PKIX
Internet-Draft
Intended status: Standards Track
Expires: July 7, 2012

M. Pritikin, Ed.
Cisco Systems, Inc.
P. Yee, Ed.
AKAYLA, Inc.

January 4, 2012

Enrollment over Secure Transport
draft-ietf-pkix-est-00

Abstract

This document profiles certificate enrollment for clients using Certificate Management over CMS (CMC) messages over a secure transport. This profile, called Enrollment over Secure Transport (EST), describes a simple yet functional certificate management protocol targeting simple Public Key Infrastructure clients that need to acquire client certificate(s) and associated Certification Authority (CA) certificate(s).

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 7, 2012.

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

This document specifies a protocol for certificate Enrollment over Secure Transport (EST). EST is a certificate enrollment protocol that operates over HTTPS, and thus should be trivially accessible by most clients. Certificate Management over CMS (CMC) [RFC5272] "Simple PKI Request" and "Simple PKI Response" messages are leveraged. EST is designed to be easily implemented by clients and servers running other common enrollment mechanisms such as the non-standard Simple Certificate Enrollment Protocol (SCEP).

"CMC: Transport Protocols" [RFC5273] provides some guidance for running CMC over HTTP [RFC2616] but notes only that "clients MAY attempt to send HTTP requests using TLS 1.0 [TLS] or later, although servers are not required to support TLS". No attempt is made in that document to specify how the client and server might take advantage of a secured transport to better leverage the Simple PKI messages. This profile specifies secure transport mechanisms and how values from the TLS exchange, the HTTP exchange, and the Simple PKI messages layers are used for authentication (and authorization) purposes by the server. For some simple operations, TLS client authentication is required. When TLS client authentication cannot be leveraged as required by a particular operation, EST also provides a conduit for full CMC operations. It is assumed that reader is familiar with the terms and concepts found in Certificate Management over CMS (CMC) [RFC5272], Certificate Request Message Format (CRMF) [RFC4211], etc. Unlike [RFC5273], EST uses both HTTPS GET and POST to support its functions.

The aspects profiled from HTTPS (HTTP over TLS) and CMC are summarized in Figure 1:

Profiled Layers:

Protocol:

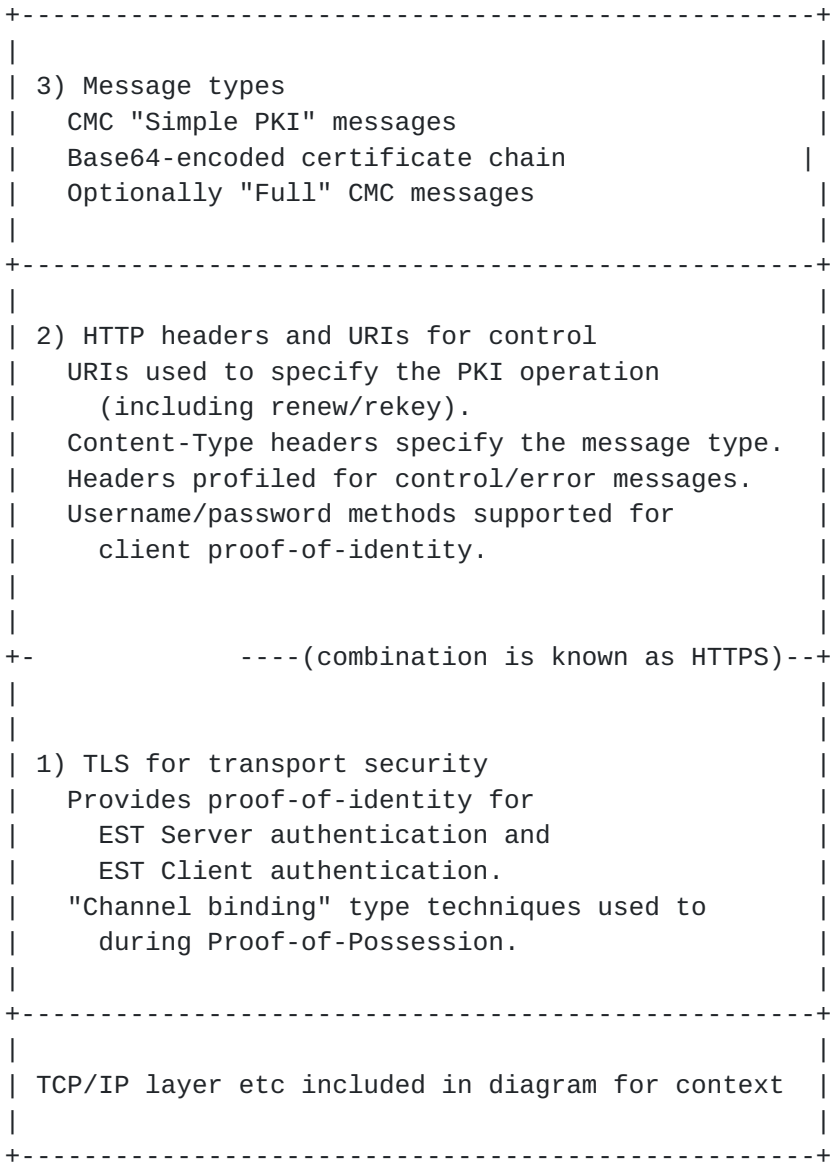

Figure 1

Base64 [RFC4648] is used as specified in Section 4 of that RFC.

