

# Creating Smart Virtual Appliances with IBM Image Construction and Composition Tool

Create smart appliances that are cloud-ready

Develop PowerVM, KVM, and ESX customized virtual appliances

Learn from experts how to cloudify your applications



Greg Hurlebaus  
Rashed Ferdous  
John Jacobson  
Li-Fang Lee

Jarek Miszczyk  
Pat Nickel  
David Peraza  
Moises Romo  
Kerry Staples





International Technical Support Organization

**Creating Smart Virtual Appliances with IBM Image  
Construction and Composition Tool**

July 2013

**Note:** Before using this information and the product it supports, read the information in “Notices” on page ix.

## **Second Edition (July 2013)**

This edition applies to Version 2, Release 2, Modification 1.1 of the IBM Image Construction and Composition Tool.

© Copyright International Business Machines Corporation 2012, 2013. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

|   |      |
|---|------|
| <b>Notices</b> .....  | ix   |
| Trademarks .....  | x    |
| <b>Preface</b> .....  | xi   |
| Authors .....   | xi   |
| Now you can become a published author, too! .....                                       | xiii |
| Comments welcome .....  | xiii |
| Stay connected to IBM Redbooks .....  | xiv  |
| <b>Chapter 1. Introduction to virtual appliance construction</b> .....                  | 1    |
| 1.1 Deploying your applications into the cloud .....                                    | 2    |
| 1.2 Solving the terminology puzzle .....  | 3    |
| 1.3 Why implement software virtual appliances .....                                     | 4    |
| 1.4 A need for a virtual appliance construction tool .....                              | 5    |
| 1.4.1 Application packaging options .....   | 5    |
| 1.4.2 Supported deployment options .....  | 5    |
| 1.5 Putting the Image Construction and Composition Tool into context .....              | 7    |
| 1.5.1 Image Construction and Composition Tool and the IBM PureSystems .....             | 7    |
| 1.5.2 Image Construction and Composition Tool and the IBM Workload Deployer .....       | 9    |
| <b>Chapter 2. Anatomy of a virtual appliance</b> .....                                  | 11   |
| 2.1 Description of a virtual appliance .....  | 12   |
| 2.2 Components of a virtual appliance .....   | 12   |
| 2.2.1 The OVF file .....  | 14   |
| 2.3 Virtual appliance self-activation .....   | 16   |
| 2.4 IBM Virtual Solutions Activation Engine .....                                       | 16   |
| 2.5 What a virtual appliance is not .....   | 20   |
| <b>Chapter 3. Architecture of the IBM Image Construction and Composition Tool</b> ..... | 21   |
| 3.1 Actors and use cases .....  | 22   |
| 3.1.1 Operating system specialist .....   | 22   |
| 3.1.2 Software specialist .....   | 23   |
| 3.1.3 Virtual appliance architect .....   | 24   |
| 3.2 Component view .....  | 25   |
| 3.2.1 The model .....   | 27   |
| 3.2.2 Package generators .....  | 27   |
| 3.2.3 Cloud providers .....   | 28   |
| 3.3 Flow of the main scenarios .....  | 29   |
| 3.3.1 Importing an image from a cloud provider .....                                    | 29   |
| 3.3.2 Importing an image from a running virtual machine .....                           | 31   |
| 3.3.3 Extending a base OS image .....   | 33   |
| 3.3.4 Capturing a synchronized virtual image instance .....                             | 34   |
| 3.3.5 Exporting a virtual appliance .....   | 34   |
| 3.4 Cloud provider concepts .....   | 36   |
| 3.5 Software bundle concepts .....  | 36   |
| 3.6 Image concepts .....  | 37   |
| 3.7 User interface .....  | 37   |
| <b>Chapter 4. Setting up the virtual appliance build environment</b> .....              | 45   |

|                   |  |           |
|-------------------|--|-----------|
| 4.1               | Installing the Image Construction and Composition Tool                             | 46        |
| 4.1.1             | Installing the tool on Linux   | 46        |
| 4.1.2             | Installing the tool on AIX   | 48        |
| 4.1.3             | Accessing the tool   | 49        |
| 4.2               | Setting up the build environment for PowerVM virtual appliances                    | 49        |
| 4.2.1             | Reference architecture   | 50        |
| 4.2.2             | Requirements and recommendations   | 51        |
| 4.2.3             | Creating the build environment   | 57        |
| 4.3               | Setting up the build environment for KVM virtual appliances                        | 65        |
| 4.3.1             | Reference architecture   | 66        |
| 4.3.2             | Nodes  | 66        |
| 4.3.3             | Tips and techniques  | 68        |
| <b>Chapter 5.</b> | <b>Product Activator Development Kit</b>   | <b>73</b> |
| 5.1               | Overview of the Product Activator Development Kit                                  | 74        |
| 5.2               | Installing PADK  | 75        |
| 5.3               | A walkthrough of the development process by using PADK                             | 77        |
| 5.3.1             | Installing the Virtual Solutions Activation Engine manually                        | 77        |
| 5.3.2             | Prototyping the activation program   | 79        |
| 5.3.3             | Creating a product activator project in Eclipse                                    | 79        |
| 5.3.4             | Coding the activation program  | 82        |
| 5.3.5             | Setting up the connection to the target virtual machine                            | 82        |
| 5.3.6             | Running and debugging the activation program                                       | 84        |
| 5.3.7             | PADK integration with Image Construction and Composition Tool                      | 86        |
| <b>Chapter 6.</b> | <b>KVM Express cloud provider</b>  | <b>91</b> |
| 6.1               | Overview of the KVM provider architecture  | 92        |
| 6.2               | Setting up the KVM provider  | 93        |
| 6.2.1             | Installing the KVM provider  | 93        |
| 6.2.2             | Managing the KVM provider  | 94        |
| 6.2.3             | Creating the KVM cloud provider  | 94        |
| 6.3               | Creating a base image from ISO   | 96        |
| 6.3.1             | Preliminary setup  | 96        |
| 6.3.2             | Importing an ISO image   | 98        |
| 6.3.3             | Building a base operating system appliance   | 99        |
| 6.4               | Importing a running virtual machine  | 102       |
| 6.4.1             | Requirements   | 102       |
| 6.4.2             | Importing a running VM from the host system of the KVM provider                    | 102       |
| 6.5               | Importing virtual images (appliances) from the KVM cloud provider                  | 104       |
| 6.5.1             | Importing virtual images   | 104       |
| 6.5.2             | Importing a virtual appliance  | 105       |
| 6.6               | Removing an image (appliance) from the virtual appliance repository                | 106       |
| 6.7               | Working with bundles   | 107       |
| 6.7.1             | Passing parameters to the reset operation  | 107       |
| 6.8               | Validating virtual images by using the OVA Runtime                                 | 109       |
| 6.8.1             | Deploying a virtual image (appliance) with OVA Runtime                             | 110       |
| 6.8.2             | Stopping and releasing a running VM (workload)                                     | 112       |
| 6.9               | Tips to override default behavior of the KVM provider                              | 112       |
| 6.9.1             | Modifying the template OVF file  | 112       |
| 6.9.2             | Changing the default number of clones  | 113       |
| 6.10              | Troubleshooting  | 113       |
| 6.10.1            | Checking the Image Construction and Composition Tool and the KVM provider versions | 113       |

|                   |   |            |
|-------------------|---|------------|
| 6.10.2            | Image Construction and Composition Tool logs . . . . .  | 114        |
| 6.10.3            | KVM provider logs. . . . .  | 114        |
| 6.10.4            | Validating that the KVM provider is active. . . . .   | 114        |
| 6.10.5            | Considerations for restarting the KVM provider services . . . . .                                       | 116        |
| 6.10.6            | Resolving common issues . . . . .   | 116        |
| <b>Chapter 7.</b> | <b>PowerVM Express cloud provider. . . . .</b>  | <b>125</b> |
| 7.1               | Overview of the PowerVM provider. . . . .   | 126        |
| 7.2               | Requirements for PowerVM provider . . . . .   | 126        |
| 7.3               | Configuring a PowerVM provider . . . . .  | 126        |
| 7.4               | Creating a virtual appliance. . . . .   | 134        |
| 7.4.1             | Creating a base image . . . . .   | 134        |
| 7.4.2             | Importing a base image . . . . .  | 137        |
| 7.4.3             | Extending, synchronizing, and capturing an image. . . . .   | 140        |
| 7.4.4             | Exporting an image as an OVA archive . . . . .  | 144        |
| 7.5               | Troubleshooting . . . . .   | 146        |
| 7.5.1             | Checking the Image Construction and Composition Tool and the PowerVM<br>provider versions . . . . .     | 146        |
| 7.5.2             | Checking the PowerVM provider logs. . . . .   | 146        |
| 7.5.3             | Checking the Image Construction and Composition Tool logs . . . . .                                     | 146        |
| 7.5.4             | Importing from a running virtual machine . . . . .  | 146        |
| 7.5.5             | Importing from a cloud provider . . . . .   | 149        |
| 7.5.6             | Extending an image . . . . .  | 150        |
| 7.5.7             | Synchronizing an image . . . . .  | 150        |
| 7.5.8             | Capturing an image. . . . .   | 153        |
| 7.5.9             | Exporting an image. . . . .   | 154        |
| <b>Chapter 8.</b> | <b>ESX cloud provider. . . . .</b>  | <b>157</b> |
| 8.1               | Overview of an ESX cloud provider . . . . .   | 158        |
| 8.2               | Requirements for a VMware ESX cloud provider . . . . .  | 158        |
| 8.3               | Configuring an ESX cloud provider. . . . .  | 158        |
| 8.4               | Creating a virtual appliance. . . . .   | 161        |
| 8.4.1             | Creating a base OS image . . . . .  | 162        |
| 8.4.2             | Importing a base image . . . . .  | 170        |
| 8.4.3             | Working with bundles . . . . .  | 173        |
| 8.4.4             | Extending, synchronizing, and capturing a virtual image . . . . .                                       | 173        |
| 8.4.5             | Exporting an image as an OVA archive . . . . .  | 176        |
| 8.5               | Troubleshooting . . . . .   | 177        |
| 8.5.1             | The Image Construction and Composition Tool version. . . . .  | 177        |
| 8.5.2             | Image Construction and Composition Tool logs . . . . .  | 178        |
| 8.5.3             | Installation logs . . . . .   | 178        |
| 8.5.4             | Image synchronization logs . . . . .  | 178        |
| 8.5.5             | Synchronization fails message: vmPath is null . . . . .   | 178        |
| 8.5.6             | Resolving common issues . . . . .   | 179        |
| <b>Chapter 9.</b> | <b>Constructing simple virtual appliances . . . . .</b>   | <b>181</b> |
| 9.1               | Scenario overview. . . . .  | 182        |
| 9.2               | Implementing the installation and configuration scripts. . . . .  | 182        |
| 9.2.1             | Developing the installation script for IBM WebSphere Application Server<br>Community Edition . . . . .  | 183        |
| 9.2.2             | Developing the configuration script for IBM WebSphere Application Server<br>Community Edition . . . . . | 185        |
| 9.2.3             | Developing a configuration script for the web application. . . . .                                      | 188        |
| 9.3               | Creating a base image . . . . .   | 189        |

|                    |  |            |
|--------------------|--|------------|
| 9.4                | Creating software bundles for the Image Construction and Composition Tool . . . . .                        | 189        |
| 9.4.1              | Creating a software bundle for IBM WebSphere Application Server<br>Community Edition . . . . .             | 189        |
| 9.4.2              | Creating a software bundle for a web application . . . . .   | 194        |
| 9.5                | Building a virtual appliance . . . . .   | 198        |
| 9.5.1              | Extending the base image to create the sample image . . . . .  | 198        |
| 9.5.2              | Adding software bundles to sample image . . . . .  | 200        |
| 9.5.3              | Synchronizing the sample image . . . . .   | 202        |
| 9.5.4              | Validating the sample virtual image . . . . .  | 203        |
| 9.5.5              | Capturing the sample virtual appliance . . . . .   | 204        |
| 9.5.6              | Exporting the sample virtual appliance . . . . .   | 205        |
| 9.5.7              | Verifying the results . . . . .  | 208        |
| 9.6                | Log file of the Image Construction and Composition Tool . . . . .  | 208        |
| 9.6.1              | Synchronizing a simple virtual appliance image . . . . .   | 209        |
| 9.6.2              | Capturing a simple virtual appliance image . . . . .   | 210        |
| 9.6.3              | Exporting a simple virtual appliance image . . . . .   | 211        |
| <b>Chapter 10.</b> | <b>Constructing complex virtual appliances . . . . .</b>   | <b>221</b> |
| 10.1               | Scenario overview . . . . .  | 222        |
| 10.2               | Importing running virtual machines . . . . .   | 223        |
| 10.3               | Preparing for activation programs . . . . .  | 224        |
| 10.3.1             | Installing PADK . . . . .  | 224        |
| 10.3.2             | Examining the configurable resources . . . . .   | 224        |
| 10.3.3             | Identifying the intercomponent dependency . . . . .  | 225        |
| 10.4               | Implementing the activation programs . . . . .   | 226        |
| 10.4.1             | Creating PADK projects in Eclipse . . . . .  | 226        |
| 10.4.2             | Coding activation programs for changeable parameters . . . . .   | 228        |
| 10.4.3             | Coding activation programs for intercomponent dependency . . . . .   | 230        |
| 10.5               | Creating software bundles . . . . .  | 234        |
| 10.6               | Building virtual appliances . . . . .  | 236        |
| 10.6.1             | Adding software bundles to base images . . . . .   | 236        |
| 10.6.2             | Synchronizing virtual appliances . . . . .   | 237        |
| 10.6.3             | Capturing a virtual appliance . . . . .  | 239        |
| 10.6.4             | Exporting virtual appliances . . . . .   | 240        |
| 10.7               | Validating the results . . . . .   | 240        |
| 10.7.1             | A deployment documentation template . . . . .  | 241        |
| 10.8               | Summary . . . . .  | 243        |
| <b>Chapter 11.</b> | <b>Virtual appliance deployment options . . . . .</b>  | <b>245</b> |
| 11.1               | Deploying KVM virtual appliances to Systems Director VMControl or IBM Flex<br>System Manager . . . . .     | 246        |
| 11.1.1             | Importing an appliance into the image repository . . . . .   | 246        |
| 11.1.2             | Deploying an appliance on a KVM host or to a KVM system pool . . . . .                                     | 248        |
| 11.1.3             | Generating a password hash value . . . . .   | 250        |
| 11.2               | Deploying PowerVM virtual appliances to Systems Director VMControl or IBM Flex<br>System Manager . . . . . | 250        |
| 11.2.1             | Importing a virtual appliance into the Systems Director VMControl<br>image repository . . . . .            | 250        |
| 11.2.2             | Deploying the virtual appliance on a PowerVM host or system pool . . . . .                                 | 252        |
| 11.3               | Deploying a virtual appliance with IBM SmartCloud Entry . . . . .  | 255        |
| 11.3.1             | Importing PowerVM and KVM virtual appliances into IBM SmartCloud Entry . . . . .                           | 255        |
| 11.3.2             | Managing PowerVM and KVM virtual appliances in IBM SmartCloud Entry . . . . .                              | 255        |
| 11.3.3             | Importing OVA into VMware vSphere with vCenter . . . . .   | 258        |



|   |            |
|---|------------|
| 11.3.4 Managing virtual appliances in IBM SmartCloud Entry . . . . .                          | 262        |
| <b>Chapter 12. Proven practices . . . . .</b>   | <b>265</b> |
| 12.1 Virtual appliance naming convention . . . . .  | 266        |
| 12.1.1 Image name . . . . .   | 266        |
| 12.1.2 Universal ID . . . . .   | 266        |
| 12.1.3 Version . . . . .  | 267        |
| 12.2 Software bundle naming convention . . . . .  | 267        |
| 12.2.1 Bundle name . . . . .  | 267        |
| 12.2.2 Universal ID . . . . .   | 268        |
| 12.2.3 Version . . . . .  | 268        |
| 12.3 Preinstalling software or creating a bundle with the installation image . . . . .        | 269        |
| 12.4 Best practices for software bundles . . . . .  | 270        |
| 12.5 Writing shell scripts for software bundle operations . . . . .                           | 271        |
| 12.5.1 Best practices when writing shell scripts for IBM i . . . . .                          | 274        |
| 12.6 Testing the product activation . . . . .   | 278        |
| 12.6.1 Testing the product activation on IBM i . . . . .                                      | 278        |
| 12.7 Split large virtual images . . . . .   | 279        |
| 12.7.1 Validating files when splitting an OVA archive . . . . .                               | 280        |
| <b>Appendix A. Sample script for KVM host setup . . . . .</b>                                 | <b>281</b> |
| <b>Appendix B. Sample scripts for simple virtual appliance . . . . .</b>                      | <b>283</b> |
| Installation script for WebSphere Application Server Community Edition . . . . .              | 284        |
| Configuration script for WebSphere Application Server Community Edition . . . . .             | 286        |
| Configuration script for PlantsByWebSphere . . . . .  | 287        |
| <b>Appendix C. Editing the generic.ovf file for IBM SmartCloud Entry deployment . . . . .</b> | <b>289</b> |
| <b>Appendix D. Additional material . . . . .</b>  | <b>293</b> |
| Locating the web material . . . . .   | 293        |
| Using the web material . . . . .  | 293        |
| System requirements for downloading the web material . . . . .                                | 293        |
| Downloading and extracting the web material . . . . .   | 294        |
| <b>Abbreviations and acronyms . . . . .</b>   | <b>295</b> |
| <b>Related publications . . . . .</b>   | <b>297</b> |
| IBM Redbooks . . . . .  | 297        |
| Online resources . . . . .  | 297        |
| Help from IBM . . . . .   | 298        |



# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.


## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

# Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

|                           |                 |   |
|---------------------------|-----------------|---|
| AIX®                      | IBM PureData™   | PureApplication™  |
| BladeCenter®              | IBM SmartCloud® | PureData™   |
| DB2®                      | Lotus®          | PureFlex™   |
| developerWorks®           | PartnerWorld®   | PureSystems™  |
| Domino®                   | POWER®          | Redbooks®   |
| Express Services™         | Power Systems™  | Redbooks (logo)  ® |
| FlashCopy®                | POWER6®         | Smarter Planet®   |
| Global Business Services® | POWER7®         | Storwize®   |
| IBM®                      | POWER7+™        | System x®   |
| IBM Flex System™          | PowerLinux™     | Systems Director VMControl™   |
| IBM Flex System Manager™  | PowerVM®        | WebSphere®  |

The following terms are trademarks of other companies:

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

# Preface

In a traditional deployment model, software is installed on a physical server, and it is configured for the particular data center environment. The cloud deployment model requires that the dependency on a specific hardware configuration is severed. This IBM® Redbooks® publication guides you through the transition from the traditional application deployment model to the cloud-friendly deployment model. It explains how to achieve these goals by packaging the software stacks into industry standard virtual appliances.

A key part of this transition involves using the IBM Image Construction and Composition Tool. This tool is the IBM tool for creating virtualized workloads that target several private cloud deployment platforms, including platforms from IBM and not from IBM. In fact, this tool is unique in its ability to support such a wide range of cloud offerings. It is also the only tool in the marketplace that can create virtual appliances for both x86 and IBM Power hardware architectures.

This book provides an in-depth look at the capabilities and internal workings of the IBM Image Construction and Composition Tool. It focuses on the capabilities of the tool, which targets the virtualization and cloud offerings of IBM Systems and Technology Group. These offerings include IBM Systems Director VMControl™, IBM SmartCloud® Entry, and IBM PureFlex™ System with IBM Flex System Manager™ appliance. The IBM Image Construction and Composition Tool also has a much richer set of capabilities. Specifically, it supports IBM Workload Deployer, IBM PureApplication™ Systems, IBM SmartCloud Provisioning, and IBM SmartCloud Orchestrator, which are addressed in the IBM publications listed in Appendix , “Related publications” on page 297.

This publication targets software architects, cloud solutions architects, and cloud administrators. Its goal is to provide the expert-level skills required to package the existing and newly created applications into self-configurable, smart virtual appliances.

## Authors

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Rochester Center.

**Greg Hurlebaus** is a Senior IT Specialist with IBM in Rochester, MN. Greg is a member of the IBM Systems and Technology Group (STG) Systems Software organization. Greg works with independent software vendors (ISVs) and is a technical evangelist for the newest technologies. In his current assignment, Greg is the worldwide technical lead for the IBM Virtual Appliance Factory (VAF) for PowerVM®. His previous roles include 12 years in development, 13 years working directly with ISVs, and the ISV Technical Enablement Strategist for IBM Smart Business and IBM initiatives with Smarter Planet®. Greg received an IBM Outstanding Technical Achievement award for his leadership. Greg is a frequent speaker at numerous IBM, IBM Business Partner, and customer conferences around the world. He has written numerous technical articles and has added nine patents to the IBM portfolio.

**Rashed Ferdous** is a Staff Software Engineer at IBM. He is currently working on cloud-computing solutions. Before this assignment, Rashed worked on IBM Life Science development, and various IBM WebSphere® products. Rashed holds a Bachelor degree and a Master of Science degree in computer science from Saint Cloud State University, in St. Cloud, MN.

