     Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)

Status of this Memo

Copyright Notice

IESG Note:

   The 40-bit ciphersuites defined in this memo are included only for
   the purpose of documenting the fact that those ciphersuite codes have
   already been assigned. 40-bit ciphersuites were designed to comply
   with US-centric, and now obsolete, export restrictions. They were
   never secure, and nowadays are inadequate even for casual
   applications. Implementation and use of the 40-bit ciphersuites
   defined in this document, and elsewhere, is strongly discouraged.

1. Abstract

   This document proposes the addition of new cipher suites to the TLS
   protocol [1] to support Kerberos-based authentication. Kerberos
   credentials are used to achieve mutual authentication and to
   establish a master secret which is subsequently used to secure
   client-server communication.

2. Introduction

   Flexibility is one of the main strengths of the TLS protocol.
   Clients and servers can negotiate cipher suites to meet specific
   security and administrative policies. However, to date,
   authentication in TLS is limited only to public key solutions. As a
   result, TLS does not fully support organizations with heterogeneous
   security deployments that include authentication systems based on
   symmetric cryptography. Kerberos, originally developed at MIT, is

based on an open standard [2] and is the most widely deployed
symmetric key authentication system. This document proposes a new
option for negotiating Kerberos authentication within the TLS
framework. This achieves mutual authentication and the establishment
of a master secret using Kerberos credentials. The proposed changes
are minimal and, in fact, no different from adding a new public key
algorithm to the TLS framework.

3. Kerberos Authentication Option In TLS

   This section describes the addition of the Kerberos authentication
   option to the TLS protocol. Throughout this document, we refer to
   the basic SSL handshake shown in Figure 1. For a review of the TLS
   handshake see [1].

```
  CLIENT                                            SERVER
  ------                                            ------
 ClientHello
                       -------------------------------->
                                                   ServerHello
                                                   Certificate *
                                                   ServerKeyExchange*
                                                   CertificateRequest*
                                                   ServerHelloDone
                       <------------------------------
 Certificate*
 ClientKeyExchange
 CertificateVerify*
 change cipher spec
 Finished
      |                -------------------------------->
      |                                             change cipher spec
      |                                             Finished
      |                                                |
      |                                                |
 Application Data     <------------------------------->Application Data
```
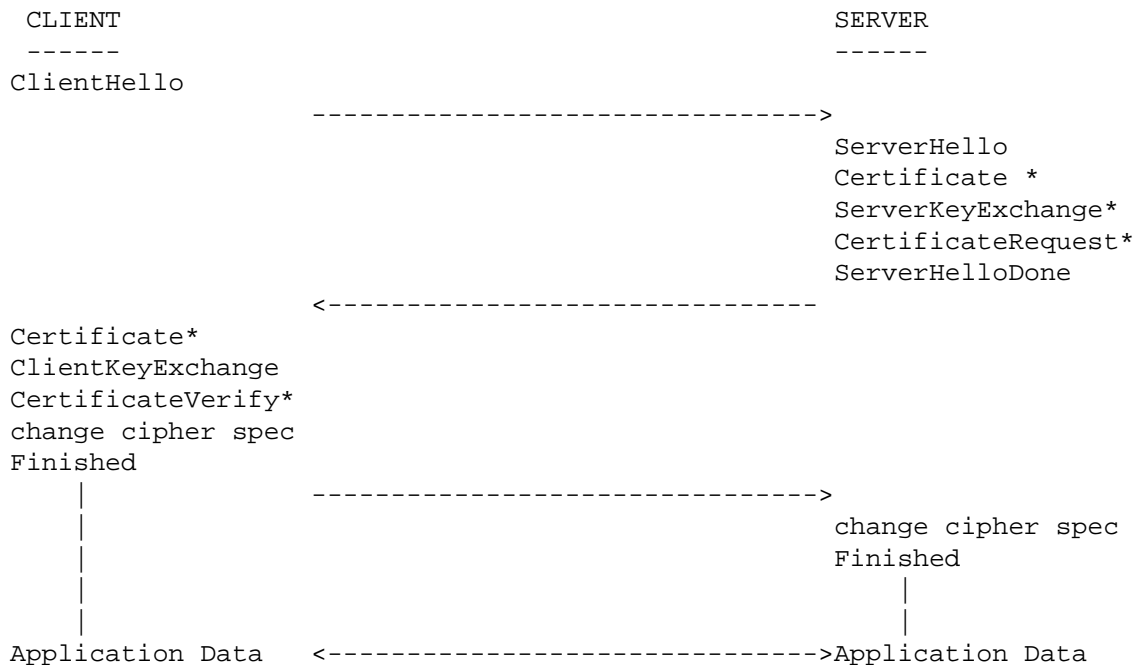
 FIGURE 1: The TLS protocol. All messages followed by a star are
           optional. Note: This figure was taken from an IETF document
           [1].

