Contents

Introduction
Background Information
CA Authentication
Request
Response
Client Enrollment
Request
Response
Client Re-enrollment
Renewal
Rollover
Building Blocks
PKCS#7
Signed Envelope (SignedData)
Enveloped Data (EnvelopedData)
PKCS#10
Related Information
Appendix
SCEP Requests
Request Message Format
Schematic View
SCEP Responses
Response Message Format
Content Types
The pkiMessage Structure
SCEP OIDs
SCEP pkiMessage
SCEP messageType
SCEP pkiStatus

Introduction

This document describes the Simple Certificate Enrollment Protocol (SCEP), which is a protocol used for enrollment and other Public Key Infrastructure (PKI) operations.

Background Information

SCEP was originally developed by Cisco, and is documented in an Internet Engineering Task Force (IETF) Draft.

Its main characteristics are:

- Request/response model based on HTTP (GET method; optional support for POST method)
- Only supports RSA-based cryptography
- Uses PKCS#10 as the certificate request format
- Uses PKCS#7 in order to convey cryptographically signed/encrypted messages
- Supports asynchronous granting by the server, with regular polling by the requester
- Has limited Certificate Revocation List (CRL) retrieval support (the preferred method is through a CRL Distribution Point (CDP) query, for scalability reasons)
- Does not support online certificate revocation (must be done offline through other means)
- Requires the use of a **challenge password** field within the Certificate Signing Request (CSR), which must be shared only between the server and the requester

Enrollment and usage of SCEP generally follows this work flow:

1. Obtain a copy of the Certificate Authority (CA) certificate and validate it.
2. Generate a CSR and send it securely to the CA.
3. Poll the SCEP server in order to check whether the certificate was signed.
4. Re-enroll as necessary in order to obtain a new certificate prior to the expiration of the current certificate.
5. Retrieve the CRL as necessary.

CA Authentication

SCEP uses the CA certificate in order to secure the message exchange for the CSR. As a result, it is necessary to obtain a copy of the CA certificate. The **GetCACert** operation is used.

Request

The request is sent as a HTTP GET request. A packet capture for the request looks similar to this:

Response

The response is simply the binary-encoded CA certificate (X.509). The client needs to validate that the CA certificate is trusted through an examination of the fingerprint/hash. This has to be done via an out-of-band method (a phone call to a system administrator or pre-configuration of the fingerprint within the trustpoint).