The following provides a high level overview describing how these layers are used. Each aspect is profiled in detail in the sections below.

1) TLS for transport security:

CMC section 3.1 notes that "the Simple PKI Request MUST NOT be used if a proof-of-identity needs to be included". This precludes use of these messages if inline authentication and/or authorization is required, unless a secured transport that provides proof-of-identity is also specified. Many simple clients engaged in certificate enrollment operations will have a TLS client implementation available for secure transport, so use of TLS is specified herein. This document specifies a method for authorizing client enrollment requests using existing certificates. Such existing certificates may have been issued by the Certification Authority (CA) (from which the client is requesting a certificate) or they may have been issued under a distinct PKI (e.g., an IEEE 802.1AR IDevID [IDevID] credential). Additionally this document specifies username/password authentication methods beyond those included in CMC. Authentication and authorization mechanisms required for certificate issuance (and renew/rekey) are provided by HTTP and HTTPS mechanisms as described in this document.

Proof-of-possession is a distinct issue from proof-of-identity and is addressed in Section 3.6.

This document also defines transport for the full CMC specification compliant with CMC Transport Protocols.

2) HTTP Headers and URIs for control:

This profile supports two operations indicated by specific URIs:

* Distribution of CA certificates

* Authorized enrollment and re-enrollment of clients

This document profiles the HTTP content-type header to indicate the message type and to provide the protocol control messages. Support for the HTTP username/password methods is profiled.

CMC does not provide explicit messages for renewal and rekey. This profile clarifies the renewal and rekey behavior of both the client and server. It does so by specifying the HTTP control mechanisms required of the client and server without requiring a distinct message type.

Various media types as indicated in the HTTP content-type header

are used to transport EST messages. For simple certificate enrollment and re-enrollment requests, application/pkcs10 (defined in [RFC5967]) is used as specified in Section 5.2. Certificate responses to enrollment and re-enrollment requests are carried as application/pkix-cert (defined in [RFC2585]) as specified in Section 5.2.2. FullCMC requests and responses are both transported as application/pkcs7-mime (as given in [RFC5273]. Requests for CA certificates generate a response with the media type multipart/parallel. Within each parallel part is an entity of media type application/pkix-cert. See Section 5.1.

3) Message Types:

Some message types used here are defined in CMC and include subsets of the PKCS#10 Certification Request [RFC2986] and the PKCS#7 [RFC2315] message specifications.

This document profiles the use of two Certificate Management over CMS messages: "Simple PKI Request" and "Simple PKI Response" and does not require full implementation of all CMC features. This is consistent with the CMC protocol specification of "simple" messages for clients to use "in the event no other services are needed". To support distribution of the CA certificate chain a simple Base 64 format is specified. Full CMC messages MAY also be used.

HTTP Content-Type headers are as specified in [RFC5273], Table 1. This document reuses media types for the simple format messages as specified by Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP [RFC2585] and The application/pkcs10 Media Type [RFC5967]. See the next section for details.

An EST server providing certificate management functions is operated by (or on behalf of) a CA or RA.

An EST server MAY provide additional, non-EST services on other URIs. The server also MAY support full CMC messages over HTTPS.

[[EDNOTE: Comments such as this one, included within double brackets and initiated with an 'EDNOTE', are for editorial use and shall be removed as the document is polished.]]

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this

document are to be interpreted as described in [RFC2119].

2. Requirements

[[EDNOTE: The following section is still included here for succinctness. It will eventually be published independently as draft-ietf-pkix-estr-00.]]

The following describes goals and technical requirements for initial PKI certificate enrollment along with a rationale for each requirement.

G1 "Completeness". Server and client implementations compliant with this document MUST be able to interoperate without reference to subsequent profiles or additional future specifications.

The goal of this enrollment protocol is to provide a simple and easy-to-implement method for end-entities to request, obtain, and update a certificate issued from a specified Certification Authority. The following certificate management operations are required. Additional operations NEED NOT be supported (via this protocol) although the protocol design SHOULD be extensible:

M1 "Distribution of current CA certificates". Clients MUST be able to obtain the current certificate for the CA under which the client's certificate was issued. Certificates have a finite lifetime and will need to be updated on a periodic basis. It must be possible for the client to obtain the updated CA certificates.

