

Transporting VIX Guest Operations to the vSphere API

vSphere 5

This document describes how to perform guest operations, that is, interaction with the guest operating system running in a virtual machine, using the vSphere API instead of the VIX API.

New in vSphere 5

VIX has always been particularly strong in the APIs it provides for guest operations. The vSphere API did not offer any equivalent functionality until the 5.0 release, which adds the following guest operations:

- GuestAuthManager – acquire credentials, release credentials, validate credentials.
- GuestFileManager – change file attributes, create temporary directory or file, delete directory or file, initiate file transfer from guest or file transfer to guest, list files, make directory, move directory or file.
- GuestProcessManager – list processes, read environment variable, start (run) program, terminate process.

You can program the VIX API in C/C++, Perl, or COM (Visual Basic or C#).

The vSphere API guest operations in release 5.0 can be used with any language that handles VMware WSDL. Java source code samples based on JAX-WS are available in the vSphere Web Services SDK. Guest scripting is possible with the vSphere SDK for Perl. The VMware PowerCLI includes some guest operations, listed under “PowerCLI for PowerShell” on page 8.

About Guest Operations

The sections below define guest operations for VIX developers who are not certain which APIs they include. Guest operations manipulate processes, files, folders, and environment variables in a guest operating system.

Virtual Machine and Datacenter Operations

Virtual machine operations affect virtual hardware state, including power on, power off, suspend, resume, take snapshot, and revert-to-snapshot. The VIX API has these, but so did the vSphere API.

Datacenter management operations perform cloud-scale configuration of compute and storage resources, and include cross-host operations such as vMotion and Fault Tolerance (FT). The VIX API never had these.

Easy Interfaces for Guest Operations

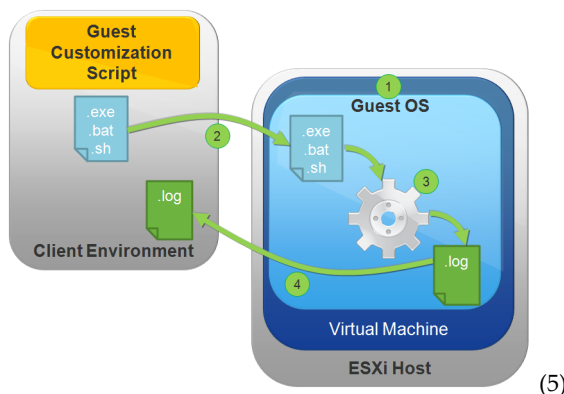
After you install the VIX package on a Windows or Linux client, many VIX guest operations are available from the `vmrun` command-line utility. The `vmrun` utility is convenient for use in shell scripts and batch files, works with ESXi hosts and vCenter, and is additionally available with Fusion for Mac OS.

VMware Labs offers an unsupported VIX-based “VMware guest console” (VGC) to manage files and processes in a guest operating system. It is available for download at the <http://labs.vmware.com/flings/vgc> Web site.

Usefulness of Guest Operations

In a datacenter, guest customization scripts can simplify management tasks for Windows and Linux systems. [Figure 1](#) shows how a script can use executable, batch, or shell script to modify the guest and return a log file.

Figure 1. Guest Customization



- 1 Power on.
- 2 Copy files to guest, including scripts, installers, and data.
- 3 Run programs in the guest, and produce a log.
- 4 Copy the log file back to the client environment.
- 5 Repeat as needed on other virtual machines across all ESXi hosts in a datacenter.

Components for Guest Operations

VIX guest operations rely on the following components:

- VIX client library loaded into your program (the vSphere API replaces this)
- VMX processes on the host to manage the run-time state of each virtual machine hosted by ESXi
- VMware Tools running in each guest operating system

The following five sections (until [“Web Services Replace the VIX Client”](#) on page 4) describe the VIX client.

Authentication by Host and Guest with VIX

VIX client programs authenticate on the host using `VixHost_Connect`, which takes as parameters a user name and password for the ESX host or vCenter Server. The user must have appropriate vSphere credentials.

In vSphere 4.1, a privilege named “Acquire guest control ticket” was established. To perform guest operations, an authenticated vSphere user must have this role assigned. Otherwise requests to run guest operations throw the error `VIX_E_HOST_USER_PERMISSIONS`.

Guest operations require further authentication, for instance using `VixVM_LoginInGuest` to specify user name and password for the guest OS. In the future, certificate-based or SSO-token authentication may be required.