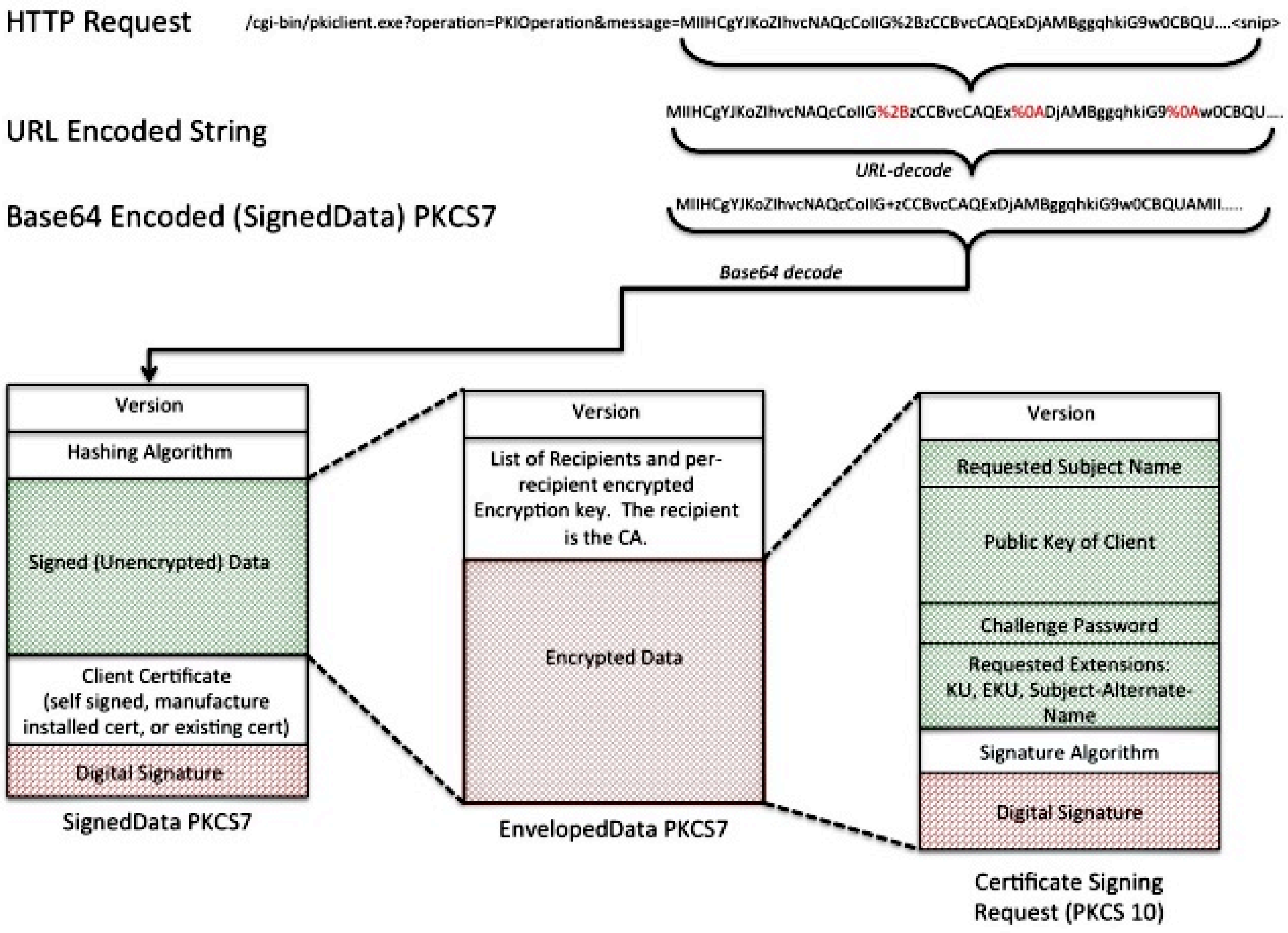
Client Enrollment

Request

The enrollment request is sent as a HTTP GET request. A packet capture for the request looks similar to this:

1. The text after the "message=" is a URL Encoded String, which is extracted from the GET request string.
2. The text is then URL Decoded into an ASCII text string. That text string is a Base64-encoded SignedData PKCS#7.
3. The SignedData PKCS#7 is signed by the client with one of these certificates; it is used to

prove that the client sent it and that it has not been altered in transit:
A self-signed certificate (used upon initial enrollment)A Manufacturer Installed Certificate (MIC)A current certification that expires soon (re-enrollment)
4. The "Signed Data" portion of the SignedData PKCS#7 is an EnvelopedData PKCS#7.
5. The EnvelopedData PKCS#7 is a container that contains "Encrypted Data" and the "decryption key." The Decryption Key is encrypted with the recipient's Public Key. In this specific case, the recipient is the CA; as a result. Only the CA can actually decrypt the "Encrypted Data."
6. The "Encrypted Data" portion of the Enveloped PKCS#7 is the CSR (PKCS#10).



HTTP Request
/cgi-bin/pkiclient.exe?operation=PKIOperation&message=MIIHCgYJKoZIhvcNAQcCoIIG%2BzCCBvcCAQExDjAMBggqhkiG9w0CBQU....<snip>
URL Encoded String
MIIHCgYJKoZIhvcNAQcCoIIG%2BzCCBvcCAQEx%0ADjAMBggqhkiG9%0Aw0CBQU....
URL-decode
Base64 Encoded (SignedData) PKCS7
MIIHCgYJKoZIhvcNAQcCoIIG+zCCBvcCAQExDjAMBggqhkiG9w0CBQUAMII.....
Base64 decode
Version
Hashing Algorithm
Signed (Unencrypted) Data
Client Certificate (self signed, manufacture installed cert, or existing cert)
Digital Signature
SignedData PKCS7
Version
List of Recipients and per-recipient encrypted Encryption key. The recipient is the CA.
Encrypted Data
EnvelopedData PKCS7
Version
Requested Subject Name
Public Key of Client
Challenge Password
Requested Extensions: KU, EKU, Subject-Alternate-Name
Signature Algorithm
Digital Signature
Certificate Signing Request (PKCS 10)

Response

The response to the SCEP enrollment request is one of three types:

- **Reject -** The request is rejected by the administrator for any number of reasons, such as: Invalid key sizeInvalid challenge passwordThe CA could not validate the requestThe request asked for attributes that the CA did not authorizeThe request was signed by an identity that the CA does not trust
- **Pending -** The CA administrator has not reviewed the request yet.
- **Success -** The request is accepted and the signed certificate is included. The signed certificate is held within a special type of PKCS#7 called a "Degenerate Certificates-Only PKCS#7," which is a special container that can hold one or more X.509 or CRLs, but does not contain a signed or encrypted data payload.

