PKI 데이터 형식

목차

소개

이 문서에서는 가장 일반적인 PKI(Public Key Infrastructure) 데이터 형식과 인코딩에 대해 설명합니다.

사전 요구 사항

요구 사항

다음 주제에 대한 지식을 보유하고 있으면 유용합니다.

- 공개 키 암호화(기본 개념)
- 공개 키 인프라(기본 개념)

사용되는 구성 요소

이 문서는 특정 소프트웨어 및 하드웨어 버전으로 한정되지 않습니다.

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다.이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다.현재 네트워크가 작동 중인 경우, 모든 명령어의 잠재적인 영향을 미리 숙지하시기 바랍니다.

표기 규칙

문서 규칙에 대한 자세한 내용은 Cisco 기술 팁 규칙을 참조하십시오.

ASN.1 표기법

추상 구문 표기법 1(ASN.1)은 데이터 유형 및 값의 정의, 이러한 데이터 유형 및 값이 다양한 데이터 구조에서 사용되고 결합되는 방식을 위한 공식 언어입니다.표준의 목표는 정보를 전송하기 위해 인코딩하는 방법을 제한하지 않고 정보의 추상 구문을 정의하는 것입니다.

*X.509 RFC*에서 가져온 예는 다음과 같습니다.

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

ITU-T(International Telecommunication Union) 표준 사이트의 다음 문서를 참조하십시오.

- X.680 ASN.1:기본 표기법 사양
- X.681 ASN.1:정보 객체 사양
- X.682 ASN.1:제약 조건 사양
- X.683 ASN.1:ASN.1 사양 매개 변수화

ITU-T 권장 사항 검색 - Rec에서 **X.509**를 **검색합니다.또는 표준**(언어가 **ASN.1**으로 **설정)**.

BER/CER/DER 인코딩

ITU-T는 ASN.1에 설명된 데이터 구조를 이진 데이터로 인코딩하는 표준 방법을 정의했습니다 .X.690은 BER(Basic Encoding Rules) 및 두 개의 하위 집합인 CER(Canonical Encoding Rules) 및 DER(Distinguished Encoding Rules)를 정의합니다. 이 세 가지 모두 계층적 구조로 포장된 **유형 길이 값** 데이터 필드를 기반으로 합니다. 이 필드는 **SEQUENCE**, **SET** 및 **CHOICE에서** 작성되며, 이러한 차이점이 있습니다.

- BER는 암호화 작업에 적합하지 않은 동일한 데이터를 인코딩하는 여러 가지 방법을 제공합니다.
- CER는 명확한 인코딩을 제공하며 특정 경우 데이터 종료 표시자와 함께 무기한 길이 데이터를 사용합니다.
- DER는 명확한 인코딩을 제공하고 특정 경우에 명시적 길이 태그를 사용합니다.
- 세 가지 중 DER는 PKI 및 암호화 페이로드를 처리할 때 일반적으로 발생하는 것입니다.

예:DER에서 20비트 값 1010 1011 1100 1101 1110은 다음과 같이 인코딩됩니다.

- **태그**:`0x03`(비트 문자열)
- **길이**:`0x04`(바이트)
- **값**:`0x04ABCDE0`

- **전체 DER 인코딩**: `0x030404ABCDE0`

선행 `04`는 인코딩된 값의 마지막 4비트(`0`자리와 같음)가 바이트 경계에서 끝나지 않으므로 삭제되어야 함을 의미합니다.

TU-T 표준 사이트에서 다음 문서를 참조하십시오.

- X.690 ASN.1 인코딩 규칙:기본 인코딩 규칙(BER), CER(Canonical Encoding Rules) 및 DER(Distinguished Encoding Rules) 사양

Wikipedia 사이트에서 다음 문서를 참조하십시오.

- 기본 인코딩 규칙
- 정식 인코딩 규칙
- 고유 인코딩 규칙

DER 16진수 덤프

Cisco IOS, ASA(Adaptive Security Appliance) 및 기타 디바이스에서는 DER 콘텐츠를 **show running-config** 명령을 사용하여 16진수 **덤프로** 표시합니다. 다음은 출력입니다.

```
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
...
```

이러한 종류의 16진수 덤프는 다양한 방법으로 DER로 다시 변환할 수 있습니다.예를 들어 스페이스 문자를 제거하고 이를 xxd **프로그램**에 파이프할 수 있습니다.

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

이 Perl 스크립트를 쉽게 사용할 수 있는 또 다른 방법입니다.

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

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

또한 **인증서 덤프**를 변환하는 간단한 방법으로, 각각 확장자가 **.hex**인 파일에 아래와 같이 **bash** 명령행에서 수동으로 복사했습니다.

```
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
```

각 파일의 결과:

- **file.hex -** 원래 파일입니다(16진수만 포함해야 함).

- **file.der -** DER(이진) 형식의 인증서
- **file.pem -** PEM(Base64 + 헤더/바닥글) 형식의 인증서
- **file.txt -** 사용자가 쉽게 읽을 수 있는 인증서 버전입니다.

Base64 인코딩

Base64 인코딩은 **uuencode**와 유사하게 64개의 인쇄 가능한 문자(`A-Za-z0-9+/`)만 있는 이진 데이터를 **나타냅니다**.바이너리에서 Base64로 변환하는 경우 원본 데이터의 6비트 블록마다 변환 테이블을 사용하여 8비트 인쇄 가능한 ASCII 문자로 인코딩됩니다.따라서 인코딩 후 데이터 크기가 33%(8을 6비트로 나눈 데이터는 1.333과 같음) 증가했습니다.