The guest agent (here, part of VMware Tools) impersonates a user using the user’s specified guest credentials, and runs with permissions of that user. This is unlike a login shell, because no session is established with the guest agent. After you call `VixVM_LoginInGuest`, credentials are stored in the client program, and sent as part of every request. The guest agent validates these credentials before processing each guest operations request. The VMware Tools guest agent executes authenticated guest operations with `vmtools.exe` on Windows or the `vmtoolsd` process on Linux.

If `VixVM_LoginInGuest` option `VIX_LOGIN_IN_GUEST_REQUIRE_INTERACTIVE_ENVIRONMENT` is specified, guest operations are redirected to a different process running in the interactive console session. You must use this option to run a program in the guest that creates a window that is visible in the guest console session, or to access certain resources in Windows guests.

Disabling Guest Operations with VIX

Guest operations can be disabled per virtual machine, or host wide. On a virtual machine, set the following attribute in the VMX configuration file. For a host, set this attribute in the host-wide configuration file. However be warned that this disables certain vSphere features that depend upon guest operations, such as the vCenter Update Manager (VUM), and vCenter guest file-system quiescing for snapshot-based backup.

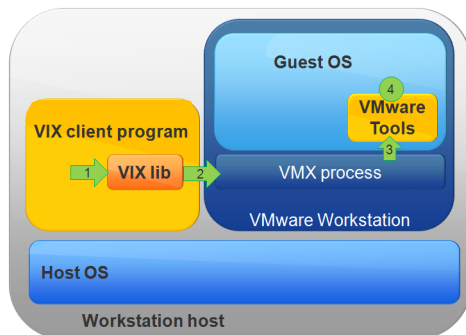
```
guest.commands.enabled = "FALSE"
```

Guest operations can be disabled per-user by removing the user's "Acquire guest control ticket" privilege.

VIX Guest Operations on Workstation

The control path for Workstation is simple: the VIX client program runs locally, on the Workstation host.

Figure 2. Control Path on Workstation and Player

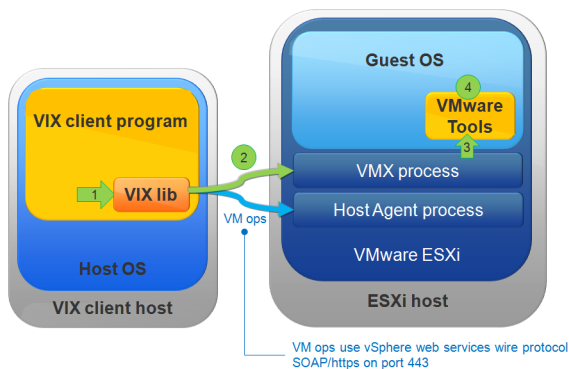


- 1 VIX client program calls a function in the VIX library.
- 2 VIX sends a command over local IPC to Workstation.
- 3 Workstation relays the command to VMware Tools in the guest.
- 4 VMware Tools has the Guest OS execute the guest operation.

VIX Guest Operations with an ESXi Host

Figure 3 illustrates the control path when a VIX client program is used with an ESXi host. Unlike Workstation, where VIX client code runs directly on the Workstation host, VIX client code runs remotely on an ESXi host.

Figure 3. Control Path on ESXi

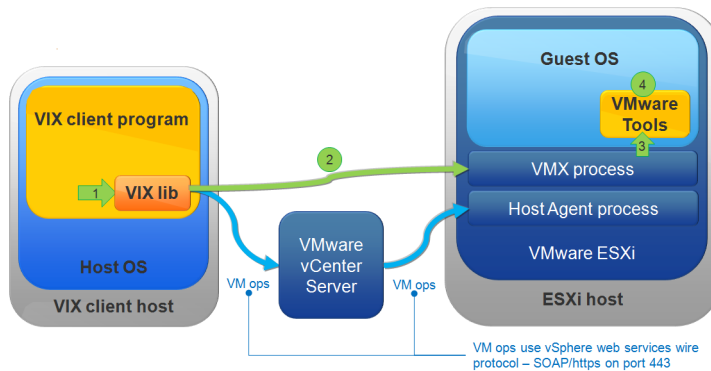


- 1 VIX client program calls a function in the VIX library.
- 2 VIX sends a command over a TCP connection (on port 902) to the virtual machine's VMX process.
By contrast, virtual machine and datacenter operations are handled by the host agent process, `hostd`.
- 3 VMX relays the guest operation command to VMware Tools in the guest.
- 4 VMware Tools has the Guest OS execute the guest operation.