**John Jacobson** is a Lead IBM Technical Consultant in the IBM ISV and Developer Relations Group at the IBM Innovation Center in Waltham, MA. He has been with IBM for six years, supporting IBM Business Partners and ISVs in technical enablement of IBM software and hardware technologies. In his current role, John helps partners to learn technical skills in new technologies and assists them in enabling their solution on IBM PureSystems™. John holds a Bachelor of Science degree in Mathematics from the University of Massachusetts at Amherst.

**Li-Fang Lee** is an IBM Senior Technical Consultant at IBM STG ISV Enablement organization. She has over 20 years of experience in software design and development, product performance, and quality improvement. Her current focus area is IBM System x® virtualization and cloud computing.

**Jarek Miszczyk** is a Cloud Solutions Lead Architect in the IBM STG ISV Enablement organization. He has been with IBM for 20 years. His mission is to provide hands-on consulting services to ISVs, large IBM clients, and other IBM organizations on cloud-related issues. He also writes extensively and teaches IBM classes in all areas of cloud and virtualization. Previously Jarek worked at the IBM ITSO in Rochester, where he was a leading author of several IBM Redbooks publications. His areas of expertise include cloud computing, virtualization, and software solutions architecture.

**Pat Nickel** is an Advisory Software Engineer at IBM. He is the lead developer at STG for the IBM Image Creation and Composition Tool. He has over 28 years of experience with IBM in software design and development. Previous roles include systems software development for IBM i and Data Discovery and Query Builder application development for Life Sciences. Pat holds a Bachelor of Computer Science degree from the University of Minnesota Institute of Technology.

**David Peraza** leads the STG development team that develops the IBM Image Construction and Composition Tool. His areas of expertise include cloud computing, virtualization, and software design and architecture. He has also worked with the IBM development team that integrates IBM WebSphere MQ with IBM i HA capabilities and has lead projects in Smart Business Platform and Cloud Computing. David is the author of several articles that have been published in the *IBM Systems* magazine and IBM developerWorks®. Before joining IBM, David wrote software for a private EDI Trade system. David holds a Master of Science degree in Computer Science from Florida International University.

**Moises Romo** is an IT Specialist with IBM in Guadalajara, Mexico. He is a member of IBM Global Business Services® (GBS) working for the ISV Solutions Enablement organization. Moises has been with IBM for six years. In his current role, Moises supports ISVs in technical enablement to build smart virtual appliances for PowerVM. He holds a Bachelor degree in Computer Science and Electronics from the Universidad del Valle de Atemajac (UNIVA) and a Master degree in Information Technology from the Universidad Interamericana para el Desarrollo (UNID).

**Kerry Staples** is an IBM Senior IT Specialist in the System and Technology Group focused on ISV technical enablement. He supports the IBM Virtual Appliance Factory, which is a structured process to help ISVs with the creation of virtual appliances by using the IBM Image Construction and Composition Tool. In this role, he creates content about KVM and the Image Construction and Composition Tool to train ISVs and IBM technical professionals on creating virtual appliances that can be deployed on IBM cloud offerings. Kerry has 14 years of experience with IBM and 20 years of experience in the industry. He has supported IBM customers, ISVs, and IBM Business Partners on IBM hardware and IBM Lotus® applications. Kerry has a Bachelor of Science degree in Applied Studies.

Thanks to the following people for their contributions to this project:

**Marcela Adan**

IBM Redbooks Project Leader, International Technical Support Organization

**Amit Dave**

IBM Distinguished Engineer, STG Technical Sales, Middle East & Africa

## Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

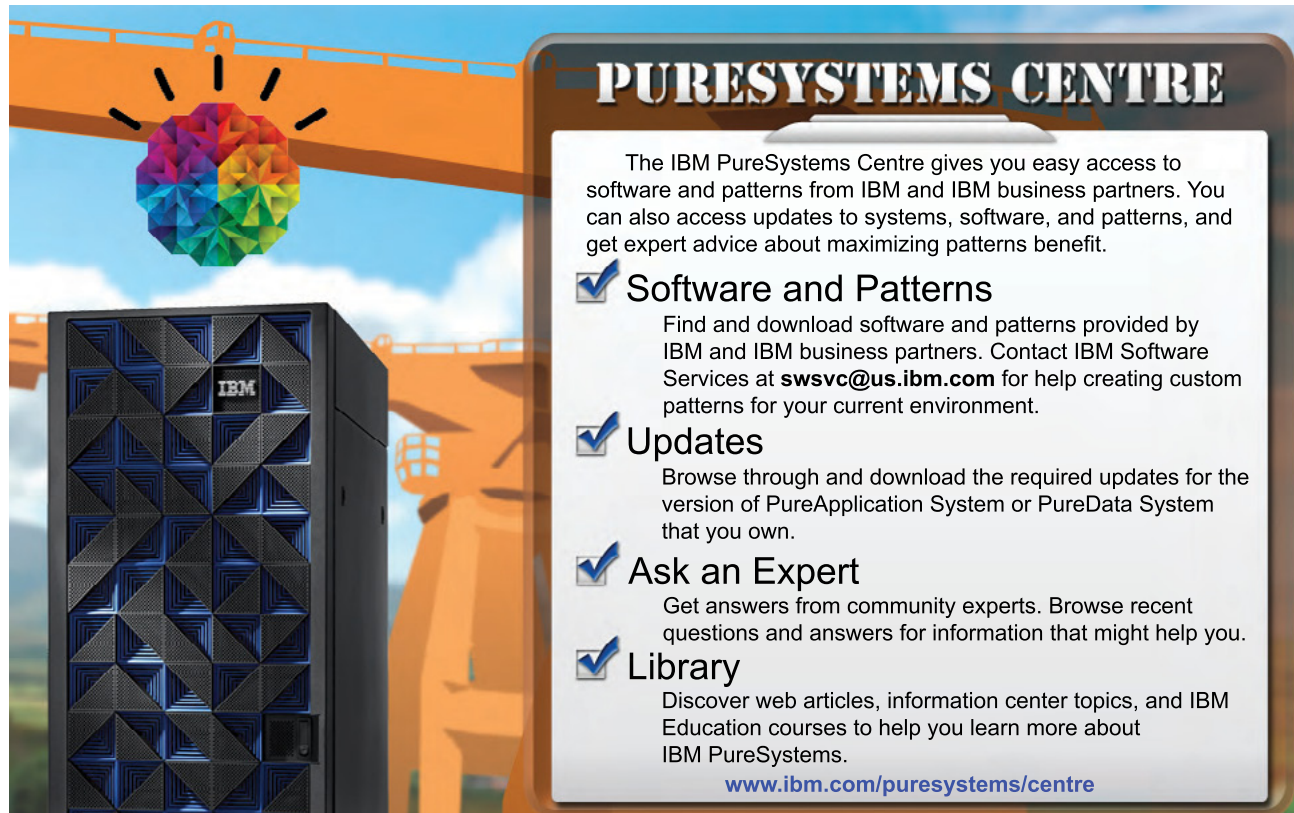
- Mail your comments to:

IBM Corporation, International Technical Support Organization  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks





- ▶ Find us on Facebook:  
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on Twitter:  
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:  
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:  
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:  
<http://www.redbooks.ibm.com/rss.html>



The graphic features a stylized background with a blue sky, white clouds, and a large orange bridge structure. A colorful, multi-faceted sphere is suspended from the bridge. In the foreground, there is a black IBM server rack with a blue geometric pattern on its front. To the right of the server is a large, dark brown rectangular box with a white border, containing the title 'PURESYSTEMS CENTRE' and a list of services.

## PURESYSTEMS CENTRE

The IBM PureSystems Centre gives you easy access to software and patterns from IBM and IBM business partners. You can also access updates to systems, software, and patterns, and get expert advice about maximizing patterns benefit.

-  **Software and Patterns**  
Find and download software and patterns provided by IBM and IBM business partners. Contact IBM Software Services at [swsvc@us.ibm.com](mailto:swsvc@us.ibm.com) for help creating custom patterns for your current environment.
-  **Updates**  
Browse through and download the required updates for the version of PureApplication System or PureData System that you own.
-  **Ask an Expert**  
Get answers from community experts. Browse recent questions and answers for information that might help you.
-  **Library**  
Discover web articles, information center topics, and IBM Education courses to help you learn more about IBM PureSystems.

[www.ibm.com/puresystems/centre](http://www.ibm.com/puresystems/centre)



# Introduction to virtual appliance construction

This chapter provides an overview of the basic concepts that are associated with the deployment of software solutions into a virtualized infrastructure. It articulates a need for a virtual appliance construction tool. In addition, it explains how to position the IBM Image Construction and Composition Tool (referred to as *the tool* in this book).

This chapter includes the following sections:

- ▶ Deploying your applications into the cloud
- ▶ Solving the terminology puzzle
- ▶ Why implement software virtual appliances
- ▶ A need for a virtual appliance construction tool
- ▶ Putting the Image Construction and Composition Tool into context

## 1.1 Deploying your applications into the cloud

Most IT professionals today understand that cloud computing constitutes a fundamental paradigm shift in how software solutions are marketed, distributed, and deployed. Three fundamental features of cloud computing contribute to its success in the marketplace:

- ▶ Self-service
- ▶ Automation
- ▶ Metering and Monitoring

The concept of *self-service* empowers users. Users can browse a catalog of prepackaged software solutions and decide when to deploy the workloads that satisfy their requirements. In this scenario, the data center provides the required services much in the same way as your public utilities company delivers electricity to your home. The data center administrators are no longer the gatekeepers; they become transparent to the users.

With a high level of *automation*, the companies can lower their data center operational expenses (OPEX) by orders of magnitude. To achieve these high levels of automation, however, the IT shops must adopt a new deployment model. In the traditional model, a solution provider, such as an independent software vendor (ISV), might send a specialist to a customer location to install or upgrade their solution. Often it is a cumbersome, time-consuming, and a tedious process that must be repeated at each customer location. Cloud computing simplifies this problem by implementing a solution that is known as a *virtual appliance*.

Another key feature in cloud computing is *metering and monitoring*. Metering and monitoring services in cloud computing track and calculate a customer's resource usage, which is the basis of the pay-per-use model. These services track how many resources and utilities each subscriber consumes so the service provider can charge subscribers appropriately.

IBM and other vendors in the virtualization space have adopted the Distributed Management Task Force (DMTF) Open Virtualization Format (OVF) specification. This specification is a standard way of building and deploying software virtual appliances.

The IBM Image Construction and Composition Tool is used by cloud solutions architects to build DMTF OVF-compliant virtual appliances. By hiding the intrinsic complexity of software stack installation and configuration, the tool has the potential to accelerate the adoption of the cloud deployment model in your organization.

## 1.2 Solving the terminology puzzle

Unfortunately, several heavily overloaded terms are used in cloud-related documentation. Their meaning differs depending on the vendor that is using them. Sometimes two teams within one company can give a term a different meaning. This Redbooks publication uses and defines the following most frequently used terms:

|                               |  |
|-------------------------------|--|
| <b>Self-configuration</b>     | A configuration that is achieved by including, in a virtual machine (VM), an activation engine or an agent that accepts the runtime parameters from the deployment platform and sets the change points to the deployment-specific values. A typical list of change points for the operating system (OS) can include the host name, IP address, and root password. Each software component in the image might require reconfiguration, not just the OS. |
| <b>Virtual image</b>          | A binary image of a VM. It can contain one or many virtual disk images. It contains an OS, and therefore, is bootable. It can also contain middleware and user applications. Typically, it is <i>not</i> self-configurable.  |
| <b>Virtual appliance</b>      | A self-configurable virtual image packaged in the DMTF OVF-compliant tape archive (tar) file. The standard file extension for the tar file is .ova. In addition to the binary images, the appliance contains an OVF file that describes the content of the appliances and all change points that require reconfiguration.  |
| <b>Virtual system</b>         | In the context of the DMTF OVF specification, a VM image in an appliance. In this book, this term is used with virtual appliance patterns.   |
| <b>Virtual system pattern</b> | A collection of virtual appliances that are logically linked and deployed as a unit. The inter-appliance links are used to properly express the relationship between two or more appliances. For example, the IBM WebSphere Application Server needs to know where to find, and how to connect to, an IBM DB2® instance. A link defines how to set up and establish such a database connection.  |

**Virtual appliances:** The IBM Image Construction and Composition Tool supports the end-to-end virtual appliance creation process. The tool creates and manipulates virtual appliances. For legacy reasons, however, the tool refers to the objects that it handles as *virtual images*. Keep in mind that the entity you eventually export from the tool for deployment on one of the supported cloud platforms is an OVF-compliant virtual appliance.

## 1.3 Why implement software virtual appliances

The software virtual appliances can encourage quicker transition to the cloud deployment model by dramatically lowering the initial requirements for in-depth virtualization skills and addressing the key issues that previously hindered adoption of the virtualized solutions. Implementing software virtual appliances has the following advantages:

- ▶ Achieve a much faster time to value. The software stacks that are packaged as virtual appliances can be deployed within minutes rather than days or weeks.
- ▶ Provide a standard way to accompany the executable image with contextual information. The metadata descriptor, called the *OVF file*, contains information about the target system hardware abstraction layer (HAL) requirements, such as how many virtual processors are needed or what the networking setup is. In addition, the OVF file can contain precise descriptions of the software components that constitute the appliance. For example, separate sections might describe the attributes of the OS, the middleware, and the user applications.
- ▶ Eliminate the need for the post-deployment application reconfiguration. Data center personnel often do not have the domain knowledge necessary to, for example, reconfigure the data source object in the WebSphere Application Server that points to the back-end database server that is available in the data center. This need is eliminated because a smart appliance can self-activate the entire software stack upon deployment.
- ▶ Support software topologies that require multiple VMs. The realistic multi-tier software architectures can be properly represented, with each tier mapped to a separate VM image. As mentioned, virtual appliances can be assembled into patterns.
- ▶ Achieve vendor independence. The appliances can be ported across management stacks, making migration among virtualization vendors and among data centers fairly easy and nondisruptive. However, the fundamental building block of an appliance, which is the virtual disk image, must comply with the target hypervisor requirements. For example, you cannot expect an IBM PowerVM image to be deployed unchanged on a VMware hypervisor. The base image is platform-dependent. Vendor independence can be achieved by adopting a software stack installation and activation methodology that works unchanged on all target deployment platforms. The Image Construction and Composition Tool shines in this area.

## 1.4 A need for a virtual appliance construction tool

By now you understand the tremendous possibilities that are offered by the software virtual appliance deployment model. Now see how the Image Construction and Composition Tool can help you to implement software virtual appliances.

### 1.4.1 Application packaging options

You might ask yourself: How do I package my applications into this format so that they become cloud-ready? Several options are possible:

- ▶ Build the software virtual appliance manually by using the open source tooling. For example, you can install, on your Linux workstation, the open source hypervisor KVM, and the virtualization tools, such as **virsh**, **virt-manager**, and **virt-viewer**. You might then use **virt-manager** to create the VM images, boot those images on KVM, and install all the necessary software components. A simple text editor can then be used to create or modify an OVF file to describe the content of the VM images you created. The final step might be to create a tar file that contains the images and the OVF.

This manual process is easier said than done. Manual virtual appliance creation requires significant virtualization and OVF skills and probably is appealing only to early adopters. Therefore, keep reading to explore other options.

- ▶ Use vendor or hypervisor-specific tools, such as VMware Studio and the OVF Toolkit, that simplify the process of virtual appliance creation. The virtual appliances created in VMware Studio can be easily imported and deployed by using vSphere Client. The process is well-documented in the VMware publications. Keep in mind that VMware tools do not support open source hypervisors, such as KVM, or architectures other than Intel.
- ▶ Use the Image Construction and Composition Tool, which is a modern web-based application that creates DMTF OVF-compliant virtual appliances. This tool is explained in depth in the following section.

### 1.4.2 Supported deployment options

The Image Construction and Composition Tool is the IBM tool for virtual appliance creation. Its intuitive user interface is easy to use, and the tool hides the complexities of image creation and packaging. In addition, it supports an impressive range of deployment options:

- ▶ Multiple virtualization and cloud management platforms:
  - IBM SmartCloud Enterprise
  - IBM SmartCloud Entry
  - IBM SmartCloud Provisioning
  - IBM Systems Director VMControl
  - IBM Workload Deployer
  - VMware vSphere or vCenter
- ▶ Multiple target hardware architectures:
  - IBM Power
  - Intel x86
- ▶ Multiple target hypervisors:
  - KVM on x86
  - VMware ESX/ESXi
  - IBM PowerVM

Figure 1-1 illustrates these concepts.

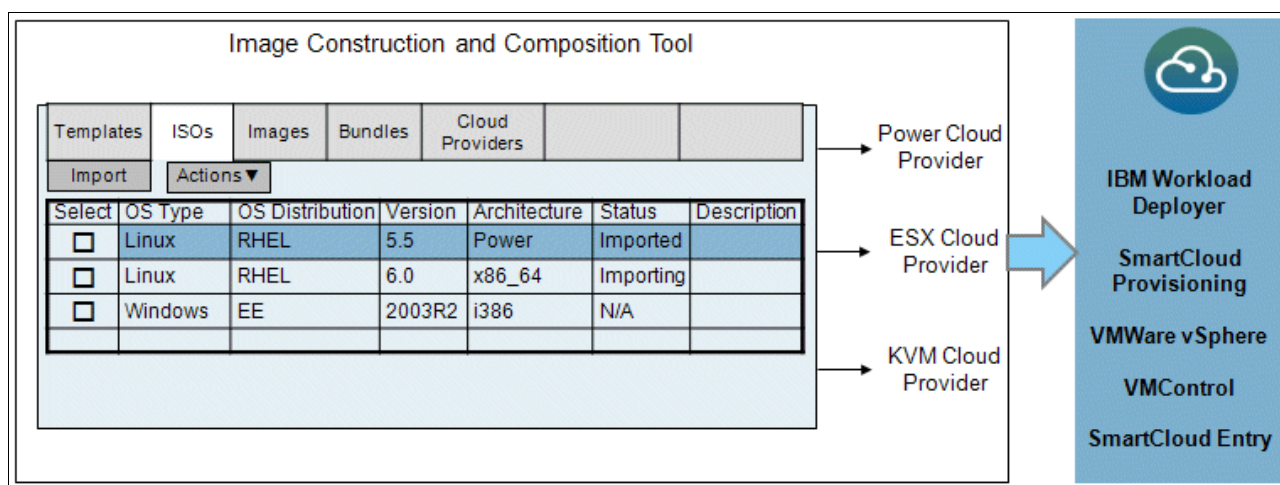


Figure 1-1 Overview of Image Construction and Composition Tool functionality

In Figure 1-1, a cloud specialist uses the Image Construction and Composition Tool UI to manipulate virtual images. The tool communicates with the target hypervisor by using an abstraction layer called a *cloud provider*. One cloud provider is available for each supported hypervisor. The cloud provider is used to build in an iterative fashion and to export virtual appliances in one of the supported formats. The precise content of a virtual appliance exported from the tool is dictated by the target platform requirements. Thus a virtual appliance that is to be deployed on IBM Systems Director with Systems Director VMControl contains OVF metadata with extensions that are required by that deployment platform. For more information about the internal workings of the tool, see Chapter 3, “Architecture of the IBM Image Construction and Composition Tool” on page 21.

By using the Image Construction and Composition Tool, you can import a running VM as a *base image*. Another option is to build a base image from an installable OS ISO image. As of this writing, only the KVM Provider supports a build from the ISO option.

In addition, by using the Image Construction and Composition Tool, you can package software components, such as middleware or a user application, into self-described containers called *bundles*. The base image is then extended with one or more bundles to create a deployable image. The tool synchronizes (merges) the content of a base image with the software components that are in bundles.

The build and composition process is always performed in the context of a cloud provider. A *cloud provider* is an abstraction layer that isolates the Image Construction and Composition Tool from peculiarities of any hypervisor. For example, the KVM cloud provider communicates with a target KVM hypervisor to perform basic image management tasks such as to provision a VM, start it, and stop it.

After the image is synchronized (bundles installed in the base image), the Image Construction and Composition Tool can capture and export the final virtual appliances. The target cloud provider dictates which metadata and possibly other necessary component are packaged into that appliance. The tool masks the syntax and semantics OVF differences between the various target deployment options. It is programmed to detect what additional, platform-specific artifacts must be included in the final appliance.

## 1.5 Putting the Image Construction and Composition Tool into context

The virtual appliances that are constructed by the Image Construction and Composition Tool (the *tool*) can be deployed directly to a virtualization platform, such as IBM PureFlex System with an IBM Flex System™ Manager node or IBM Systems Director with Systems Director VMControl. VMware images can also be imported into other tools, such as IBM Workload Deployer, to create more complex and functional deployment landscapes. The tool can also create base PowerVM images, with the Activation Engine installed, to import into IBM Workload Deployer. The following sections examine these options more closely.