M2 "Enrollment". A client MUST be able to use the protocol to submit a certificate request and obtain a certificate.

M3 "Renew/Rekey". Certificates have a finite lifetime and will need to be updated on a periodic basis. A client MUST be able to use the protocol for certificate renewal or rekey operations.

End-Entity Proof of Identity authentication mechanisms:

A1 "Username/Password". It MUST be possible to identify a username specified client as being in possession of an associated symmetric key. This allows users currently in possession of a username and password to obtain a certificate.

A2 "Password". It MUST be possible to identify a client without reference to a "username". A common operational model is to distribute a "one-time password" that is presented to a CA or RA to authorize enrollment.

A3 "Existing Certificate". It MUST be possible to authenticate a client by making use of an existing certificate associated with the client. A certificate used for client identification need not be issued under the same PKI as the certificate that is being requested. This allows clients that are already in a PKI to use a certificate from that PKI to obtain additional certificates. Additionally this capability allows a client who has a certificate issued by a 3rd party, such as a certificate issued by a device manufacturer, to leverage that credential during initial enrollment.

A4 "Username/password and Certificate". It MUST be possible to authenticate a client by using a combination of a username and password and an existing certificate. This is a combination of A1 and A3. This supports "two-factor authentication" where the client proves possession of the private keys for an existing certificate stored within a hardware device and knowledge of a username/password.

A5 "Password and certificate". It MUST be possible to authenticate a client by using a combination of a secret value that is not associated with a "username" and an existing certificate. This is a combination of A2 and A3. This requirement is similar to A4 except that the client is in possession of a "one-time password".

End-Entity Proof of Possession:

P1 Proof-of-Possession of subject keys MUST be supported. As discussed in Appendix C of [RFC4211], Proof-of-Possession "means that the CA is adequately convinced that the entity requesting a certificate for the public key Y, has access to the corresponding private key X".

Key algorithms:

K1 "Algorithm agility". The protocol MUST support algorithm agility. It must be possible to employ different cryptographic algorithms for securing the transport or for signing the certificates. The protocol SHOULD demonstrate this agility by being shown to work with existing RSA-based solutions as well as providing for other algorithms such as Elliptic Curve cryptography.

Server Identity mechanism:

I1 "RA certificate". It MUST be possible for a client to verify authorization of the EST server as a representative of the CA. The protocol operations allow the client to send a username/password or (one-time) password to the EST server. These values cannot be safely transmitted to an unauthorized server.

3. Secure Transport

HTTPS MUST be used. TLS 'session resumption' SHOULD be supported.

HTTPS is defined in HTTP Over TLS [RFC2818] and is a definition of how HTTP messages are carried over TLS. HTTPS is a commonly used secure transport and can be easily layered on top of extremely simple client or server code. In some environments HTTPS can be utilized through an external process. Specifying HTTPS as the secured transport for PKI enrollment messages introduces two potential 'layers' for communication of authorization data or for status/informative responses during the protocol exchange: TLS and HTTPS. This profile specifies when information is used from each layer.

3.1. TLS-Based Server Authentication

The client MUST validate the TLS server certificate presented during the TLS [RFC5246] exchange-defined Server Certificate message or the client MUST independently validate the response contents. Validation is performed as given in [RFC5280] and [RFC6125]. The cipher suites are detailed in Section 6.

There are multiple methods of validation depending on the current state of the client:

1. If the client has a store of trust anchors, which may be in the form of certificates, for validating TLS connections the client MAY validate the TLS server certificate using the standard HTTPS logic of checking the server's identity as presented in the server's Certificate message against the URI provisioned for the EST server (see HTTP Over TLS, Section 3.1 Server Identity and [RFC6125]). This method makes it possible for clients with a store of trust anchors, possibly in the the form of certificates, to securely obtain the CA certificate by leveraging the HTTPS security model. The EST server URI MUST be made available to the client in a secure fashion so that the client only obtains EST functions from a desired server.

2. If the client already has one or more trust anchors associated with this EST server, the client MUST validate the EST server certificate using these trust anchors. The EST server URI MAY be

made available to the client in an insecure fashion. The EST server certificate MUST contain the id-kp-cmcRA [CMC RFC5272bis] extended key usage extension.

3. If the client does not yet have a trust anchor associated with this EST server then the client MAY provisionally accept the TLS connection, but the HTTP content data MUST be accepted manually as described in Section 5.1. HTTP authentication requests MUST NOT be responded to.

Methods 1 and 2 are essentially validation as given in [RFC5280] with the addition of authorization. Method 1 is as described in [RFC6125] Section 6.6.1 "Match Found". Method 2 is described in [RFC6125] as "No Match Found, Pinned Certificate". Method 3 is described in [RFC6125] as "Fallback" and describes the process of "pinning" the recieved certificate.

