TLS Working Group Will Price
INTERNET-DRAFT Michael Elkins
Network Associates, Inc.
18 February 2002

draft-ietf-tls-openpgp-02.txt

Extensions to TLS for OpenPGP keys

Status of this Memo

Abstract

This document builds upon the TLS Protocol Specification [TLS]. The extensions described herein are intended to apply to Version 1.0 of the TLS specification.

The purpose of this document is to update the TLS protocol with extensions to support the certificates, public key algorithms, symmetric ciphers, hash algorithms, and trust model used by OpenPGP [OpenPGP].

This document uses the same notation used in the TLS Protocol draft. 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 RFC 2119.

Certificate

The X.509.v3 [X509] certificates recommended for use with TLS will

not be used in conjunction with OpenPGP keys. An implementation SHOULD be able to support both TLS with X509 and TLS with OpenPGP keys. It is NOT REQUIRED that an implementation support both. The "peer certificate" in the session state of TLS MAY refer to either X509 or OpenPGP.

The contents of the Certificate (11) message sent from server to client and vice versa are determined by the Cipher Suite. Many new Cipher Suites are defined in the Cipher Suites section of this document for use with OpenPGP keys. All OpenPGP Cipher Suites REQUIRE a OpenPGP key in the Certificate messages. A OpenPGP key appearing in the Certificate message will be sent in binary OpenPGP format. The Certificate message includes a OpenPGP key where the X509 certificate would normally appear. The option is also available to send a 64 bit OpenPGP Key ID instead of sending the entire key. The client will respond with a fatal alert no_certificate if the Key ID cannot be found. If the key is not valid, expired, revoked, or corrupt, the appropriate fatal alert message is sent from section A.3 of the TLS specification. If a key is valid and neither expired nor revoked, it is accepted by the protocol. A particular OpenPGP compatible TLS implementation MAY wish to allow users to designate certain keys specifically as Trusted Server Keys. This is a local matter outside the scope of this document.

```
enum {
     keyid (0), key (1), (255)
} PGPKeyDescriptorType;
opaque PGPKeyID<8>
opaque PGPKey<1..2^24-1>
struct {
     PGPKeyDescriptorType descriptorType;
     select (descriptorType) {
          case keyid: PGPKeyID;
          case key: PGPKey;
     }
} Certificate;
```

Certificate Request

The server is allowed to request a client certificate from the client. The meaning of this message is essentially unchanged to allow OpenPGP keys.

The rsa_sign and dss_sign certificate types may be requested in conjunction with OpenPGP keys. The ClientCertificateType field is used identically to the TLS specification. The subsequent DistinguishedName field is NOT included when using Cipher Suites

based on OpenPGP keys.

The Client Certificate message is sent using the same formatting as the server Certificate message in response to the Certificate Request message. If no OpenPGP key is available from the client, the no_certificate alert is returned. The server MAY then respond with a fatal alert if appropriate. This transaction follows the TLS specification.

Server Key Exchange

When using ephemeral Diffie-Hellman, the Server Key Exchange message is sent to pass the Diffie-Hellman public value to the client. The client and server then use this value to establish the shared secret. This is identical to the operation of standard TLS.

Certificate Verify

The Certificate Verify message for OpenPGP key types is identical to the specification. The signature is made using either DSS or RSA depending on the Cipher Suite.

Finished

The Finished message for OpenPGP key types is identical to the description in the specification.

Cipher Suites

Since TLS does not have a certificate type field, the Cipher Suites field is used to perform the same function. A number of additional Cipher Suites are defined for use with OpenPGP keys.

```
CipherSuite TLS_PGP_DHE_DSS_WITH_CAST_CBC_SHA          = { 0x01, 0x01 };
CipherSuite TLS_PGP_DHE_DSS_WITH_IDEA_CBC_SHA          = { 0x01, 0x02 };
CipherSuite TLS_PGP_DHE_DSS_WITH_3DES_EDE_CBC_SHA      = { 0x01, 0x03 };
CipherSuite TLS_PGP_DHE_DSS_WITH_CAST_CBC_RMD          = { 0x01, 0x04 };
CipherSuite TLS_PGP_DHE_DSS_WITH_IDEA_CBC_RMD          = { 0x01, 0x05 };
CipherSuite TLS_PGP_DHE_DSS_WITH_3DES_EDE_CBC_RMD      = { 0x01, 0x06 };
CipherSuite TLS_PGP_DHE_RSA_WITH_CAST_CBC_SHA          = { 0x01, 0x10 };
CipherSuite TLS_PGP_RSA_WITH_CAST_CBC_SHA              = { 0x01, 0x20 };
CipherSuite TLS_PGP_RSA_WITH_IDEA_CBC_SHA              = { 0x01, 0x21 };
CipherSuite TLS_PGP_RSA_WITH_3DES_EDE_CBC_SHA          = { 0x01, 0x22 };
CipherSuite TLS_PGP_RSA_WITH_CAST_CBC_RMD              = { 0x01, 0x23 };
CipherSuite TLS_PGP_RSA_WITH_IDEA_CBC_RMD              = { 0x01, 0x24 };
CipherSuite TLS_PGP_RSA_WITH_3DES_EDE_CBC_RMD          = { 0x01, 0x25 };
CipherSuite TLS_PGP_DSA_WITH_NULL_SHA                  = { 0x01, 0xF0 };
```

Note: The above numeric definitions for Cipher Suites have not yet been registered.

All of the above Cipher Suites use either the CAST, IDEA, or TripleDES block ciphers in CBC mode. The choice of hash is either SHA-1 or RIPEMD-160. Use of any of these Cipher Suites REQUIRES an OpenPGP key in any Certificate and Client Certificate messages. MD5 MAY NOT be used with OpenPGP keys. "Export" algorithms also MAY NOT be used with OpenPGP keys.

To be considered compliant with support for OpenPGP in TLS, an implementation MUST support TLS_PGP_DHE_DSS_WITH_3DES_EDE_CBC_SHA.

References

[TLS] T. Dierks, and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[OpenPGP] J. Callas, L. Donnerhacke, H. Finney, R. Thayer, "OpenPGP Message Format", RFC 2440, November 1998.

Authors

Will Price <wprice@cyphers.net>
Network Associates, Inc.
Michael Elkins <michael_elkins@nai.com>
Network Associates, Inc.