VIX Guest Operations with vCenter Server

Figure 4 illustrates the control path when a VIX client program is directed through vCenter Server. Note that VIX guest operations do not follow the same data path as virtual machine and datacenter operations.

Figure 4. Control Path with vCenter Server



- 1 VIX client program calls a function in the VIX library.
- 2 VIX sends a command over a TCP connection (on port 902) to the virtual machine’s VMX process.

The TCP connection for VIX guest operations goes directly to VMX on the ESXi host, bypassing the vCenter Server, and the host agent process. By contrast, virtual machine and datacenter operations are proxied by vCenter Server using SOAP/https on port 443, and relayed to the host agent.

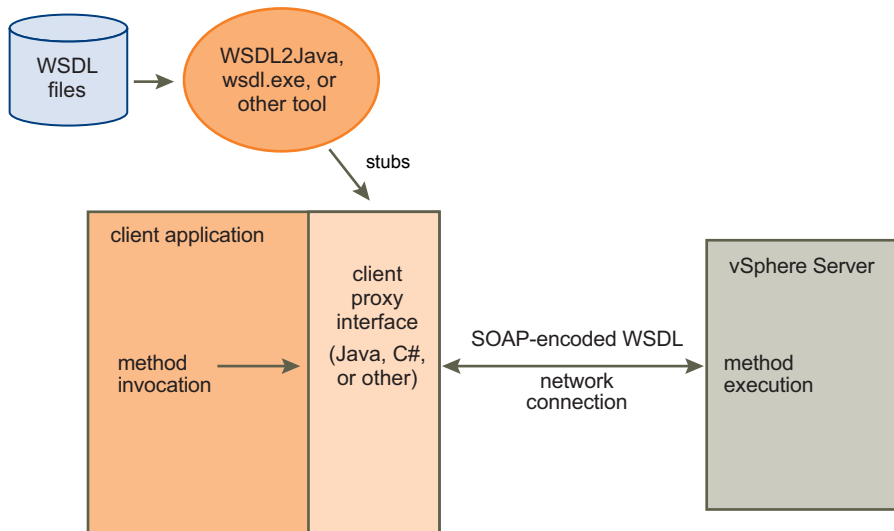
- 3 VMX relays the command to VMware Tools in the guest.
- 4 VMware Tools has the Guest OS execute the guest operation.

Web Services Replace the VIX Client

For VIX guest operations in Workstation, on ESXi hosts, or through vCenter, you run programs on a VIX client using the “wrapper” library, a stub library that dynamically loads a suitable VIX implementation for the client. On Windows, the wrapper library is implemented as `VixAllProducts.lib`, a static library. On Linux, the wrapper library is implemented as `libVixAllProducts.so`, a dynamic (shared object) library.

Figure 5 shows vSphere guest operations. Using Java or C# methods for example, your client application calls a client proxy interface that provides language-specific WSDL bindings (stubs). The client proxy encapsulates your method invocation in a SOAP layer. Responding on an HTTPS port, vSphere executes the method, often asynchronously. The WSDL bindings and client proxy interface can be identical on Windows and Linux.

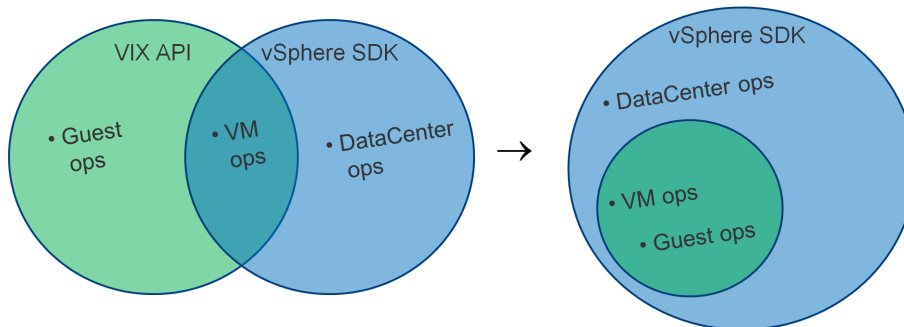
Figure 5. SOAP communications with WSDL, a different model



New Guest Operations for the vSphere API

Previously, writing application software to run on VMware platform products was complicated by having to use the vSphere API for certain operations, and the VIX API for other operations. This complication has been eliminated in vSphere 5 by folding VIX guest operations into the vSphere API.