If one of these validation methods succeeds the CA certificates are stored and made available for future use. If none of these validation methods succeeds the client MUST reject the EST server response and SHOULD log and/or inform the end user.

The EST server MUST present an end-entity certificate such as a CMC Registration Authority (RA) certificate.

3.2. Server Authentication and Authorization

The client MUST check the EST server authorization before accepting the server's response.

If the client has a securely configured and authorized URI for the EST server it MUST check the URI "against the server's identity as presented in the server's Certificate message" (Section 3.1 Server Identity [RFC2818] and [RFC6125]). The securely configured URI provides the authorization statement and the server's authenticated identity confirms it is the authorized server.

If this check fails, or if the URI was configured using an insecure method, then the client MUST verify the server's authorization by checking that the [RFC5280] defined certificate policy extension sequence contains the 'RA Authorization' policy OID.

The RA Authorization policy OID is defined as: id-cmc [[EDNOTE: TBD, perhaps 35]]. The RA Authorization policy information MUST NOT contain any optional qualifiers.

3.3. TLS-Based Client Authentication

Clients MUST support client-side certificate authentication [RFC5246]. To authenticate the client, the server sends the certificate request message to the client, the client returns a client certificate and a certificate verify message to the server, and the server verifies the certificate verify message with the certificate in the client certificate message. As required by [RFC5246], the client certificate needs to indicate support for digital signatures.

The certificate presented by the client MAY be from the same PKI as the Server Certificate, from a completely different PKI, or as a last resort the client MAY respond with a self-signed certificate. The certificate supplied during authentication is used during client authorization (Section 3.5).

3.4. HTTP-Based Client Authentication

As specified in CMC: Transport Protocols [RFC5273] the server "MUST NOT assume client support for any type of HTTP authentication such as cookies, Basic authentication, or Digest authentication". Clients intended for deployments where password authentication is advantageous SHOULD support the Basic and Digest authentication mechanism. Servers MAY provide configuration mechanisms for administrators to enable Basic [RFC2616] and Digest [RFC2617] authentication methods. Basic and Digest authentication MUST be performed over TLS [RFC5246].

Servers that support Basic and Digest authentication methods MAY reject requests using the HTTP defined WWW-Authenticate response-header (Section 14.47). At that point the client SHOULD repeat the request, including the appropriate HTTP [RFC2617] Authorization Request Header (Section 3.2.2).

Support for Basic authentication as specified in HTTP allows the server access to the client's cleartext password. This provides integration with legacy username/password databases but requires exposing the plaintext password to the EST server. The client MUST NOT respond to this request unless the client has authenticated the EST server (as per Section 3.2).

Clients MAY set the username to the empty string ("") if they wish to present a "one-time password" or "PIN" that is not associated with a username.

3.5. Client Authorization

When the EST server receives a CMC Simple PKI Request or rekey/renew message, the decision to issue a certificates is always a matter of local policy. Thus the CA can use any data it wishes in making that determination. The EST protocol exchange provides the EST server access to the TLS client certificate in addition to any HTTP user authentication credentials to help in that determination. The communication channel between the TLS server implementation and the EST software implementation is out-of-scope of this document.

3.6. Proof-of-Possession

As discussed in Appendix C of CRMF [RFC4211], Proof-of-Possession "means that the CA is adequately convinced that the entity requesting a certificate for the public key Y, has access to the corresponding private key X".

The signed enrollment request provides a "Signature"-based proof-of-posession. The mechanism described in Section 3.7 strengthens this by optionally including identity linking information within the data covered by the enrollment request signature (thus ensuring that the enrollment request was signed after the TLS tunnel was established).

3.7. Linking Identity and POP information

This specification provides a method of linking identity and proof-of-possession by including information specific to the current authenticated TLS session within the signed certification request. This proves to the server that the authenticated TLS client has possession of the private key associated with the certification request and that the client was able to sign the certification request after the TLS session was established. This is an alternative to the [RFC5272] Section 6.3-defined "Linking Identity and POP information" method available if fullCMC messages are used.

The client generating the request SHOULD obtain the tls-unique value as defined in Channel Bindings for TLS [RFC5929] from the TLS subsystem, encode it using base64 encoding, and place the resulting string in the certification request challenge password field.

The tls-unique specification includes a synchronization issue as described in Channel Bindings for TLS [RFC5929] section 3.1. This problem is avoided for EST implementations. The tls-unique value used MUST be from the first TLS handshake. EST client and servers must use their tls-unique implementation specific synchronization methods to obtain this first tls-unique value.

Any TLS renegotiation MUST use "secure_renegotiation" [RFC5746] (thus maintaining the binding). Mandating secure renegotiation secures this method of avoiding the synchronization issues encountered when using the most recent tls-unique value (which is defined as the the value from the most recent TLS handshake).

