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

M. VanHeyningen
Aventail Corporation
20 March 1997

Challenge-Response Authentication Method for SOCKS V5

Status of this Memo

Abstract

This document specifies a general Challenge-Response Authentication Method (CRAM) for use with SOCKS Version 5 [RFC 1928]. It is intended to support various challenge-response mechanisms, such as S/KEY and OTP [RFC 1938] as well as authentication tokens.

Introduction

The protocol specification for SOCKS Version 5 [RFC 1928] specifies a generalized framework for the use of arbitrary authentication protocols in the initial SOCKS connection setup. This document suggests a general framework for a Challenge-Response Authentication Method (CRAM) as it fits into the SOCKS Version 5 authentication "subnegotiation."

Initial Negotiation

During initial SOCKS V5 negotiation, the client and server negotiate the authenticiation method. The METHOD for this protocol shall be X'05'.

Challenge-Reponse Framework

Subnegotiation begins after the client has selected the CRAM authentication method.

Message Format

In general, messages exchanged consist of a version identifier and a list of attribute-value assertions, where attributes are single octets and values are sequences of 0-255 octets.

VER	NAVAS	ATT1	VAL1LEN	VAL1	ATT2	...
1	1	1	1	0-255	1	...

VER contains the current version of the subnegotiation, which is X'01'. NAVAS contains the number of attribute-value assertions to follow. Each AVA includes ATT_i, containing the attribute, VAL_iLEN, containing the length of VAL_i, and VAL_i. In general, robust implementations should ignore assertions with attributes they do not understand. This provides a powerful and general mechanism for future extensions while allowing backward compatibility.

Notationally, a single message with a list of n assertions shall be represented as:

ATT_1(VAL_1), ATT_2(VAL_2), ... ATT_n(VAL_n)

Attributes

The following attribute definitions apply to all messages:

ATT	Label	Meaning
X'00'	STATUS	0 = success
X'01'	TEXT-MESSAGE	Informational text
X'02'	USER-IDENTITY	
X'03'	CHALLENGE	
X'04'	RESPONSE	
X'05'	CHARSET	

The TEXT-MESSAGE attribute may always be included in any message (except submethod negotiation.) Implementations should display its contents to the user if applicable; it should be used for advisory information (e.g. warnings of pending password expiration, explanations accompanying a failure.) If there is no user,

implementations may log its contents.

The CHARSET attribute provides advisory infomration about the character set in use; it, too, may be present in any message. Implementations should use it to guide presentation of information to users. The semantics are identical to that of the charset parameter in MIME [RFC 1521]; if absent, a default of ISO-8859-1 should be assumed.

Submethods may also define their own additional attributes, but must not redefine the above standard ones.

Protocol Exchange

Generic challenge-response simply authenticates the client's identity by sending a textual challenge from the server which the client displays to the user. The user somehow (e.g. by using an external application or a security token) computes the appropriate response and enters it.

First, the client asserts its identity to the server:

USER-IDENTITY(<username>)

The server then responds with a textual challenge to be displayed to the user:

CHALLENGE(<challenge>)

The client displays this challenge, prompts for a response, and sends it:

RESPONSE(<response>)

The server may respond with a new challenge method, for which a new response is

required. Arbitrarily many challenges may be issued.
When the server finished issuing challenges, it sends a status message:

STATUS(success|failure)

and the subnegotiation terminates. If the authentication did not succeed, the server must drop the connection.

Security Considerations

Challenge-response protocols are generally designed to provide protection from passive attacks such as sniffing passwords. The security mechanisms here generally offer only limited protection from

real-time active attacks.

In most challenge-response security mechanisms, it is important that challenges be produced in a fashion an adversary cannot predict or duplicate. As with all negotiation-based security, implementations may be vulnerable to downgrade attacks. Clients and servers should refuse to operate with methods and algorithms considered insufficiently secure.

Acknowledgements

Thanks to Dave Blob, Wei Lu, Craig Metz and William Perry for assistance with this document.

References

[RFC 1521] Borenstein, N, & Freed, N., "MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies," September 1993.

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

[RFC 1938] Haller, N., & Metz, C., "A One-Time Password System," May 1996.

[RFC 2058] Rigney, C., Rubens, A., Simpson, W. A., & Willens, S., "Remote Authentication Dial In User Service (RADIUS)", January 1997.

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