

## Telnet Encryption: DES 64 bit Cipher Feedback

### Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

### Copyright Notice

Copyright (C) The Internet Society (2000). All Rights Reserved.

### Abstract

This document specifies how to use the DES encryption algorithm in cipher feedback mode with the telnet encryption option.

### 1. Command Names and Codes

#### Encryption Type

DES_CFB64	1
-----------	---

#### Suboption Commands

CFB64_IV	1
CFB64_IV_OK	2
CFB64_IV_BAD	3

### 2. Command Meanings

```
IAC SB ENCRYPT IS DES_CFB64 CFB64_IV <initial vector> IAC SE
```

The sender of this command generates a random 8 byte initial vector, and sends it to the other side of the connection using the CFB64\_IV command. The initial vector is sent in clear text. Only the side of the connection that is WILL ENCRYPT may send the CFB64\_IV command.

```
IAC SB ENCRYPT REPLY DES_CFB64 CFB64_IV_OK IAC SE  
IAC SB ENCRYPT REPLY DES_CFB64 CFB64_IV_BAD IAC SE
```

The sender of these commands either accepts or rejects the initial vector received in a CFB64\_IV command. Only the side of the connection that is DO ENCRYPT may send the CFB64\_IV\_OK and CFB64\_IV\_BAD commands. The CFB64\_IV\_OK command MUST be sent for backwards compatibility with existing implementations; there really isn't any reason why a sender would need to send the CFB64\_IV\_BAD command except in the case of a protocol violation where the IV sent was not of the correct length (i.e., 8 bytes).

### 3. Implementation Rules

Once a CFB64\_IV\_OK command has been received, the WILL ENCRYPT side of the connection should do keyid negotiation using the ENC\_KEYID command. Once the keyid negotiation has successfully identified a common keyid, then START and END commands may be sent by the side of the connection that is WILL ENCRYPT. Data will be encrypted using the DES 64 bit Cipher Feedback algorithm.

If encryption (decryption) is turned off and back on again, and the same keyid is used when re-starting the encryption (decryption), the intervening clear text must not change the state of the encryption (decryption) machine.

If a START command is sent (received) with a different keyid, the encryption (decryption) machine must be re-initialized immediately following the end of the START command with the new key and the initial vector sent (received) in the last CFB64\_IV command.

If a new CFB64\_IV command is sent (received), and encryption (decryption) is enabled, the encryption (decryption) machine must be re-initialized immediately following the end of the CFB64\_IV command with the new initial vector, and the keyid sent (received) in the last START command.

If encryption (decryption) is not enabled when a CFB64\_IV command is sent (received), the encryption (decryption) machine must be re-initialized after the next START command, with the keyid sent (received) in that START command, and the initial vector sent (received) in this CFB64\_IV command.

#### 4. Algorithm

Given that  $V[i]$  is the initial 64 bit vector,  $V[n]$  is the  $n$ th 64 bit vector,  $D[n]$  is the  $n$ th chunk of 64 bits of data to encrypt (decrypt), and  $O[n]$  is the  $n$ th chunk of 64 bits of encrypted (decrypted) data, then:

```
V[0] = DES(V[i], key)
O[n] = D[n] <exclusive or> V[n]
V[n+1] = DES(O[n], key)
```

#### 5. Integration with the AUTHENTICATION telnet option

As noted in the telnet ENCRYPTION option specifications, a keyid value of zero indicates the default encryption key, as might be derived from the telnet AUTHENTICATION option. If the default encryption key negotiated as a result of the telnet AUTHENTICATION option contains less than 8 bytes, then the DES\_CFB64 option must not be offered or used as a valid telnet encryption option. If the encryption key negotiated as a result of the telnet AUTHENTICATION option is greater than 16 bytes the first 8 bytes of the key should be used as keyid 0 for data sent from the telnet client to the telnet server, and the second 8 bytes of the key should be used as keyid 0 for data sent by the telnet server to the telnet client. Otherwise, the first 8 bytes of the encryption key is used as keyid zero for the telnet ENCRYPTION option in both directions (with the client as WILL ENCRYPT and the server as WILL ENCRYPT).

In all cases, if the key negotiated by the telnet AUTHENTICATION option was not a DES key, the key used by the DES\_CFB64 must have its parity corrected after it is determined using the above algorithm.

Note that the above algorithm assumes that it is safe to use a non-DES key (or part of a non-DES key) as a DES key. This is not necessarily true of all cipher systems, but we specify this behaviour as the default since it is true for most authentication systems in popular use today, and for compatibility with existing implementations. New telnet AUTHENTICATION mechanisms may specify alternative methods for determining the keys to be used for this cipher suite in their specification, if the session key negotiated by that authentication mechanism is not a DES key and where this algorithm may not be safely used.

## 6. Security Considerations

Encryption using Cipher Feedback does not ensure data integrity; the active attacker has a limited ability to modify text, if he can predict the clear-text that was being transmitted. The limitations faced by the attacker (that only 8 bytes can be modified at a time, and the following 8-byte block of data will be corrupted, thus making detection likely) are significant, but it is possible that an active attacker still might be able to exploit this weakness.

The tradeoff here is that adding a message authentication code (MAC) will significantly increase the number of bytes needed to send a single character in the telnet protocol, which will impact performance on slow (i.e. dialup) links.

## 7. Acknowledgments

This document was originally written by Dave Borman of Cray Research with the assistance of the IETF Telnet Working Group.

### Author's Address

Theodore Ts'o, Editor  
VA Linux Systems  
43 Pleasant St.  
Medford, MA 02155

Phone: (781) 391-3464  
EMail: tytso@mit.edu

## Full Copyright Statement

Copyright (C) The Internet Society (2000). 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.