24비트 버퍼는 8비트(8비트)의 3개 그룹을 6비트(6비트)의 4개 그룹으로 변환하는 데 사용됩니다 .따라서 입력 데이터 스트림의 끝에 패딩이 1바이트 또는 2바이트가 필요할 수 있습니다.Padding은 Base64로 인코딩된 데이터의 끝에 표시되며, 인코딩하는 동안 입력에 추가된 8개의 패딩 비트의 각 그룹에 대해 등호(`=`) 기호로 표시됩니다.

Wikipedia의 이 예를 참조하십시오.

RFC 4648의 최신 정보를 참조하십시오.Base16, Base32 및 Base64 데이터 인코딩입니다.

PEM 인코딩

PEM(Privacy Enhanced Mail)은 보안 메시지를 교환하기 위한 완전한 IETF(Internet Engineering Task Force) PKI 표준입니다.더 이상 널리 사용되지 않지만, Base64로 인코딩된 PKI 관련 데이터를 포맷하고 교환하기 위해 캡슐화 구문이 널리 대여되었습니다.

PEM RFC 1421, 섹션 4.4:Encapsulation Mechanism은 RFC 934를 기반으로 하는 EB(Encapsulation Boundaries)로 구분된 PEM 메시지를 다음 형식으로 정의합니다.

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

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

실제로 오늘날 PEM 형식의 데이터가 배포될 때 이 경계 형식이 사용됩니다.

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

유형은 다음과 같은 다른 키 또는 인증서와 함께 사용할 수 있습니다.

- `RSA`
-
-
-
- `X509 CRL`

참고:RFC는 이 작업을 필수 항목으로 만들지 않지만 EB의 선행 및 후행 대시(-) 수는 중요하며 항

상 5여야 합니다. 그렇지 않으면 OpenSSL과 같은 일부 애플리케이션이 입력에서 차단됩니다.반면 Cisco IOS와 같은 다른 애플리케이션은 EB가 전혀 필요하지 않습니다.

자세한 내용은 다음 최신 RFC를 참조하십시오.

- RFC 1421:PEM Part I:메시지 암호화 및 인증 절차
- RFC 1422:PEM Part II:인증서 기반 키 관리
- RFC 1423:PEM Part III:알고리즘, 모드 및 식별자
- RFC 1424:PEM 4부:주요 인증 및 관련 서비스

X.509 인증서 및 CRL

X.509는 X.500의 하위 집합으로, Open Systems Interconnection에 대한 확장된 ITU 사양입니다 .IETF에서 인증서와 공개 키를 다루며 인터넷 표준으로 채택되었습니다.X.509는 인증서 정보 및 인증서 해지 목록을 저장하기 위해 ASN.1 표기법으로 RFC에 표현된 구조와 구문을 제공합니다.

X.509 PKI에서 CA는 공개 키를 바인딩하는 인증서를 발급합니다. 예를 들면 다음과 같습니다.특정 DN(Distinguished Name)에 대한 RSA(Rivest-Shamir-Adleman) 또는 DSA(Digital Signature Algorithm) 키 또는 이메일 주소 또는 FQDN(Fully Qualified Domain Name)과 같은 대체 이름 DN은 X.500 표준의 구조를 따릅니다.예를 들면 다음과 같습니다.

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

ASN.1 정의 때문에 X.509 데이터는 바이너리 형태로 교환하기 위해 DER로 인코딩될 수 있으며, 선택적으로 터미널에 복사 붙여넣기 등의 텍스트 기반 통신 수단을 위해 Base64/PEM으로 변환할 수 있습니다.

- 이 ITU-T 표준 문서 X.509 Open Systems Interconnection - The Directory:공개 키 및 특성 인증서 프레임워크.
- RFC 5280 참조:자세한 내용은 X.509 인증서 및 CRL(Certificate Revocation List) 프로필을 참조하십시오.

PKCS 표준

PKCS(Public-Key Cryptography Standards)는 업계 표준으로 일부 발전된 RSA Labs의 사양입니다 .가장 자주 접하는 사람들은 이러한 주제를 다루어야 합니다.그러나 모두 데이터 형식을 다루는 것은 아닙니다.

PKCS#1(RFC 3347) - RSA 기반 암호화(암호화 프리미티브, 암호화/서명 체계, ASN.1 구문)의 구현 측면을 다룹니다.

PKCS#5(RFC 2898) - 비밀번호 기반 키 파생에 대해 설명합니다.

PKCS#7(RFC 2315) 및 **S/MIME RFC 3852 -** 서명 및 암호화된 데이터와 관련 인증서를 전송하기 위한 메시지 구문을 정의합니다.X.509 인증서의 컨테이너로 사용되는 경우가 많습니다.

PKCS#8- 일반 텍스트 또는 암호화된 RSA 키 쌍을 전송하기 위한 메시지 구문을 정의합니다.

PKCS#9(RFC 2985) - 추가 개체 클래스와 ID 특성을 정의합니다.

PKCS#10(RFC 2986) - CSR(Certificate Signing Requests)에 대한 메시지 구문을 정의합니다 .CSR은 엔티티에서 CA로 전송되며 공개 키 정보, ID, 추가 특성 등 CA에서 서명하는 정보를 포함 합니다.

PKCS#12 - 단일 파일에서 관련 PKI 데이터(일반적으로 **엔티티 키 쌍** + **엔티티 인증서** + **루트 및 중간 CA 인증서**)를 패키지하기 위한 컨테이너를 정의합니다.Microsoft의 PFX(개인 정보 교환) 형식의 진화입니다.

다음 리소스를 참조하십시오.

- PKCS에 대한 위키백과 문서
- PKCS의 RSA Labs 페이지

관련 정보

- **기술 지원 및 문서 – Cisco Systems**