The server SHOULD verify the tls-unique information. This ensures that the authenticated TLS client is in possession of the private key used to sign the certification request.

The tls-unique value is encoded into the certification request by the client but back-end infrastructure elements that process the request after the EST server might not have access to the initial TLS session. Such infrastructure elements validate the source of the certification request to determine if POP checks have already been performed. For example if the EST client authentication results in an authenticated client identity of an RA that is known to independently verify the proof-of-possession then the back-end infrastructure does not need to perform proof-of-possession checks a second time. If the EST server forwards a request to a back-end process it SHOULD communicate the authentication results. This communication might use the CMC "RA POP Witness Control" in a CMC Full PKI Request message or other mechanisms which are out-of-scope of this document.

4. HTTP URIs

EST uses the HTTPS "GET" and "POST" messages to communicate with the EST server. The following describes the syntax of these messages:
"GET" BASEPATH OPERATIONPATH
"POST" BASEPATH OPERATIONPATH

where:

- BASEPATH is a common path for all EST operations

- OPERATIONPATH specifies the specific operation.

When a URI is formed, the BASEPATH and OPERATIONPATH are combined to form the abs_path [RFC2616]. The means by which clients acquire the BASEPATH URI are outside the scope of this document. The following are two example base URIs:

- With BASEPATH having the value of /arbitrary/base/path

- https://example.org/arbitrary/base/path

- https://example2.org:8080/arbitrary/base/path

These can be conveniently distributed as they are in a form with which many end users are already familiar. The following operation paths for clients to access are defined relative to the EST base URL:

- /CACerts - The server responds to an HTTPS GET with the CA certificates as defined in Distribution of CA certificates (Section 5.1).

- /simpleEnroll - The client sends a CMC Simple PKI Request message as specified in Enrollment of Clients (Section 5.2), the response is a CMC Simple PKI Response message as specified in Enroll Response (Section 5.2.2).

- /simpleReEnroll - Exactly the same as 'simpleEnroll' except that the request is for re-enrollment or re-issuance purposes. Only certificates that are suitable for TLS client authentication can be re-enrolled using this operation because of the reliance on the TLS authentication. For other types of certificates, use of the fullCMC operation is required.

- /fullCMC - Provides for Full CMC messages (OPTIONAL).

The following examples are valid complete URIs under this specification:

- With BASEPATH having the value /base/path

- https://example.org/base/path/CACerts

- https://example2.org:8080/base/path/simpleEnroll

- https://example3.org/base/path/fullCMC

The mechanisms by which the EST server interacts with an HTTPS server to handle GET and POST operations at these URIs is outside the scope of this document. The use of distinct URIs simplifies implementation for servers that do not perform client authentication when distributing "CACerts" responses.

Implementation note: A simple Common Gateway Interface (CGI) application at each fully specified path, with the HTTPS server configured to provide Section 3.3, is sufficient for a working example (the web service can forward the Section 3.6 proof-of-possession information to the application using the CGI interface). Additional dicussion regarding the use of CGI can be found in Appendix C.

[[EDNOTE: This does not use the mechanism specified in "Defining Well-Known Uniform Resource Identifiers (URIs)" [RFC5785]. That would be a possibility here for a base URL of "https://example.org/.well-known/EST" but such would preclude the flexibility associated with multiple base URLs being handled by the same server unless some form of "?designator=value" is included.]]

5. Messages

5.1. Distribution of CA certificates

Before engaging in enrollment operations, clients MUST request trust anchor information (in the form of certificates) by sending an HTTPS GET message to the EST base URI with the relative path extension '/CACerts'. Clients SHOULD request an up-to-date response before stored information has expired.

The EST server SHOULD NOT require client authentication or authorization to reply to this request.

The client MUST authenticate the EST server as specified in Authentication and Authorization (Section 3). If the authentication and authorization is successful, the client accepts the response and stores it. If the authentication and authorization is not successful, then when the response is received the client MUST extract the CA certificate and engage the end-user or otherwise authorize the credential using out-of-band pre-configuration data such as a CA certificate "fingerprint" (e.g., a SHA-1, SHA-256, SHA-512, or MD5 hash on the whole CA certificate).

The client MUST NOT accept the CA certificate without validating it via one of the mechanisms described in Section 3.1.

Subsequent connections to the EST server validate the TLS server certificate using the stored CA certificates as described in Authentication and Authorization (Section 3).

5.1.1. Distribution of CA certificates response

The EST server MUST respond to the client HTTPS GET message with CA trust anchor information in the form of a certificate. Additionally the server MUST include any "Root CA Key Update" CMP certificates.

The response format is the media type multipart/parallel. Within each parallel part is an entity of media type application/pkix-cert. One part MUST be the the current self-signed CA certificate. Additional parts MAY be included. If additional parts are included

they MUST be the three additional CMP-defined Root CA Key Update certificates: OldWithOld, OldWithNew, and NewWithOld.