### 1.5.1 Image Construction and Composition Tool and the IBM PureSystems

The IBM PureSystems are a new breed of computer systems that combine the flexibility of general-purpose systems and the simplicity of an appliance with integrated expertise. IBM PureSystems are cloud-ready, expert, integrated systems. They provide a new computing paradigm that consolidates workloads across IBM Power, IBM System x, systems management, networking, and storage with a unified management system that allows integration of the infrastructure and the application layer. As of this writing, the PureSystems family has three members:

- ▶ IBM PureFlex System

When included with the Flex System Manager node, this system constitutes an infrastructure as a service (IaaS) platform. The virtual appliances that are built into the Image Construction and Composition Tool can be imported easily through the Flex System Manager console to the Appliance Repository. The appliances that are built by the tool are fully compliant with Flex System Manager and the underlying management stack (IBM Systems Director with the Systems Director VMControl plug-in). Therefore, they are ready for deployment as soon as the import successfully completes.

- ▶ IBM PureApplication System

This system constitutes the IBM platform as a service (PaaS) offering, with built-in platform management capability, and preinstalled middleware and preintegrated deployment patterns. The virtual appliances that are built by the tool can be imported into the image library and then be used to assemble sophisticated and highly functional virtual appliance patterns.

- ▶ IBM PureData™ System

The PureData System is optimized exclusively for delivering data services to today's demanding applications. It offers built-in expertise, integration by design, and a simplified experience throughout its lifecycle. It is offered in four designs (PureData System for Transactions, PureData System for Analytics, PureData System for Operational Analytics, and PureData System for Hadoop) that are specifically optimized to address specific data workload requirements.



Figure 1-2 illustrates, in the context of PureSystems, a typical progression from simple images through more functional appliances to highly functional virtual appliance and virtual application patterns.

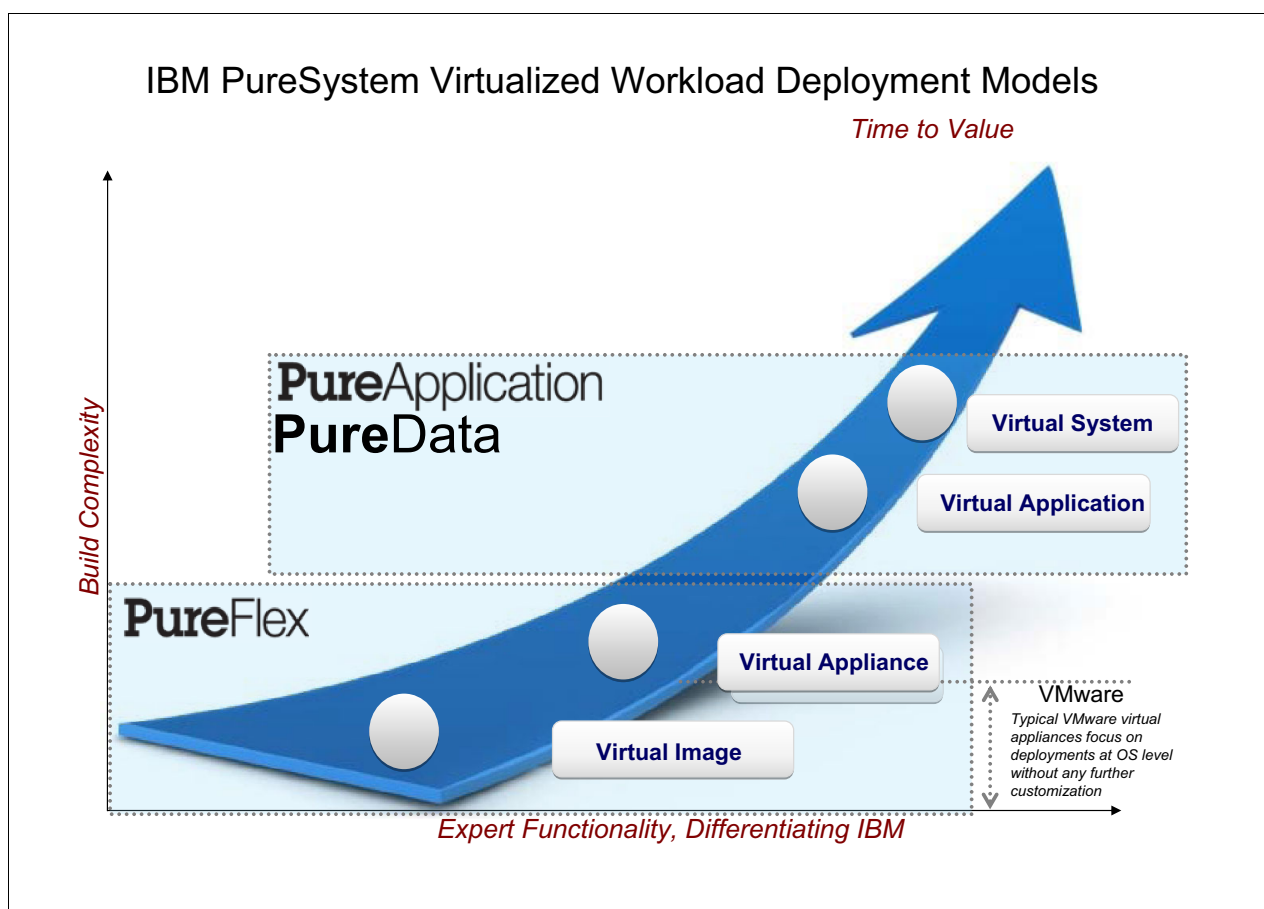


Figure 1-2 PureSystem virtualized workload deployment models

Virtual images can be built manually or with tools, such as VMware Studio, and then deployed to a PureFlex system directly through a vSphere Client. Typically, such images have the VMware tools installed so that the OS-level change points can be automatically reconfigured. The rest of the software stack (middleware, user applications) will have to be configured after deployment.

Alternatively, the virtual appliances that are created in the Image Construction and Composition Tool are self-configurable. As mentioned previously, the tool can build virtual appliances that are deployable through vSphere or IBM Flex System Manager™.

The PureApplication System offers the ability to construct and deploy finely tuned and highly functional virtual appliance and virtual application patterns. The virtual application patterns constitute an application-centric approach to cloud workload creation and deployment. Therefore, they are not covered in this book. For more information about this concept, see *IBM Workload Deployer: Pattern-based Application and Middleware Deployments in a Private Cloud*, SG24-8011.

## 1.5.2 Image Construction and Composition Tool and the IBM Workload Deployer

With all its strengths and rich functionality, the Image Construction and Composition Tool might not be the only tool that you will ever need to create cloud workloads.

The tool architects made a conscious decision not to support virtual appliance patterns. As mentioned, virtual appliance patterns are needed when you want to deploy complex software architectures such as WebSphere Application Server. In a typical use case, a WebSphere Application Server deployment might require a Deployment Manager node, several custom nodes, HTTP Server nodes, and one or more DB2 nodes. Each node can be implemented as a single-virtual-machine appliance. Then a solution architect uses a pattern editor to assemble a collection of virtual appliances into a virtual appliance pattern.

The IBM Workload Deployer implements a fully functional pattern editor that can use virtual appliances created in the Image Construction and Composition Tool.

The following example illustrates a typical use case for virtual appliance patterns:

1. Import a running VM image into the Image Construction and Composition Tool, which creates a base virtual appliance (image) that can be extended.
2. Extend the imported images with bundles to add the desired software components such as middleware and user applications.
3. Export the resulting, synchronized image in a standard compliant Open Virtual Appliance (OVA) archive (an OVA tar file).
4. Import the OVA tar file into the IBM Workload Deployer virtual image catalog.
5. Use the IBM Workload Deployer Pattern Editor to compose a virtual appliance deployment pattern.
6. Deploy the newly created pattern to the private cloud managed by IBM Workload Deployer.

Figure 1-3 on page 10 illustrates this typical scenario.

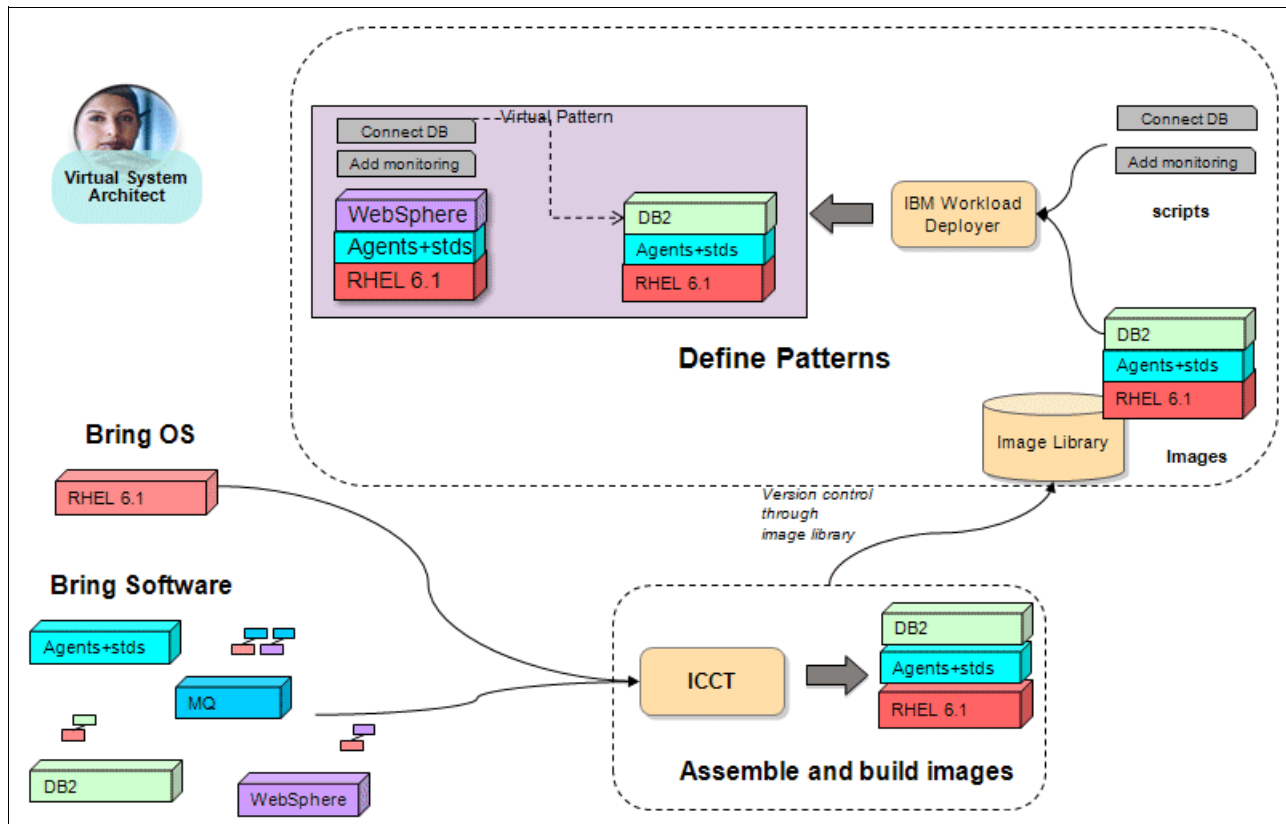


Figure 1-3 Creating virtual appliance patterns

After tackling the broader topics associated with the cloud workloads construction and deployment, the following chapter highlights the details of a virtual appliance. After all, the Image Construction and Composition Tool is in the business of virtual appliance creation.



## Anatomy of a virtual appliance

Virtual appliances are a key component of the cloud deployment model. The virtual images that are created by the IBM Image Construction and Composition Tool are virtual appliances, from the Distributed Management Task Force (DMTF) Open Virtualization Format (OVF) specification point of view.

The Image Construction and Composition Tool (referred to as *the tool* in this book) automatically handles most of the mundane and error-prone tasks such as creating an OVF, installing an activation engine, and packaging an appliance into a virtual appliance. This chapter explains the processes handled by the tool. Consequently, this chapter helps you to resolve even the most complex and esoteric issues.

This chapter describes the anatomy of virtual appliances. It highlights the key artifacts in a virtual appliance and how they relate to each other. This chapter includes the following sections:

- ▶ Description of a virtual appliance
- ▶ Components of a virtual appliance
- ▶ Virtual appliance self-activation
- ▶ IBM Virtual Solutions Activation Engine
- ▶ What a virtual appliance is not

**Attribution:** Parts of this chapter are reprinted with permission by MC Press Online, LLC in Ketchum, Idaho:

<http://www.mcpressonline.com>

## 2.1 Description of a virtual appliance

A *virtual appliance* is a prepackaged software stack that combines the operating system (OS), middleware, and applications in one package. Virtual appliances facilitate a quicker transition to cloud and require much less installation and configuration than traditional deployment methods. Virtual appliances address key issues to cloud computing and standardization. They apply to traditional independent software vendors (ISVs) and software as a service (SaaS) providers.

As an example, at one end of the spectrum, with little effort, you can package an existing matured, market-tested COBOL application as a single-image virtual appliance, which immediately becomes deployable into the cloud. At the other end of the spectrum, you can package a newly implemented, highly distributed, service-oriented application and integrate with the services provided by the cloud. By using this method, ISVs can respond more rapidly to the changing business needs of their customer, with flexibility and agility.

IBM and many other vendors in the virtualization space have adopted the DMTF OVF specification as a standard way to build and deploy software virtual appliances.

## 2.2 Components of a virtual appliance

According to the DMTF OVF specification, a software virtual appliance can consist of one or many virtual systems. Each virtual system can contain one or many virtual disks. The relationships between the components and their attributes are described in the OVF file. All of the components, including the binary images of the virtual disks and the corresponding OVF file, are packaged in a tape archive (tar) file. The recommended file name extension for the tar file is .ova, which refers to the Open Virtualization Format Appliance (OVA). This tar file becomes a software virtual appliance. In its most basic form, a software virtual appliance consists of a single virtual server that contains one virtual disk image and a corresponding OVF.

**Composite virtual appliances:** Currently the Image Construction and Composition Tool and the deployment platforms, such as Systems Director VMControl, do not support composite virtual appliances, which are appliances with multiple virtual machine (VM) images. Rather, IBM implemented a more flexible architecture based on virtual system patterns. A virtual system pattern consists of one or many single virtual appliances. For more information, see 1.2, “Solving the terminology puzzle” on page 3.

Figure 2-1 illustrates the content of a sample virtual appliance that targets KVM as the deployment hypervisor.

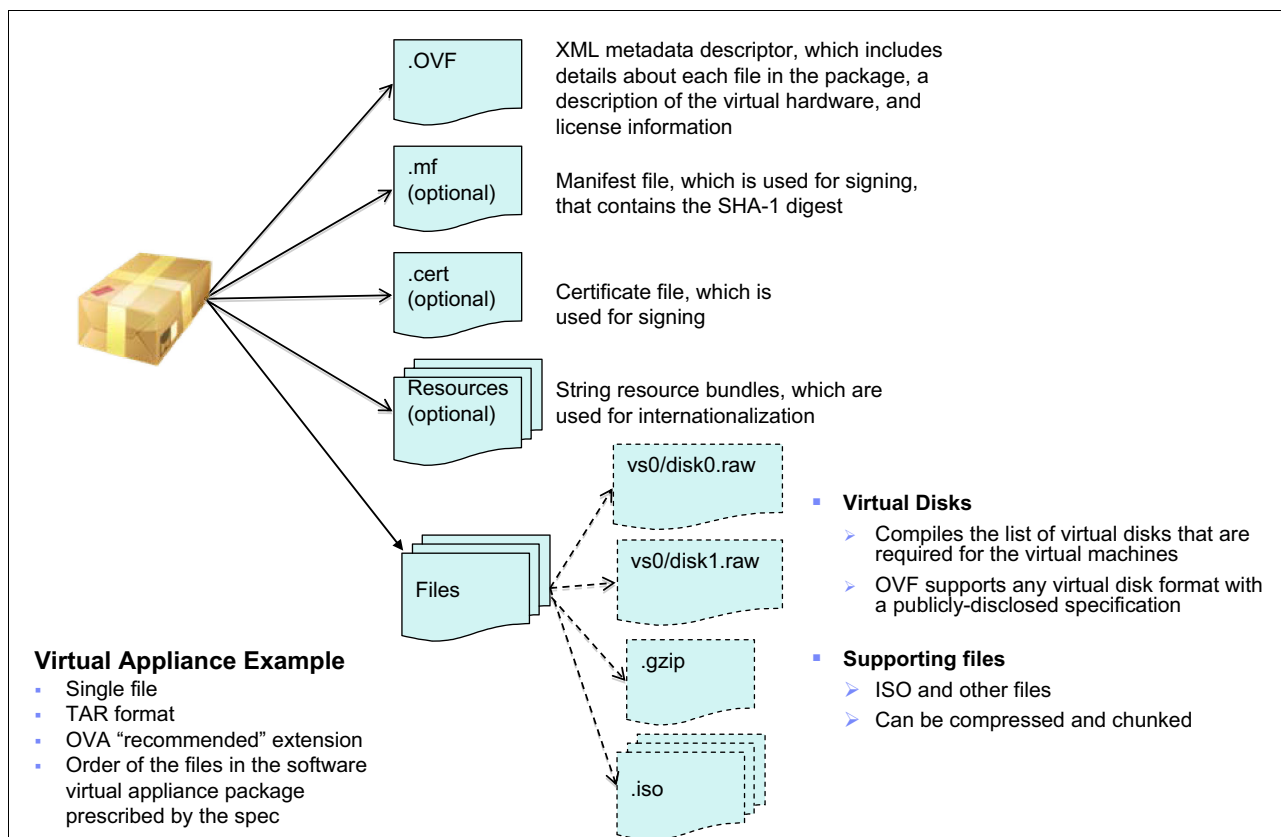


Figure 2-1 Anatomy of a sample DMTF OVF-compliant virtual appliance

The virtual appliance archive shown in Figure 2-1 consists of several components, including the following key components:

|                         |  |
|-------------------------|--|
| <b>OVF file</b>         | An XML document that contains the metadata that describes the content of the software virtual appliance. This file must be present as the first file in the root directory of the archive.   |
| <b>Manifest file</b>    | A file that is used to digitally sign the software virtual appliance. A digital signature guarantees the integrity and the security of the content. The cloud deployment platform rejects the deployment of an appliance with a compromised digital signature.   |
| <b>Virtual disks</b>    | A binary representation of a disk that is needed by a virtual system to boot and function properly. For example, the first disk is a bootable image of the OS, and the second disk is the data disk. Each virtual system might require one or many virtual disk images. The DMTF OVF standard does not deal with the virtual disk formats. It requires only that the format is published. In the previous example, the raw disk format is used, which is supported by the open source hypervisor called KVM. |
| <b>Supporting files</b> | A set of software artifacts that need to be loaded (in certain cases) when the software virtual appliance is started for the first time. These artifacts can be packaged with the software virtual appliance as ISO images or as compressed file archives.   |

The virtual appliances that are created by Image Construction and Composition Tool consist of one or many virtual disk images and the corresponding metadata. The certificate and the manifest files are currently not used, which might change in future versions of the tool.

## 2.2.1 The OVF file

The OVF file is a critical part of every DMTF-compliant software virtual appliance. The cloud deployment management system parses its content to retrieve the information that is necessary for successful activation of the software solution in an appliance. The OVF file is an XML document that contains several sections. Each section describes a different aspect of the appliance. The following sections highlight the most important parts of a typical OVF file.

### Disk Section

The Disk Section defines all virtual disk images at the global (virtual appliance) level. Example 2-1 is an XML snippet that shows a typical disk section for a simple, single virtual-system, single-disk appliance.

*Example 2-1 OVF Disk Section*

---

```
<ovf:References>
  <ovf:File ovf:href="vs0/disk0.raw" ovf:id="disk0"
ovf:size="10737418240" />
</ovf:References>
<ovf:DiskSection>
  <ovf:Info>Describes the set of virtual disks</ovf:Info>
  <ovf:Disk ovf:capacity="10737418240"                                1
ovf:capacityAllocationUnits="byte"
ovf:diskId="vs0/vd0"
ovf:fileRef="disk0 "                                                  2
    ovf:format="http://www.ibm.com/xmlns/ovf/diskformat/qemu.raw" 3
    ovf:populatedSize="10737418240" />
</ovf:DiskSection>
```

---

The numbered lines in Example 2-1 correspond to the following explanations:

- 1.** The virtual disk capacity is specified.
- 2.** A file reference to the disk image location is defined. The deployment platform uses this information to locate the file within the appliance tar file that contains the binary image of a given virtual disk.
- 3.** A link to the disk format specification is provided.

### Network Section

The Network Section provides all network definitions at the global level, listing the required private and public networks that are used in the software virtual appliance.

**Collection element:** Because the Image Construction and Composition Tool does not support composite virtual appliances, the Collection element of the virtual system is not present in the OVFs that are created by the tool.

## Virtual System

The Virtual System Section defines the metadata for each of the VMs that make up the appliance. The Virtual System Section contains the following sections:

**EULA Section** Provides the license agreement for the virtual server.

**Support by Systems Director VMControl:** As of writing this book, the EULA section is not supported by Systems Director VMControl.

