Using a Proxy with vSphere Virtual Serial Ports

VMware vSphere 7.0
vCenter Server 7.0
VMware ESXi 7.0

vmware®

You can find the most up-to-date technical documentation on the VMware website at:

https://docs.vmware.com/

If you have comments about this documentation, submit your feedback to

docfeedback@vmware.com

Contents

About This Book

The VMware® *Using a Proxy with vSphere Virtual Serial Ports* manual provides information about developing a virtual serial port concentrator for use in the vSphere environment.

VMware provides different APIs and SDKs for various applications and goals. This manual describes the capabilities of a proxy implementation that supports the VMware telnet extension. It also includes information about configuring a virtual serial port for a proxy connection.

To view the current version of this book as well as all VMware API and SDK documentation, go to http://www.vmware.com/support/pubs/sdk_pubs.html.

Revision History

This book is revised with each release of the product or when necessary. A revised version can contain minor or major changes. Table 1-1. Revision History summarizes the significant changes in each version of this book.

Table 1-1. Revision History

Revision Date	Description
02 APR 2020	Support for VMware vSphere 7.0
26 APR 2016	Support for VMware vSphere 6.5
12 MAR 2015	Support for VMware vSphere 6.0
11 MAR 2014	Supported on VMware vSphere 6.0 Beta.
19 SEP 2013	Supported in VMware vSphere 5.5; added information about TCP keepalive configuration and time limit for VMOTION-GOAHEAD response (vmotionLimit).
10 SEP 2012	Supported in VMware vSphere 5.1; added syntax for authentication parameters used with encrypted telnet and telnets connections.
24 AUG 2011	Supported in VMware vSphere 5.0.
13 JUL 2010	Initial release, supported in VMware vSphere 4.1.

Intended Audience

This book is intended for anyone developing a virtual serial port proxy for the vSphere environment. Proxy developers should have experience with the telnet protocol and network communication.

VMware Technical Publications Glossary

VMware Technical Publications provides a glossary of terms that might be unfamiliar to you. For definitions of terms as they are used in VMware technical documentation go to http://www.vmware.com/support/pubs.

Document Feedback

VMware welcomes your suggestions for improving our documentation. Send your feedback to docfeedback@vmware.com.

Remote Serial Port Access with a Virtual Serial Port Proxy

1

The VMware vSphere™ API supports the use of virtual serial ports on virtual machines that run on ESX™ hosts. A virtual serial port represents a serial port on a virtual machine. A virtual serial port can operate as a network serial port by using a network socket on the host to provide access to remote systems. A remote system is any system on the network that supports a serial port connection with a virtual machine. This feature supports remote console login for system management using an out-of-band connection that bypasses the vSphere Client.

You can also use a virtual serial port for communication through a software pipe, for output to a file, and for communication through a physical serial port on the host. For information about the vSphere API support for these applications, see the description of the `VirtualSerialPort` object in the *vSphere API Reference*.

In a VMware® data center, a vCenter Server uses vMotion™ to move virtual machines across hosts as part of resource management. vSphere does not support vMotion operations on a virtual machine that has a direct network serial port connection with a remote system. Network serial ports use the host IP address. If the server attempts to move a virtual machine with a network serial port connection, the host rejects the vMotion request to maintain the connection.

To support vMotion for virtual machines that use network serial ports, you can develop a proxy that operates between a virtual machine and a remote system. The vSphere API supports the use of a third-party virtual serial port concentrator in the data center environment. A virtual serial port concentrator acts as a proxy or an access server between virtual machines and remote systems. The proxy maintains communication with virtual machines that are using network serial ports. When a vMotion event occurs, the virtual machine connects to the proxy from its new host so that the connection with the remote system persists.

Virtual machines use the telnet protocol for virtual serial port proxy connections. VMware defines a telnet extension that includes commands for proxy notification of vMotion events.

See Chapter 2 VMware Telnet Extension Commands for Proxy Communication for reference information about the VMware telnet extension commands.

This chapter includes the following topics:

- Virtual Serial Port Proxy Capabilities

- Network Serial Ports

- VMware Telnet Session

- vMotion and Persistent Serial Port Connections

Virtual Serial Port Proxy Capabilities

A virtual serial port proxy operates as an RFC2217-compatible telnet server. A proxy provides the following capabilities in the vSphere environment.

- Accepts telnet connections from virtual machines

- Supports VMware telnet extension commands and responds to VMware telnet extension messages from virtual machines

- Forwards network traffic between virtual machines and remote systems

- Handles extension commands for notification of vMotion events and redirects traffic accordingly

Proxy as a Virtual Serial Port Concentrator

Figure 1-1. Virtual Serial Port Concentrator shows a proxy that operates as a virtual serial port concentrator. The concentrator has internal connections to several virtual machines in a data center and an external connection to a remote system.

