessários no final do fluxo de dados de entrada. O preenchimento é indicado no final dos dados codificados em Base64, por um sinal de igual (`=`) para cada grupo de oito (8) bits de enchimento adicionados à entrada durante a codificação.

Consulte este exemplo da Wikipédia.

Consulte as informações mais recentes no RFC 4648: Codificações de dados Base16, Base32 e Base64.

Codificação PEM

O Privacy Enhanced Mail (PEM) é um padrão completo de PKI da Internet Engineering Task Force (IETF) para trocar mensagens seguras. Ela não é mais usada como tal, mas sua sintaxe de encapsulamento foi amplamente emprestada para formatar e trocar dados relacionados a PKI codificados na Base64.

PEM RFC 1421, seção 4.4: Mecanismo de encapsulamento, define as mensagens PEM como delimitadas pelos limites de encapsulamento (EBs), que são baseados no RFC 934, com este formato:

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

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

Na prática atual, quando os dados formatados por PEM são distribuídos, esse formato de limite é usado:

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

o tipo pode estar com outras chaves ou certificados, como:

- `CHAVE PRIVADA RSA`
- `CHAVE PRIVADA CRIPTOGRAFADA`
- `CERTIFICADO`
- `CERTIFICATE REQUEST`
- `CRL X509`

Note: Embora os RFC não tornem isso obrigatório, o número de traços dianteiros e finais (-) nos EBs é significativo e deve ser sempre cinco (5). Caso contrário, alguns aplicativos, como o OpenSSL, ficam bloqueados na entrada. Por outro lado, outros aplicativos, como o Cisco IOS, não exigem EBs.

Consulte os RFCs mais recentes para obter mais informações:

- RFC 1421: PEM Parte I: Procedimentos de criptografia e autenticação de mensagens
- RFC 1422: PEM Parte II: Gerenciamento de chaves baseado em certificado
- RFC 1423: PEM Parte III: Algoritmos, modos e identificadores
- RFC 1424: PEM Parte IV: Certificação principal e serviços relacionados

Certificados X.509 e CRLs

O X.509 é um subconjunto do X.500, que é uma especificação ITU estendida sobre a Interconexão de sistemas abertos. Ele trata especificamente de certificados e chaves públicas e foi adaptado como um padrão da Internet pela IETF. O X.509 fornece uma estrutura e sintaxe, expressas no RFC com notação ASN.1, para armazenar informações de certificado e listas de revogação de certificado.

Em um PKI X.509, uma CA emite um certificado que vincula uma chave pública, por exemplo: uma chave Rivest-Shamir-Adleman (RSA) ou Digital Signature Algorithm (DSA) para um nome distinto (DN) específico, ou para um nome alternativo, como um endereço de e-mail ou nome de domínio totalmente qualificado (FQDN). O DN segue a estrutura nos padrões X.500. Aqui está um exemplo:

```
CN=nome comum,OU=unidade organizacional,O=organização,L=local,C=país
```

Devido à definição ASN.1, os dados X.509 podem ser codificados em DER para serem trocados na forma binária e, opcionalmente, convertidos em Base64/PEM para meios de comunicação baseados em texto, como colagem de cópia em um terminal.

- Consulte este documento de padrões ITU-T X.509 Open Systems Interconnection - The Diretory: Estruturas de certificado de atributos e chaves públicas.
- Consulte o RFC 5280: Perfil de lista de certificados e certificados revogados (CRL) X.509 para obter mais informações.

PPKCS Padrões

Os PKCS (Public-Key Cryptography Standards, Padrões de Criptografia de Chave Pública) são especificações dos RSA Labs que evoluíram parcialmente para padrões do setor. Os mais frequentemente encontrados abordam estes tópicos; no entanto, nem todos lidam com formatos de dados.

PKCS#1 (RFC 3347) - Abrange os aspectos de implementação da criptografia baseada em RSA (primitivos de criptografia, esquemas de criptografia/assinatura, sintaxe ASN.1).

PKCS#5 (RFC 2898) - Abrange derivação de chave baseada em senha.

PKCS#7 (RFC 2315) e **S/MIME RFC 3852 -** Define uma sintaxe de mensagem para transmitir dados assinados e criptografados e certificados relacionados. Frequentemente usado simplesmente como um contêiner para certificados X.509.

PKCS#8 - Define uma sintaxe de mensagem para transportar texto claro ou pares de chaves RSA criptografados.

PKCS#9 (RFC 2985) - Define classes de objetos e atributos de identidade adicionais.

PKCS#10 (RFC 2986) - Define uma sintaxe de mensagem para solicitações de assinatura de certificado (CSRs). Um CSR é enviado por uma entidade a uma CA e contém as informações a serem assinadas pela CA, como informações de chave pública, identidade e atributos adicionais.

PKCS#12 - Define um contêiner para dados de PKI relacionados a embalagens (normalmente, **par de chaves de entidade + certificado de entidade + certificados CA raiz e intermediários**) em um único arquivo. É uma evolução do formato do Intercâmbio de informações pessoais (PFX) da Microsoft.

Consulte estes recursos:

- Artigo da Wikipedia sobre o PKCS
- Página do RSA Labs no PKCS

Informações Relacionadas

- **Suporte Técnico e Documentação - Cisco Systems**