**Hardware Section** Defines the virtual hardware to be used by a virtual driver. It needs to be in the scope of the hardware abstraction layer (HAL) of the hypervisor that is chosen for deployment. Typically, this section describes the following aspects of HAL, which include network (virtual or physical), capacity of virtual servers that will boot from images (memory and CPU), storage controller, and devices.

**Product Section** Defines the application, service, and static information that needs to be configured at the deployment of the appliance. For example, the local host name must be changed in the DB2 server configuration. Example 2-2 shows an example of this section.

*Example 2-2 Product Section*

---

```
<ovf:ProductSection ovf:class="com.ibm.vsa.2_1.db2-ese95-config"> 1
  <ovf:Info>DB2 Version 9.5 Enterprise edition</ovf:Info>
  <ovf:Product>activate-db2-base</ovf:Product>
  <ovf:Version>9.5</ovf:Version>
  <ovf:Property ovf:key="db2_hostname" ovf:type="string" 2
    ovf:userConfigurable="true" ovf:value="localhost">
      <ovf:Description>DB2 host name</ovf:Description>
      <ovf:Label>DB2 host name</ovf:Label>
    </ovf:Property>
    <ovf:Property ovf:key="db2inst1_username" ovf:type="string"
      ovf:userConfigurable="false" ovf:value="db2inst1">
      <ovf:Description>Instance user</ovf:Description>
      <ovf:Label>Instance user</ovf:Label>
    </ovf:Property>
    <ovf:Property ovf:key="db2inst1_password" ovf:type="string"
      ovf:userConfigurable="true" ovf:value="">
      <ovf:Description>Instance password</ovf:Description>
      <ovf:Label>Instance password</ovf:Label>
    </ovf:Property>
  </ovf:ProductSection>
```

---

The numbered lines in Example 2-2 correspond to the following explanations:

- 1.** A Product Section is uniquely identified within the scope of an OVF with the class attribute. In this section, the activation engine can locate the proper activation program to which to pass the hostname parameter and avoid the name conflicts for the same-named parameters that are used by different activation programs. The Product Section also contains the elements that describe the product that is being configured. You can provide a product description, version, and information.
- 2.** The **db2\_hostname** property is defined. This property is passed by the virtualization platform, such as Systems Director VMControl, to the VM at the time that the appliance is deployed. Notice the presence of the **userConfigurable** attribute. This attribute controls whether this property is shown at deployment time so that the deployer can provide a runtime-specific value.



## 2.3 Virtual appliance self-activation

The true value of the OVF-compliant software virtual appliances lies in the ability to automatically install and configure all of the software components, including the OS, middleware, and user applications. Unfortunately, the prevailing approach of many appliance architects is to ship them incomplete, without automating the activation of the entire software stack. Often the appliance architect uses, for example, the VMware tools to ensure the reconfiguration of the OS-level settings, such as host name, domain name, and root password. Appliance architects tend not to address the reconfiguration needs of the software components above the OS at the initial deployment time. For example, they do not reconfigure the database settings. Therefore, at deployment time, the host name at the OS level is set to the correct value, but the database configuration shows the original host name. This conflict can lead to unpredictable results and often requires manual debug and repair.

This approach is problematic for several reasons:

- ▶ The need for a manual configuration of deployed workloads defeats the key value proposition of cloud computing, namely automation.
- ▶ The deployer (a cloud administrator) might not have the necessary domain knowledge to properly reconfigure a third-party application or middleware or the interdependencies of a multi-tiered application construct.

## 2.4 IBM Virtual Solutions Activation Engine

A preferred solution to address these issues is to take advantage of the IBM Virtual Solutions Activation Engine (VSAE). This VSAE is a software tool that is used during the appliance creation and deployment phases. The Image Construction and Composition Tool automatically installs the VSAE in a virtual image, when the image is created from ISO or when it is imported from a running VM. (For more information, see Chapter 6, “KVM Express cloud provider” on page 91, Chapter 7, “PowerVM Express cloud provider” on page 125, and Chapter 8, “ESX cloud provider” on page 157.) Then, when the appliance is deployed, the VSAE runs in each of the virtual appliances and reconfigures the OS, the middleware, and the installed applications.

Essentially, the VSAE is a scripting engine that starts on the first boot before the application services are activated. For example, the data source configuration for WebSphere Application Server requires the DB2 server host name, the port on which DB2 is listening, the database name, and the DB2 user credentials. These settings might be different for each instance and are runtime environment-specific. They need to be set before the WebSphere Application Server application startup to prevent conflicts and security exposure.

The OVF standard recommends that the runtime parameters are passed to the VSAE by using an XML configuration file (with a default name of `ovf-env.xml`) in a virtual CD drive. The virtual CD drive is attached to a VM at the boot time. The VSAE parses the XML configuration file to retrieve the parameters and then calls a specific activation program that sets points of variability (changeable parameters) in the application stack.

Figure 2-2 illustrates these concepts of the VSAE.

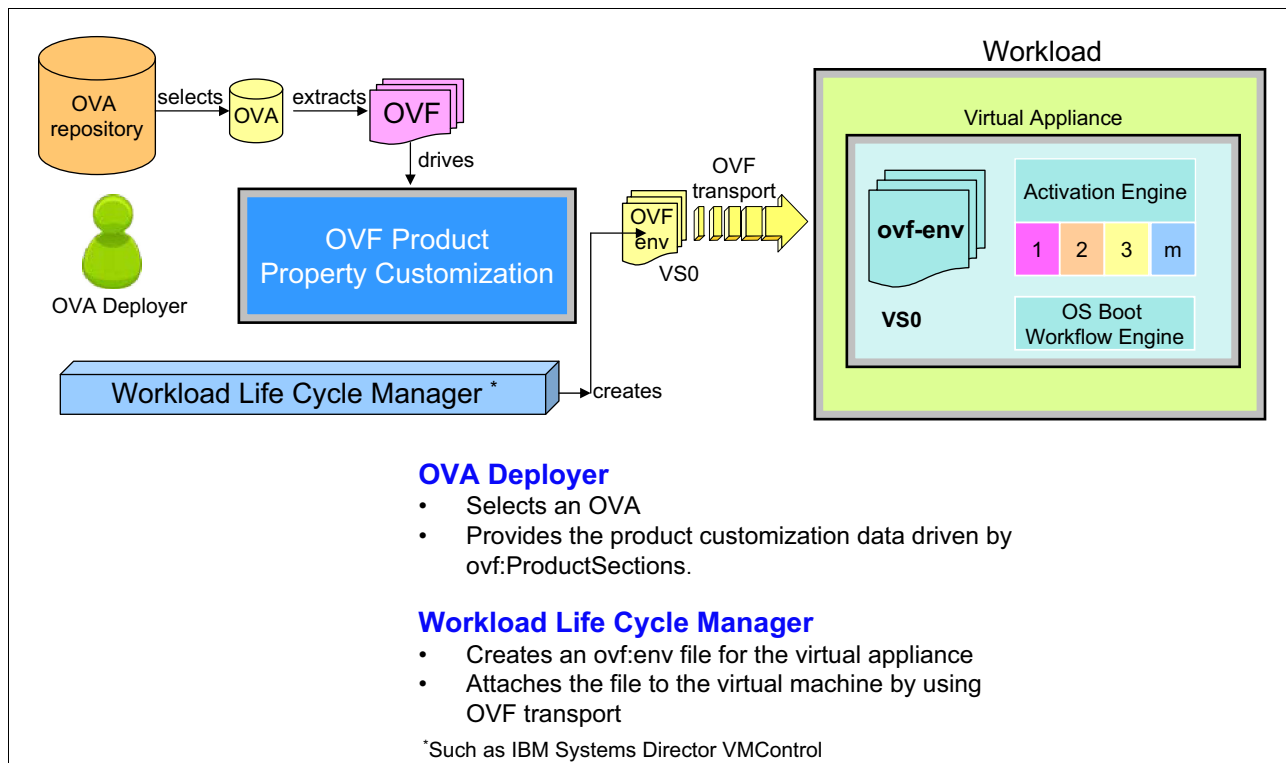


Figure 2-2 Product activation flow

In the activation flow, for example, an activation program that reconfigures a DB2 instance requires several runtime-specific parameters that need to be passed by the deployment platform using the `ovf-env.xml` transport mechanism. Example 2-3 shows an excerpt from the `ovf-env.xml` file that illustrates how the DB2 runtime parameters are passed to the VSAE.

Example 2-3 Content from the `ovf-env.xml` file

```
<PropertySection>
<Property ovfenv:key="com.ibm.vsa.2_1.db2-ese95-config.db2inst1_password"
  ovfenv:value="s3cr3t" />
<Property ovfenv:key="com.ibm.vsa.2_1.db2-ese95-config.db2inst1_username"
  ovfenv:value="db2inst1" />
<Property ovfenv:key="com.ibm.vsa.2_1.db2-ese95-config.db2_hostname"
  ovfenv:value="vs102" />
/PropertySection>
```

The line in bold highlighting in Example 2-3 shows that the **db2\_hostname** parameter is passed with the "vs102" value. The fully qualified parameter name is used, which allows the VSAE to locate the correct activation program to which to pass the hostname parameter. Therefore, it avoids name conflicts for the same-named parameters that are used by different activation programs. For example, the WebSphere Application Server activation might also require a same-named **db2\_hostname** parameter to properly reset the server name in the data source object.

Generally, the value of the parameter can be set at appliance construction time or deployment time:

- ▶ At the appliance construction time, the appliance architect can use the OVF Product Section for DB2 to preset the parameters to their default values. See Example 2-3 on page 17.
- ▶ At the appliance deployment time, the deployment platform identifies all the parameters that have the **userConfigurable** attribute set to true. The parameters are presented to the deployer at one of the deployment steps, at which time they can be changed to the values that reflect the current runtime environment. For example, the **db2\_hostname** is set to `vs102`, which overrides the default value of `localhost`. The deployer can be a human being or the cloud management software, such as the appliance placement services.

As mentioned, regardless of the method used to set the parameter, the parameter is eventually copied into the `ovf-env.xml` file. If the recommended transport mechanism is used, this file is in the root directory of a virtual CD drive that is attached to the VM at deployment time (see Figure 2-2 on page 17).

Example 2-4 shows an excerpt from the `env-ovf.xml` file that corresponds to the **db2\_hostname** parameter (property) as explained in this section.

*Example 2-4 Excerpt from the env-ovf.xml file that corresponds to the db2\_hostname parameter*

---

```
<Property ovfenv:key="com.ibm.vsa.2_1.db2-ese95-config.db2_hostname"
  ovfenv:value="vs102" />
```

---

The value of the **ovfenv:key** attribute matches the value of **ovf:class** attribute in the corresponding OVF file (see Example 2-2 on page 15).

The next step is for the VSAE to retrieve this value and pass it as a parameter to the activation program. But how does the VSAE know which program to call and how to pass the parameter? The activation logic controls this process.

The Image Construction and Composition Tool automatically generates the activation logic for the standard activation programs that are installed into a VM along with the core VSAE module. The standard activation programs provide functions that are roughly equivalent to the VMware Tools. They are responsible for reconfiguring the critical aspects of the guest OS such as the network interface and root password. For example, the following overview shows the activation programs that are installed by the tool in a KVM-based virtual appliance:

|                          |   |
|--------------------------|---|
| <b>network-interface</b> | Configures the network interface, <code>eth0</code> , by default. It supports both static and dynamic IP.     |
| <b>system-host</b>       | Sets the system host name and domain name.  |
| <b>dns-client</b>        | Sets up the Domain Name System (DNS) server IP addresses and the domain search list.                          |
| <b>system-user</b>       | Sets up the administrative user (root), changes the password, and sets the home directory, among other tasks. |
| <b>ntp-client</b>        | Points the ntp client to the ntp servers to synchronize the guest OS clock.                                   |

In addition, each software component in the virtual appliance software stack might require a corresponding activation program that knows how to reconfigure that component.

As mentioned, the sequence in which these programs are started by the VSAE, and the required parameters, are specified in an activation logic file. Example 2-5 shows an activation logic entry for the activation program that handles the reconfiguration of the DB2 instance.

*Example 2-5 Activation logic file*

---

```
<?xml version="1.0" encoding="UTF-8"?>
<a12:Activation xmlns:a12="http://www.ibm.com/xmlns/ovf/activation/a1/2">
  <VirtualSystem id="vs0">
    <ProductActivation class="com.ibm.vsa.2_1.db2-ese95-config">
      <Service name="activate.db2-config">
        <RunLevel value="5" />
        <RunLevel value="3" />
      </Service>
      <Program cmdOptionStyle="longSpace" envVars="false"
        href="AS/db2-config/activate.py" />
      <OSServiceDependency serviceName="network" start="after" />
      <ResetProgram href="AS/db2-config/reset.py" />
      <Properties>
        <Property key="db2inst1_password" />
        <Property key="db2inst1_username" />
        <Property key="db2_hostname" />
      </Properties>
    </ProductActivation>
  </VirtualSystem>
</a12:Activation>
```

---

The numbered lines in Example 2-5 correspond to the following explanations:

- 1.** The virtual system ID attribute value must match the **ovf:id** attribute in the Virtual System element in the corresponding OVF file. If these two values do not match, the VSAE does not start any of the activation programs that are defined in this activation logic.
- 2.** The activation section is defined for a Product Section in the OVF. Again, the match is made by using the value of the **class** attribute.
- 3.** A reference to the activation program is specified.
- 4.** The invocation order is defined. In this case, the activation program is called by VSAE *after* the network service is started. Name the activation program *activate* and place it into a separate directory, *db2\_config*, in this case.
- 5.** The reset program is specified. This program is called to reset the virtual appliance to its original state. Generally, reset undoes the actions that are performed by *activate*, where applicable.
- 6.** The list of input parameters is defined. These parameters are passed by VSAE to the activation and reset programs defined earlier in lines **4** and **5**. The current parameter values are, as stated, provided in the *env-ovf.xml* file. For example, to find the value for the **db2\_hostname** parameter, the VSAE constructs its unique identifier by concatenating the value of the class attribute in line **2** with the **db2\_hostname** property, which results in **com.ibm.vsa.2\_1.db2-ese95-config.db2\_hostname**. This identifier matches the **ovfenv:key** in the *env-ovf.xml* file. Therefore, the parameter value is set to *vs102* and passed to *activate.py* as an input parameter.

Consequently, the VSAE runs a command, such as the following example, to start the activation program that reconfigures DB2 instance:

```
AS/db2-config/activate.py --db2inst1_password s3cr3t --dbinst1_username db2inst1
--db2_hostname vs102
```

Following the methodology described in this section, several software virtual appliances, were built for IBM partners. These software virtual appliances fully automated the deployment of real-life and sometimes quite-complex solutions.

## **2.5 What a virtual appliance is not**

The software virtual appliance model governs the packaging and deploying of software solutions into cloud environments. It does not have dependencies on any specific cloud architecture or cloud services.

We believe that the software virtual appliances standard, as codified by the DMTF OVF specification, is the best option for simplifying the move from the traditional deployment model to the cloud deployment model. It speeds the process of moving a range of applications to the cloud and instrumenting them in a cloud-specific manner. The software components in a virtual appliance might integrate with the services that are provided by the cloud, such as database services or queuing services. However, this integration with cloud services is neither required nor governed by the DMTF OVF specification. Which underlying services to use is determined by the solution architect much in the same way as the solution architect might choose to use Java 2 Platform, Enterprise Edition rather than .NET. Choosing which cloud services to use has nothing to do with how the solution is packaged.



# Architecture of the IBM Image Construction and Composition Tool

The Image Construction and Composition Tool (referred to as *the tool* in this book) is built by using IBM WebSphere sMash. This tool uses the Derby embedded database for persistence. The Representational State Transfer (REST) API decouples the graphical user interface (GUI) from the business logic. The REST API is built on a layer of Groovy scripts that enables interoperability with the Java sMash projects. The GUI is built by using the Dojo framework with JavaScript Object Notation (JSON) messages that bridge the GUI and the REST API. The command-line interface (CLI) is built by using python scripts that communicate with the REST API by using JSON messages. The tool is packaged as an IBM Installation Manager repository with support for both interactive and silent installation.

The architecture of the IBM Image Construction and Composition Tool is described in this chapter. The chapter provides a high-level view of all the parts and flows of the tool to explain what occurs in the background when you use the tool to build software solutions into cloud-ready virtual appliances. It starts with a description of the main actors and the use cases that they realize. Then it highlights the components of the tool and the interaction among them. Next, it describes the main scenario flows in the core functionality of the tool. The scenario flows are followed by a description of the cloud provider, image, and bundle concepts. Finally, this chapter describes the GUI.

This chapter includes the following sections:

- ▶ Actors and use cases
- ▶ Component view
- ▶ Flow of the main scenarios
- ▶ Cloud provider concepts
- ▶ Software bundle concepts
- ▶ Image concepts
- ▶ User interface

Reading this chapter can help you understand the basic concepts of the tool so that you can apply this knowledge when installing, using, troubleshooting, and upgrading the tool. This chapter is essential to understanding the details in Chapter 6, “KVM Express cloud provider” on page 91, Chapter 7, “PowerVM Express cloud provider” on page 125, and Chapter 8, “ESX cloud provider” on page 157.

## 3.1 Actors and use cases

Although the Image Construction and Composition Tool does not currently support multiple user profiles with a corresponding access control and authorization, its design takes multiple roles into consideration. Even when you use the tool as a single user, a specific set of skills is needed to run a domain of the application. This section describes the different roles or actors that you play as you go through the set of features in the tool.

Figure 3-1 shows a Unified Modeling Language (UML) use-case diagram with actors and corresponding use cases as implemented by the Image Construction and Composition Tool.

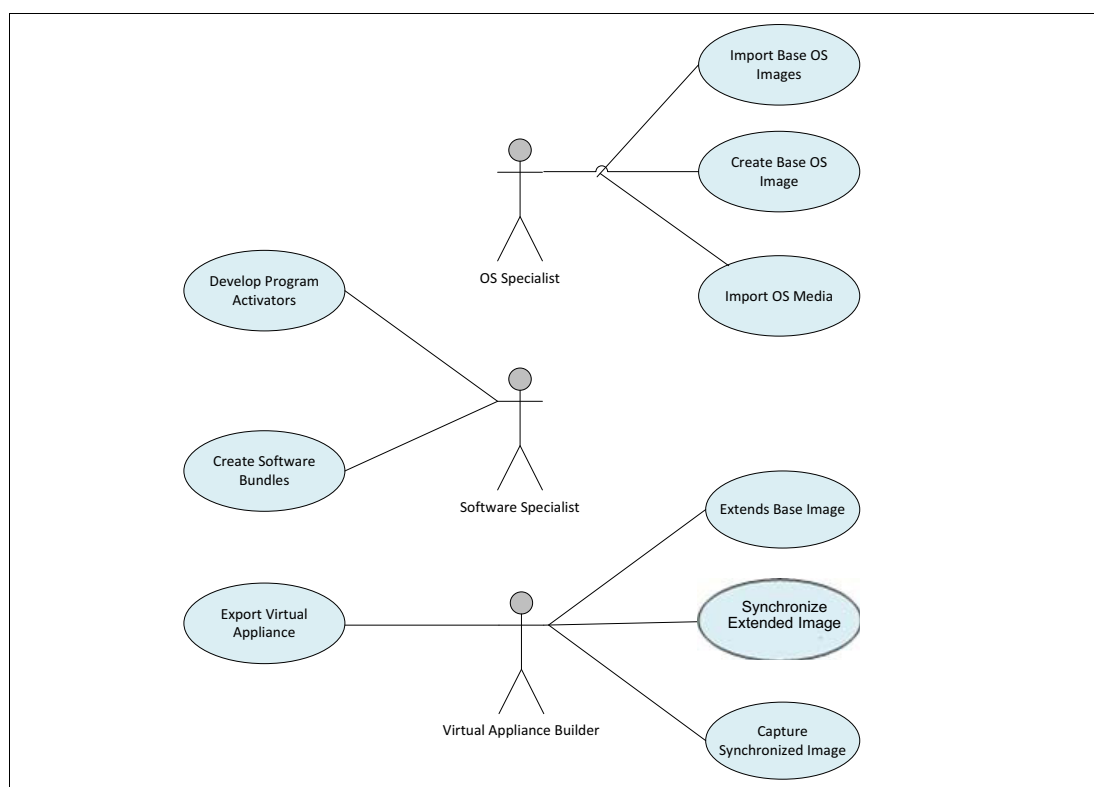


Figure 3-1 Use-case diagram from the Image Construction and Composition Tool

### 3.1.1 Operating system specialist

The operating system (OS) specialist is responsible for generating base OS disk images, which are one of the most important raw materials used in the Image Construction and Composition Tool build process. The OS specialist needs the following skill set:

- ▶ Knowledge of the OS installation and configuration
- ▶ Knowledge of the OS boot cycle and services
- ▶ Knowledge of the OS scripting languages

- ▶ Knowledge of the OS update methods
- ▶ A basic understanding of the hypervisors that supports the OS

OS specialists have the main role of bringing OS virtual disk images to the tool environment, which they can do in two main ways:

- ▶ Creating the images from scratch
- ▶ Importing a running virtual machine (VM) image into the tool environment

The OS specialist can create base OS images by using OS media images in the form of ISO files. The ISO files must be imported to the tool, and they are referenced during the creation process. The tool can do a remote installation by pointing to the desired OS ISO.