The concentrator provides a remote system with access to multiple virtual machines that act as serial port servers. The virtual machines connect to the concentrator, indicating that they will accept connections from remote systems. See Backing Options and Backing Information for Network Serial Ports for more information about serial port servers. The concentrator presents the choice of virtual machines to the remote system and maintains any subsequent connection. In Figure 1-1. Virtual Serial Port Concentrator, a solid line represents an active connection between the remote system and a virtual machine, and dashed lines represent potential connections.
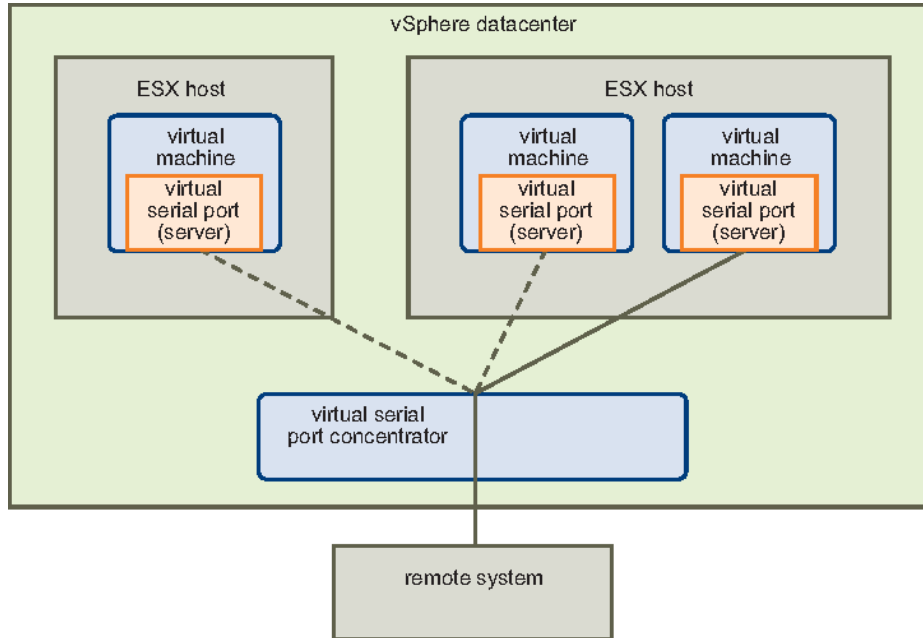
Figure 1-1. Virtual Serial Port Concentrator

Figure 1-1. Virtual Serial Port Concentrator shows the concentrator as part of the vSphere data center. For this kind of application, you can deploy the concentrator as a virtual appliance. A virtual appliance is software on a platform of one or more virtual machines (in this case, a single virtual machine). With a virtual appliance implementation, you can take advantage of resource management that vCenter Server provides. The vCenter Server uses vMotion technology to migrate the virtual appliance proxy through the data center without interrupting remote serial port service. The virtual appliance uses virtual machine IP addresses that persist across vMotion events.

Although the primary purpose of the proxy is to preserve connections during vMotion events, it also can provide a gateway or firewall to enforce security. The proxy serves as the point of authentication for remote systems that use virtual serial ports to access virtual machines.

Network Serial Ports

A network serial port is a virtual serial port that uses a network socket on the host computer. A network socket is the endpoint of a network connection and is represented by an IP address and a port number. The vSphere API uses Uniform Resource Identifiers (URI) to identify the protocol, IP address, and port number for endpoints of network serial port connections. See RFC3986 for information about Uniform Resource Identifiers.

To use a network serial port, you must verify that the ESX host supports the feature and then create or reconfigure a virtual machine to configure the port. For information about writing an application that uses the vSphere API, see the *vSphere Web Services SDK Programming Guide*. As an alternative, you can use the vSphere Client to configure a network serial port. For more information, see the online help for the VMware vSphere Client.

Backing Options and Backing Information for Network Serial Ports

The vSphere API uses backing objects to represent the host device that is associated with a virtual device. The API offers two types of backing objects for virtual devices: option objects and information objects.

Backing option objects identify host device support for virtual machines. An ESX host defines the set of virtual device backing options that are available for virtual machines. To use a network serial port, the ESX host must provide the virtual serial port URI backing option, which is represented by the `VirtualSerialPortURIBackingOption` object. For information about how to find out if the host supports the URI backing option, see Determining Backing Options.

Backing information objects supply data for virtual device configuration. Use a `VirtualSerialPortURIBackingInfo` object to bind a virtual serial port to a network socket. You can configure a virtual serial port to use URI backing when you create or reconfigure a virtual machine. For a description of how to configure URI backing for a network serial port, see Create a Virtual Machine with a Network Serial Port.

When you specify URI backing for the network serial port connection, you define these characteristics.

- Direction – defines the role that the virtual machine plays in a connection with a remote system (client or server). When a virtual machine acts as a serial port client, it initiates the connection with the remote system. When a virtual machine acts as a serial port server, it listens for connections that are initiated by remote clients.

- Service URI – identifies the target of connections. When the virtual machine acts as a client, the service URI identifies the remote system. When the virtual machine acts as a server, the service URI identifies proxy access to the host on which the virtual machine runs. See RFC3986 for information about URI syntax.

- Proxy URI – identifies the virtual serial port proxy. The presence of the proxy URI directs the virtual machine to establish a connection with the proxy. The virtual machine always initiates the connection with the proxy.

  You can use either telnet or secure telnet (telnets) for the proxy connection. The proxy URI specification must use one of the following formats. See RFC3986 for information about URI syntax. See the description of the `VirtualSerialPortURIBackingInfo` data object in the *vSphere API Reference* for information about URI schemes and authentication parameters for network serial port configuration.

  - telnet://<host>:<port>[#key[=value][&key[=value] ...]]

  - telnets://<host>:<port>[#key[=value][&key[=value] ...]]

  When the virtual machine performs telnet negotiation with the proxy, it sends the direction and service URI information to the proxy. (See the description of the DO-PROXY command in Proxy Operation.)

Network Connections Between a Virtual Machine, a Proxy, and a Remote System

[Figure 1-2. Virtual Serial Port Connections to a Remote System Through a Proxy](#) shows two virtual machines. Each machine uses a proxy for its connection to a remote system.

- One virtual machine acts as a virtual serial port server. This virtual machine uses the service URI to identify proxy access to the host machine. The virtual machine initiates a telnet connection with the proxy. The proxy, on behalf of the virtual machine, listens for connection requests from remote systems and forwards the requests to the virtual machine.

