0. Abstract

This document describes the telnet authentication option as a generic method for negotiating an authentication type and mode including whether encryption should be used and if credentials should be forwarded. While this document summarizes currently utilized commands and types it does not define a specific authentication type. Separate documents are to be published defining each authentication type.

1. Command Names and Codes

<table>
<thead>
<tr>
<th>Command</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHENTICATION</td>
<td>37</td>
</tr>
</tbody>
</table>

Authentication Commands

- IS          | 0    |
- SEND        | 1    |
- REPLY       | 2    |
- NAME        | 3    |

Authentication Types

- NULL        | 0    |
- KERBEROS_V4 | 1    |
2. Command Meanings

This document makes reference to a "server" and a "client". For the purposes of this document, the "server" is the side of the connection that did the passive TCP open (TCP LISTEN state), and the "client" is the side of the connection that did the active open.

**IAC WILL AUTHENTICATION**

The client side of the connection sends this command to indicate that it is willing to send and receive authentication information.

**IAC DO AUTHENTICATION**

The servers side of the connection sends this command to indicate...
that it is willing to send and receive authentication information.

IAC WONT AUTHENTICATION

The client side of the connection sends this command to indicate that it refuses to send or receive authentication information; the server side must send this command if it receives a DO AUTHENTICATION command.

IAC DONT AUTHENTICATION

The server side of the connection sends this command to indicate that it refuses to send or receive authentication information; the client side must send this command if it receives a WILL AUTHENTICATION command.

IAC SB AUTHENTICATION SEND authentication-type-pair-list IAC SE

The sender of this command (the server) requests that the remote side send authentication information for one of the authentication types listed in "authentication-type-pair-list". The "authentication-type-pair-list" is an ordered list of "authentication-type" pairs. Only the server side (DO AUTHENTICATION) is allowed to send this.

IAC SB AUTHENTICATION IS authentication-type-pair <auth data> IAC SE

The sender of this command (the client) is sending the authentication information for authentication type "authentication-type-pair". Only the client side (WILL AUTHENTICATION) is allowed to send this.

IAC SB AUTHENTICATION REPLY authentication-type-pair <auth data> IAC SE

The sender of this command (the server) is sending a reply to the authentication information received in a previous IS command. Only the server side (DO AUTHENTICATION) is allowed to send this.

IAC SB AUTHENTICATION NAME remote-user IAC SE

This optional command is sent to specify the account name on the remote host that the user wishes to be authorized to use. Note that authentication may succeed, and the authorization to use a particular account may still fail. Some authentication mechanisms may ignore this command.

The "authentication-type-pair" is two octets, the first is the authentication type, and the second is a modifier to the type. The au-
authentication type may or may not include built-in encryption. For instance, when the Kerberos 4 authentication type is negotiated encryption must be negotiated with the telnet ENCRYPT option. However, the SSL and KEA_SJ authentication types provide an encrypted channel as part of a successful telnet AUTH option negotiation.

There are currently five one bit fields defined in the modifier. The first two of these bits are processed as a pair, the AUTH_WHO_MASK bit and the AUTH_HOW_MASK bit. There are four possible combinations of these two bits:

AUTH_CLIENT_TO_SERVER
AUTH_HOW_ONE_WAY

The client will send authentication information about the local user to the server. If the negotiation is successful, the server will have authenticated the user on the client side of the connection.

AUTH_SERVER_TO_CLIENT
AUTH_HOW_ONE_WAY

The server will authenticate itself to the client. If the negotiation is successful, the client will know that it is connected to the server that it wants to be connected to.

AUTH_CLIENT_TO_SERVER
AUTH_HOW_MUTUAL

The client will send authentication information about the local user to the server, and then the server will authenticate itself to the client. If the negotiation is successful, the server will have authenticated the user on the client side of the connection, and the client will know that it is connected to the server that it wants to be connected to.