**Image creation from the ISO:** Not all cloud providers and operating systems support image creation from the ISO. However, the strategy of the Image Construction and Composition Tool is to support this method of image creation for all operating systems that support it. As of this writing, only the Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES) operating systems in the KVM cloud provider environment support this feature.

OS specialists can also employ their skills to create a base OS image, by using methods other than the tool, and then import the image. For example, OS specialists can create an RHEL image by a “kickstart install” using an FTP or HTTP site in their shop. They can create an IBM i image by installing from a virtual optical media, a Linux on Power image using a network installation, or an IBM AIX® image by using traditional Network Installation Management (NIM) methods. OS specialists can import images into the tool by using either of the following approaches:

- ▶ Import images that are already stored in the corresponding cloud provider image store.
- ▶ With a running VM, point the tool to it by using its IP address. This way the tool can inject the activation engine artifacts and capture that running VM as a new image that can then be extended into virtual appliances.

Your company can have one or more OS specialists. Each OS specialist should focus on building base OS images only for the operating systems in which they have expertise. For example, your x86 support organization might have two OS specialists, one that manages Windows images and another one that handles Linux images.

For more information about images, see 3.6, “Image concepts” on page 37.

### 3.1.2 Software specialist

The software specialist creates software artifacts, called *software bundles*. The software bundle is another raw material that the virtual appliance architect (*builder*) uses to generate complete solutions as virtual appliances. The software specialist needs the following skills around the software component that is packaged into a software bundle:

- ▶ Installation knowledge
- ▶ Configuration knowledge
- ▶ Scripting knowledge
- ▶ Knowledge of software dependencies and prerequisites
- ▶ Eclipse integrated development environment (IDE) skills



You can create the software bundle in either of the following ways:

- Using the Product Activator Development Kit (PADK) Eclipse plug-in

The PADK automatically generates a software bundle with the configuration and the reset scripts with the corresponding metadata (knowledge of Python is recommended when using the PADK). After the software bundle is created this way, the software specialist must still add the software installation packages. For information about PADK, see Chapter 5, “Product Activator Development Kit” on page 73.

- Directly using the GUI of the Image Construction and Composition Tool

With the tool, software specialists can generate a software bundle skeleton that can then be used to add the installation, the configuration, and the reset artifacts that are needed to complete the software bundle. For more information about the artifacts inside a software bundle, see 3.5, “Software bundle concepts” on page 36.

### 3.1.3 Virtual appliance architect

The virtual appliance architect assembles the base image and the software bundles that, together, constitute a solution into a self-contained virtual appliance. The virtual appliance is the end product that the Image Construction and Composition Tool constructs. It is a cloud artifact that can then be used to deploy software solutions into an IBM public or private infrastructure as a service (IaaS) environment. For more information about virtual appliances, see Chapter 2, “Anatomy of a virtual appliance” on page 11.

A virtual appliance architect must have the following required knowledge to create software solutions and package them into virtual appliances by using the tool:

- Solution architecture knowledge
- An understanding of the relationships among all components
- Solution configuration knowledge

An example of the type of software solutions that a virtual appliance architect builds is an enterprise customer relationship management (CRM) system that is built on top of a J2EE framework with a DB2 database as the persistence component. In this example, the environment is installed on an AIX operating system. The virtual appliance architect selects the AIX image version that is built by an OS specialist and then adds the WebSphere Application Server bundle, the DB2 bundle, and the CRM bundle that are built by software specialists. The action to add bundles to a base OS image is called in the tool image extension. After the image is extended, it must be synchronized. By synchronizing the image, the tool provisions the base OS image, waits for it to boot as a new virtual system, and injects the three bundles that are added during the extension process. After the image is synchronized, the virtual appliance architect can capture the VM as a new virtual appliance. The new virtual appliance can then be extended further or exported to be transported to the target cloud environment.

## 3.2 Component view

This section highlights the high-level components at work when running Image Construction and Composition Tool scenarios. The tool has many more components than are shown here, but the high-level abstraction that is used is sufficient to explain the main interactions.

Figure 3-2 on page 26 shows a component-level view of the tool (in the figure, ICCT is the Image Construction and Composition Tool).

**Current version support:** The current version of Image Construction and Composition Tool supports more implementations of cloud providers than are shown in the component diagram. Figure 3-2 shows only the implementations that are within the scope of this book.

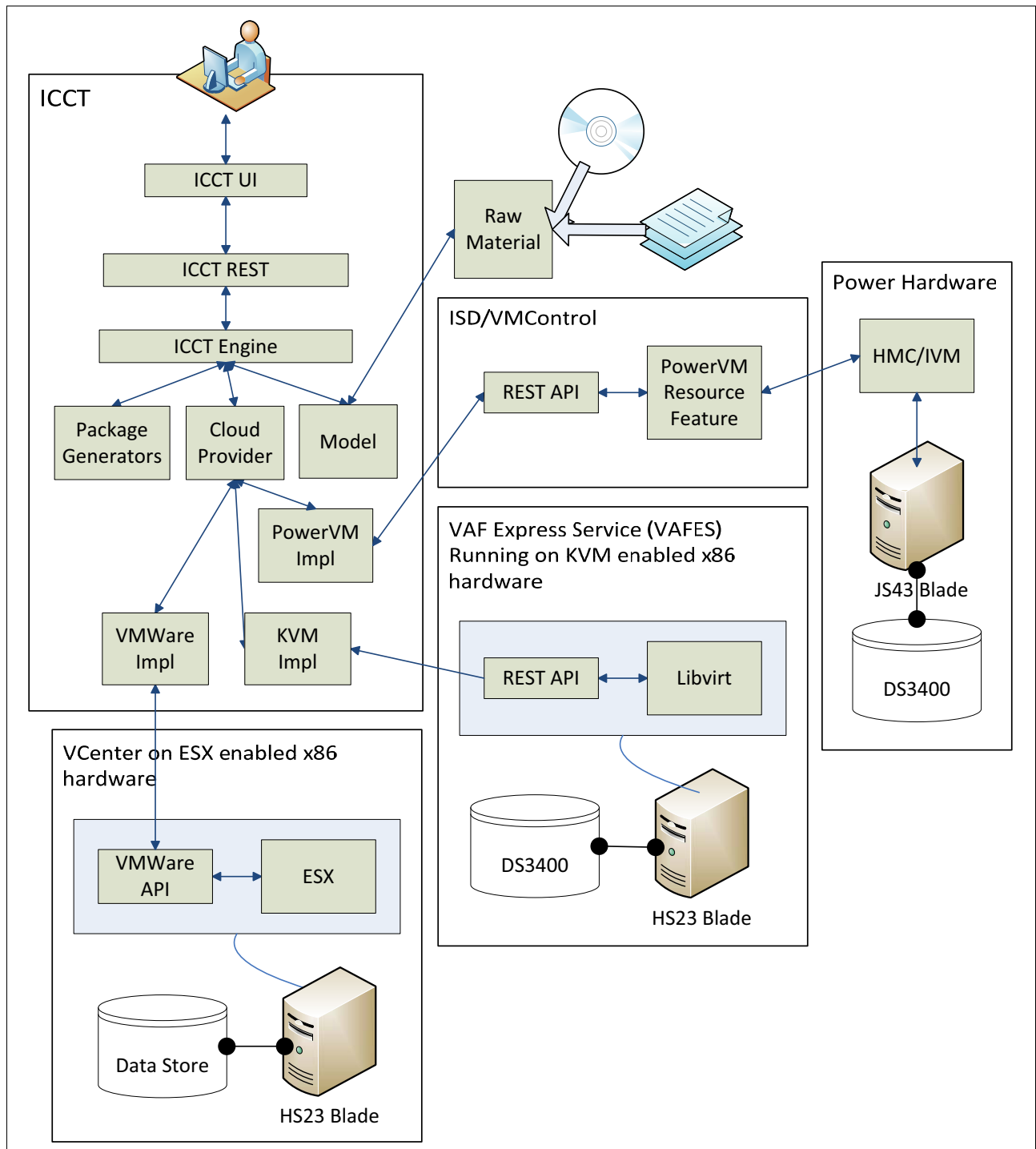


Figure 3-2 High-level component view of the Image Construction and Composition Tool

The Image Construction and Composition Tool has three components:

- ▶ The GUI, which is a web-based interface and what the tool user interacts with
- ▶ The REST API, which is implemented by a RESTful web service that runs on top of HTTP and connects the GUI with the engine of the tool
- ▶ The engine, which is the brain of the tool and where all the business logic runs

The engine has two roles:

- Provision the base images.
- Connect to the guest OS after the VM is running to install the packages that are described in the model that is associated with the image.

The following subsections describe the components that make up the engine of the tool.

### 3.2.1 The model

The Image Construction and Composition Tool is a model-driven tool. It relies on the model to tie the tool artifacts into a logical entity. Through these relationships, the tool can determine which media file to use to build base OS images in a cloud provider. It can also determine which software bundle to install in an instance of a base OS image and which base OS image to use in a cloud provider.

The following artifacts of the tool are described by the model:

- ▶ Base OS images
- ▶ Software bundles
- ▶ Cloud providers
- ▶ Media images
- ▶ Base OS image instances

You might think of the model as an in-memory database with a persistence capability that drives all the decisions made in the tool. The model is accessed through special objects that know how to query this in-memory database. Every time a new image is imported, a new bundle is created, and when an image is extended, the model is updated with the new metadata that describes those artifacts.

### 3.2.2 Package generators

In the Image Construction and Composition Tool, the package generator is a piece of code that can automatically generate a package that contains all the artifacts in all the bundles that are added to a base OS image. For a list of all artifacts in the bundle that will be packaged by the package generators, see 3.5, “Software bundle concepts” on page 36. The package generators have a hierarchy: a base object with a common installation and packaging logic, and one extension per shell engine and per operating system.

Most package generators are OS-specific. For example, one package generator might be for Linux, and another might be for AIX. The package generator for IBM i is basically the same as for AIX. These package generators can convert a set of bundles that are assigned to a base OS image and create a package that can be installed. They automatically generate OS-compatible bundle installation scripts, given the relationships in the model. The scripts have intelligence about how to install software in a running OS instance and about how to extend the activation engine in the instance. Special bundles are labeled as enablement bundles that also contain the activation engine software. The engine of the tool recognizes when this enablement bundle was installed in the base image. If the bundle is not installed, a

request is sent to the package generator so that the necessary bundle is included in the package to install in the image that is under construction.

The package generators are independent of the cloud providers. For example, the same package that is generated for Linux might be sent to a Linux VM that is running on ESX or on KVM. This method allows for a separate object hierarchy that can share logic across cloud providers. The same package generator can also be used for different operating systems in certain situations. For example, the AIX package generator can be shared with RHEL and IBM i because of the similarities between the Korn shell (ksh) interpreters in AIX and the RHEL operating systems, and the IBM Portable Application Solutions Environment for IBM i (PASE).

### 3.2.3 Cloud providers

Because this book covers only the VMware, KVM, and PowerVM cloud providers, Figure 3-2 on page 26, shows implementations of these cloud providers. A cloud provider is responsible for all necessary operations to manage the lifecycle of a VM.

A cloud provider must have the following minimum capabilities:

- ▶ Deploy a VM from a base OS image.
- ▶ Start or stop a VM.
- ▶ Capture a VM as an extended base OS image.
- ▶ Delete a VM.

The three cloud provider implementations in this section meet the following basic criteria:

- ▶ The VMware cloud provider implementation communicates with a vCenter server through the VMware APIs that are specified in the VMware SDK client.
- ▶ The KVM implementation uses a REST API (developed by IBM) called the *IBM Virtual Appliance Factory Express Services* (VAFES), which runs on top of the `libvirt` libraries in a KVM host.

**Virtual appliance for the KVM cloud provider:** The KVM cloud provider does not use Systems Director VMControl for the basic provisioning operations that are required by the Image Construction and Composition Tool. However, the virtual appliances created with this provider are fully compatible with Systems Director VMControl and the Flex System Manager appliance for IBM PureFlex System.

- ▶ The PowerVM cloud provider requires an IBM Systems Director system with the Systems Director VMControl and Systems Director Storage Control advanced managers. The communication between the PowerVM cloud provider implementation and Systems Director VMControl is implemented in a RESTful web service.

## 3.3 Flow of the main scenarios

A virtual appliance architect employs main features to accomplish the primary goal of the Image Construction and Composition Tool. Recall that the objective of the tool is to put the artifacts together to create a virtual appliance that contains a complete software solution. Remember also that the artifacts are typically created by other actors. For example, a software specialist creates software bundles, an operating system specialist creates base OS images.

A virtual appliance architect must be familiar with the following tasks:

- ▶ Importing an image from a cloud provider
- ▶ Importing an image from a running virtual machine
- ▶ Extending a base OS image
- ▶ Capturing a synchronized virtual image instance
- ▶ Exporting a virtual appliance

### 3.3.1 Importing an image from a cloud provider

The OS specialist for the Image Construction and Composition Tool must import existing images from a cloud provider when possible. This approach is the fastest way to import a base OS image into the environment of the tool.

Figure 3-3 shows a flow diagram for importing an image from a cloud provider (in the figure, ICCT is the Image Construction and Composition Tool).

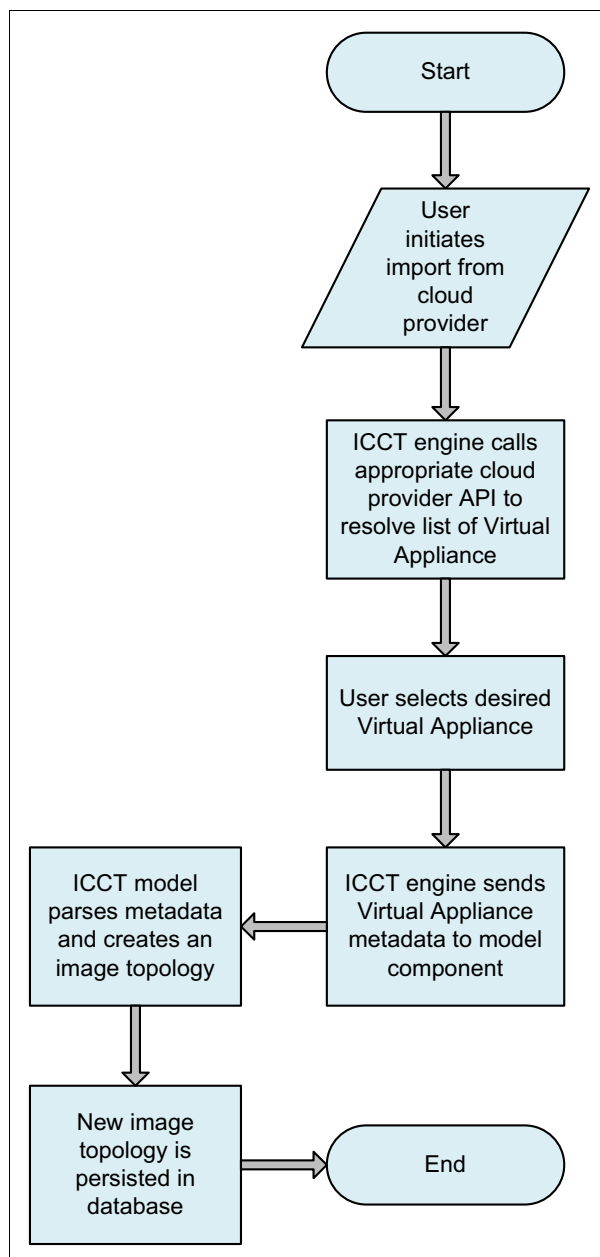


Figure 3-3 Importing an image from a cloud provider

However, do not import an image from a cloud provider if its origin is unknown. If you are unsure about whether a valid activation engine is in the images that are stored in the cloud provider, use the method that is described in 3.3.2, “Importing an image from a running virtual machine” on page 31. When importing from a cloud provider, the tool imports only the metadata of the image, which in this case is the Open Virtualization Format (OVF) metadata. The tool can parse this OVF XML and add the image to the model so that it can be used as a base OS image during the build process.

**Metadata transformations:** The Image Construction and Composition Tool keeps a different format to store metadata of the images. The model of the tool creates a topology of the metadata that follows an IBM proprietary schema. However, the tool supports transformation from OVF to the image topology of the tool and vice versa. Transformations from OVF to the topology schema are done when importing image metadata, and transformations from an image topology to OVF are done during export. For more information, see 3.3.5, “Exporting a virtual appliance” on page 34.

The binary images stay in the cloud provider image repository. The tool makes only a reference to the binary image in its model.

### 3.3.2 Importing an image from a running virtual machine

In many cases, the OS specialists might want to bring their own image with the preferred software stack already installed and configured. For example, the cloud provider does not offer an image with LDAP client capabilities. Therefore, the OS specialist can deploy an existing image with the chosen capabilities and then instruct the Image Construction and Composition Tool to import the running VM.

The tool injects, into the image, the activation engine that is tailored for the OS that is imported and then captures the running VM. It also generates the metadata for the image that is being imported in the form of a topology object to be persisted in the database.

In some cases, such as with the VMware cloud provider, the tool also brings over the imported image to the internal storage of the tool in the /drouter directory. However, in other cases, such as with the KVM and PowerVM cloud providers, the tool stores only a reference to the captured image and allows the cloud providers to store the image in their repositories.



Figure 3-4 shows the flow diagram for importing an image from a running VM. This figure illustrates a user request to import from a running VM. The request triggers the communication flow between the engine of the tool, the package generator, the cloud provider, and the model components, resulting in a new base OS image in the Image Construction and Composition Tool. (In the figure, ICCT is the Image Construction and Composition Tool.)

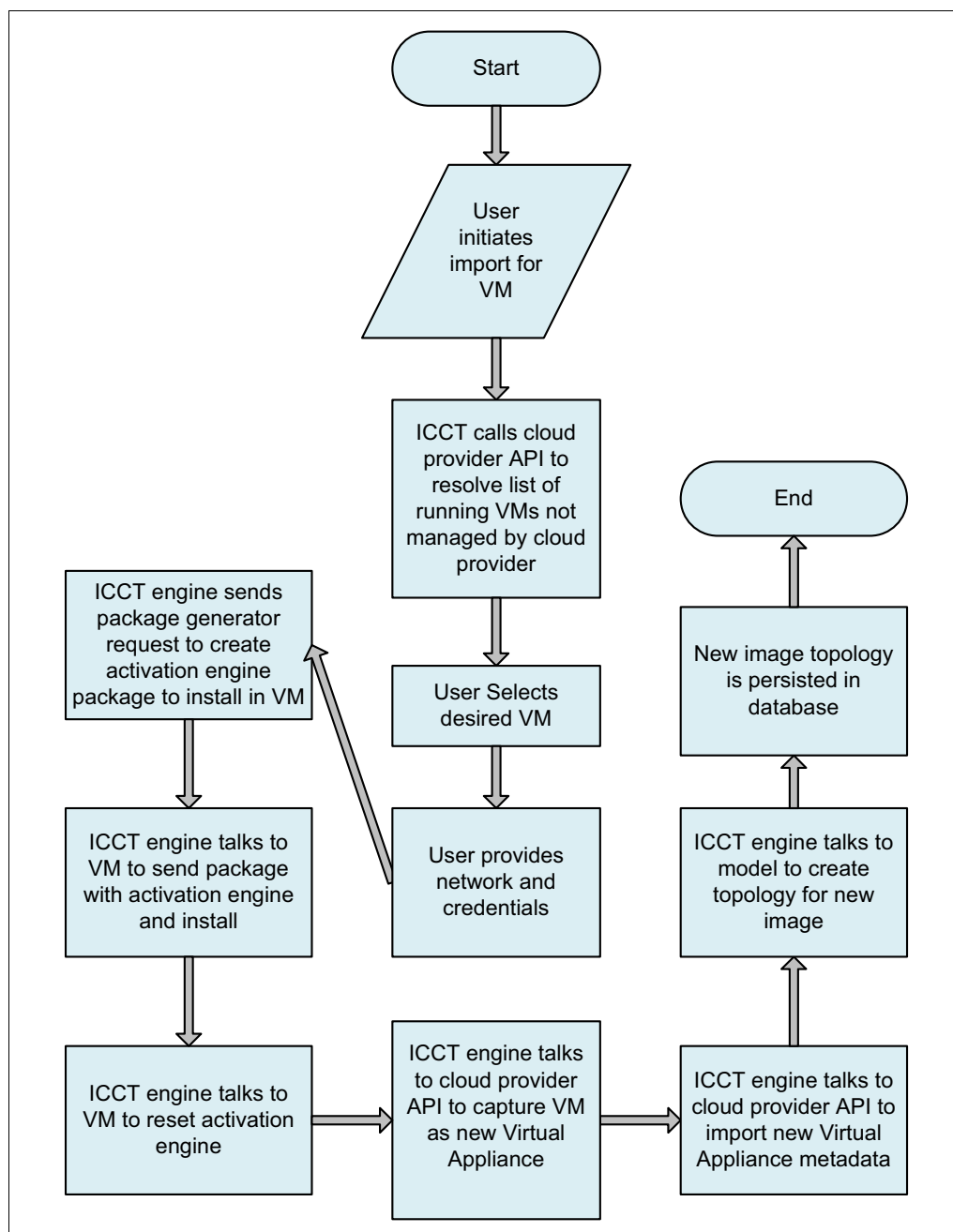


Figure 3-4 Importing an image from a running VM

### 3.3.3 Extending a base OS image

Extending the base OS image process involves the following steps:

1. Copy the metadata of the base OS image into a new image object.
2. Add the software bundle definition to the new image topology object.
3. Synchronize a copy of the base OS image by installing the software bundle content.