- One virtual machine acts as a virtual serial port client. This virtual machine uses the service URI to identify the remote system. The virtual machine initiates a telnet connection with the proxy. The proxy forwards the connection request to the remote system.

The proxy has two network sockets. It uses one socket (proxy URI) for the telnet connections with the virtual machine and the other for the connection with the remote system. The external connection between the proxy and a remote system uses the mutually agreed upon protocol. For a connection initiated by the remote system, the person operating the remote system must obtain the service URI by email or other independent communication.

Figure 1-2. Virtual Serial Port Connections to a Remote System Through a Proxy
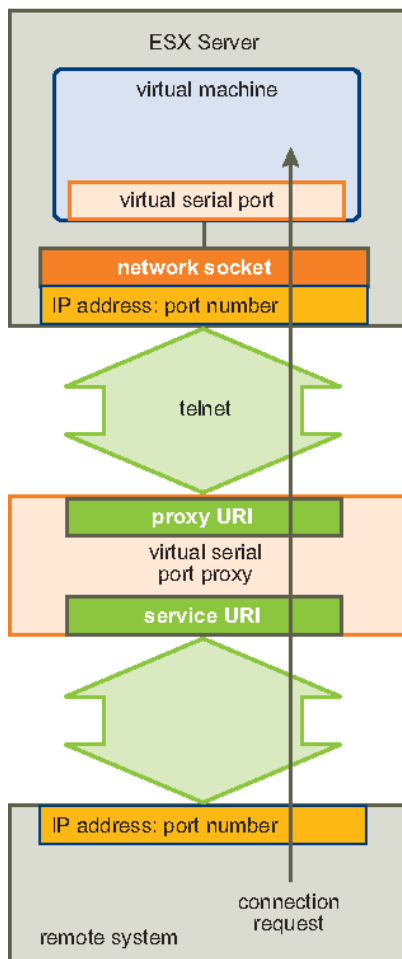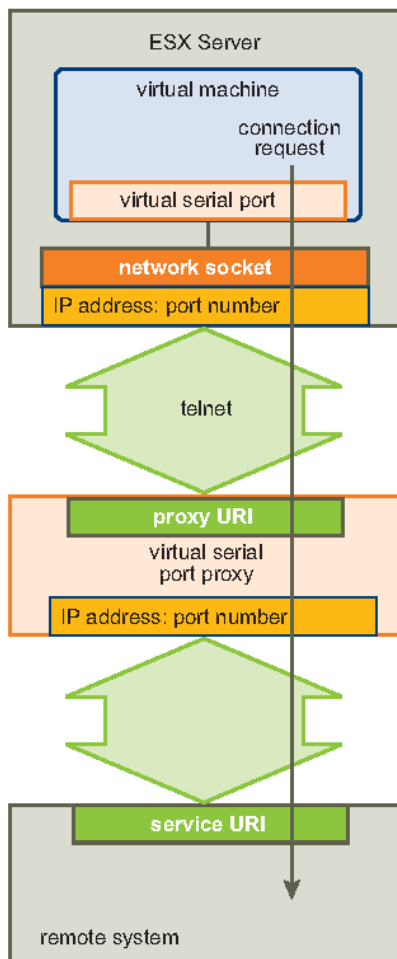
Determining Backing Options

Use the `QueryConfigOption` method to determine which backing options are available on the host on which you will create the virtual machine. The method returns a `VirtualMachineConfigOption` data object. The virtual machine configuration data includes a a list of virtual device options and a list of backing options. The following pseudocode shows the path to the backing options.

```
VirtualMachineConfigOption.hardwareOptions.VirtualDeviceOption[].backingOption[]
```

The virtual device option array must include the `VirtualSerialPortOption` data object. For network backing, the array of serial port backing options must include the `VirtualDeviceURIBackingOption` data object. The `VirtualDeviceURIBackingOption.type` property specifies the name of the backing object you should create when you configure network backing.

Create a Virtual Machine with a Network Serial Port

You can use the `CreateVM_Task` method to create a virtual machine and configure a network serial port. The virtual machine uses the virtual serial port configuration to connect to a proxy and to accept a connection from a remote system.

Procedure

1   Create a `VirtualMachineConfigSpec` object for the method's `config` parameter.

2   Create a `VirtualDeviceConfigSpec` object for the `deviceChange` array in the `VirtualMachineConfigSpec` object.

3   Create a `VirtualSerialPort` object for the `device` property in the `VirtualDeviceConfigSpec` object.

4   To control proxy connection behavior, create a `VirtualDeviceConnectInfo` data object for the `VirtualSerialPort.connectable` property.

   The following pseudocode shows the resulting path to the connection information.

   ```
   VirtualMachineConfigSpec.deviceChange[].device.connectable
   ```

5   Set the `startConnected` property in the `connectable` object to TRUE.

   The `startConnected` property determines whether the virtual machine will open a connection with the proxy when the virtual machine starts. When the virtual machine powers on, it copies the `startConnected` value to the `VirtualDeviceConnectInfo.connected` property. While the virtual machine is running, you can use the `ReconfigVM_Task` method to set the `connected` property to open or close the proxy connection. If you reconfigure a port with `connected` set to TRUE, the virtual machine closes the existing connection and opens a new one.

6 Create a `VirtualDeviceURIBackingInfo` object for the `backing` property in the `VirtualSerialPort` object.

The following pseudocode shows the resulting path to the backing information.

```
VirtualMachineConfigSpec.deviceChange[].device.backing
```

Use the backing information object to supply the following information:

- Set the `direction` property to "server" to direct the virtual machine to accept remote connections.

- Set the `serviceURI` property to the URI for the host on which the virtual machine runs.

- Set the `proxyURI` property to the URI for the virtual serial port concentrator. You cannot include a user/password sequence in the proxy URI.