   The TLS security context is negotiated in the client and server hello
   messages. For example: TLS_RSA_WITH_RC4_MD5 means the initial
   authentication will be done using the RSA public key algorithm, RC4
   will be used for the session key, and MACs will be based on the MD5
   algorithm. Thus, to facilitate the Kerberos authentication option,
   we must start by defining new cipher suites including (but not
   limited to):

```
CipherSuite        TLS_KRB5_WITH_DES_CBC_SHA          = { 0x00,0x1E };
CipherSuite        TLS_KRB5_WITH_3DES_EDE_CBC_SHA     = { 0x00,0x1F };
CipherSuite        TLS_KRB5_WITH_RC4_128_SHA          = { 0x00,0x20 };
CipherSuite        TLS_KRB5_WITH_IDEA_CBC_SHA         = { 0x00,0x21 };
CipherSuite        TLS_KRB5_WITH_DES_CBC_MD5          = { 0x00,0x22 };
CipherSuite        TLS_KRB5_WITH_3DES_EDE_CBC_MD5     = { 0x00,0x23 };
CipherSuite        TLS_KRB5_WITH_RC4_128_MD5          = { 0x00,0x24 };
CipherSuite        TLS_KRB5_WITH_IDEA_CBC_MD5         = { 0x00,0x25 };


CipherSuite        TLS_KRB5_EXPORT_WITH_DES_CBC_40_SHA = { 0x00,0x26 };
CipherSuite        TLS_KRB5_EXPORT_WITH_RC2_CBC_40_SHA = { 0x00,0x27 };
CipherSuite        TLS_KRB5_EXPORT_WITH_RC4_40_SHA     = { 0x00,0x28 };
CipherSuite        TLS_KRB5_EXPORT_WITH_DES_CBC_40_MD5 = { 0x00,0x29 };
CipherSuite        TLS_KRB5_EXPORT_WITH_RC2_CBC_40_MD5 = { 0x00,0x2A };
CipherSuite        TLS_KRB5_EXPORT_WITH_RC4_40_MD5     = { 0x00,0x2B };
```

   To establish a Kerberos-based security context, one or more of the
   above cipher suites must be specified in the client hello message.
   If the TLS server supports the Kerberos authentication option, the
   server hello message, sent to the client, will confirm the Kerberos
   cipher suite selected by the server. The server's certificate, the
   client

   CertificateRequest, and the ServerKeyExchange shown in Figure 1 will
   be omitted since authentication and the establishment of a master
   secret will be done using the client's Kerberos credentials for the
   TLS server. The client's certificate will be omitted for the same
   reason. Note that these messages are specified as optional in the
   TLS protocol; therefore, omitting them is permissible.

   The Kerberos option must be added to the ClientKeyExchange message as
   shown in Figure 2.

```
 struct
 {
    select (KeyExchangeAlgorithm)
    {
        case krb5:            KerberosWrapper;        /* new addition */
        case rsa:             EncryptedPreMasterSecret;
        case diffie_hellman:  ClientDiffieHellmanPublic;
    } Exchange_keys;

 } ClientKeyExchange;

 struct
 {
    opaque Ticket;
    opaque authenticator;            /* optional */
    opaque EncryptedPreMasterSecret; /* encrypted with the session key
                                        which is sealed in the ticket */
 } KerberosWrapper;                  /* new addition */
```

FIGURE 2: The Kerberos option in the ClientKeyExchange.

To use the Kerberos authentication option, the TLS client must obtain
a service ticket for the TLS server. In TLS, the ClientKeyExchange
message is used to pass a random 48-byte pre-master secret to the
server.