Figure 6. Old and New API Designs

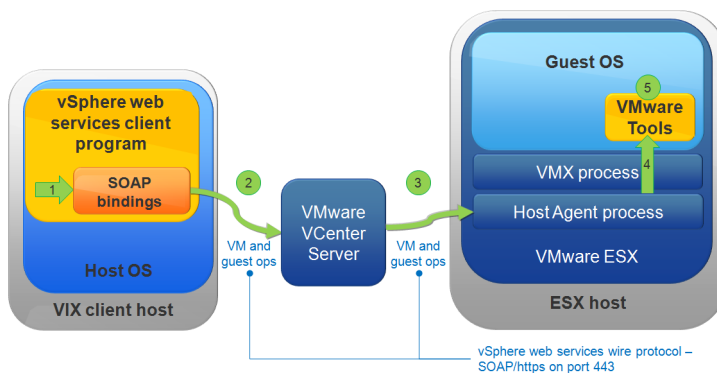


Benefits of the New API Set

The benefits are that you can use the same language bindings, a unified object model, and single-sign-on to the host. Security is enhanced because of better integration with the vSphere user-and-role permissions model. Monitoring is improved by alignment with the events and auditing features of vSphere. The new model makes it possible to follow guests after vMotion of their virtual machines.

Most importantly, network connectivity is simplified by having everything go over https as a Web service, rather than requiring datacenters to open TCP port 902 for VIX communications. In [Figure 7](#), compare the former VIX control path ([Figure 4](#)) to the new vSphere control path through vCenter Server. Guest operations now follow the same control path as virtual machine and datacenter operations.

Figure 7. Simplified Control Path with vSphere 5



- 1 A vSphere Web services client program calls a function in the vSphere API.
- 2 The client sends a SOAP command over https (port 443) to vCenter Server.
- 3 The vCenter Server passes the command to the host agent process `hostd`, which sends it to VMX.
- 4 VMX relays the command to VMware Tools in the guest.
- 5 VMware Tools has the Guest OS execute the guest operation.

Comparison of Guest Operations

[Table 1, “Guest Operations in VIX and vSphere,”](#) on page 6 compares methods of the new vSphere API with `vmrun` and the VIX API function calls for C/C++. The VIX API calls for Perl and COM have similar names but without the `Vix` and `VixVM` preface, respectively. Authorization methods are not the same. VIX programs call `login` once, run guest operations, then `logout`. In vSphere, the credentials are sent with every request.

In the third column, items ending in “Manager” are managed objects that have the methods listed under them.

Table 1. Guest Operations in VIX and vSphere

VIX API for C/C++	vmrun Command	vSphere API
VixHost_RegisterVM	register	Inventory
Connections		GuestAuthManager
VixHost_Connect (somewhat equivalent)	-h <i>host</i>	AcquireCredentialsInGuest
VixVM_LoginInGuest	-u <i>user</i> -p <i>password</i>	ValidateCredentialsInGuest
VixHost_Disconnect	end of script	ReleaseCredentialsInGuest
VixHost_OpenVM supersedes VixVM_Open	<i>path/to/vm.vmx</i>	
File Operations		GuestFileManager
—	—	ChangeFileAttributesInGuest
—	—	CreateTemporaryDirectoryInGuest
VixVM_CreateTempFileInGuest	—	CreateTemporaryFileInGuest
VixVM_DeleteDirectoryInGuest	deleteDirectoryInGuest	DeleteDirectoryInGuest
VixVM_DeleteFileInGuest	deleteFileInGuest	DeleteFileInGuest
VixVM_CopyFileFromGuestToHost	copyFileFromGuestToHost	InitiateFileTransferFromGuest
VixVM_CopyFileFromHostToGuest	copyFileFromHostToGuest	InitiateFileTransferToGuest
VixVM_ListDirectoryInGuest	listDirectoryInGuest	ListFilesInGuest
VixVM_CreateDirectoryInGuest	createDirectoryInGuest	MakeDirectoryInGuest
(use file rename)	(use file rename)	MoveDirectoryInGuest
VixVM_RenameFileInGuest	renameFileInGuest	MoveFileInGuest
VixVM_DirectoryExistsInGuest	directoryExistsInGuest	—
VixVM_FileExistsInGuest	fileExistsInGuest	—
VixVM_WaitForToolsInGuest	—	vix.vm.GuestInfo.guestOperationsReady ...interactiveGuestOperationsReady
VixVM_InstallTools	installTools	—
Processes and Variables		GuestProcessManager
VixVM_ListProcessesInGuest	listProcessesInGuest	ListProcessesInGuest
VixVM_ReadVariable	readVariable <i>guestEnv</i>	ReadEnvironmentVariableInGuest
VixVM_RunProgramInGuest	runProgramInGuest	StartProgramInGuest
VixVM_KillProcessInGuest	killProcessInGuest	TerminateProcessInGuest
VixVM_RunScriptInGuest	runScriptInGuest	(use StartProgramInGuest)
Shared Folder Operations		not supported on vSphere