VMware Telnet Session

To establish a VMware telnet session for remote serial port communication, a virtual machine and a proxy participate in a message exchange that follows the model of a standard telnet protocol sequence. The proxy handles messages from the virtual machine and sends the appropriate responses. Before the proxy can forward content, the virtual machine and the proxy participate in the following progression.

1 The virtual machine and the proxy negotiate the use of the VMware telnet extension option.

Both sides of the connection must agree to use the option before continuing with any additional extension message exchange.

2 The virtual machine and the proxy identify the VMware telnet suboption commands that they support.

The proxy must support the `KNOWN-SUBOPTIONS-2` and `UNKNOWN-SUBOPTION-RCVD-2` commands. During a session, the proxy can send the `UNKNOWN-SUBOPTION-RCVD-2` command as a valid response to virtual machine messages.

3 The proxy agrees to act as a VMware telnet extension proxy for remote serial port communication.

VMware Telnet Commands

Table lists the VMware telnet extension commands and their brief descriptions. Commands in the table are grouped by purpose: VMware telnet extension and suboptions, proxy, vMotion, and identification. The table uses command symbols. A VMware telnet extension command is a two- or three-byte code embedded in a telnet command sequence. For more information about commands and the code equivalents, see Chapter 2 VMware Telnet Extension Commands for Proxy Communication.

Command	Source	Description
VMWARE-TELNET-EXT	virtual machine, proxy	The VMware telnet extension option command
KNOWN-SUBOPTIONS-1	virtual machine	Suboptions supported by the virtual machine
KNOWN-SUBOPTIONS-2	proxy	Suboptions supported by the proxy
UNKNOWN-SUBOPTION-RCVD-1	virtual machine	Unknown suboption received by the virtual machine

Command	Source	Description
UNKNOWN−SUBOPTION−RCVD−2	proxy	Unknown suboption received by the proxy
DO−PROXY	virtual machine	Virtual machine request for proxy support
WILL−PROXY	proxy	Positive proxy response
WON'T−PROXY	proxy	Negative proxy response
VMOTION−BEGIN	virtual machine	Notification of a pending vMotion operation for a virtual machine
VMOTION−GOAHEAD	proxy	Positive proxy response
VMOTION−NOTNOW	proxy	Negative proxy response
VMOTION−PEER	virtual machine	Virtual machine request identifying the new virtual machine instance
VMOTION−PEER−OK	proxy	Proxy response accepting the new virtual machine instance
VMOTION−COMPLETE	virtual machine	Notification that vMotion operation is complete
VMOTION−ABORT	virtual machine	Notification that vMotion operation was aborted
GET−VM−VC−UUID	proxy	Proxy request for the virtual center representation of virtual machine UUID
VM−VC−UUID	virtual machine	Virtual machine response
GET−VM−NAME	proxy	Proxy request for the virtual machine name
VM−NAME	virtual machine	Virtual machine response
GET-VM-BIOS-UUID	proxy	Proxy request for the virtual machine BIOS UUID
VM-BIOS-UUID	virtual machine	Virtual machine response
GET-VM-LOCATION-UUID	proxy	Proxy request for the virtual machine location UUID
VM-LOCATION-UUID	virtual machine	Virtual machine response

Establish a VMware Telnet Extension Session

If you specify a proxy URI in the virtual serial port configuration, the virtual machine will initiate a telnet connection with the proxy. After the connection is established, the virtual machine initiates a message exchange to establish a VMware telnet extension session. The following progression describes the model for the message exchange and uses symbols to represent the codes. For the code equivalents, see the command descriptions in Chapter 2 VMware Telnet Extension Commands for Proxy Communication.

Procedure

1   Start the VMware telnet session.

- Virtual Machine – Sends a message that indicates it will send VMware telnet extension commands to the proxy.

```
IAC WILL VMWARE−TELNET−EXT
```

- Proxy – Responds that it will receive VMware telnet extension commands.

  ```
  IAC DO VMWARE-TELNET-EXT
  ```

2 Identify known suboptions.

- Virtual Machine – Sends a message that indicates the set of suboption commands that it can send and receive. The suboptions are expressed as a sequence of codes that correspond to suboption commands.

  ```
  IAC SB VMWARE-TELNET-EXT KNOWN-SUBOPTIONS-1 suboptions IAC SE
  ```

- Proxy – Responds with a message that indicates the set of suboption commands that it can send and receive.

  ```
  IAC SB VMWARE-TELNET-EXT KNOWN-SUBOPTIONS-2 suboptions IAC SE
  ```

3 Negotiate proxy support.

- Virtual Machine – Sends a request for the proxy to act as a VMware telnet extension proxy. The proxy request includes direction and URI information from the network backing for the virtual serial port. (For information about using the API to specify this information, see Create a Virtual Machine with a Network Serial Port.)

  ```
  IAC SB VMWARE-TELNET-EXT DO-PROXY direction uri IAC SE
  ```

- Proxy – Responds, indicating that it will perform proxy operations.

  ```
  IAC SB VMWARE-TELNET-EXT WILL-PROXY IAC SE
  ```

Proxy Failure Detection (TCP keepalive)

The ESX Server uses a TCP keepalive timer to determine if a serial port connection between a virtual machine and a virtual serial port proxy is still active. By default, the virtual serial port keepalive time is 300 seconds.

To configure the keepalive time for a serial port connection, set the `keepIdle` advanced option for the virtual machine. The option name uses the following format:

```
serialn.keepIdle
```

n is an integer in a sequence starting with zero, corresponding to the sequence of serial ports on the virtual machine. Examples of `keepIdle` options include `serial0.keepIdle`, `serial1.keepIdle`, and `serial2.keepIdle`.