The client and server then use the pre-master secret to independently
derive the master secret, which in turn is used for generating
session keys and for MAC computations. Thus, if the Kerberos option
is selected, the pre-master secret structure is the same as that used
in the RSA case; it is encrypted under the Kerberos session key and
sent to the TLS server along with the Kerberos credentials (see
Figure 2). The ticket and authenticator are encoded per RFC 1510
(ASN.1 encoding). Once the ClientKeyExchange message is received,
the server's secret key is used to unwrap the credentials and extract
the pre-master secret.

Note that a Kerberos authenticator is not required, since the master
secret derived by the client and server is seeded with a random value
passed in the server hello message, thus foiling replay attacks.
However, the authenticator may still prove useful for passing
authorization information and is thus allotted an optional field (see
Figure 2).

Lastly, the client and server exchange the finished messages to
complete the handshake. At this point we have achieved the
following:

   1) A master secret, used to protect all subsequent communication, is
      securely established.

   2) Mutual client-server authentication is achieved, since the TLS
      server proves knowledge of the master secret in the finished
      message.

   Note that the Kerberos option fits in seamlessly, without adding any
   new messages.

4. Naming Conventions:

   To obtain an appropriate service ticket, the TLS client must
   determine the principal name of the TLS server. The Kerberos service
   naming convention is used for this purpose, as follows:

     host/MachineName@Realm
       where:
          - The literal, "host", follows the Kerberos convention when not
            concerned about the protection domain on a particular machine.
          - "MachineName" is the particular instance of the service.
          - The Kerberos "Realm" is the domain name of the machine.

5. Summary

   The proposed Kerberos authentication option is added in exactly the
   same manner as a new public key algorithm would be added to TLS.
   Furthermore, it establishes the master secret in exactly the same
   manner.

6. Security Considerations

   Kerberos ciphersuites are subject to the same security considerations
   as the TLS protocol. In addition, just as a public key
   implementation must take care to protect the private key (for example
   the PIN for a smartcard), a Kerberos implementation must take care to
   protect the long lived secret that is shared between the principal
   and the KDC. In particular, a weak password may be subject to a
   dictionary attack. In order to strengthen the initial authentication
   to a KDC, an implementor may choose to utilize secondary
   authentication via a token card, or one may utilize initial
   authentication to the KDC based on public key cryptography (commonly
   known as PKINIT - a product of the Common Authentication Technology
   working group of the IETF).

7. Acknowledgements

   We would like to thank Clifford Neuman for his invaluable comments on
   earlier versions of this document.

8. References

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

   [2] Kohl J. and C. Neuman, "The Kerberos Network Authentication
       Service (V5)", RFC 1510, September 1993.

9. Authors' Addresses

   Ari Medvinsky
   Excite
   555 Broadway
   Redwood City, CA 94063

   Phone: +1 650 569 2119
   EMail: amedvins@excitecorp.com
   http://www.excite.com

   Matthew Hur
   CyberSafe Corporation
   1605 NW Sammamish Road
   Issaquah WA 98027-5378

   Phone: +1 425 391 6000
   EMail: matt.hur@cybersafe.com
   http://www.cybersafe.com

10. Full Copyright Statement

   Copyright (C) The Internet Society (1999). All Rights Reserved.

   This document and translations of it may be copied and furnished to
   others, and derivative works that comment on or otherwise explain it
   or assist in its implementation may be prepared, copied, published
   and distributed, in whole or in part, without restriction of any
   kind, provided that the above copyright notice and this paragraph are
   included on all such copies and derivative works. However, this
   document itself may not be modified in any way, such as by removing
   the copyright notice or references to the Internet Society or other
   Internet organizations, except as needed for the purpose of
   developing Internet standards in which case the procedures for
   copyrights defined in the Internet Standards process must be
   followed, or as required to translate it into languages other than
   English.

   The limited permissions granted above are perpetual and will not be
   revoked by the Internet Society or its successors or assigns.

   This document and the information contained herein is provided on an
   "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING
   TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING
   BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION
   HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF
   MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

   Funding for the RFC Editor function is currently provided by the
   Internet Society.