Network Working Group
Request for Comments: 2840
Category: Informational

J. Altman
F. da Cruz
Columbia University

May 2000

TELNET KERMIT OPTION

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

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

ABSTRACT

This document describes an extension to the Telnet protocol to allow the negotiation, coordination, and use of the Kermit file transfer and management protocol over an existing Telnet protocol connection.

CONTENTS

1. MOTIVATION

The Kermit protocol [KER] performs error-corrected file transfer and management over many types of connections, including terminal connections, among diverse hardware and software platforms. It is supported by a large number of Telnet clients and is also widely available on the Internet hosts to which Telnet connections are made.

Traditionally, the Kermit protocol connection is started manually by a user, or perhaps by an automated script. It is the user's responsibility to start the Kermit server on one end of the connection and the Kermit client on the other, or to start a Kermit "send" operation on one end and a Kermit "receive" on the other.

This procedure grew out of necessity on ordinary direct-dial connections, and serves its purpose within the limitations of that context. But it introduces timing and dexterity problems, and lacks an effective way for each Kermit program to determine the "mode" of the other, or even its very presence, and therefore to know with certainty which operations and procedures are legal on the connection at any given time.

When Kermit services are offered on the Internet, however, a strong coupling can be established between the two end applications by having the Telnet protocol [TEL] serve as a supervisor for Kermit sessions, ensuring that a valid and known relationship is always obtained. Kermit sessions are, in effect, embedded within Telnet sessions, with Telnet providing the mechanism for starting and stopping them and defining which end is the Kermit client and which is the Kermit server, possibly changing the relationship in response to user actions.

2. DEFINITIONS

Kermit server
: A software program that is ready to accept and act upon commands in the form of well-defined Kermit packets [KER].

Kermit client
: A software program that receives requests through its user interface from a human agent (or a script or other source) and translates them to command packets, which it sends to a Kermit server, thus initiating a Kermit protocol transaction such as the transfer of one or more files.

Availability of Kermit server
For the purposes of this document, a Kermit server is said to be available if, through the negotiations described herein, its Telnet partner knows that it is a Kermit server.

3. COMMANDS AND CODES

Support for a Kermit server is negotiated separately in each direction, allowing Kermit service to be embedded in the Telnet client, the Telnet server, or in both. The proposed Telnet extensions are, therefore, symmetrical.

When the connection is first opened, Kermit service is unavailable in both directions.

The availability of Kermit service is negotiated using the following Telnet option:

KERMIT	47 (assigned by IANA)

The state of the connection is controlled by the following Telnet subnegotiation function codes:

START-SERVER	0
STOP-SERVER	1
REQ-START-SERVER	2
REQ-STOP-SERVER	3
SOP	4
RESP-START-SERVER	8
RESP-STOP-SERVER	9

4. COMMAND MEANINGS

The KERMIT OPTION is negotiated using the standard Telnet mechanisms:

IAC WILL KERMIT
The sender of this command incorporates a Kermit server and is willing to negotiate its use.

IAC WONT KERMIT
The sender of this command does not incorporate a Kermit server or refuses to negotiate its use.

IAC DO KERMIT
The sender of this command requests that the receiver negotiate use of a Kermit server.

IAC DONT KERMIT
The sender of this command refuses to negotiate the use of a Kermit server.

Once WILL KERMIT is negotiated in a particular direction, subnegotiations are used to indicate or request a change in state of the connection, or to convey other information. Subnegotiations may be sent at any time.

IAC SB KERMIT START-SERVER
This command is sent by the WILL side to indicate that the Kermit server is now active; that is, that client-initiated Kermit packets will be accepted.

IAC SB KERMIT STOP-SERVER
This command is sent by the WILL side to indicate that the Kermit server is no longer active, and therefore that it is not ready to accept Kermit packets.

IAC SB KERMIT REQ-START-SERVER
This command is sent by the DO side to request that the Kermit server be started. It must be responded to with either RESP-START-SERVER or RESP-STOP-SERVER depending upon whether the request was accepted.

IAC SB KERMIT REQ-STOP-SERVER
This command is sent by the DO side to request that the Kermit server be stopped. It must be responded to with either RESP-START-SERVER or RESP-STOP-SERVER depending upon whether the request was accepted.

IAC SB KERMIT RESP-START-SERVER
This command is sent by the WILL side in response to REQ-START-SERVER or REQ-STOP-SERVER to indicate that the Kermit server is active after the request was accepted or denied.

IAC SB KERMIT RESP-STOP-SERVER
This command is sent by the WILL side in response to REQ-START-SERVER or REQ-STOP-SERVER to indicate that the Kermit server is not active after the request was accepted or denied.