Client Re-enrollment

Prior to certificate expiration, the client needs to get a new certificate. There is a slight behavioral difference between renewal and rollover. Renewal happens when the ID certificate of the client approaches expiration, and its expiration date is not the same (earlier than) as the expiration date of the CA certificate. Rollover happens when the ID certificate approaches expiration, and its expiration date is the same as the CA's certificate expiration date.

Renewal

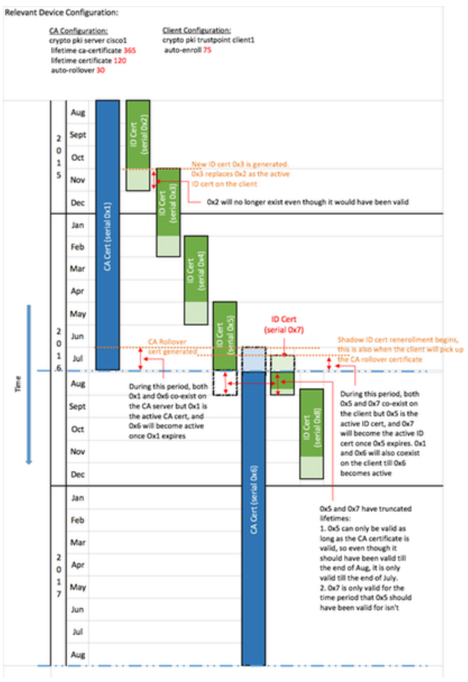
As the expiration date of an ID certificate approaches, a SCEP client might want to obtain a new certificate. The client generates a CSR and goes through the Enrollment process (as defined previously). The current certificate is used in order to sign the SignedData PKCS#7, which in turn proves identity to the CA. Upon reciept of the new certificate, the client immediately deletes the current certificate and replaces it with the new one, whose validity starts immediately.

Rollover

Rollover is a special case where the CA certificate expires and a new CA certificate is generated. The CA generates a new CA certificate which becomes valid once the current CA certificate expires. The CA usually generates this "Shadow CA" certificate some time prior to rollover time, because it is needed in order to generate "Shadow ID" certificates for the clients.

When the SCEP client's ID certificate approaches expiration, the SCEP client queries the CA for the "Shadow CA" Certificate. This is done with the **GetNextCACert** operation as shown here:

Once the SCEP client has the "Shadow CA" certificate, it requests a "Shadow ID" certificate after the normal enrollment procedure. The CA signs the "Shadow ID" certificate with the "Shadow CA" certificate. Unlike a normal renewal request, the "Shadow ID" certificate that is returned becomes valid at the time of CA certificate expiration (rollover). As a result, the client needs to keep a copy of the pre- and post-rollover certificates for both the CA and the ID certificate. At the time of CA expiration (rollover), the SCEP client deletes the current CA certificate and ID certificate and replaces them with the "Shadow" copies.


Relevant Device Configuration:

CA Configuration:
```
crypto pki server cisco1
 lifetime ca-certificate 365
 lifetime certificate 120
 auto-rollover 30
```

Client Configuration:
```
crypto pki trustpoint client1
 auto-enroll 75
```

New ID cert 0x3 is generated. 0x3 replaces 0x2 as the active ID cert on the client

0x2 will no longer exist even though it would have been valid

During this period, both 0x1 and 0x6 co-exist on the CA server but 0x1 is the active CA cert, and 0x6 will become active once 0x1 expires

Shadow ID cert renerollment begins, this is also when the client will pick up the CA rollover certificate

During this period, both 0x5 and 0x7 co-exist on the client but 0x5 is the active ID cert, and 0x7 will become the active ID cert once 0x5 expires. 0x1 and 0x6 will also coexist on the client till 0x6 becomes active

0x5 and 0x7 have truncated lifetimes:
1. 0x5 can only be valid as long as the CA certificate is valid, so even though it should have been valid till the end of Aug, it is only valid till the end of July.
2. 0x7 is only valid for the time period that 0x5 should have been valid for isn't

Building Blocks

This structure is used as the building blocks of SCEP.

> **Note**: PKCS#7 and PKCS#10 are not SCEP-specific.

PKCS#7

PKCS#7 is a defined data format that allows data to be signed or encrypted. The data format includes the original data and the associated metadata necessary in order to perform the cryptographic operation.

Signed Envelope (SignedData)

The signed envelope is a format that carries data and confirms that the encapsulated data is not altered in transit via digital signatures. It includes this information:

- Version number - With SCEP, version 1 used.
- List of Digest Algorithms Used - With SCEP, there is only one Signer and thus only one Hashing Algorithm.
- Actual data that is signed - With SCEP, this is a PKCS#7 Enveloped-data format (Encrypted Envelope).
- List of certificates of the signers - With SCEP, this is a self-signed certificate on initial enrollment or the current certificate if you re-enroll.
- List of the signers and the fingerprint generated by each signer - With SCEP, there is only one signer.