AUTH_SERVER_TO_CLIENT
AUTH_HOW_MUTUAL

The server will authenticate itself to the client, and then the client will authenticate itself to the server. If the negotiation is successful, the client will know that it is connected to the server that it wants to be connected to, and the server will know that the client is who it claims to be.

The third and fifth bits in the modifier are the ENCRYPT_MASK bits. These bits are used to determine if and how encryption should be enabled. Of the four possible combinations only three are currently defined:
ENCRIPT_OFF

Encryption will not be used for this session.  TELOPT ENCRYPTION SHOULD NOT be negotiated.  This mode MUST be used with all AUTH types that do not provide a shared secret to be used as a session key.

ENCRIPT_USING_TELOPT

Encryption will be negotiated via the use of TELOPT ENCRYPT.  Immediately after authentication has completed TELOPT ENCRYPT MUST be negotiated in both directions.  This is required to occur before credentials forwarding; other telnet options are negotiated; or any user data is transmitted.  A failure to successfully negotiate TELOPT ENCRYPT in either direction MUST result in immediate session termination.

ENCRIPT_AFTER_EXCHANGE

Encryption will be activated in both directions immediately after the successful exchange of the shared secret to be used as the session key.  The encryption algorithm to be used MUST be implied by the AUTH type.

The fourth bit field in the modifier is the INI_CRED_FWD_MASK bit.  This bit is either set to INI_CRED_FWD_ON or INI_CRED_FWD_OFF.  This bit is set by the client to advise the server to expect forwarded credentials from the client.

INI_CRED_FWD_OFF

The client will not be forwarding credentials to the server.  This mode must be used if the selected authentication method does not support credentials forwarding.

INI_CRED_FWD_ON

Once authentication, and perhaps encryption, completes, the client will immediately forward authentication credentials to the server.

The motivation for this advisory bit is that the server may wish to wait until the forwarded credentials have been sent before starting any operating system specific login procedures which may depend on these credentials.  Note that credentials forwarding may not be supported by all authentication mechanisms.  It is a protocol error to set this bit if the underlying authentication mechanism does not support credentials forwarding.

Credentials forwarding MUST NOT be performed if
AUTH_CLIENT_TO_SERVER|AUTH_HOW_ONE_WAY was used since the identity of the server can not be assured. Credentials SHOULD NOT be forwarded if the telnet connection is not protected using some encryption or integrity protection services.

Note that older implementations of the telnet authentication option will not understand the ENCRYPT_MASK and INI_CRED_FWD_MASK bits. Hence an implementation wishing to offer these bits should offer authentication type pairs with these bits both set and not set if backwards compatibility is required.

3. Default Specification

The default specification for this option is

WONT AUTHENTICATION
DONT AUTHENTICATION

meaning there will not be any exchange of authentication information.

4. Motivation

One of the deficiencies of the Telnet protocol is that in order to log into remote systems, users have to type their passwords, which are passed in clear text through the network. If the connections goes through untrusted networks, there is the possibility that passwords will be compromised by someone watching the packets as they go by.

The purpose of the AUTHENTICATION option is to provide a framework for the passing of authentication information through the TELNET session, and a mechanism to enable encryption of the data stream as a side effect of successful authentication or via subsequent use of the telnet ENCRYPT option. This means that: 1) the users password will not be sent in clear text across the network, 2) if the front end telnet process has the appropriate authentication information, it can automatically send the information, and the user will not have to type any password. 3) once authentication has succeeded, the data stream can be encrypted to provide protection against active attacks.

It is intended that the AUTHENTICATION option be general enough that it can be used to pass information for any authentication and encryption system.

5. Security Implications

The ability to negotiate a common authentication mechanism between client and server is a feature of the authentication option that should be used with caution. When the negotiation is performed, no authentication has yet occurred. Therefore each system has no way of
knowing whether or not it is talking to the system it intends. An intruder could attempt to negotiate the use of an authentication system which is either weak, or already compromised by the intruder.