Notes on New vSphere APIs

ChangeFileAttributesInGuest and CreateTemporaryDirectoryInGuest are new APIs with no VIX equivalents.

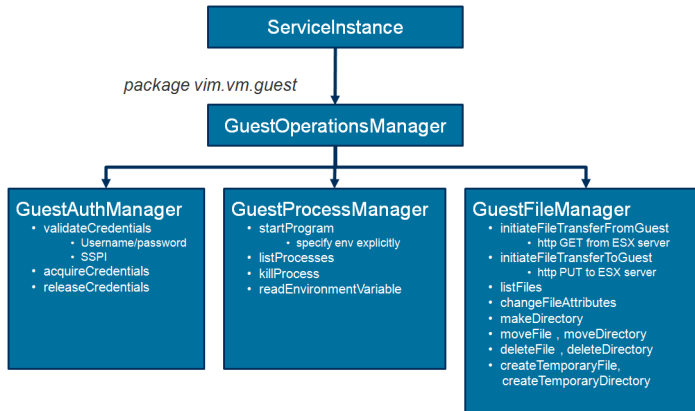
AcquireCredentialsInGuest is the method to get a session security ticket. Currently the only available ticket is from the Security Support Provider Interface (SSPI). SSPI performs a challenge-response authentication of the credentials. If valid, the virtual machine’s guest operations component issues a session ticket, type *sspiToken* in data object *SSPIAuthentication*. Clients authenticate with the ticket until it expires. SSPI is useful for when the client environment and guest operating system are both Windows instances joined to the same domain, so SSPI can pass a Windows domain login session token from the client to the guest agent without requiring an explicit user name and password.

ValidateCredentialsInGuest checks for valid credentials. You pass it a managed object reference to the guest's virtual machine, and the GuestAuthentication data object with sspiToken, or user name and password. You pass the same auth (authorization) credential in every vSphere guest operation method.

Object Class Diagram

Figure 8 shows the objects that inherit from GuestOperationsManager.

Figure 8. Guest Operations Class Design



In the vSphere API, the VirtualMachine and GuestInfo managed objects contain information about what guest operations might be running and relevant virtual machine state:

```

vim.VirtualMachine.guest()
vim.vm.GuestInfo.guestOperationsReady
vim.vm.GuestInfo.interactiveGuestOperationsReady
  
```

VMware Tools must still be present to run guest operations, as before. To perform interactive guest operations, the user must be logged into the console, for example through the vSphere Client.

Other guest interfaces include the Guest SDK for guest monitoring and the HA Application Monitoring SDK.

Permissions for vSphere Guest Operations

VIX stores guest authorization state for you, so the typical usage is for a program to LoginInGuest, run some guest operations, then LogoutFromGuest.

With the vSphere guest operations APIs, no state is stored on the client side, so authorization information must be included as part of every guest operations method call. VMware Tools do not cache guest credentials, so even after VixVM_LoginInGuest, authorization information goes with each request, but the VIX client library stores your credentials between requests. However because vSphere communicates by WSDL, guest operation methods cannot cache your credentials, so the authorization API must be stateless.

A guest credential is encapsulated by the abstract class GuestAuthentication, and the client passes an object of this type with each guest operation call. To authenticate with the guest using a user name and password, you instantiate an object of type NamePasswordAuthentication (child class of GuestAuthentication) and set the user name and password fields appropriately. The other currently supported type of guest authentication is SSPI passthrough, from a Windows client to a Windows guest instance. For details on how to implement this, see the *vSphere API Reference* for the GuestAuthManager's acquireCredentialsInGuest method.