The data encapsulated is not encrypted or obfuscated. This format simply provides protection against the message that is altered.

Enveloped Data (EnvelopedData)

The Enveloped Data format carries data that is encrypted and can only be decrypted by the specified recipient(s). It includes this information:

- Version number - With SCEP, version 0 is used.
- List of each of the recipients and the related encrypted data-encryption key - With SCEP, there is only one recipient (for requests: the CA server; for responses: the client).
- The encrypted data - This is encrypted with a randomly generated key (that has been encrypted with the recipient's public key).

PKCS#10

PKCS#10 describes the format of a CSR. A CSR contains the information that clients request be included within their certificates:

- Subject Name
- A copy of the public key

- A challenge password (optional)
- Any certificate extensions reqested, such as:
  Key Usage (KU)Extended Key Usage (EKU)Subject Alternative Name (SAN)Universal Principal Name (UPN)
- A fingerprint of the request

Here is an example of a CSR:

Related Information

- SCEP IETF Draft
- Legacy SCEP using the CLI Configuration Guide
- Configuring SCEP Support for BYOD

Appendix

SCEP Requests

Request Message Format

Requests are sent with an HTTP GET of the form :

```
GET CGI-path/pkiclient.exe?operation=operation&message=message HTTP/version
```

Where:

- **CGI-path** is dependent on the server and points to the Common Gateway Interface (CGI) program that handles SCEP requests: Cisco IOS® CA uses an empty path string.Microsoft CA uses **/certsrv/mscep/mscep.dll**, which points to the MSCEP/ Network Device Enrollment Service (NDES) IIS service.
- **Operation** identifies the operation that is performed.
- **Message** carries additional data for that operation (and it can be empty if no actual data is required).

With the GET method, the **message** part is either plain text, or Distinguished Encoding Rules (DER)-encoded PKCS#7 converted to Base64. If the POST method is supported, content that would be sent in Base64 encoding with GET might be sent in binary format with POST instead.

Schematic View

Possible values for **operations** and their associated **message** values:

- **operation** = `PKIOperation`: **message**is a SCEP **pkiMessage** structure, based on PKCS#7 and encoded with DER and Base64.the **pkiMessage** structure can be of these types: **PKCSReq**: PKCS#10 CSR**GetCertInitial**: polling for CSR granting status**GetCert** or **GetCRL**: certificate or CRL retrieval
- **operation** = **GetCACert**, **GetNextCACert**, or (optional) **GetCACaps**: **message** can be omitted, or can be set to a name that identifies the CA.

SCEP Responses

Response Message Format

SCEP responses are returned as standard HTTP content, with a **Content-Type** that depends on the original request and the type of data returned. DER content is returned as binary (not in Base64 as for the request). PKCS#7 content might or might not contain encrypted/signed enveloped data; if it does not (only contains a set of certificates), it is referred to as a **degenerate** PKCS#7.

Content Types

Possible values for **Content-Type**:

application/x-pki-message:

- in response to the **PKIOperation** operation, with **pkiMessage** of type: **PKCSReq, GetCertInitial, GetCert or GetCRL**
- response body is a **pkiMessage** of type: **CertRep**

application/x-x509-ca-cert:

- in response to the **GetCACert** operation
- response body is the DER-encoded X.509 CA certificate

application/x-x509-ca-ra-cert:

- in response to the **GetCACert** operation
- response body is a DER-encoded degenerate PKCS#7 that contains the CA and RA certificates

application/x-x509-next-ca-cert:

- in response to the **GetNextCACert** operation
- response body is a variation of a **pkiMessage** of type: **CertRep**

The pkiMessage Structure

SCEP OIDs

```
GET CGI-path/pkiclient.exe?operation=operation&message=message HTTP/version
```

SCEP pkiMessage

- PKCS#7 **SignedData**
- PKCS#7 EnvelopedData (called **pkcsPKIEnvelope**; optional, encrypted to message recipient) **messageData** (CSR, cert, CRL, ...)
- **SignerInfo** with **authenticatedAttributes**: **transactionID**, **messageType**, **senderNoncepkiStatus**, **recipientNonce** (response only)**failInfo** (response + failure only)

SCEP messageType

- request:

  PKCSReq (19): PKCS#10 CSR**GetCertInitial** (20): certificate enrollment polling**GetCert** (21): certificate retrieval**GetCRL** (22): CRL retrieval
- response:
  CertRep (3): response to certificate or CRL request

SCEP pkiStatus

- **SUCCESS** (0): request granted (response in pkcsPKIEnvelope)
- **FAILURE** (2): request rejected (details in failInfo attribute)
- **PENDING** (3): request awaits manual approval