If the authentication type requires that encryption be enabled as a separate optional negotiation (the ENCRYPT option), it will provide a window of vulnerability from when the authentication completes, up to and including the negotiation to turn on encryption by an active attacker. An active attack is one where the underlying TCP stream can be modified or taken over by the active attacker. If the server only offers authentication type pairs that include the ENCRYPT_USING_TELOPT set in the ENCRYPT_MASK field, this will avoid the window of vulnerability, since both parties will agree that telnet ENCRYPT option must be successfully negotiated immediately following the successful completion of telnet AUTH.

Other authentication types links the enabling of encryption as a side effect of successful authentication. This will also provide protection against the active attacker. The ENCRYPT_AFTER_EXCHANGE bit allows these authentication types to negotiate encryption so that it can be made optional.

Another opportunity for active attacks is presented when encryption may be turned on and off without re-authentication. Once encryption is disabled, an attacker may hijack the telnet stream, and interfere with attempts to restart encryption. Therefore, a client SHOULD NOT support the ability to turn off encryption. Once encryption is disabled, if an attempt to re-enable encryption fails, the client MUST terminate the telnet connection.

It is important that in both cases the authentication type pair be integrity protected at the end of the authentication exchange. This must be specified for each authentication type to ensure that the result of the telnet authentication option negotiation is agreed to by both the client and the server. Some authentication type suboptions may wish to include all of the telnet authentication negotiation exchanges in the integrity checksum, to fully protect the entire exchange.

6. Implementation Rules

WILL and DO are used only at the beginning of the connection to obtain and grant permission for future negotiations.

The authentication is only negotiated in one direction; the server must send the "DO", and the client must send the "WILL". This restriction is due to the nature of authentication; there are three possible cases; server authenticates client, client authenticates server, and server and client authenticate each other. By only negotiating the option in one direction, and then determining which of
Once the two hosts have exchanged a DO and a WILL, the server is free to request authentication information. In the request, a list of supported authentication types is sent. Only the server may send requests ("IAC SB AUTHENTICATION SEND authentication-type-pair-list IAC SE"). Only the client may transmit authentication information via the "IAC SB AUTHENTICATION IS authentication-type ... IAC SE" command. Only the server may send replies ("IAC SB AUTHENTICATION REPLY authentication-type ... IAC SE"). As many IS and REPLY suboptions may be exchanged as are needed for the particular authentication scheme chosen.

If the client does not support any of the authentication types listed in the authentication-type-pair-list, a type of NULL should be used to indicate this in the IS reply. Note that if the client responds with a type of NULL, the server may choose to close the connection. When the server has concluded that authentication cannot be negotiated with the client it should send IAC DONT AUTH to the client.

The order of the authentication types MUST be ordered to indicate a preference for different authentication types, the first type being the most preferred, and the last type the least preferred.

As long as the server is WILL AUTH it may request authentication information at any time. This is done by sending a new list of supported authentication types. Requesting authentication information may be done as a way of verifying the validity of the client's credentials after an extended period of time or to negotiate a new session key for use during encryption.

7. User Interface

Normally protocol specifications do not address user interface specifications. However, due to the fact that the user will probably want to be able to specify the things about authentication and encryption and also know whether or not things succeeded, some guidance needs to be given to implementors to provide some minimum level of user control.

The user MUST be able to specify whether or not authentication is to be used, and whether or not encryption is to used if the authentication succeeds. There SHOULD be at least four settings, REQUIRE, PROMPT, WARN and DISABLE. Setting the authentication switch to REQUIRE means that if the authentication fails, then an appropriate error message must be displayed and the TELNET connection must be terminated. Setting the authentication switch to PROMPT means that if
the authentication fails, then an appropriate error message must be displayed and the user must be prompted for confirmation before continuing the TELNET session. Setting the authentication switch to WARN means that if the authentication fails, then an appropriate error message must be displayed before continuing the TELNET session. Setting the authentication switch to DISABLE means that authentication will not be attempted. The encryption switch SHOULD have the same settings as the authentication switch; however its settings are only used when authentication succeeds. The default setting for both switches should be WARN. Both of these switches may be implemented as a single switch, though having them separate gives more control to the user.