IAC SB KERMIT SOP <octet>
Kermit Start Of Packet. The sender of this command specifies the octet it will use to mark the beginning of the Kermit packets it sends. This command must be sent by each connection partner upon the first WILL/DO pair to allow unambiguous identification of Kermit packets in the data stream. This subnegotiation must be sent whenever the Start of Packet character changes. The values

are restricted to ASCII C0 control characters other than Carriage Return and NUL. The normal value is 1 (ASCII SOH). The two Kermit partners normally use the same SOP, but may use distinct ones if desired.

IAC SB KERMIT SOP is necessary to allow each Telnet partner to recognize subsequent incoming Kermit packets. Data following the SOP is processed by the Kermit packet analyzer. All other Kermit protocol parameters are automatically negotiated within the Kermit protocol upon the initial exchange of Kermit packets [KER].

START-SERVER and STOP-SERVER commands must be sent by the WILL side whenever the state of the Kermit server changes. When WILL is successfully negotiated the state of the WILL side is assumed to be STOP-SERVER. If the server is active, the WILL side must send a START-SERVER to indicate the change in state.

The receiver of a REQ-START-SERVER or REQ-STOP-SERVER is not required to agree to the request to change state. The receiver must respond with either RESP-START-SERVER or RESP-STOP-SERVER to indicate the state of the Kermit Server subsequent to the request. RESP-xxx-SERVER is sent instead of xxx-SERVER to enable the sender of REQ-xxx-SERVER to distinguish between the WILL side's spontaneous change in state and the response to the DO side's request.

If the Kermit server receives a Kermit packet commanding it to cease Kermit service (such as a FINISH, REMOTE EXIT or BYE packet [KER]), it must send IAC SB KERMIT STOP-SERVER if the command is accepted.

These rules ensure that the Telnet client's user interface always knows whether (and on which end) a Kermit server is available, and can therefore present the user only with valid choices, and that changes in state of one Telnet partner automatically switch the other to a complementary and valid state.

While it is possible for a traditional telnet service (port 23) to implement this option while at the same time supporting the existing remote shell access functionality, it is not expected that this option will be used in that manner. Instead, this option is primarily meant for use with dedicated Kermit services such as the Internet Kermit Service (port 1649) [IKS].

5. KERMIT PROTOCOL IMPLICATIONS

The Kermit protocol is described elsewhere [KER]. It is an extensible and self-configuring protocol, like Telnet, and thus any two proper Kermit implementations should interoperate automatically.

In Kermit, as in Telnet, one particular octet is distinguished. In Telnet's case, it is IAC (decimal 255); in Kermit's it is the character specified by the IAC SB KERMIT SOP negotiation, normally SOH (decimal 1, Ctrl-A). All Kermit packets must begin with the SOP and should not contain the SOP character in an unquoted form.

Telnet protocol takes precedence over Kermit protocol; whenever an IAC is detected, it is processed as the beginning of a Telnet command unless quoted by another IAC. Telnet commands can contain any characters at all, including the SOP octet, transparently to the Kermit protocol, and in fact Telnet commands are not seen by the Kermit protocol at all.

Kermit protocol must follow Telnet NVT rules in each direction when Telnet binary mode is not negotiated for that direction.

If 8-bit transparency is desired, Telnet binary mode may be negotiated upon entry to Kermit protocol in the appropriate direction, and the previous mode (NVT or binary) restored upon exit from Kermit protocol. Telnet binary mode can result in more efficient transfers, but is not required for data transfer, since Kermit protocol does not require a transparent path.

6. EXAMPLES

6.1. EXAMPLE 1

The Telnet server contains a Kermit server. The Telnet client includes Kermit protocol but does not implement the Telnet KERMIT Option.

```
Telnet Server                  Telnet Client
-----------------------------  -----------------------------
<starts negotiations>
WILL KERMIT
DO KERMIT
                               <responds to negotiations>
                               DONT KERMIT
                               WONT KERMIT
```

From this point, no subnegotiations take place, and the Kermit client/server relationship is under manual control of the user of the Telnet client.

6.2. EXAMPLE 2

The Telnet server contains a Kermit server and starts a Kermit server immediately after a connection is made. The Telnet client does not offer a Kermit server.

Telnet Server	Telnet Client
<starts negotiations>	
WILL KERMIT	
DO KERMIT	
	<responds to negotiations>
	DO KERMIT
	SB KERMIT SOP <0x01>
	WONT KERMIT
SB KERMIT SOP <0x01>	
<starts Kermit Server>	
SB KERMIT START-SERVER	

At this point the Telnet client knows that a Kermit server is on the other end of the connection, and so may customize its command set or menus to allow only those commands that are valid as a client of a Kermit server.