The client can always find the current self-signed CA certificate by examining the certificates received. The NewWithNew certificate is self-signed and has the latest NotAfter date.

The NewWithNew certificate is the certificate that is extracted and authorized using out-of-band information as described in Section 5.1. When out-of-band validation occurs each of the other three certificates MUST be validated using normal [RFC5280] certificate path validation (using the NewWithNew certificate as the trust anchor) before they can be used to build certificate paths during peer certificate validation.

5.2. Simple Enrollment of Clients

At any time the client MAY request a certificate from the EST base URI with the OPERATIONPATH "/simpleEnroll'.

When HTTPS POSTing to the 'SimpleEnroll' location the client MUST include a CMC Simple PKI Request as specified in CMC Section 3.1 (i.e., a PKCS#10 Certification Request).

The content-type of "application/pkcs10" MUST be specified. The format of the request is as specified in Section 6.4 of [RFC4945].

The server MUST authenticate the client as specified in Authentication and Authorization (Section 3). The server applies whatever authorization or policy logic it chooses determining if the certificate should be issued. The client MAY request an additional certificate even when using an existing certificate in the TLS client authentication.

The client MUST authenticate the EST server as specified in Section 3.1.

5.2.1. Simple Re-Enrollment of Clients

At any time a client MAY request renew/rekey of its certificate from the EST base URI with the OPERATIONPATH "/simpleReEnroll'. The certificate request is the same format as for the "simpleEnroll" path extension with the same content-type.

The EST server MUST handle enrollment requests submitted to the "simpleReEnroll" URI as renewal or rekey requests rather than depending only on the method of identifying a renewal or rekey request specified in Section 2 of [RFC5272], that "renewal and rekey

requests look the same as any certification request, except that the identity proof is supplied by existing certificates from a trusted CA". The proof of client identity is supplied by client authentication during the HTTPS handshake. When attempting to renew or rekey the client MUST use its existing certificate for TLS client authentication.

[[EDNOTE: draft-turner-suiteb-cmc defines a method of recognizing a re-enroll based on PKCS10 contents, see section 4.1. The method described herein is explicit.]]

5.2.2. Simple Enroll and Re-Enroll Response

The server responds to a 'simpleEnroll' or 'simpleReEnroll' request with the client's newly issued certificate or it provides an error response.

If the enrollment is successful the server response MUST have a response code of 200 with a content-type of "application/pkix-cert". The response data is the certificate formatted as specified in Section 6.1 of [RFC4945].

When rejecting a request the server MUST specify either an HTTP 4xx/ 401 error, or an HTTP 5xx error. A CMC Simple PKI Response with an HTTP content-type of "application/pkcs7-mime" MAY be included in the response data for any error response. If the content-type is not set the response data MUST be a plain text human-readable error message. A client MAY elect not to parse a CMC error response in favor of a generic error message.

If the server responds with an HTTP 202 this indicates that the request has been accepted for processing but that a response is not yet available. The server MUST include a Retry-After header as defined for 503 responses and MAY include informative human-readable content. The client MUST wait at least the specified 'retry-after' time before repeating the same request. The client repeats the initial enrollment request after the appropriate 'retry-after' interval has expired. The client SHOULD log or inform the end user of this event. The server is responsible for maintaining all state necessary to recognize and handle retry operations as the client is stateless in this regard (it simply sends the same request repeatedly until it receives a different response code).

All other return codes are handled as specified in HTTP.

5.3. Full CMC

At any time the client MAY request a certificate from the EST base URI with the OPERATIONPATH "/fullCMC".

The client MUST authenticate the server as specified in Server Authentication (Section 3.1).

The server SHOULD authenticate the client as specified in Authentication and Authorization (Section 3). The server MAY depend on CMC client authentication methods instead.

5.3.1. Full CMC Request

When HTTPS POSTing to the "fullCMC" location the client MUST include a valid CMC message. The content-type MUST be set to "application/pkcs7-mime" as specified in [RFC5273].

5.3.2. Full CMC Response

The server responds with the client's newly issued certificate or provides an error response.

If the enrollment is successful the server response MUST have a response code of 200 with a content-type of "application/pkcs7-mime" as specified in [RFC5273]. The response data includes either the CMC Simple PKI Response or the CMC Full PKI Response.

When rejecting a request the server MAY specify either an HTTP 4xx/401 error, an HTTP 5xx error, or a response code 200. A CMC response with content-type of "application/pkcs7-mime" MUST be included in the response data for any error response. The client MUST parse the CMC response to determine the current status.

All other return codes are handled as specified in Section 5.2.2 or HTTP [RFC2616].

6. Cryptographic Algorithms

This section details the specific cryptographic algorithms and cipher suite requirements.