8. Example

The following is an example of use of the option:

Client                           Server
IAC WILL AUTHENTICATION
  [ The server is now free to request authentication information. ]
    IAC SB AUTHENTICATION SEND
    KERBEROS_V4 CLIENT|MUTUAL
    KERBEROS_V4 CLIENT|ONE WAY IAC SE
    [ The server has requested mutual Kerberos authentication, but is willing to do just one-way Kerberos authentication. The client will now respond with the name of the user that it wants to log in as, and the Kerberos ticket. ]
    IAC SB AUTHENTICATION NAME "joe"
    IAC SE
    IAC SB AUTHENTICATION IS
    KERBEROS_V4 CLIENT|MUTUAL AUTH 4
    7 1 67 82 65 89 46 67 7 9 77 0
    48 24 49 244 109 240 50 208 43
    35 25 116 104 44 167 21 201 224
    229 145 20 2 244 213 220 33 134
    148 4 251 249 233 229 152 77 2
    109 130 231 33 146 190 248 1 9
    31 95 94 15 120 224 0 225 76 205
    70 136 245 190 199 147 155 13
    IAC SE
    [ The server responds with an ACCEPT command to state that the authentication was successful. ]
    IAC SB AUTHENTICATION REPLY
    KERBEROS_V4 CLIENT|MUTUAL ACCEPT
    IAC SE
    [ Next, the client sends across a CHALLENGE to verify that it is really talking to the right server. ]
IAC SB AUTHENTICATION IS
KERBEROS_V4 CLIENT|MUTUAL
CHALLENGE xx xx xx xx xx xx xx
xx IAC SE
[ Lastly, the server sends across a RESPONSE to prove that it really is the right server.

IAC SB AUTHENTICATION REPLY
KERBEROS_V4 CLIENT|MUTUAL
RESPONSE yy yy yy yy yy yy yy
IAC SE

The following is an example of use of the option with encryption negotiated via telnet ENCRYPT:

<table>
<thead>
<tr>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAC WILL AUTHENTICATION</td>
<td>IAC DO AUTHENTICATION</td>
</tr>
<tr>
<td>[ The server is now free to request authentication information. ]</td>
<td></td>
</tr>
<tr>
<td>IAC SB AUTHENTICATION SEND</td>
<td>IAC SE</td>
</tr>
<tr>
<td>KERBEROS_V4</td>
<td>IAC SB AUTHENTICATION IS</td>
</tr>
<tr>
<td>ENCRYPT_USING_TELOPT</td>
<td>CLIENT</td>
</tr>
<tr>
<td>AUTH 4 7 1 67 82 65 89 46 67 7 9</td>
<td>ENCRYPT_USING_TELOPT</td>
</tr>
<tr>
<td>77 0 48 24 49 244 109 240 50 208</td>
<td>ENCRYPT_USING_TELOPT</td>
</tr>
<tr>
<td>43 35 25 116 104 44 167 21 201</td>
<td>ENCRYPT_USING_TELOPT</td>
</tr>
<tr>
<td>224 229 145 20 2 244 213 220 33</td>
<td>[ The server has requested mutual Kerberos authentication, but is willing to do just one-way Kerberos authentication. In both cases it is willing to encrypt the data stream. The client will now respond with the name of the user that it wants to log in as, and the Kerberos ticket. ]</td>
</tr>
</tbody>
</table>
| 134 148 4 251 249 233 229 152 77 | IAC SB AUTHENTICATION NAME "joe"
| 2 109 130 231 33 146 190 248 1 9 | IAC SE
| 31 95 94 15 120 224 0 225 76 205 | IAC SB AUTHENTICATION IS
| 70 136 245 190 199 147 155 13 | KERBEROS_V4
| ENCRYPT_USING_TELOPT|CLIENT|MUTUAL | KERBEROS_V4 |CLIENT|ONE_WAY |
| IAC SE | ENCRYPT_USING_TELOPT|CLIENT|MUTUAL | KERBEROS_V4 |CLIENT|ONE_WAY |
| [ The server responds with an ACCEPT command to state that the authentication was successful. ] | IAC SB AUTHENTICATION REPLY
| KERBEROS_V4 | CLIENT|MUTUAL|ENCRYPT_USING_TELOPT
| ACCEPT IAC SE | [ Next, the client sends across a CHALLENGE to verify that it is