The `keepIdle` value is the duration in seconds on an idle connection, after which the ESX Server sends up to five keepalive probes to the virtual serial port proxy. If there is no response after five probes, the connection is considered inactive. The Server uses a keepalive interval of 1/100th of the `keepIdle` value, and a retry value of five. Using the default `keepIdle` value (300 seconds), the Server will detect a proxy failure no later than 315 seconds after the last activity (300 + 5*300/100).

vMotion and Persistent Serial Port Connections

A virtual serial port proxy provides support for persistent remote serial port connections when a vCenter Server moves a virtual machine to a different host. To support the persistent connection during the vMotion operation, the proxy must support two telnet connections for the virtual machine.

A proxy buffers data while it is forwarding content from the remote system to a virtual machine. A proxy must respond to the `VMOTION-BEGIN` command, which is notification of a pending vMotion operation. ESX Server support for the VMware telnet extension allows the proxy to postpone the vMotion event until it finishes forwarding content. After sending the `VMOTION-GOAHEAD` command in response to `VMOTION-BEGIN`, the proxy must buffer any additional data it receives from the remote system. When the vMotion event is complete, the proxy continues the content transmission to the new instance of the virtual machine.

By default, the proxy must respond to a `VMOTION-BEGIN` command within 5000ms (5 seconds). If the proxy does not respond in time, the vMotion operation is terminated andthe virtual machine remains on the original server. You can specify a different time limit for the vCenter Server to wait for a `VMOTION-GOAHEAD` response from the proxy. To configure the time limit, set the `serialn.vmotionLimit` advanced option for the virtual machine. The *n* corresponds to the sequence of serial ports on the virtual machine. Specify the `vmotionLimit` value in milliseconds. Examples of `vmotionLimit` options include `serial0.vmotionLimit`, `serial1.vmotionLimit`, and `serial2.vmotionLimit`.

Figure 1-3. VMOTION-BEGIN and VMOTION-GOAHEAD shows two ESX hosts and a proxy at the start of a vMotion operation. Host A has a virtual machine that will be moved to Host B.

Figure 1-3. VMOTION-BEGIN and VMOTION-GOAHEAD
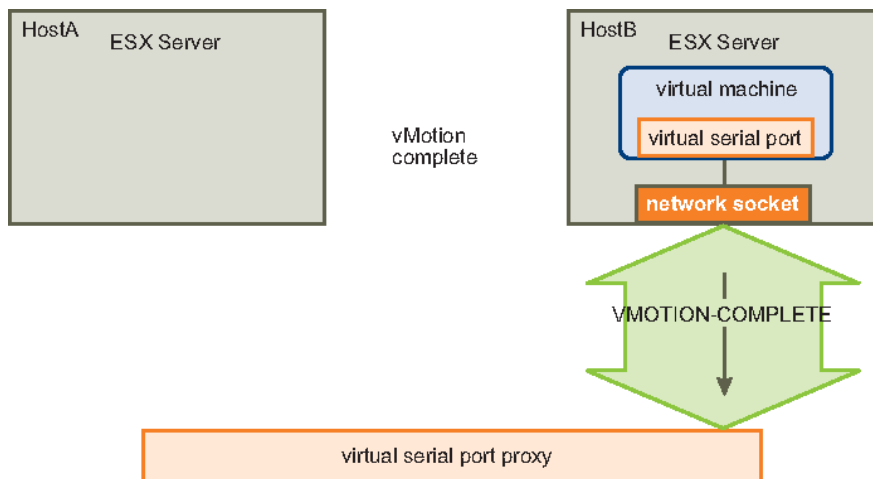
- Virtual machine – At the beginning of the vMotion operation, the ESX server (Host A) uses the virtual machine's serial port connection to send a `VMOTION-BEGIN` request to the proxy. To identify the virtual machine, the host provides an opaque `sequence` value for the message.

```
IAC SB VMWARE-TELNET-EXT VMOTION-BEGIN sequence IAC SE
```

- Proxy – After sending pending data, the proxy replies by sending a VMOTION–GOAHEAD message to indicate that the vMotion operation can continue. The message includes the VMOTION–BEGIN sequence value and an opaque secret value, which the proxy provides.

```
IAC SB VMWARE-TELNET-EXT VMOTION-GOAHEAD sequence secret IAC SE
```

After the proxy replies with a VMOTION–GOAHEAD response, the vMotion operation begins. The proxy has a single connection to the virtual machine.

The vCenter server creates a second instance of the virtual machine on Host B. When the virtual machine boots on the Host B, it configures network backing for the virtual serial port and establishes a second telnet connection with the proxy. Before continuing with the vMotion operation, the new virtual machine and the proxy renegotiate the telnet COM–PORT–OPTION. They do not renegotiate the Com Port configuration. The proxy should be prepared to support the same Com Port configuration that was established for the original telnet connection. See RFC2217 for information about the telnet Com Port control option. The proxy then maintains one telnet connection for each instance of the virtual machine.

To start the VMware telnet extension session for the new connection, the new virtual machine instance negotiates the VMware telnet extension option (VMWARE–TELNET–EXT).

Figure 1-4. VMOTION-PEER and VMOTION-PEER-OK shows the systems during the vMotion operation.

