Network Working Group
Internet Draft

Bellovin and Moskowitz
AT&T Labs Research; ICSA Labs

Expiration Date: May 2001

November 2000

Client Certificate and Key Retrieval for IKE

draft-ietf-ipsra-getcert-00.txt

1. Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

2. Abstract

IKE was designed for use with certificates. In a remote access scenario, that implies that clients must possess their own certificates. We leverage off of work already done to fast-start certificate use with IPsec via the Simple Certificate Enrollment Protocol [SCEP]. We use only parts of SCEP over a client authenticated TLS/HTTP connection to a CA. By using TLS, the client can trust a CA root certificate it receives, without an out-of-band verification and the CA can perform automatic enrollment. We replace the out-of-band client identification process for a certificate enrollment with a legacy authentication, like RADIUS. Further, since the certificates issued here are short-lived, there is no need to support client-based revocation or rekeying. Also, there is typically no need for CRL support.

3. Introduction

IKE was designed for use with certificates. In a remote access scenario, that implies that clients must possess their own certificates. Unfortunately, that is not always practical. Apart from the lack of a suitable PKI in many companies, there is often the need (or desire) to use other forms of personal identification, especially some sort of authentication token in conjunction with a RADIUS server.

We consider inadvisable to change IKE [RFC2409] to meet these needs. IKE is a complex protocol; adding more features to it is a bad idea. Instead, we propose a layered approach: use standard IKE, with certificates, but provide a simple mechanism to provide clients with keys and certificates.

A number of objections have been raised to using certificates. The most important is that we lack a public key infrastructure (PKI). We do not agree that this is an obstacle. Our proposal provides a simple mechanism for certificate generation and retrieval, while still relying on legacy authentication infrastructures. Furthermore, we provide for an easy migration path to certificate use once organizational PKIs are deployed.

In the interests of simplicity, we have chosen to reuse standard protocols and components. In particular, we use HTTP [RFC2616] for transport, HTML [RFC1866] as a data representation and TLS [RFC2246] for confidentiality. However, we do not mandate (or even necessarily encourage) use of a actual Web browser for certificate retrieval.

The client authentication mechanism is not specified, but we assume that a legacy authentication, like a challenge/response authentication to a RADIUS server, will be preformed by the client after the server-based TLS session is set up. Thus the certificate server will have access to adequate information from the authentication server about the client to create a certificate.

We further concluded that our scheme is semantically equivalent to a general certificate enrollment protocol. That is, we suggest that our preferred mechanism is equivalent to what is needed for any sort of certificate issuance; the only difference is that our certificates are very short-lived, which generally eliminates the need for detailed, comprehensive record-keeping and CRLs.

The actual mechanism to get certificates will be the Simple Certificate Enrollment Protocol [SCEP]. This will enable client-side certificate generation that is currently used by many in the IPsec community.

3.1. Requirements Keywords

The keywords "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT", and "MAY" that appear in this document are to be interpreted as described in [RFC2119].

4. Protocol Definition

As noted, we suggest using SCEP. Apart from certificate issuance, there are two other functions that may be needed, root certificate retrieval and CRL retrieval. Both of these functions are provided for in SCEP.

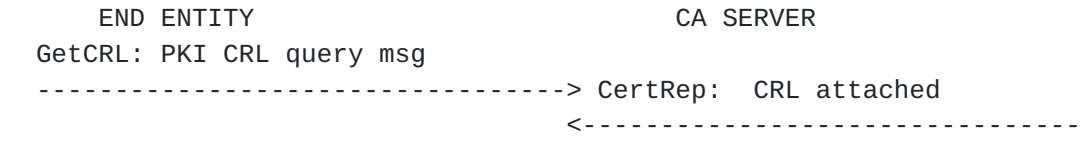
4.1. CA root certificate retrieval

A client MAY need to retrieve the CA's root certificate, if it does not have it cached. The client will use the SCEP Get CA/RA Cert transaction for this as follows:

There is no need for out-of-band root certificate validation here, as the client has an authenticated connection to the server.

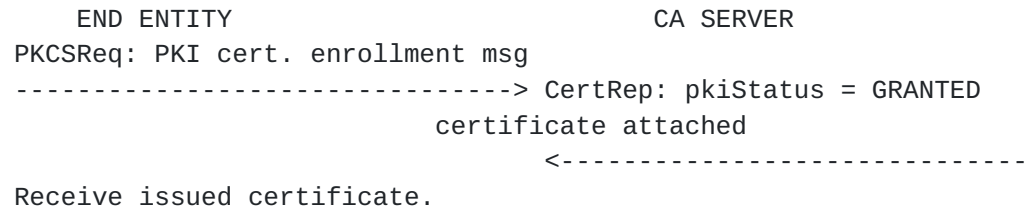
4.2. CRL retrieval

CRL support in GETCERT is OPTIONAL. In many client-to-gateway environments, there is no need for the client to do CRL checking on the gateway. However, in some peer-to-peer IPSRA environments, CRLs could be important. If CRL support is desired, the CRL retrieval guidelines in SCEP SHOULD be followed. The GETCRL transaction is:



4.3. Client certificate request and retrieval (enrollment)

The SCEP 'automatic mode' is used:



4.4. HTTP "GET" Message Format

In the protocol, CA certificates are send to the end entity in clear, whereas the end entity certificates are send out using the PKCS#7 secure protocol. This results in two types of GET operations. The type of GET operation is specified by augmenting the GET message with OPERATION and MESSAGE parameters in the Request-URL. OPERATION identifies the type of GET operation, and MESSAGE is actually the PKI message encoded as a text string.