As a result, the extended image contains the base OS image software, plus the new software added by using software bundles. The Image Construction and Composition Tool also adds the metadata that corresponds to the software bundles for the base OS metadata.

Figure 3-5 shows the flow of extending the base OS image (in the figure, ICCT is the Image Construction and Composition Tool).

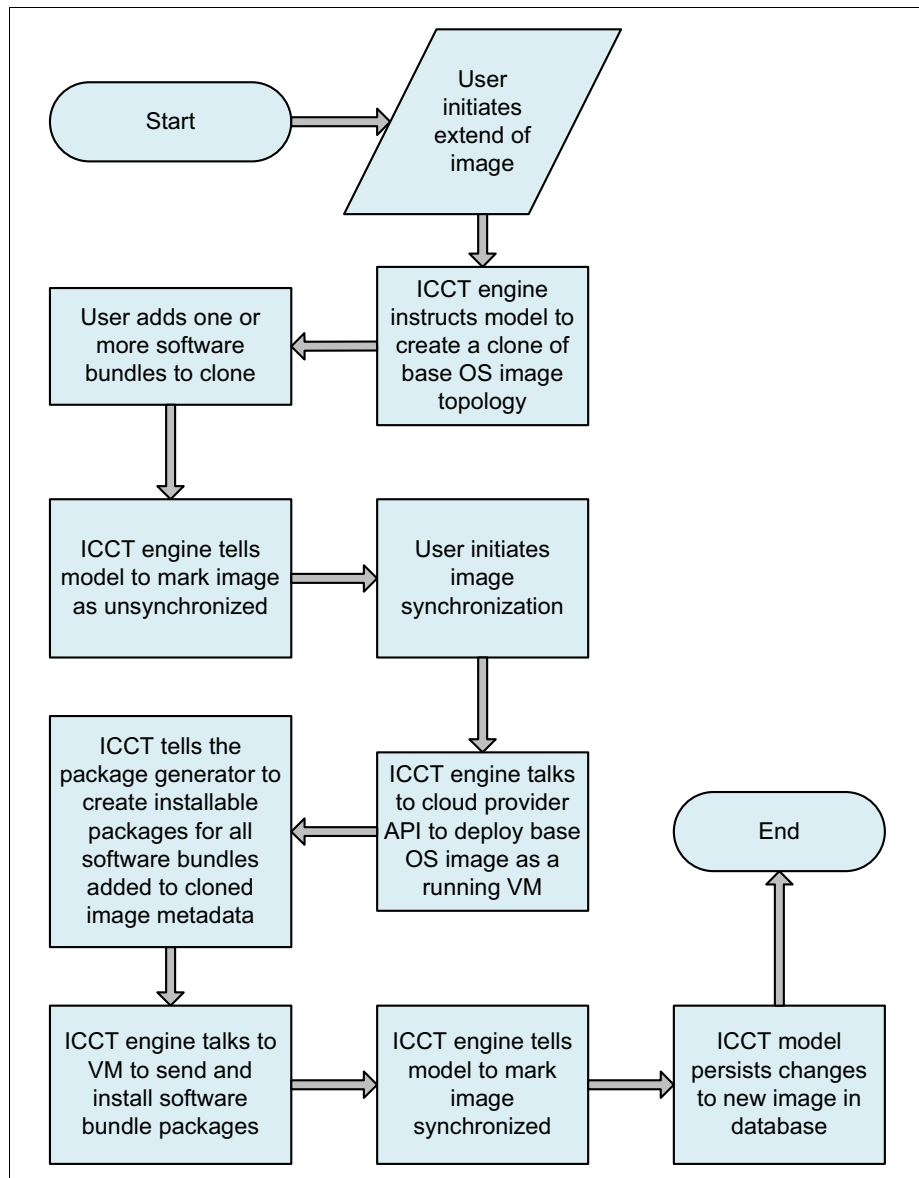


Figure 3-5 Extending a base OS image

### 3.3.4 Capturing a synchronized virtual image instance

After extended images are in a synchronized state, by using the Image Construction and Composition Tool, the user can capture an image. Capturing an extended image results in a new base OS image, with extended packages, that are ready to be extended again or exported to the cloud. The tool alerts the cloud provider to capture the running VM with the extended content as a virtual appliance. It also modifies the metadata that is maintained by the provider so that the new software added by using bundles can now be accounted for in the virtual appliance that is stored in the cloud provider image repository.

Figure 3-6 illustrates the flow for capturing a synchronized virtual image instance (in the figure, ICCT is the Image Construction and Composition Tool).

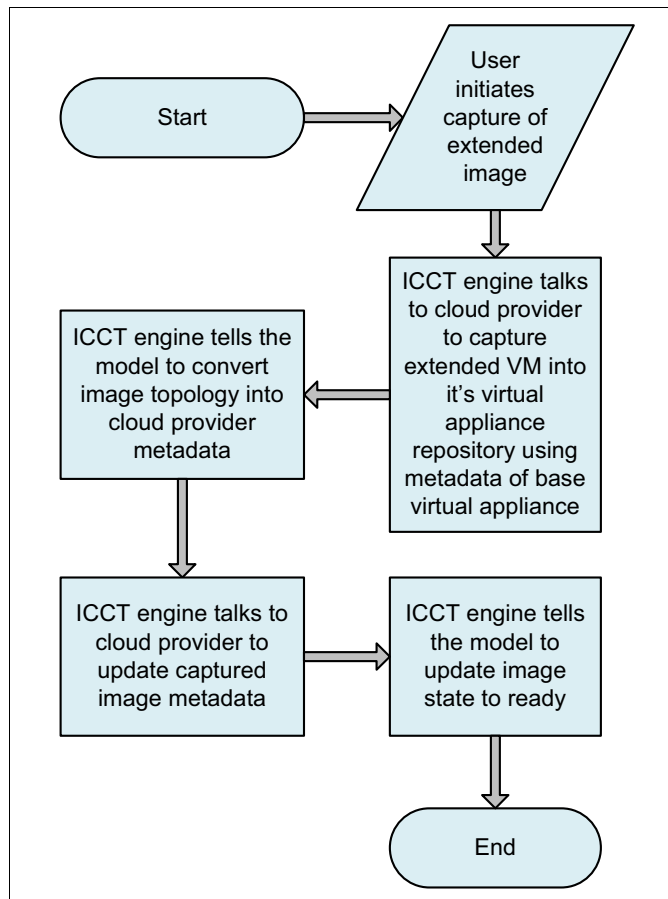


Figure 3-6 Capturing a synchronized virtual image instance

The flow diagram shows how the engine of the tool, the model, and the cloud provider components interact to capture the extended image into a new image that is ready for use as a base OS image. This diagram also shows how the image metadata in the cloud provider is updated with new bundle metadata. As mentioned previously, the model of the tool keeps an image topology that needs to be transformed to OVF.

### 3.3.5 Exporting a virtual appliance

Up to this point, all the images that are imported and extended are in the Image Construction and Composition Tool and in the cloud provider space. This section describes how you can

extract those images from the tool and cloud providers as a file that can be transferred to the cloud.

Each cloud provider has different types of image repositories. For example, the VMware cloud provider keeps a copy of the images in the tool store. KVM and PowerVM cloud providers depend on the capabilities of the cloud provider image repository to store images. Although internal differences exist about how cloud providers handle image storage, the tool externalizes an image that is referenced in the model as an artifact that the cloud can import and run regardless of these differences.

Figure 3-7 shows a flow diagram for exporting a virtual appliance. This diagram considers the various cloud providers in the scope of this book. (In the figure, ICCT is the Image Construction and Composition Tool.)

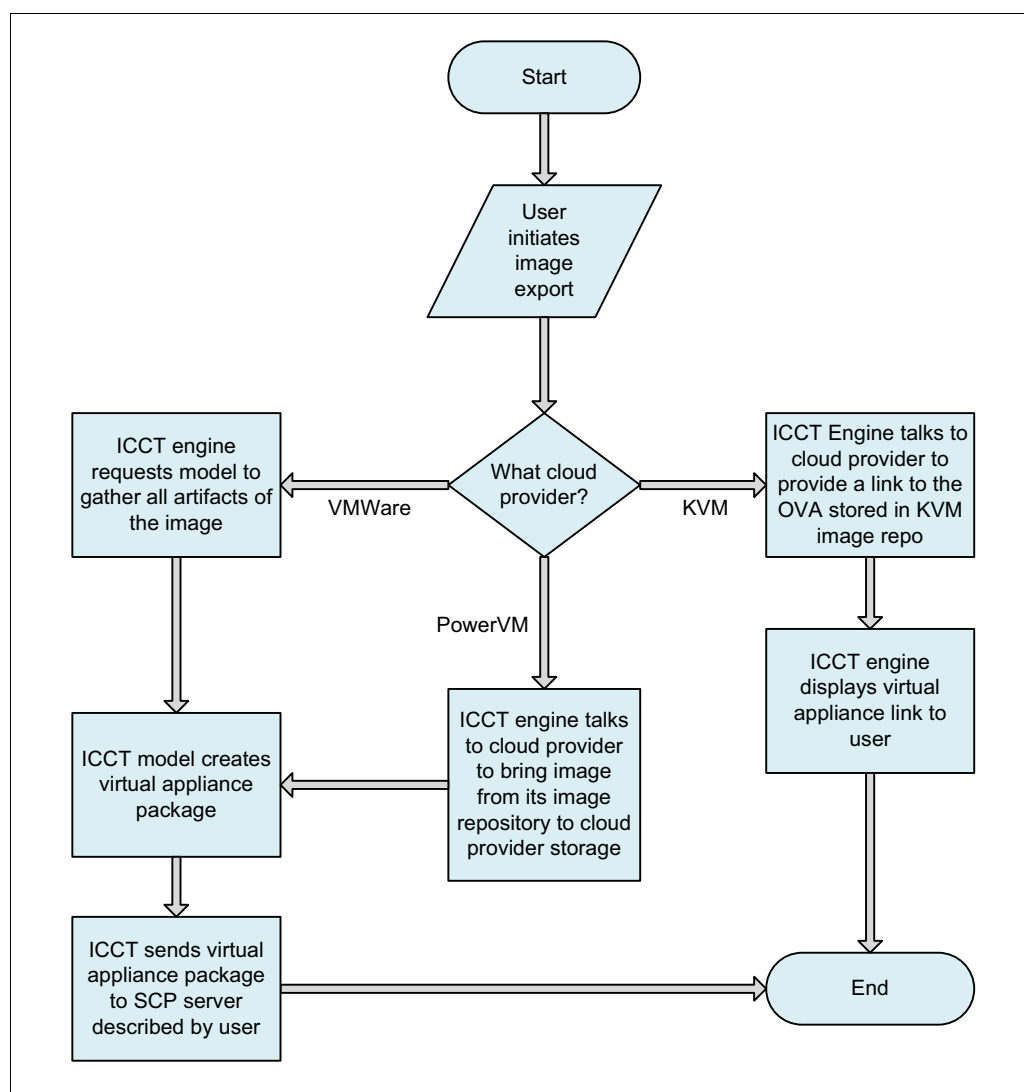


Figure 3-7 Exporting a virtual appliance

The flow diagram shows the interactions between the engine, cloud provider, and model components. It also shows how the tool interacts with the different cloud provider implementations to create an OVA that can then be transferred to a cloud domain and can be used as a self-activating workload.

## 3.4 Cloud provider concepts

A cloud provider is an artifact of the Image Construction and Composition Tool that is stored in the model of the tool. The cloud provider hierarchy allows for an abstraction level that keeps the tool independent from hypervisors and their specific APIs. Each cloud provider implementation connects to a specific hypervisor to provision images that can then be extended by the tool.

All cloud providers keep the following set of common properties in the model:

- ▶ Connection information to the hypervisor manager
- ▶ Credentials for the hypervisor manager
- ▶ Network information within the domain of the hypervisor manager
- ▶ A client API that is used to communicate with the hypervisor manager

Although most properties are common for all cloud providers, a cloud provider might require or allow specifying additional properties to enable its functions. For example, the PowerVM cloud provider allows specifying the storage pool to be used for deployment during synchronization operations, and requires additional information about the image repository storage of the hypervisor manager to bring image files to the domain of the tool, and package virtual appliances during the export operation.

In the scope of this Redbooks publication, the hypervisor managers are vCenter for VMware cloud provider, Systems Director VMControl for PowerVM cloud provider, and the VAFES API for the KVM provider.

## 3.5 Software bundle concepts

*Software bundles* are other types of artifacts in the Image Construction and Composition Tool model. They allow for a standard format of software distribution that can be converted by the package generator component of the tool into an installation package that can be used to extend base OS images. Software bundles are compatible with most installation technologies if a script is available that can drive the installer without user interaction. An installer must support silent installation.

A software bundle has the following parts:

- ▶ Identity information
  - Name of the bundle
  - Publisher
  - Date of creation
  - Unique identifier
- ▶ Requirements information
  - Supported OS
  - Required software
  - Required bundle
- ▶ Installation information
  - Software packages to install
  - Script that runs the installer in silent mode
  - Parameters to be passed to a script during installation

- ▶ Configuration information
  - Product Activator packages (IBM Virtual Solutions Activation Engine (VSAE) extensions)
  - Script that can configure software pieces that are in a bundle
  - Parameter names and default values to be passed to the VSAE at boot time
- ▶ Reset information
  - Script that can reset software pieces that are in a bundle
  - Parameter names and default values to be passed to the VSAE at reset time.

This rich set of parts and properties of software bundles allows for the level of automation that the package generators of the tool need to create the installation software packages. For example, the package generator checks for the supported OS information to generate installation scripts that are compatible with the target OS. Similarly, the package generator must be aware of any software or other bundle requirements, so that those dependencies can be added to the final installation package.

A software bundle called the *enablement bundle* is automatically injected into all images that the tool finds to be missing the activation engine package. No version of the activation engine is checked. When images are started within a VM, the operating system version and distribution are discovered and matched to the well-known enablement bundles that contain an activation engine version that is compatible with an OS version and distribution.


## 3.6 Image concepts


*Images* are other types of artifacts that are kept in the model of the IBM Image Construction and Composition Tool. In previous sections, you might have noticed usage of the terms *image* and *virtual appliance* interchangeably. However, the term *image* refers to disk images in the domain of the tool. The term *virtual appliance* refers to images that contain cloud provisioning metadata and are ready to be used as a cloud workload. From the perspective of the tool, an *image* is a raw material with minimum provisioning metadata to be used in the virtual appliance building process.

Because the Image Construction and Composition Tool supports multiple hypervisors, the tool must support different disk image formats. This book focuses on two main formats: the VMDK format for VMware and RAW files from KVM and PowerVM.

## 3.7 User interface

This section helps you to visualize the main flows from the GUI. From the Images panel (Figure 3-8 on page 38), you can import or add an image.

When you click the **Import from cloud provider** icon () , you see a window with a list of images in the cloud provider image repository. Images that are already imported into the Image Construction and Composition Tool are not displayed in the list. When you select an image to import, you can also add a user name and password for the image so that the tool can connect to the image during the image synchronization process.

When you click the **Add image** icon () , a window opens from where you select VMs that are not instances of existing images in the cloud provider image repository. You can select a VM, its network connection, and its credentials. This way, the tool can connect to the VM and inject the corresponding enablement bundle content.

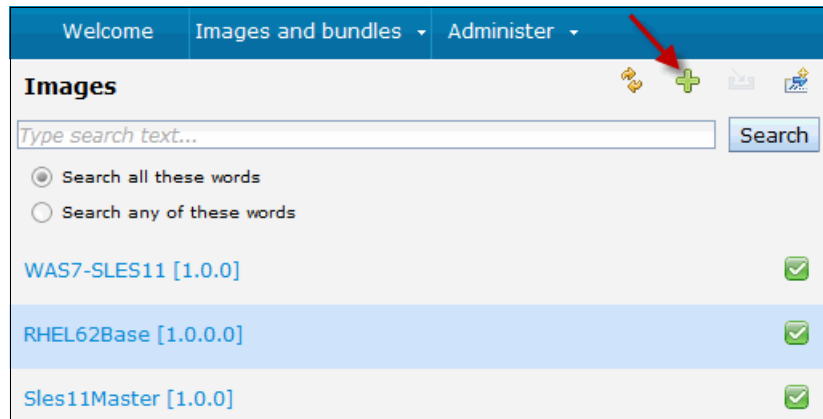


Figure 3-8 Add image icon

The following panels drive the flow that is explained in 3.3.3, “Extending a base OS image” on page 33. As shown in Figure 3-9, when you click the **Extend** icon, you signal the need for a new image that is derived from the current image that is selected. A clone of the image metadata is created so that you can extend the clone.

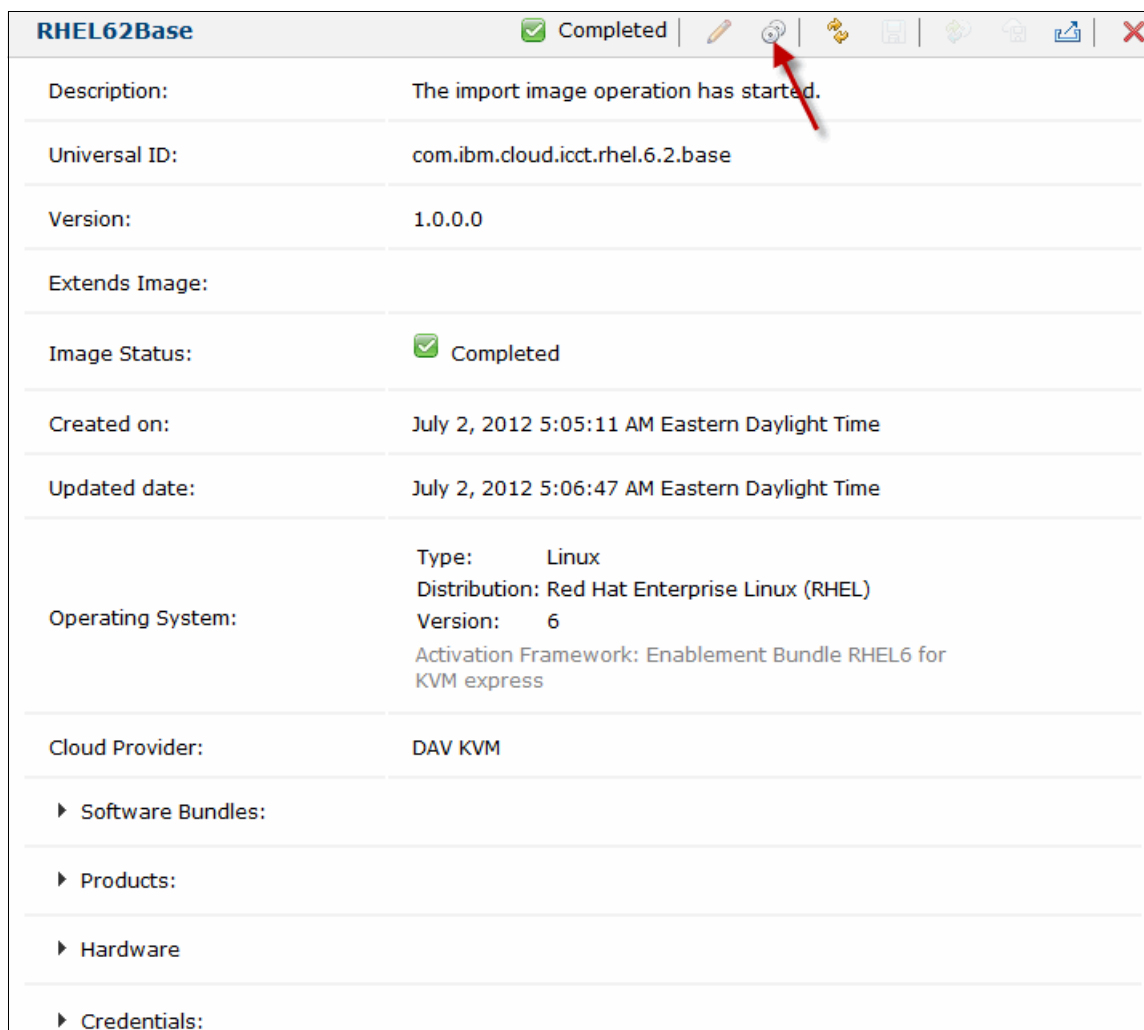


Figure 3-9 The Extend GUI icon

When you have a clone of a base OS image, you can edit the clone and add bundles to it. First, click the **Start Editing** icon and then click **Add bundle** (Figure 3-10).