The vSphere guest operations require three additional vCenter privileges that affect authentication, which are documented in the *vSphere API Reference* as being required by the vm parameter, a managed object reference to the virtual machine on which a guest operation is performed.

- VirtualMachine.GuestOperations.Query – required for authorization on a virtual machine.
- VirtualMachine.GuestOperations.Modify – required for file management on a virtual machine.
- VirtualMachine.GuestOperations.Execute – required for process execution on a virtual machine.

Permissions for vSphere guest operations follow the read/write/execute model. Each API requires a different privilege to fit its function: for example query for listFiles, modify for deleteFile, execute for terminateProcess. As stated in “Authentication by Host and Guest with VIX” on page 2, VIX required the “Acquire guest control ticket” privilege. This is also required for vSphere guest operations.

- `VirtualMachine.Interact.GuestControl` – same as “Acquire guest control ticket” in the UI.

Licensing

A paid ESXi or vSphere license (not free vSphere Hypervisor) is required to use the VIX API or vSphere API. No additional license is required to use these APIs.

PowerCLI for PowerShell

Guest operations in PowerCLI 5.0 use the new vSphere APIs. This change does not affect script compatibility. PowerCLI still packages VIX libraries for backward compatibility. PowerCLI guest operations include:

- `Copy-VMGuestFile` – same functionality as `copyFileFromGuestToHost` and `copyFileFromHostToGuest`.
- `Get-VMGuest` – provides information about the type of guest operating system.
- `Invoke-VMScript` – same functionality as `runScriptInGuest` or `runProgramInGuest`.
- `Mount-Tools` and `Update-Tools` – same functionality as `installToolsInGuest`.

Perl Script for Guest Operations

The SourceForge “vghetto” project contains the `guestOpsManagement.pl` Perl script, which implements the majority of vSphere guest operations for management by a centralized script. It requires a client system with vCLI 5.0 installed, or the vMA 5.0 virtual appliance, and ESXi 5.0 must be running virtual machines with the latest VMware Tools installed. Here is an introduction to the script and the 12 guest operations it supports:

<http://www.virtuallyghetto.com/2011/07/automating-new-integrated-vixguest.html>

The `guestOpsManagement.pl` script is written and owned by a third party. It is not supported by VMware. However you can look at the Perl code to get an idea how to work with the vSphere API.

Java Source Code Samples

Four Java code samples based on JAX-WS are available in the vSphere SDK for Web services, in this directory:

`SDK/vsphere-ws/java/JAXWS/samples/com/vmware/guest`

CreateTemporaryFile.java

This sample creates a temporary file inside a virtual machine, by calling the following method:

```
vimPort.createTemporaryFileInGuest(fileManagerRef, vmMOR, auth, prefix, suffix, directoryPath);
```

DownloadGuestFile.java

This sample downloads a file from the guest to a specified path on the host where the client is running.

```
vimPort.initiateFileTransferFromGuest(fileManagerRef, vmMOR, auth, guestFilePath);
```

The destination, a local file on the client host, is specified on the command line as `--localfilepath`.

RunProgram.java

This sample runs a specified program inside a guest operating system, with output re-directed to a temporary file, and downloads the resulting output to a file on the local client.

```
vimPort.startProgramInGuest(processManagerRef, vmMOR, auth, spec);
```

The program must already exist on the guest, and is specified on the command line as `--guestprogrampath`. The output file to store on the client host is specified on the command line as `--localoutputfilepath`.

UploadGuestFile.java

This sample uploads a file from the client machine to a specified location inside the guest.

```
vimPort.initiateFileTransferToGuest(fileManagerRef, vmMOR, auth, guestFilePath,  
    guestFileAttributes, fileSize, optionsmap.containsKey("overwrite"));
```

The source, a local file on the client host, is specified on the command line as `--localfilepath`.

If you have comments about this documentation, submit your feedback to: docfeedback@vmware.com

VMware, Inc. 3401 Hillview Ave., Palo Alto, CA 94304 www.vmware.com

Copyright © 2011 VMware, Inc. All rights reserved. This product is protected by U.S. and international copyright and intellectual property laws. VMware products are covered by one or more patents listed at <http://www.vmware.com/go/patents>. VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions. All other marks and names mentioned herein may be trademarks of their respective companies.

Item: EN-000703-00

9/7/11
