Jason Herne **02/12/04**

z/VM Telnet Interaction Study

The purpose of this exercise is to examine telnet session trace to a z/VM instance in order to learn the underlying protocols that will allow the COSI z/VM GUI team to write software to communicate with z/VM via a telnet session.

While capturing packets with Ethereal, I connected to z/VM via port 23 and executed the following commands:

l maint pass
q n
log

The results of the capture were saved and examined.

Communication Observed

Here is a transcription of the communications that took place. The data that actually was transferred (in some cases) is written in terms of Telnet commands for clarity.

<-- = Data sent from host to z/VM
--> = Data sent from z/VM to host

--> Do TN3270E

<-- Do Suppress Go Ahead
Will Terminal Type
Will Terminal Speed
Will Remote Flow Control
Will Linemode
Will New Environment Option
Do Status
Will X Display Location

<-- Wont TN3270E

--> Will Suppress Go Ahead
Do Suppress Go Ahead
Do Terminal Type

Sub begin: Terminal Type
Send your Terminal Type
Sub end:
Don't Negotiate About Window Size
Don't Terminal Speed
Don't Remote Flow Control
Don't linemode
Don't New Environment Option
Won't Status
Don't X Display Location

<-- Will Suppress Go Ahead
Sub begin: Terminal Type
Here's my Terminal Type: xterm
Sub end:

--> Sub begin: Terminal Type
Send your Terminal Type
Sub end:

<-- Sub begin: Terminal Type
Here's my Terminal Type: xterm
Sub end:

--> Will Suppress Go Ahead

--> “z/VM ONLINE --ZVMV4R30--PRESS BREAK KEY TO BEGIN SESSION”

--> .\021

<-- \r\000

--> \r\n
\023\177

--> .\021

<-- l
<--
<-- m
<-- a
<-- i
<-- n
<-- t
<--

```
<--    p
<--    a
<--    s
<--    s
<--    \r\000

-->    \r\n
       \023\177

-->    10:54:51 HCPLNM102E DASD 0123 forced R/O; R/W by
       DIRMAINT \a\a\a\a\r\n
       \023\177

-->    10:54:51 z/VM Version 4 Release 3.0, Service Level 0000 (64-
       bit), \r\n
       \023\17710:54:51 built on IBM Virtualization Technology \r\n
       \023\177

-->    10:54:51 There is no logmsg data\r\n
       \023\177

-->    10:54:51 FILES: 0016 RDR, 0003 PRT, NO PUN\r\n
       \023\17710:54:51 LOGON AT 10:54:51 EST THURSDAY
       02/12/04\r\n
       \023\17710:54:51 TCPIP0 LOGON AS MAINT USERS=17\r\n
       \023\177

-->    z/VM V4.3.0  2002-04-04  10:28\r\n
       \023\177

-->    .\021

<--    q
<--
<--    n
<--    \r\000

-->    \r\n
       \023\177
```

And communications continues in similar fashion. I stopped translation at this point because nothing new was being learned.

Examination Results

1) Messages like WILL, WONT, DONT, and DO are used at the beginning of the session to set up exactly how the communication will take place.
2) Each line of returned Data is ended by a \r\n\023\177
3) A \021 ends each Data Message sent.

Things I still don't know.
1. This is a \r \n \023 \177 and \021?
2. How to work with Do, Won't, Don't and Do options.

Examination of RFC 854

Lets turn to RFC 854 “Telnet Protocol Specification” for possible answers to these questions.

Notes about Telnet options

- An option may only be requested.. not announced.. This means that if I am in mode X I cannot sent a request to change to mode X.
- A host may not answer a request to change to Mode X if he is already in Mode X.
- All other requests should always be answered.
- A sub option is used to further refine a configuration option or to give extra data, but only after the basic syntax has been fully enabled. After that, the sub options can be sent that depend on that option.
- May refuse unwanted and/or unknown options by sending a rejection to that option.

WILL XXXX	- Request permission to start XXXX or confirms it is beginning XXXX.
WONT XXXX	- Refusal to comply, or to continue to comply with XXXX.
DO XXXX	- Requests XXX from other side, or confirms that it is expected that XXX should be enabled.
DON'T XXXX	- Refusal to comply, or to continue to comply with XXXX.

Go Ahead option – This is used to transfer sending control from end to end on a half duplex connection. We will not need this functionality and can always request the “Suppress Go Ahead” option.

The characters that we encountered earlier are in octal (except the \r and \n which are canonical representations).

ASCII/Octal	Function	ASCII/DEC
\r	CR	13
\n	LF	10
\023	DC3	19
\021	DC1	17
\177	DEL	127

Examining Options From Our Trace

1. Suppress Go Ahead

This is defined above and can be requested in all cases.

2. Terminal Type: xterm

I can't find any detailed information on this but it seems as though it simply sets which characters are sent to mean what. We will use “xterm” since the data we got while using xterm seems very easy to work with.

Plan of Action

I think we have enough information to start an interaction.

I will connect and send:

Will Terminal Type
Do Suppress Go Ahead

If answered with “Do Terminal Type” I'll answer:

Sub begin: Terminal Type
Here's my Terminal Type: xterm
Sub end:

If answered with

Will Suppress Go Ahead

I'll answer with (only if “Suppress Go Ahead” is not already confirmed) After answering this, it will be confirmed:

Will Suppressed Go Ahead

If answered with

Do Suppress Go Ahead:

I'll answer will:

Will Suppress Go Ahead

If I get Data , I'll parse that data as needed. \r\n will be treated as a newline and \023\177 will be ignored. A \021 will be treated as an EOR (End Of Record) symbol which alerts me that all of the data for this particular message has been received.

Specific details regarding the encoding/decoding of all messages and options will be taken from RFC 854 Telnet Protocol Specification.