- Virtual Machine: To continue the vMotion operation, Host B sends a VMOTION–PEER message to the proxy to identify the new instance as the same virtual machine that started the vMotion operation. The message includes both the sequence and secret values to identify the virtual machine instance.

```
IAC SB VMWARE-TELNET-EXT VMOTION-PEER sequence secret IAC SE
```

- Proxy: The proxy replies with a VMOTION–PEER–OK message to indicate that it accepts the peer connection.

```
IAC SB VMWARE-TELNET-EXT VMOTION-PEER-OK sequence IAC SE
```

Figure 1-4. VMOTION-PEER and VMOTION-PEER-OK
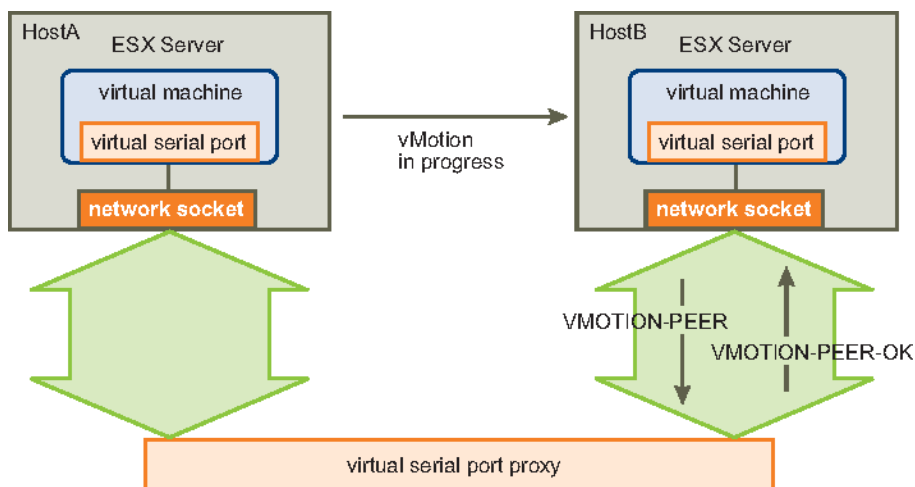
Figure 1-5. VMOTION-COMPLETE shows the completed vMotion operation. After the proxy accepts the new virtual machine instance as a peer, the ESX server on Host B sends a VMOTION–COMPLETE message to the proxy. The message indicates that the proxy should use the new connection for all traffic to the serial port. Now, the proxy can terminate the original telnet connection to Host A.

```
IAC SB VMWARE–TELNET–EXT VMOTION–COMPLETE sequence IAC SE
```

Figure 1-5. VMOTION-COMPLETE
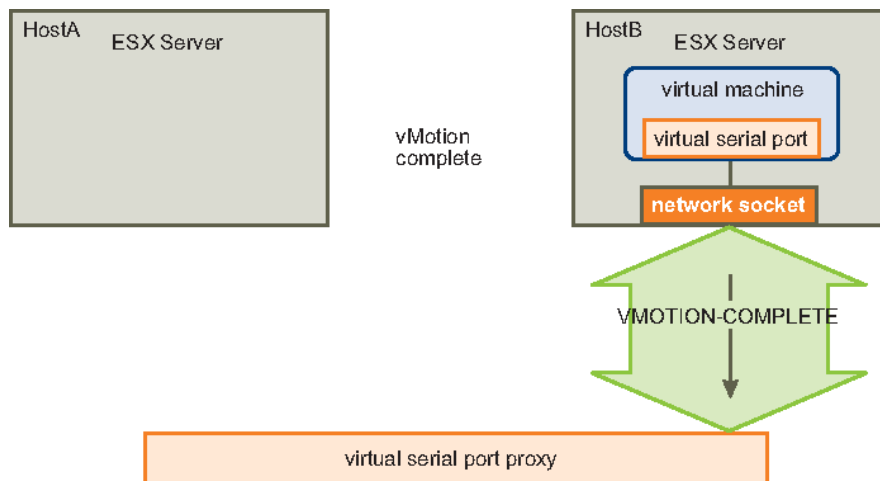
VMware Telnet Extension Commands for Proxy Communication

2

This chapter includes the following topics:

Command Syntax

The VMware telnet extension defines a telnet proxy option for sending notification of vSphere vMotion events to a virtual serial port proxy. A VMware telnet extension command sequence uses the following general format:

```
IAC telnet-command VMWARE-TELNET-EXT [ext-command [parameters] IAC SE]
```

Where

- `IAC` is the telnet escape character prefix (Interpret as Command).

- `telnet-command` is one of the following telnet commands: `WILL`, `WON'T`, `DO`, `DON'T`, and `SB`.

- `VMWARE-TELNET-EXT` is the VMware telnet extension command.

- `ext-command` is a VMware telnet extension command that indicates one of the extension suboption commands. A suboption command sequence begins with the `VMWARE-TELNET-EXT` command and is enclosed within `IAC SB ... IAC SE` telnet subnegotiation commands.

- `parameters` indicates one or more parameter values.

- `IAC SE` is the telnet sequence for completing a subnegotiation command sequence.

The following sections contain tables of VMware telnet extension commands. A table entry shows the command, the corresponding integer code, a description, and the command and code sequences. A code sequence for a VMware telnet extension command has the following characteristics.

- The code sequence begins with code 255, the telnet escape character prefix IAC (Interpret as Command).

- VMware telnet extension commands use two- or three-byte code sequences.

- The VMWARE-TELNET-EXT command code is 232.

- Subnegotiation command sequences include begin (SB) and end (SE) codes (250 and 240).

For example, the DO-PROXY command requires the following code sequence with embedded arguments for direction ("S" for server) and URI:

```
IAC SB   VMWARE-TELNET-EXT DO-PROXY direction        uri          IAC SE
255 250      232               70        "S" "telnet://example.com" 255 240
[R]VM
```

VMware Telnet Extension Option

Table 2-1. lists the commands that establish the VMware telnet extension communication between a virtual machine and a proxy. The virtual machine initiates the connection by sending a WILL VMWARE-TELNET-EXT message. If the proxy accepts by responding with a DO VMWARE-TELNET-EXT message, the virtual machine and the proxy can continue negotiation.