Expires January 2000
really talking to the right server. ]
IAC SB AUTHENTICATION IS
KERBEROS_V4
CLIENT|MUTUAL|ENCRYPT_USING_TELOPT
CHALLENGE xx xx xx xx xx xx xx
xx IAC SE
[ The server sends across a RESPONSE to prove that it really is
the right server. ]
IAC SB AUTHENTICATION REPLY
KERBEROS_V4
CLIENT|MUTUAL|ENCRYPT_USING_TELOPT
RESPONSE yy yy yy yy yy yy yy
IAC SE
[ At this point, the client and server begin to negotiate the
telnet ENCRYPT option in each direction for a secure channel.
If the option fails in either direction for any reason the
connection must be immediately terminated. ]

The following is an example of use of the option with integrated en-
cryption:

Client                           Server
IAC DO AUTHENTICATION
IAC WILL AUTHENTICATION
[ The server is now free to request authentication information. ]
IAC SB AUTHENTICATION SEND
KEA_SJ
CLIENT|MUTUAL|ENCRYPT_AFTER_EXCHANGE
IAC SE
[ The server has requested mutual KEA authentication with
SKIPJACK encryption. The client will now respond with the name
of the user that it wants to log in as and the KEA cert. ]
IAC SB AUTHENTICATION NAME "joe"
IAC SE IAC SB AUTHENTICATION IS
KEA_SJ
CLIENT|MUTUAL|ENCRYPT_AFTER_EXCHANGE
[ The server responds with its KEA Cert. ]
IAC SB AUTHENTICATION REPLY
KEA_SJ
CLIENT|MUTUAL|ENCRYPT_AFTER_EXCHANGE
IAC SE
[ Next, the client sends across a CHALLENGE to verify that it is
really talking to the right server. ]
IAC SB AUTHENTICATION IS KEA_SJ
CLIENT|MUTUAL|ENCRYPT_AFTER_EXCHANGE
'3' IV'||Encrypt(
NonceB+1||NonceA ) IAC SE
[ At this point, the client begins to encrypt the outgoing data
stream, and the server, after receiving this command, begins to

Expires January 2000                [Page 11]
decrypt the incoming data stream. Lastly, the server sends across a RESPONSE to prove that it really is the right server.

```
IAC SB AUTHENTICATION REPLY
KEA_SJ
CLIENT|MUTUAL|ENCRYPT_AFTER_EXCHANGE
IAC SE
```

[ At this point, the server begins to encrypt its outgoing data stream, and the client, after receiving this command, begins to decrypt its incoming data stream. ]

It is expected that any implementation that supports the Telnet AUTHENTICATION option will support all of this specification.

9. Security Considerations

This memo describes a general framework for adding authentication and encryption to the telnet protocol. The actual authentication mechanism is described in the authentication suboption specifications, and the security of the authentication option is dependent on the strengths and weaknesses of the authentication suboption.

It should also be noted that the negotiation of the authentication type pair is not protected, thus allowing an attacker to force the result of the authentication to the weakest mutually acceptable method. (For example, even if both sides of the negotiation can accept a "strong" mechanism and a "40-bit" mechanism, an attacker could force selection of the "40-bit" mechanism.) An implementation should therefore only accept an authentication mechanism to be negotiated if it is willing to trust it as being secure.

10. References


Author’s Address

Theodore Ts’o, Editor
VA Linux Systems
43 Pleasant St.
Medford, MA 02155
Phone: (781) 391-3464
EMail: tytso@valinux.com

Jeffrey Altman
Columbia University
Watson Hall Room 716