The client SHOULD offer the Suite B compliant cipher suites as indicated in [RFC5430], Section 4 "Suite B Compliance and Interoperability Requirements". For greatest interoperability the client SHOULD also offer TLS_RSA_WITH_AES_128_CBC_SHA.

When the client accesses the "simpleReEnroll" method the TLS cipher suite in use MUST be appropriate for the existing certificate. The certificate type used determines the appropriate signatureAlgorithm for the PKCS#10 Certification Request. For example if a [RFC5430] cipher suite is used the signatureAlgorithm MAY be ecdsa-with-sha256 for P-256 certification requests, or MAY be ecdsa-with-sha384 for P-384 certification requests.

[[EDNOTE: This is in alignment with draft-turner-suitb-cmc-03 section 4.1. To encourage algorithm agility, discussions of the MUST/SHOULD algorithms should be in a distinct document.]]

7. Contributors/Acknowledgements

The editors would like to thank Stephen Kent, Vinod Arjun, Jan Vilhuber, Sean Turner, and others for their feedback and prototypes of early drafts.

8. IANA Considerations

(This section is incomplete)

The following aspects should be registered with IANA Considerations:

The RA Authorization certificate policy extension OID as discussed in Section 3.2 requires registration with IANA.

[[EDNOTE: The URLs specified in Section 1 probably do not need to be registered with IANA.]]

9. Security Considerations

(This section is incomplete)

"Badges? We ain't got no badges. We don't need no badges! I don't have to show you any stinkin' badges!" -- The Treasure of the Sierra Madre.

As described in CMC Section 6.7, "For keys that can be used as signature keys, signing the certification request with the private key serves as a POP on that key pair". The inclusion of tls-unique within the certification request provides timeliness to the proof-of-possession. For support of keys that can not be used for signing the certification request the full CMC specification MUST be used.

As described in Section 3.3 clients use an existing certificate for TLS client authentication. If a certificate with appropriate key usage is not available the client MAY generate one. If a self-signed certificate with appropriate key usage is used the server SHOULD require HTTP-based client authentication according to server policy as described in Section 3.3 and Section 3.5. The server MAY fall back on manual authorization by the server administrator.

As described in Section 3.1 servers use an existing certificate for TLS server authentication. When the server certificate is issued by a mutually trusted PKI hierarchy validation proceeds as specified in Section 3.2. In this situation the client has validated the server as being a valid responder for the URI configured but can not directly verify that the responder is authorized as an RA within the to-be-enrolled PKI hierarchy. A client may thus be enticed to expose username/password or certificate enrollment requests to an unauthorized server (if the server presents a valid HTTPS certificate for an erroneous URL that the client has been tricked into using). Proof-of-Identity and Proof-of-Possession checks by the CA prevent an illegitimate RA from leveraging such misconfigured clients to act as a man-in-the-middle during client authenticated operations but it is possible for such illegitimate RAs to send the client doctored messages or erroneous CA certificate lists. If the illegitimate RA has successfully phished a username/password or PIN from the client it might try to use these values to enroll its own keypair with the real PKI hierarchy. EST servers identified with an externally issued server certificate SHOULD require HTTPS-based client authentication (Section 3.3). Similarly EST clients SHOULD use an existing client certificate to identify themselves and otherwise prevent "private data" (obviously including passwords but also including private identity information) from being exposed during the enrollment exchange a weak server authorization method is used.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2315] Kaliski, B., "PKCS #7: Cryptographic Message Syntax Version 1.5", RFC 2315, March 1998.

[RFC2585] Housley, R. and P. Hoffman, "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP", RFC 2585, May 1999.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, November 2000.

[RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, September 2005.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

[RFC4945] Korver, B., "The Internet IP Security PKI Profile of IKEv1/ISAKMP, IKEv2, and PKIX", RFC 4945, August 2007.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, June 2008.

[RFC5273] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC): Transport Protocols", RFC 5273, June 2008.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5430] Salter, M., Rescorla, E., and R. Housley, "Suite B Profile for Transport Layer Security (TLS)", RFC 5430, March 2009.

[RFC5746] Rescorla, E., Ray, M., Dispensa, S., and N. Oskov, "Transport Layer Security (TLS) Renegotiation Indication Extension", RFC 5746, February 2010.

[RFC5929] Altman, J., Williams, N., and L. Zhu, "Channel Bindings for TLS", RFC 5929, July 2010.

[RFC5967] Turner, S., "The application/pkcs10 Media Type", RFC 5967, August 2010.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, March 2011.

10.2. Informative References

[IDevID] IEEE Std, "IEEE 802.1AR Secure Device Identifier", December 2009, <http://standards.ieee.org/findstds/standard/802.1AR-2009.html>.

[RFC4211] Schaad, J., "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, September 2005.