Table 2-1.

Command	Codes
WILL VMWARE-TELNET-EXT A virtual machine request indicating that it is willing to send VMware telnet extension messages to the proxy. IAC WILL VMWARE-TELNET-EXT 255 251 232	251 232
WON'T VMWARE-TELNET-EXT A virtual machine notification that it is unwilling to send VMware telnet extension messages to the proxy. The virtual machine can use this command to turn off the extension option. To resume message exchange, the virtual machine and proxy must negotiate to turn on the option again (WILL VMWARE-TELNET-EXT and DO VMWARE-TELNET-EXT). IAC WON'T VMWARE-TELNET-EXT 255 252 232	252 232

Table 2-1. (continued)

Command	Codes

`DO VMWARE-TELNET-EXT`
 	
A proxy response that indicates it is willing to receive VMware telnet extension messages from the virtual machine.
 	
`IAC DO VMWARE-TELNET-EXT`
 	
255 253 232
	253 232

`DON'T VMWARE-TELNET-EXT`
 	
A proxy response that indicates it is unwilling to receive VMware telnet extension messages. The proxy can send this message at any time to stop remote serial port communication. To resume message exchange, the virtual machine and proxy must negotiate to turn on the option again (`WILL VMWARE-TELNET-EXT` and `DO VMWARE-TELNET-EXT`).
 	
`IAC DON'T VMWARE-TELNET-EXT`
 	
255 254 232
	254 232

Option Subnegotiation

The `VMWARE-TELNET-EXT` telnet option requires various suboptions to support remote serial port access. The virtual machine and proxy use subnegotiation to indicate the command sets that they can support. Subnegotiation identifies all commands for negotiation as well as commands for proxy support, vMotion notification, and virtual machine identification. Table 2-2 lists the commands for option subnegotiation.

Command	Code

`KNOWN-SUBOPTIONS-1 suboptions`
 	
A virtual machine notification that indicates the `VMWARE-TELNET-EXT` suboptions that it can generate and accept. *suboptions* is a sequence of one or more codes, each of which corresponds to a VMware telnet extension command.
 	
If the virtual machine sends this message more than once during a session, the list of suboptions is always the same. The presence of a command in the list does not imply that the virtual machine will use the command during the session.
 	
<pre>IAC SB VMWARE-TELNET-EXT KNOWN-SUBOPTIONS-1 suboptions IAC SE
255 250 232 0 suboptions 255 240</pre>
	0

`KNOWN-SUBOPTIONS-2 suboptions`
 	
A proxy response that indicates the `VMWARE-TELNET-EXT` suboptions that it can generate and accept. *suboptions* is a sequence of one or more codes, each of which corresponds to a VMware telnet extension command. The proxy must support this command.
 	
If the proxy sends this message more than once during a session, the list of suboptions must be the same each time.
 	
<pre>IAC SB VMWARE-TELNET-EXT KNOWN-SUBOPTIONS-2 suboptions IAC SE
255 250 232 1 suboptions 255 240</pre>
	1

Unknown Command Response

Table lists the commands to indicate reception of an unknown command. The proxy must send UNKNOWN‑SUBOPTION‑RCVD‑2 any time it receives a command that it does not support.

Command	Code

UNKNOWN‑SUBOPTION‑RCVD‑1 code
 	
A virtual machine response to indicate that it received an unknown suboption command from the proxy. *code* identifies the unknown suboption.
 	
`IAC SB VMWARE‑TELNET‑EXT UNKNOWN‑SUBOPTION‑RCVD‑1 code IAC SE`
`255 250 232 2 code 255 240`
	2

UNKNOWN‑SUBOPTION‑RCVD‑2 code
 	
A proxy response to indicate that it received an unknown suboption command from the virtual machine. *code* identifies the unknown suboption. The proxy must support this command.
 	
`IAC SB VMWARE‑TELNET‑EXT UNKNOWN‑SUBOPTION‑RCVD‑2 code IAC SE`
`255 250 232 3 code 255 240`
	3

Proxy Operation

Table lists the extension commands to negotiate proxy support.

Command	Code
`DO-PROXY direction serviceUri`	70

A virtual machine request for the proxy to do forwarding operations over the connection. The virtual machine sends a `DO-PROXY` request before any content is exchanged between the virtual machine and the remote system.

`direction` and `serviceURI` correspond to the `VirtualSerialPortURIBackingInfo` properties in the virtual serial port configuration. (See the *vSphere API Reference* for information about the `VirtualSerialPortURIBackingInfo` data object.)

`direction` is either "C" (client) or "S" (server). If the direction indicates that the virtual machine is acting as a client, `serviceUri` identifies a remote system and the proxy initiates a connection with the remote system. If the direction indicates that the virtual machine is acting as a server, `serviceUri` identifies the network socket on the host on which the virtual machine runs. In this case, the virtual machine will accept a connection initiated by a remote system (using the proxy).

If possible, the proxy should also handle port control options, as specified by RFC 2217.

```
IAC SB VMWARE-TELNET-EXT DO-PROXY direction uri IAC SE
255 250 232 70 direction uri 255 240
```

`WILL-PROXY`	71

A proxy response to indicate that it will do forwarding operations.

```
IAC SB VMWARE-TELNET-EXT WILL-PROXY IAC SE
255 250 232 71 255 240
```

`WON'T-PROXY`	73

A proxy response to indicate that it will not do forwarding operations.

```
IAC SB VMWARE-TELNET-EXT WON'T-PROXY IAC SE
255 250 232 73 255 240
```

vMotion Notification

Table lists the commands that hosts and proxy use during a vMotion operation. In the following descriptions, the *source* host contains the original virtual machine instance, and the *target* host is the host on which the new instance is created. A host uses the virtual machine serial port connection with the proxy to send vMotion messages.