The following is the syntax definition of a HTTP GET message send from an end entity to a certificate authority server:

Request = "GET " PKI-PATH "?operation=" OPERATION "&message=" MESSAGE where: PKI-PATH defines the actual path to invoke the program which parses the request. This is intended to be the program that the CA will use to handle the SCEP transactions, (Note: the original SCEP specification requires this to end with "pkiclient.exe". We choose to permit the actual path to vary, though of course nothing precludes use of that string.) OPERATION is set to be the string "PKIOperation" when the GET message carries a PKI message to request certificates or CRL; OPERATION is set to be the string "GetCACert" or "GetCACertChain" when the GET operation is used to get CA/RA certificate or the CA Cert chain (respectively). When OPERATION is "PKIOperation", MESSAGE is a base64-encoded PKI message when OPERATION is "GetCACert" or "GetCACertChain", MESSAGE is a string which represents the certificate authority issuer identifier.

4.5. Response Message Format

For each GET operation, the CA/RA server will return a MIME object via HTTP. For a GET operation with PKIOperation as its type, the response is tagged as having a Content Type of application/x-pki-message. The body of this message is a DER encoded binary PKI message. The following is an example of the response:

"Content-Type:application/x-x509-ca-cert\n\n"<DER-encoded X509>

5. Authentication Techniques

Although this document is carefully agnostic about the user authentication techniques to be used, there are two underlying assumptions. First, we assume that the authentication is actually being performed by a back-end RADIUS server, over the TLS HTTP connection, with its accompanying database. Second, we wish to support a variety of common authentication techniques, including ordinary passwords, time-varying tokens, and challenge/response tokens. All are believed to be accommodated by this framework. Finally, we rely on this client and server authentication so that the SCEP assumptions on CA root certificate distribution and subjectName checking are automated.

Requests for certificates must be authenticated. Since we prescribe HTTP, HTTP authentication mechanisms -- under protection of TLS -- MUST be used. Specifically, the request will generally include an appropriate Authorization: line, using whatever form of authentication is locally preferred. If it does not, the server MUST return a 401 error line, per [RFC2616] and [RFC2617]; the client MUST then resubmit the request with the appropriate Authentication: line. (This two-phase process is permitted in order to support challenge/response forms of authentication.)

6. Certificate Characteristics

Signed or generated certificates should, as noted, have a distinctive name format that can be recognized and accepted by the IPsec servers. The expiration time of the certificates is limited by local policy on reuse. In some cases, these certificates will valid for several hours (and hence several sessions, if needed); in other cases, they will expire within a very few minutes and are thus practically usable only for a single IKE exchange. (Note that this also requires tight time synchronization between the authentication server, the IPsec servers, and -- if they care -- the IPsec clients.)

7. Process Flow

The client establishes a TLS secured HTTP connection to the CA service.

The client authenticates the server certificate. The subject is the desired server.

The client provides its authentication over this connection.

The client requests via SCEP the CA root certificate. Note: The server certificate might be a commercial SSL certificate. This way the client might be expected to have that certificate's signing certificate for validation.

The client requests a certificate via SCEP.

The server sends the certificate via SCEP after validating the request against the RADIUS database information.

The client uses the issued certificate in its IKE negotiation with a gateway.

8. Security Considerations

The client -- the program and the ultimate human -- MUST check the server's TLS certificate to guard against man-in-the-middle attacks.

<<more>>

9. Acknowledgements

Paul Hoffman supplied considerable encouragement to the production of this document.

10. References

[RFC1866] "Hypertext Markup Language - 2.0". T. Berners-Lee, D. Connolly. November 1995.

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels". S. Bradner. March 1997.

[RFC2246] "The TLS Protocol Version 1.0". T. Dierks, C. Allen. January 1999.

[RFC2401] "Security Architecture for the Internet Protocol". S. Kent, R. Atkinson. November 1998.

[RFC2409] "The Internet Key Exchange (IKE)". D. Harkins, D. Carrel. November 1998.

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

[SCEP] "Cisco Systems' Simple Certificate Enrollment Protocol". draft-nourse-scep-03.txt, X. Liu, C. Madsen, D. McDrew, A. Nourse. August, 2000.

11. Author Information

Steven M. Bellovin
AT&T Labs Research
Shannon Laboratory
180 Park Avenue
Florham Park, NJ 07974
USA
Phone: +1 973-360-8656
Email: smb@research.att.com

Robert G. Moskowitz
ICSA Labs, a division of TruSecure Corporation
1200 Walnut Bottom Rd.
Carlisle, PA 17013
Email: rgm@icsa.net