6.3. EXAMPLE 3

Telnet server and Telnet client both contain a Kermit server. Telnet client Kermit server is active whenever its terminal emulator is active, and not active at other times. The Telnet server is used for shell access and does not start a Kermit Server unless requested.

Telnet Server	Telnet Client
<starts negotiations>	
WILL KERMIT	
DO KERMIT	
	<responds to negotiations>
	DO KERMIT
	SB KERMIT SOP <0x01>
	WILL KERMIT
SB KERMIT SOP <0x01>	
	<telnet client enters terminal emulator>
	SB KERMIT START-SERVER
	<client leaves terminal emulator>
	SB KERMIT STOP-SERVER
	<client requests Kermit service>
	SB KERMIT REQ-START-SERVER
<starts Kermit server>	
SB KERMIT RESP-START-SERVER	
	<client sends Kermit FINISH packet>
<stops Kermit server>	
SB KERMIT STOP-SERVER	
	<client returns to terminal emulator>
	SB KERMIT START-SERVER

6.4. EXAMPLE 4

Telnet server and Telnet client both contain a Kermit server. Telnet client's Kermit server is active whenever the terminal emulator is active. Telnet server is used solely for Kermit protocol and automatically starts a Kermit Server upon accepting the connection.

```
Telnet Server                Telnet Client
-------------------------    -----------------------------
<starts negotiations>
WILL KERMIT
DO KERMIT
                             <responds to negotiations>
                             DO KERMIT
                             SB KERMIT SOP <0x01>
                             WILL KERMIT

SB KERMIT SOP <0x01>
                             <client enters terminal emulator>
                             SB KERMIT START-SERVER

<in response to DO>
SB KERMIT SOP <0x01>
SB KERMIT START-SERVER
                             <client restricts command set to
                              Kermit protocol commands>
                             SB KERMIT STOP-SERVER

                             <client performs Kermit protocol
                              operations>

                             <client want to enter terminal mode>
                             SB KERMIT REQ-STOP-SERVER

<Kermit Server refuses>
SB KERMIT RESP-START-SERVER
```

6.5. EXAMPLE 5

This is an example of something that should not be allowed to happen. Some Telnet clients that implement file transfer capabilities are designed to accept incoming connections. In this situation the Telnet Client acts as a pseudo Telnet Server but without the ability to provide shell access or many of the other functions associated with Telnet. If both Telnet clients support this option and contain a Kermit server that is active during terminal emulation there is the potential for a deadlock situation if scripting is also supported. This is because Telnet clients that support a script language do not process input while waiting for the next command to be issued.

```
Telnet Client One             Telnet Client Two
--------------------------    ------------------------------
<starts negotiations>
WILL KERMIT
DO KERMIT
                              <responds to WILL>
                              DO KERMIT
                              SB KERMIT SOP <0x01>

<in response to DO>
SB KERMIT SOP <0x01>
SB KERMIT START-SERVER
                              <responds to DO>
                              WILL KERMIT
                              SB KERMIT START-SERVER

<client one restricts command
 set to Kermit protocol and
 disables Kermit Server>
SB KERMIT STOP-SERVER
                              <client two restricts command
                               set to Kermit protocol and
                               disables Kermit Server>
                              SB KERMIT STOP-SERVER
```

At this point both clients have restricted their command set to Kermit Protocol commands. However, in both cases neither side is processing input. Therefore the following restriction MUST be enforced: A Telnet partner may not restrict the command set if it accepted the incoming connection.

7. SECURITY

Implementors of this Telnet Option must enforce appropriate user authentication and file system access restrictions in conjunction with their implementation of the Kermit file transfer protocol. These issues are beyond the scope of this document.

8. REFERENCES

[BCP] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[KER] da Cruz, Frank, "Kermit, A File Transfer Protocol", Digital Press/ Butterworth Heinemann, Newton, MA, ISBN 0-932376-88-6 (1987).

[IKS] da Cruz, F. and J. Altman, "Internet Kermit Service", RFC 2839, May 2000.

[TEL] Postel, J. and J. Reynolds, "Telnet Protocol Specification", STD 8, RFC 854, May 1983.

[TEL] Postel, J. and J. Reynolds, "Telnet Option Specification", STD 8, RFC 855, May 1983.

9. AUTHORS' ADDRESSES

Jeffrey E. Altman

EMail:jaltman@columbia.edu

Frank da Cruz

EMail: fdc@columbia.edu

The Kermit Project
Columbia University
612 West 115th Street
New York NY 10025-7799
USA
http://www.columbia.edu/kermit/
http://www.kermit-project.org/

10. Full Copyright Statement

Acknowledgement

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