Command	Code
`VMOTION—BEGIN sequence`	40

A source host request to indicate that a vMotion operation is pending. The operation requires a VMOTION-GOAHEAD response to start. The intent is to flush all pending data transmission on the connection.

The host provides an opaque `sequence` value. All subsequent vMotion messages related to the connection between the virtual machine and this proxy use this sequence value.

```
IAC SB VMWARE—TELNET—EXT VMOTION—BEGIN sequence IAC SE
255 250 232 40 sequence 255 240
```

Command	Code
`VMOTION—GOAHEAD sequence secret`	41

A proxy response to indicate that all queued data has been transmitted, and that the vMotion operation can start. The proxy sends this message only in response to a `VMOTION—BEGIN` request.

The proxy returns the sequence value that it received in the `VMOTION—BEGIN` request. The proxy also creates and returns an opaque secret value. The virtual machine uses the sequence and secret to identify itself when it establishes a proxy connection from the new host. See the description of `VMOTION—PEER`.

```
IAC SB VMWARE—TELNET—EXT VMOTION—GOAHEAD sequence secret IAC SE
255 250 232 41 sequence secret 255 240
```

Command	Code
`VMOTION—NOTNOW sequence`	43

A proxy response to indicate that the vMotion operation should not start at this time. This message includes the sequence value that the proxy obtained in the `VMOTION—BEGIN` request. The proxy can send this message only in response to a `VMOTION—BEGIN` request.

```
IAC SB VMWARE—TELNET—EXT VMOTION—NOTNOW sequence IAC SE
255 250 232 43 sequence 255 240
```

Command	Code
`VMOTION—PEER sequence secret`	44

A target host request to indicate that this connection belongs to the virtual machine associated with `sequence` and `secret`. The sequence and secret identify the duplicate connections between the proxy and virtual machine instances during a vMotion operation.

The original host generates the sequence value for the `VMOTION—BEGIN` message. The proxy generates the secret value for the `VMOTION—GOAHEAD` message.

```
IAC SB VMWARE—TELNET—EXT VMOTION—PEER sequence secret IAC SE
255 250 232 44 sequence secret 255 240
```

Command	Code
`VMOTION—PEER-OK sequence`	45

A proxy response to indicate that the secret in the `VMOTION—PEER` message was accepted. The proxy sends this message only in response to a `VMOTION—PEER` request.

```
IAC SB VMWARE—TELNET—EXT VMOTION—PEER-OK sequence IAC SE
255 250 232 45 sequence 255 240
```

Command	Code

`VMOTION-COMPLETE sequence`
 	
A target host notification that the vMotion operation is complete. The proxy must send all subsequent communication for the virtual machine on the connection associated with this `VMOTION-COMPLETE` message. The host sends this message only after sending a `VMOTION-PEER` message.
 	
`IAC SB VMWARE-TELNET-EXT VMOTION-COMPLETE sequence IAC SE`
`255 250 232 46 sequence 255 240`
	46

`VMOTION-ABORT`
 	
A source host notification that the vMotion operation failed. The proxy must send all communication for the virtual machine on the connection associated with this `VMOTION-ABORT` message. The host can send `VMOTION-ABORT` only if it has sent `VMOTION-BEGIN` previously on this connection.
 	
`IAC SB VMWARE-TELNET-EXT VMOTION-ABORT IAC SE`
`255 250 232 48 255 240`
	48

Virtual Machine Identification

Table lists the suboption commands for virtual machine identification.

Command	Code

`VM-VC-UUID vm-uuid`
 	
A virtual machine response that contains its universally unique identifier.
 	
`IAC SB VMWARE-TELNET-EXT VM-VC-UUID vm-uuid IAC SE`
`255 250 232 80 vm-uuid 255 240`
	80

`GET-VM-VC-UUID`
 	
A proxy request for the virtual machine's universally unique identifier.
 	
`IAC SB VMWARE-TELNET-EXT GET-VM-VC-UUID IAC SE`
`255 250 232 81 255 240`
	81

`VM-NAME vm-name`
 	
A virtual machine response that contains its name.
 	
`IAC SB VMWARE-TELNET-EXT VM-NAME vm-name IAC SE`
`255 250 232 82 vm-name 255 240`
	82

`GET-VM-NAME`
 	
A proxy request for the virtual machine's name.
 	
`IAC SB VMWARE-TELNET-EXT GET-VM-NAME IAC SE`
`255 250 232 83 255 240`
	83

`VM-BIOS-UUID vm-uuid`
 	
A virtual machine response that contains its universally unique identifier stored in the virtual machine BIOS.
 	
`IAC SB VMWARE-TELNET-EXT VM-BIOS-UUID-IS vm-uuid IAC SE`
`255 250 232 84 vm-uuid 255 240`
	84

Command	Code
GET–VM–BIOS–UUID	85

A proxy request for the virtual machine's universally unique identifier stored in its BIOS.

```
IAC SB VMWARE–TELNET–EXT GET–VM–BIOS–UUID IAC SE
255 250 232 85 255 240
```

Command	Code
VM–LOCATION–UUID vm–uuid	86

A virtual machine response that contains its universally unique identifier.

```
IAC SB VMWARE–TELNET–EXT VM–LOCATION–UUID vm–uuid IAC SE
255 250 232 86 vm–uuid 255 240
```

Command	Code
GET–VM–LOCATION–UUID	87

A proxy request for the virtual machine's location universally unique identifier.

```
IAC SB VMWARE–TELNET–EXT GET–VM–LOCATION–UUID IAC SE
255 250 232 87 255 240
```