Appendix A. Server Discovery

(informative)

(This section is incomplete)

Clients MAY use DNS-SD or similar discovery algorithms to determine the EST base URL. In such cases it is expected that method 2 (Section 3.1) be used during server authentication.

Appendix B. External TLS concentrator

(informative)

In some deployments it may be beneficial to use a TLS concentrator to offload the TLS processing from the server. In such a deployment the TLS client authentication result must, in some way, be forwarded to the server.

The TLS server SHOULD NOT reject the connection based on PKIX validation of the client certificate. The client certificate SHOULD be passed to the EST layer for verification and authorization. This allows support of external TLS concentrators, or an external web server, that might provide an independent TLS implementation.

The TLS concentrator MUST validate the TLS Section 7.4.8 'Certificate Verify'.

A TLS concentrator MUST insert the client certificate into the HTTP header. The TLS concentrator MUST first remove any existing client certificates, possibly inserted by a nefarious client, from the HTTP headers before forwarding the HTTP connection to the server.

[TBD - need to better understand what would happen in the case of proxy's or multiple concentrators. Or specifically state that as out of scope.]

[TBD - the HTTP header field names etc shall be specified here]

The EST server MUST be specifically configured by the administrator to accept this mechanism.

Appendix C. CGI Server implementation

(informative)

In some deployments it may be beneficial to use a HTTPS server that runs the EST server as a CGI application. In such a deployment the HTTPS server client authentication result must, in some way, be forwarded to the server.

An HTTPS server MUST insert the client certificate into environment variables before calling a server CGI application.

[TBD - describe the CGI environment variables here. Can likely follow the apache example].

An HTTP server MUST insert the client certificate into environment variables before calling a server CGI application.

[TBD - describe the CGI environment variables here. Can likely follow the apache example].

Appendix D. Updating SCEP implementations

(informative)

SCEP has been used instead of a full implementation of CMC for the same simplicity reasons discussed in Section 1. Such implementations would benefit from being updated to this specification in the following ways:

- Implementing a subset of CMC provides an enhancement path if the full CMC functionality is required.

- The use of HTTPS as a transport is often perceived as more secure. Although the SCEP protocol specification includes mechanisms (and complexity) to address security issues avoiding a vendor requirement to educate systems administrators is beneficial. Implementors can benefit from the wide availability of existing HTTPS/TLS libraries.

- SCEP servers can use their CA certificate to protect SCEP traffic in ways that are not appropriate. (See SCEP draft Section 8.2). This specification precludes those misuses.

- The SCEP draft Appendix D renew and rekey functionalities imply a 'flag moment' where the PKI infrastructure transitions from an (expired) CA certificate to a new CA certificate. This specification specifies the better mechanism defined in CMP.

Updating an SCEP client implementation to support this protocol involves the following changes to the SCEP implementation. There is no server-side indication that SCEP clients should be so modified so this depends on a client-side configuration:

- The SCEP client supports HTTPS server authentication and authorization as detailed Section 3.1.

- The SCEP client supports HTTPS client authentication as detailed in Section 3.3.

- When performing the "Get CA Cert" SCEP transaction the client supports the Section 5.1 described CMC Simple PKI Response (ref CMC 4.1, which is extremely similar to the SCEP "CA/RA Certificate Response Message Format" if not exactly the same).

- When performing the certificate enrollment via SCEP PKCSReq the outgoing message is simplified to be only the inner PKCS10 (ref CMC section 3.2.1.2.1).

- When handling the certificate enrollment response the response format is simplified to be only the SCEP inner 'messageData' containing the actual certificates in the degenerate PKCS7 form. (ref CMC 4.1) The only 'authenticatedAttributes' value of remaining importance is the 'pkiStatus' and this value is now found in the HTTP header as defined in Section 5.2.2.

- Polling is simplified with clients repeatedly establishing the full HTTPS connection; no polling specific state information is encoded into the EST messages.

- GetCert is deprecated.

- GetCRL is deprecated.

These simplifications to an existing SCEP implementation result in an SCEP client that is compliant with CMC when using the EST transport.

Implementation note: The use of tls-unique-securerenegotiation precludes the use of SCEP 'challenge-password' within the pkcs10 for password/PIN assertion. Instead these values must be asserted with the Section 3.4 described mechanism. A side effect of this is that a client communicating with an EST server can not embed an SCEP 'challenge-password' within the PKCS#10. An EST service running as an RA thus can not forward the PKCS#10 using SCEP to an SCEP server that expects the 'challenge-password' to be populated.

Authors' Addresses

Max Pritikin (editor)
Cisco Systems, Inc.
510 McCarthy Drive
Milpitas, CA 95035
USA

Email: pritikin@cisco.com

Peter E. Yee (editor)
AKAYLA, Inc.
7150 Moorland Drive
Clarksville, MD 21029
USA

Email: peter@akayla.com