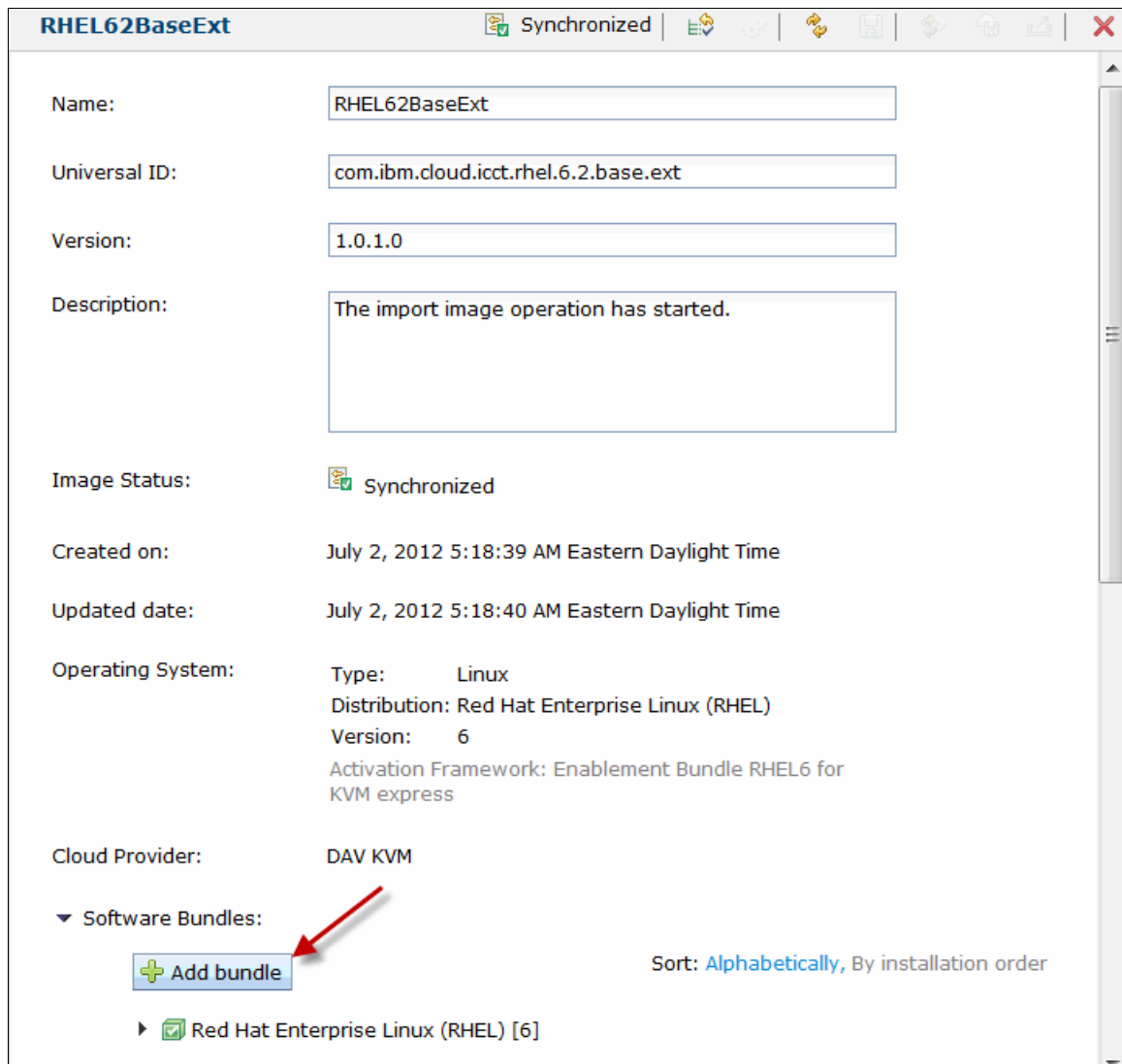


Figure 3-10 Add bundle icon



The Add bundle to image window (Figure 3-11) opens; it shows a list of existing bundles to select. From this window, you can filter the bundle list by OS compatibility. That is, the window shows only those bundles with metadata that matches the OS of the image that you are extending.

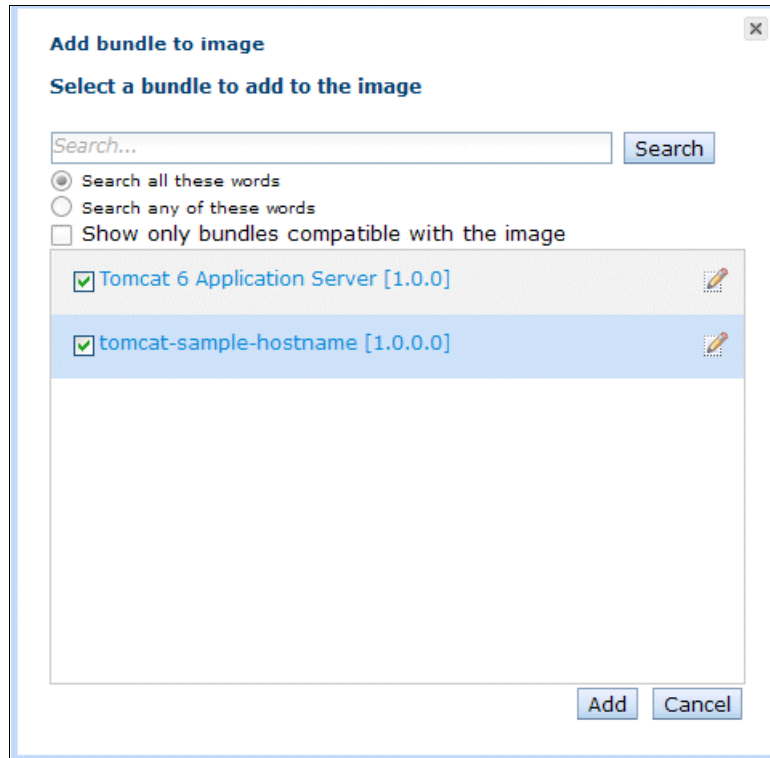


Figure 3-11 Add bundle to image window

After you add all the bundles you want in the new image, click the **Done Editing** icon and then the **Save** icon; the tool shows the new image as being out of sync, as indicated in the Image Status field (Figure 3-12). This status enables the Synchronize icon. If you click the **Synchronize** icon, the tool deploys an instance of the base OS image that is linked to this clone. Then, it initiates creation of the synchronization package so that it can be sent to the running instance. In the synchronization window, you can customize credentials if you did not set them when importing the base OS image.

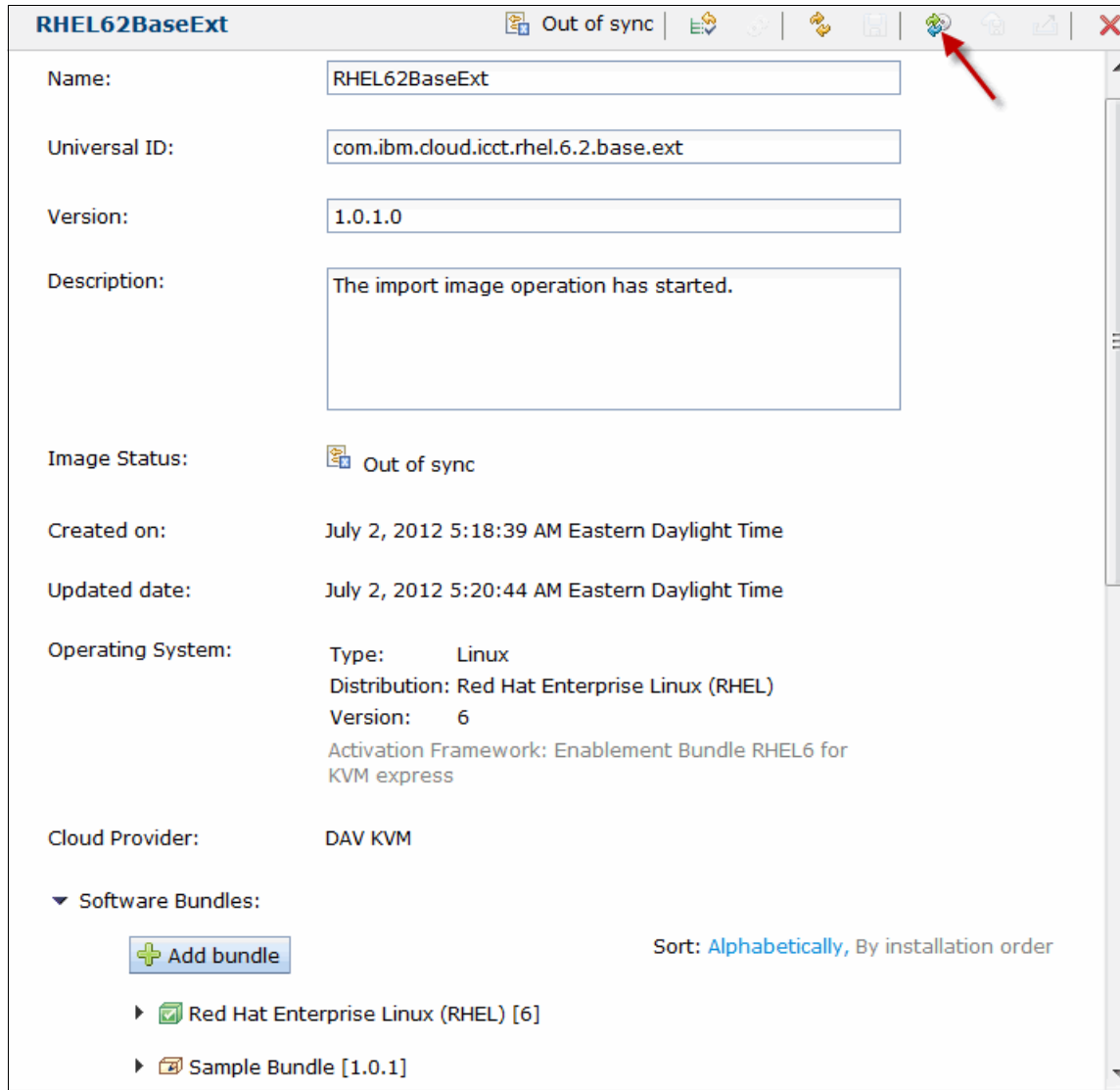


Figure 3-12 Synchronize button

The panel shown in Figure 3-13 controls the flow that is described in 3.3.4, “Capturing a synchronized virtual image instance” on page 34. By clicking the **Capture** icon, you can capture the extended image that is already synchronized as another base OS image. When capturing with the PowerVM cloud provider, you are prompted to select the repository where the new base image should be created, if more than one repository is available. The image goes from a *Synchronized* state to a *Captured* state when the capture process is completed.

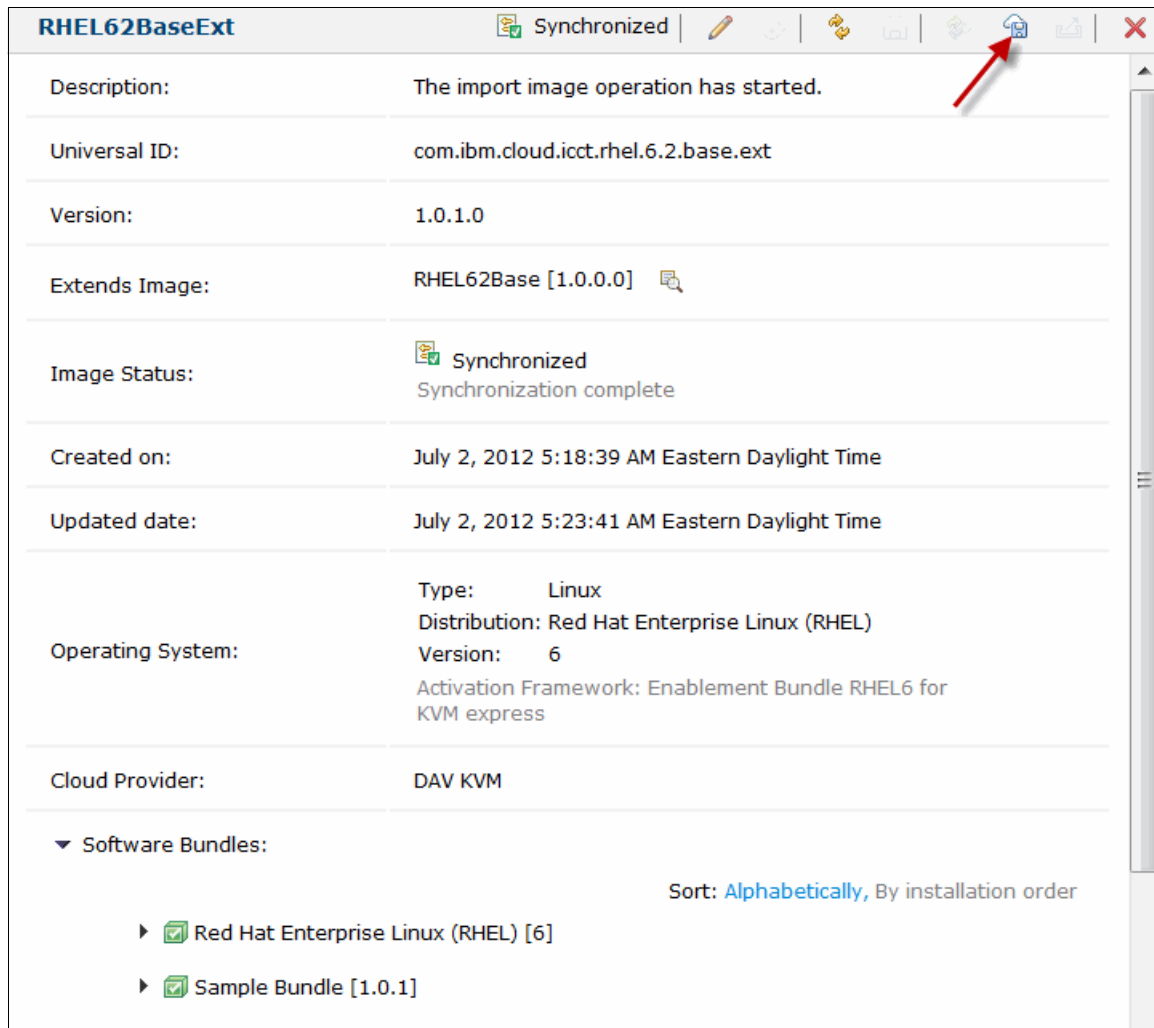


Figure 3-13 Capture GUI icon

The panel in Figure 3-14 controls the flow that is described in 3.3.5, “Exporting a virtual appliance” on page 34. By clicking the **Export** icon, you can extract an image in the cloud provider as a portable file that is packaged as an OVA. Depending on the cloud provider, you see different windows. If you use the KVM cloud, the window shows only the link to the OVA tar file. If you use the PowerVM or VMware cloud provider, you are prompted for a system that supports secure copy command (**scp**) as a target connection for sending the OVA archive.

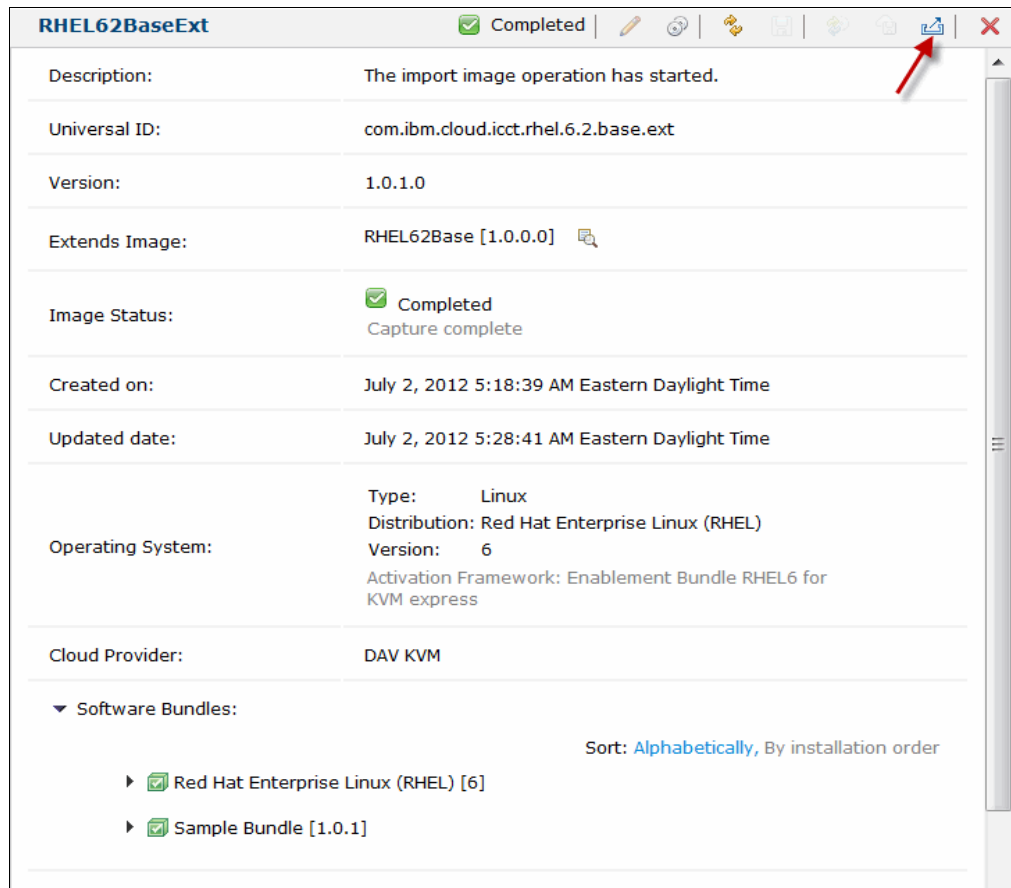


Figure 3-14 Export icon





## Setting up the virtual appliance build environment

This chapter explains how to set up a robust virtual appliance build environment. It begins by explaining how to install the Image Construction and Composition Tool (referred to as *the tool* in this book). Then it shows how to configure the PowerVM environment. Finally, it explains the KVM build environment.

This chapter includes the following sections:

- ▶ Installing the Image Construction and Composition Tool
- ▶ Setting up the build environment for PowerVM virtual appliances
- ▶ Setting up the build environment for KVM virtual appliances

## 4.1 Installing the Image Construction and Composition Tool

The IBM Image Construction and Composition Tool is included with other IBM cloud management offerings, such as IBM SmartCloud Provisioning, IBM SmartCloud Entry, IBM SmartCloud Orchestrator, and IBM Workload Deployer, but it is a separately installable tool. Consequently, the way that you can obtain the installable image of the tool depends on the cloud product management that you implement in your organization.

The IBM Installation Manager is used to install the Image Construction and Composition Tool. Typically, the installable image contains both components. However, the Installation Manager is platform-dependent. Therefore, to successfully install it, you need a version that supports the OS on which you are installing the tool. The following instructions take this consideration into account so they should be applicable to most environments.

Therefore, the installation process is split into the following parts:

- ▶ Installing the IBM Installation Manager
- ▶ Installing the Image Construction and Composition Tool

This way, debugging potential issues that are associated with the missing IBM Installation Manager prerequisites can be easier.

### 4.1.1 Installing the tool on Linux

The following instructions document the installation on a 64-bit RHEL. Installation on other environments might differ slightly.

To install the IBM Image Construction and Composition Tool on Linux, use the following steps:

1. Install the IBM Installation Manager prerequisites as follows (several dependencies exist for the IBM Installation Manager):
  - a. Run the following command to install the prerequisites:

```
yum install libXtst.i686 gtk2.i686
```

Several additional dependencies are pulled from the yum repository.
  - b. When installing on a 64-bit distribution, make sure that the 32-bit `libgcc` is installed with the matching 64-bit version, which you can verify by running the following command:

```
rpm -qa | grep libgcc
```

The following example shows the results:

```
libgcc-4.4.6-4.el6.x86_64  
libgcc-4.4.6-4.el6.i686
```
  - c. If the 32-bit library is missing, you must install it. Remember that the library versions must match. To install the correct version, use the exact name of the library as the `yum install` parameter:

```
yum.sh install libgcc-4.4.6-4.el6.i686
```
  - d. If you receive the missing `libstdc++` message, install the `libstdc++` compatibility library:

```
yum install compat-libstdc++
```

## 2. Install the IBM Installation Manager:

- a. Go to the directory where the compressed installable image is saved, for example, the /tmp/im directory, and extract the compressed file.

Typically, the Installation Manager installation files are in the top-level directory (/tmp/im), and the Image Construction and Composition Tool installation files are in the corresponding subdirectory (/tmp/im/icon).

