INTERNET-DRAFT
<draft-ietf-aft-socks-ssl-00>
Expires 20 September 1997

M. VanHeyningen
Aventail Corporation
20 March 1997

Secure Sockets Layer for SOCKS Version 5

STATUS OF THIS MEMO

This document is an Internet-Draft. 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.''

ABSTRACT

This document specifies the use of SSL 3.0 and possible successor protocols as an authentication method for SOCKS Version 5. The design is similar to, and largely derived from, the integration of GSS-API into SOCKS5 [RFC 1961]. A framework is provided for future extensions, and the use of other "subauthentication" methods inside SSL is supported.

SSL METHOD IDENTIFIER

During the initial SOCKS V5 negotiation, an authentication method is negotiated. The identifier used in the SOCKS5 authentication handshake for this method shall be X'86' (134 decimal.)

SSL RECORD FRAMING

The SSL data stream is wrapped for transport over the SOCKS5 data stream as follows:

VER	STATE	LEN	DATA
1	1	2	0 - 65535

The VER shall be X'01 for this revision. The STATE may have four different values:

```
Initial handshake     X'01'
Option negotiation    X'02'
Data flow             X'03'
Closing handshake     X'04'
```

LEN specifies the number of bytes in the DATA, and shall be MSB-first.

SSL PROCEDURE

After the server selects SSL as its authentication method, an initial handshake to establish an SSL-secured connection commences with STATE of X'01'.

SSL specific options

After this handshake all subsequent data is secured via SSL, then in the framing specified above. With STATE of X'02', the client sends a list of SSL-specific options it can support:

```
+-----+----------
| LEN | OPTIONS
+-----+----------
|  1  |  0-255
+-----+----------
```

The server responds with an identically formatted list of the subset of options to be used for this connection.

At this time, only two options are defined:

```
Subauthentication   X'01'
UDP-Naked           X'10'
```

UDP-Naked

With SSL integration, the handling of UDP data is a tricky issue. Unlike GSSAPI-based solutions, SSL is not designed to handle datagram traffic. UDP-Naked specifies that SSL shall be used only to protect the control connection of the UDP association, and UDP traffic will receive no enhancements.

It is expected that other options for UDP processing will be defined; in general it is expected that the server will choose one UDP method from those offered by the client.

Subauthentication

Subauthentication provides a mechanism for embedding a simpler

authentication mechanism inside SSL. For example, some environments may use SSL with server-side certificates to establish a secure connection to a server, then use cleartext passwords [RFC 1929] inside the encrypted connection.

If the server's option handshake specifies that subauthentication is to be used, the client responds with a message specifying the authentication methods it supports in this environment; the format is identical to that of [RFC 1928, sec 3].

After negotiating a subauthentication method, subauthentication proceeds exactly as it would normally, but inside the SSL-secured data stream. It is expected that only primitive authentication techniques, such as cleartext passwords or CHAP [SOCKS-CHAP], will be used as subauthentitcation methods.

Command processing

After option processing is complete, the authentication method finishes. STATE is set to X'03', and the SOCKS command commences.

Orderly closure

Upon connection termination, the terminating side should change STATE to X'04' and send an SSL closure handshake; the other side should also change its STATE and respond appropriately.

SECURITY CONSIDERATIONS

As with any negotiation-based mechanism, SOCKS5 is subject to downgrade attacks. Both clients and servers should operate only with security parameters consistent with the given security policy.

The UDP-Naked option, as its name suggests, provides very minimal security for UDP traffic. It allows subauthentication and initial commands which may specifiy destination and other use restrictions on a UDP association to be established securely, but it does nothing to preserve the privacy or integrity of the bulk data. More secure methods of proxying UDP traffic may be added as additional SSL-specific options.

All SSL-based considerations, such as the importance of properly seeding random generators used to generate keying information and storing persistent keying information in secure ways, apply here.

SSL 2.0 has known weakness and limitations. Since there is no existing base of software using this specification based on SSL 2.0, applications may support SSL 3.0 (or later) only without compromising

interoperability.

REFERENCES

[RFC 1928] Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., & Jones, L., "SOCKS Protocol V5," April 1996.

[RFC 1929] Leech, M., "Username/Password Authentication for SOCKS V5," March 1996.

[RFC 1961] McMahon, P., "GSS-API Authentication Method for SOCKS Version 5," June 1996.

[SOCKS-CHAP] VanHeyningen, M., "Challenge-Handshake Authentication Protocol for SOCKS V5," <draft-ietf-aft-socks-chap-00>, March 1997, work in progress.

[SSL] Freier, A., Karlton, P., Kocher, P., "The SSL Protocol: Version 3.0," <draft-ietf-tls-ssl-version3-00.txt>, November 1996, work in progress.

AUTHOR'S ADDRESS

Marc VanHeyningen
Aventail Corporation
117 S. Main Street, Suite 400
Seattle, WA 98104

Phone: +1 206 777-5600

Fax: +1 206 777-5656
Email: marcvh@aventail.com