- b. Open the install.xml file in a text editor and remove the following lines:

```
<repository location='icon' />
<offering id='com.ibm.cloud.icon' />
```

- c. To install IBM Installation Manager in the /opt/IBM/InstallationManager/eclipse directory, enter the following command from the /tmp/im directory:

```
./installc -log <log_file> -acceptLicense
```

Where <log\_file> is the full path and name of the log file on the Image Construction and Composition Tool system.

After the installation is complete, you see a message similar to the following example:

```
Installed com.ibm.cic.agent_1.5.0.20110909_1200 to the
/opt/IBM/InstallationManager/eclipse directory.
```

**Tip:** If the installable image does not contain the IBM Installation Manager, you can download the version specific to Linux from the following location:

[http://public.dhe.ibm.com/software/rational/sdp/v7/im/15/zips/agent.installer.linux.gtk.x86\\_1.5.0.20110909\\_1200.zip](http://public.dhe.ibm.com/software/rational/sdp/v7/im/15/zips/agent.installer.linux.gtk.x86_1.5.0.20110909_1200.zip)

## 3. Install the IBM Image Construction and Composition Tool:

- a. In the directory with the binary files of the Image Construction and Composition Tool (/tmp/im/icon), open the icon\_silent\_install\_response\_file.xml response file.
- b. Edit the response file so that the repository location points to the directory where the compressed file is extracted as shown in the following example (/tmp/im/icon). This directory must contain the repository.config file.

```
<repository location='/tmp/im/icon' />
```

To change the installation directory from the default location, update the **installLocation** parameter to point to the location that you want by updating the following line in the file:

```
<profile id="Image Construction and Composition Tools"
installLocation="/opt/IBM/icon" />
```

The installation location can contain English characters only.

## 4. Enter the following command to start the installation:

```
/opt/IBM/InstallationManager/eclipse/tools/imcl input
/tmp/im/icon/icon_silent_install_response_file.xml -acceptLicense
```

After the installation is complete, you see a message similar to the following example:

```
Installed com.ibm.cloud.icon_2.2.1.1.x to the /opt/IBM/icon directory.
```



**Uninstallation:** To uninstall the Image Construction and Composition Tool, run the following command:

```
/opt/IBM/InstallationManager/eclipse/tools/imcl uninstall com.ibm.cloud.icon
```

Deleting drouter (rm -rf /drouter/\*) removes the image and bundle repositories.

## 4.1.2 Installing the tool on AIX

To install the Image Construction and Composition Tool on AIX, use the following steps:

1. Install the IBM Installation Manager. In most cases, download the AIX version of the Installation Manager. As of writing this book, you can find the installable image at the following address:

[http://public.dhe.ibm.com/software/rational/sdp/v7/im/15/zips/agent.installer.aix.motif.ppc\\_1.5.0.20110909\\_1200.zip](http://public.dhe.ibm.com/software/rational/sdp/v7/im/15/zips/agent.installer.aix.motif.ppc_1.5.0.20110909_1200.zip)

- a. Go to the directory where the compressed file is saved, for example the /tmp/im directory, and extract the compressed file:

```
unzip agent.installer.aix.motif.ppc_1.5.0.20110909_1200.zip
```

- b. Run the following command to install IBM Installation Manager in the /opt/IBM/InstallationManager/eclipse directory:

```
./installc -log <log_file> -acceptLicense
```

Where <log\_file> is the full path and name of the log file on the Image Construction and Composition Tool system.

After the installation is complete, you see a message similar to the following example:

```
Installed com.ibm.cic.agent_1.5.0.20110909_1200 to the
/opt/IBM/InstallationManager/eclipse directory.
```

2. Install the IBM Image Construction and Composition Tool.

- a. In the directory with the binary files of the Image Construction and Composition Tool, for example /tmp/icct, open the icon\_silent\_install\_response\_file.xml response file.

- b. Edit the file so that the repository location points to the directory where the compressed file is extracted. This directory must contain the repository.config file.

```
<repository location='/tmp/icct'/>
```

- c. In addition to the repository location, edit the following parameters in the icon\_silent\_install\_response\_file.xml file:

```
<data key='cic.selector.os' value='aix'/>
```

```
<data key='cic.selector.arch' value='ppc64'/>
```

Remove the following line:

```
<data key='cic.selector.ws' value='gtk'/>
```

- d. If you want to change the installation directory from the default location, update the **installLocation** parameter to point to the location that you want by updating the following line in the file:

```
<profile id="Image Construction and Composition Tools"
installLocation="/opt/IBM/icon"/>
```

The installation location can contain English characters only.

3. Run the following command to start the installation:

```
/opt/IBM/InstallationManager/eclipse/tools/imcl input  
/tmp/icct/icon_silent_install_response_file.xml -acceptLicense
```

After the installation is complete, you see a message similar to the following example:

Installed com.ibm.cloud.icon\_2.2.1.1.x to the /opt/IBM/icon directory.

### 4.1.3 Accessing the tool

The Image Construction and Composition Tool is now installed and accessible from the following URL:

`https://<IP_address>/icn/ui/`

You can log in to the tool by using the credentials that were used in the response file, which has the following default values:

- ▶ User ID: admin
- ▶ Password: Passw0rd (capital P with a zero)

## 4.2 Setting up the build environment for PowerVM virtual appliances

This section focuses on the suggested requirements for the PowerVM Express cloud provider (referred to as *PowerVM provider*) and IBM Image Construction and Composition Tool to successfully implement your build environment. As of writing, two management platforms support the PowerVM provider:

- ▶ The IBM Flex System Manager (FSM), as part of IBM PureFlex System and Flex System offerings, is a Systems Management appliance that provides a preintegrated and virtualized management environment across servers, storage, and networking. Because the FSM is an appliance, you can begin using the FSM as the PowerVM provider with minimal setup and configuration of the management platform.
- ▶ IBM Systems Director is a platform-management foundation that streamlines the way you manage physical and virtual systems across heterogeneous environments. Unlike the FSM, IBM Systems Director does not come in an appliance form and requires the setup and configuration of servers, storage, networking, and software to support this management platform.

The Systems Director management platform can be implemented in several ways, but this section does not explain them in detail. IBM does offer documentation that provides reference configurations that are used to deliver private cloud solutions on IBM POWER® Systems. For more information, see the following documents:

- ▶ *Implementing IBM SmartCloud Entry on POWER Systems using the PS703 BladeCenter H Reference Configuration*  
<ftp://public.dhe.ibm.com/common/ssi/ecm/en/poo03077usen/P0003077USEN.PDF>
- ▶ *Implementing IBM SmartCloud Entry on POWER Systems using the POWER 740 Express Reference Configuration*  
<ftp://public.dhe.ibm.com/common/ssi/ecm/en/poo03078usen/P0003078USEN.PDF>

This chapter covers the setup and configuration of the build environment for PowerVM virtual appliances. The build environment consists of the IBM Systems Director management

platform, as the PowerVM provider, and IBM Image Construction and Composition Tool. Although this section focuses on the Systems Director management platform, some of these steps are also required for the PureFlex and Flex System environment and are noted where applicable. The recommendations in this chapter are based on the authors' experience with creating these environments.

This section contains the following topics:

- ▶ Reference architecture
- ▶ Requirements and recommendations
- ▶ Creating the build environment

## 4.2.1 Reference architecture

Figure 4-1 illustrates the reference architecture diagram that defines the components of the build environment.

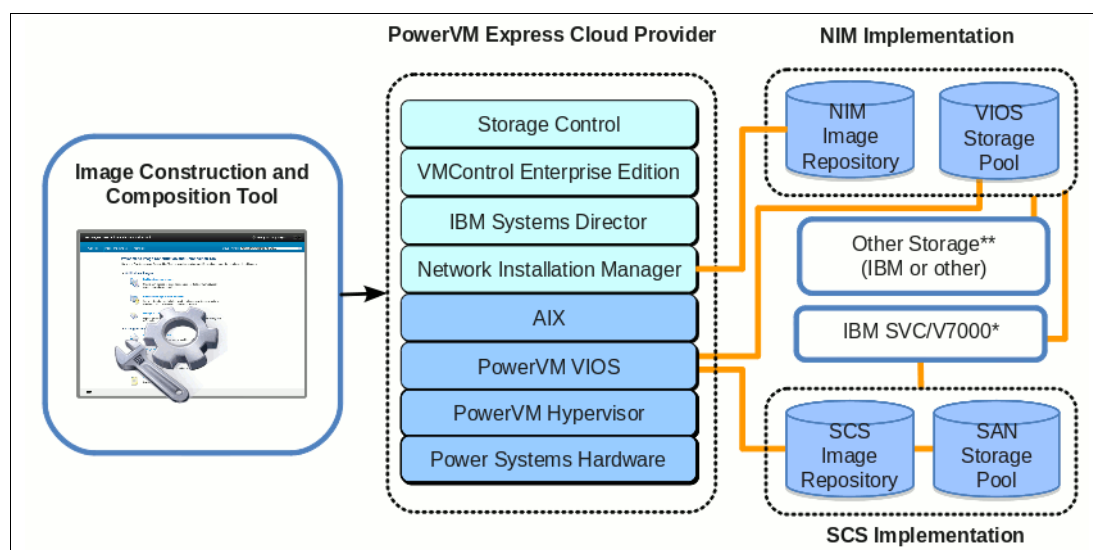


Figure 4-1 PowerVM virtual appliance build environment reference architecture

The Image Construction and Composition Tool can be installed on Linux (x86) or AIX (IBM POWER). In our environment, we prepared an additional AIX partition to install Image Construction and Composition Tool.

At its core, the PowerVM provider has IBM Systems Director, IBM Systems Director VMControl, and IBM Systems Director Storage Control components that manage IBM Power Systems™ hardware and storage. The IBM Flex System Manager, part of the IBM PureFlex System offering, is a self-contained appliance. It provides key components with the capability to manage the IBM Flex System chassis, nodes, switches, and storage. You can use IBM PureFlex System and Flex System to build virtual appliances. However, if you plan to create a build environment by using existing hardware and software resources, follow the guidelines provided in this section. A reference architecture for the PowerVM provider in an AIX environment is shown in Figure 4-1.

In this scenario, the PowerVM provider is an AIX, Virtual I/O Server-hosted partition created on IBM Power Systems hardware with the required management software stack:

- ▶ IBM Systems Director
- ▶ IBM Systems Director VMControl
- ▶ IBM Systems Director Storage Control

Optionally, Network Installation Management (NIM) can be installed on the same partition if you want to use NIM for your deployments or initial builds.

The storage in the environment is external fiber attached to Virtual I/O Server (VIOS). Although Systems Director supports several storage devices, we suggest using the SAN Volume Controller or IBM Storwize® V7000 storage. By using this storage, you can take advantage of the VMControl Storage Copy Services and build appliances for AIX, Linux, and IBM i.

The following sections describe the details of each component and the corresponding requirements.

## 4.2.2 Requirements and recommendations

This section describes the requirements and recommendations for the build environment. In the creation of the build environment, ensure that the PowerVM cloud provider is set up and configured. The cloud provider consists of IBM Power Systems hardware and software and IBM Systems Director for the management software foundation.

You must be familiar with the following components and concepts:

- ▶ Power Systems
  - Hardware Management Console (HMC) to manage Power Systems
  - Integrated Virtualization Manager (IVM) to manage IBM BladeCenter® Power Blades
  - Virtual I/O Server
  - AIX operating system
- ▶ Storage
  - Storage Volume Controller or Storwize V7000
  - Storage Fabric Zoning
  - Storage Copy Services
- ▶ Internet Protocol network
  - Private and public network concepts
- ▶ Management software stack
  - IBM Systems Director
  - Systems Director VMControl
  - Systems Director Storage Control
  - Optional: NIM on AIX

This section also highlights the requirements and recommendations for hosting the Image Construction and Composition Tool in an AIX partition. In addition, it provides a high-level task list to guide you through the implementation of the build environment.

**Important:** The recommendations in this section are based on our experience with creating the build environment. You might be required to assign more resources, depending on the complexity of your environment and the needs of your team.

### Hardware

This section describes the hardware components of the reference architecture and provides a list of requirements and recommendations.

## **IBM Power Systems hardware**

To create the build environment, use one of the following IBM Power Systems:

- ▶ Power Systems (IBM POWER6®, IBM POWER7® and IBM POWER7+™) managed by HMC
- ▶ BladeCenter POWER processor-based blade (POWER6, POWER7 and POWER7+) managed by IVM
- ▶ Flex POWER processor-based nodes (POWER7 and POWER7+) managed by FSM

To determine the supported combination of HMC and server firmware, access one of the following links:

- ▶ HMC and POWER6 processor-base systems:  
<http://www-304.ibm.com/webapp/set2/sas/f/power5cm/supportedcode.html>
- ▶ HMC and POWER7, POWER7+ processor-base systems:  
<http://www-304.ibm.com/webapp/set2/sas/f/power5cm/supportedcodep7.html>

Non-managed or stand-alone IBM Power Systems are not supported as the base hardware component of the cloud provider. When determining the hardware resources for the cloud provider, pay attention to the number and size of the virtual images that you will create and manage. At a minimum, you need a system with an 8-core processor and 64 GB of memory for the cloud provider.

## **Storage hardware**

Use external storage to host partitions and the repositories. Two types of storage pools can be configured for capture and deployment:

- ▶ PowerVM shared storage pool

With PowerVM shared storage pools, you can create storage pools that can be accessed by any Virtual I/O Server (VIOS) virtual servers that are members of the shared storage pool. You can edit these shared storage pools by adding or removing VIOS virtual servers. You can also edit the storage pool that is associated with the shared storage pool by adding or replacing physical volumes, or changing the storage pool threshold. The storage pool is created by directly mapping a volume to the VIOS and creating a shared storage pool by using the HMC or IVM GUI interfaces or command-line interface (CLI). When capturing to the VIOS storage pool, a logical unit is created and disk block copies to be stored in the common repository on the VIOS server. When you deploy to the VIOS storage pool, a linked clone is created for the deployed partition.

**Note:** As of writing, the Image Construction and Composition Tool does not support using PowerVM shared storage pools.

- ▶ SAN storage pool

You can use SAN storage pools with both the NIM and VMControl Storage Copy Services deployment methods. A Storage Management Initiative Specification (SMI-S) provider and the Systems Director Storage Control plug-in are required. The storage pool is created by zoning together a SAN storage device and a VIOS fiber adapter. Any arrays or pools created are detected by Systems Director VMControl when configuring the pool. When deploying to the SAN storage pool, a volume is created on the storage device for the deployed partition. When capturing to the SAN storage pool, a volume is created on the storage device to store the image in the common repository. With a SAN storage pool, you use VMControl Storage Copy Services to copy and create the volumes.

A recommendation is to use SAN storage pools, so you can capture and deploy virtual appliances for AIX, Linux, and IBM i. This section describes the setup of an environment that is based on VMControl Storage Copy Services (SCS) by using Storwize V7000 and SAN storage pools. We typically define three types of storage pools. The *management storage pool* is used to host partitions that are used to manage or build appliances in the environment. These partitions are typically the management, NIM and the Image Construction and Composition Tool LPARs. The *deployment storage pool* is used to host deployed appliance instances. These instances are created when you deploy a virtual appliance using VMControl or synchronizing an extended image using the Image Construction and Composition Tool (the *tool*). The *SCS repository storage pool* is used to store virtual appliances that are captured by VMControl or the tool. In addition to the storage pools, we can also define a NIM repository, which is a volume that is created in the management storage pool.

Use the following guidelines to size your storage:

|                                    |  |
|------------------------------------|--|
| <b>Management storage pool</b>     | 200 GB + (3 x largest virtual appliance) + (size of NIM repository volume) |
| <b>Deployment storage pool</b>     | 2 x number of deployed partitions x average size                           |
| <b>SCS repository storage pool</b> | 3 x number of virtual appliances x average size                            |
| <b>NIM repository volume</b>       | 1 x number of virtual appliances x average <code>mksysb</code> size        |

**Notes:**

- ▶ Storage pools: Place all storage arrays or pools in a RAID 5 configuration for best capacity and performance. A single pool can be configured for the management, deployment, and SCS repository storage pools; separate pools are not required.
- ▶ Additional storage: Additional storage is required for the VIOS, management, and Image Construction and Composition Tool partitions and described later in this section.

In addition to the storage setup, implement SAN zoning to map the storage to hosts, which is done by using the Fibre Channel switches. At a high level, the VIOS and the storage subsystem must be able to communicate with each other on the SAN storage fabric.

The Systems Director Storage Control plug-in also requires that an SMI-S provider is set up and configured within the environment for the appropriate Fibre Channel Switches. For more information about managing SMI-S providers, see “Managing SMI-S providers” in the IBM Systems Director information center:

[http://publib.boulder.ibm.com/infocenter/director/pubs/topic/com.ibm.director.storage.helps.doc/fqm0\\_t\\_sm\\_managing\\_smis\\_providers.html](http://publib.boulder.ibm.com/infocenter/director/pubs/topic/com.ibm.director.storage.helps.doc/fqm0_t_sm_managing_smis_providers.html)

## Software

The reference architecture consists of several software components, which have several requirements.

### *Virtual I/O Server*

After your Power Systems hardware and storage are in place, install the VIOS. Use the following configuration when creating the partition:

|                            |  |
|----------------------------|--|
| <b>Assigned processing</b> | A 0.1 processing unit with a single core virtual processor, and a weight factor of 255.  |
| <b>Assigned memory</b>     | 4 GB.  |
| <b>Disk</b>                | 50 GB + (2 x largest image size).<br>The disk can be a local or external SAN LUN.  |
| <b>Physical adapters</b>   | At least one network adapter to be used for shared Ethernet adapter (SEA) and at least one fiber adapter to attach to the external SAN storage.                            |
| <b>Virtual adapters</b>    | One virtual SCSI adapter for the Management LPAR, one virtual SCSI adapter for the Image Construction and Composition Tool LPAR, and one virtual Ethernet adapter for SEA. |
| <b>VIOS/IVM version</b>    | Version 2.2.2.1  |

**Disk size:** The reason for the large disk size for VIOS rootvg is that the export function of the Image Construction and Composition Tool temporarily stores the disk image in the /home/padmin directory.

### *Management Server*

Figure 4-1 on page 50 illustrates the management software stack for the cloud provider that consists of IBM Systems Director, Systems Director VMControl, and Systems Director Storage Control. Before installing the software stack, you must configure and set up a “manager” AIX LPAR. You can also use the management LPAR as a NIM server if one does not exist in your environment.

**Note:** The management LPAR is not required for IBM PureFlex System or Flex System because it is prepackaged as the Flex System Manager appliance. In the following sections, the setup and configuration of a management LPAR are described. These steps can be ignored if you are using PureFlex System or Flex System. If you require a NIM server in your environment, use the configuration details described in “Base AIX image” on page 55 and size the disk appropriately as described in “Storage hardware” on page 52.

Use the following configuration for the management LPAR:

|                            |   |
|----------------------------|---|
| <b>Assigned processing</b> | A 2.0 processing unit, a 4-core virtual processor, and a weight factor of 128.  |
| <b>Assigned memory</b>     | 8 GB.   |
| <b>Disk</b>                | Disk 1 with 60 GB (rootvg), which can be a local or external SAN LUN; optional Disk 2 for NIM repository, which can be a local or external SAN LUN. |
| <b>Virtual adapter</b>     | At least one virtual SCSI adapter and at least one virtual Ethernet adapter.  |

Table 4-1 shows the software levels.

Table 4-1 Software levels

| Software                                      | Level      |
|---|------------|
| AIX   | 7100-02-01 |
| IBM Systems Director                          | 6.3.2      |
| Systems Director VMControl Enterprise Edition | 2.4.2      |
| Systems Director Storage Control              | 4.2.2.143  |

**More information:** For information about installing IBM Systems Director on AIX, see *Installing IBM Systems Director with managed IBM DB2 database on AIX*:

[http://publib.boulder.ibm.com/infocenter/director/pubs/topic/com.ibm.director.main.helps.doc/fqp0\\_bk\\_install\\_server\\_aix\\_managed\\_db2.pdf](http://publib.boulder.ibm.com/infocenter/director/pubs/topic/com.ibm.director.main.helps.doc/fqp0_bk_install_server_aix_managed_db2.pdf)

### **IBM Image Construction and Composition Tool**

Before installing the Image Construction and Composition Tool, you must configure and set up an additional AIX LPAR.

Use the following partition configuration to host the Image Construction and Composition Tool:

|                            |  |
|----------------------------|--|
| <b>Assigned processing</b> | A 0.2 processing unit, with a 2-core virtual processor and a weight factor of 128. |
| <b>Assigned memory</b>     | 4 GB.  |
| <b>Disk</b>                | 100 GB + (3 x largest image size)<br>Can be a local or external SAN LUN.           |
| <b>Virtual adapter</b>     | At least one virtual SCSI adapter and one virtual Ethernet adapter.                |
| <b>Versions</b>            | AIX 7.1 TL02 SP01 and Image Construction and Composition Tool 2.2.1.1.             |

### **Base AIX image**

An initial base AIX image might be required for your initial appliance builds. This base image has the following software requirements:

|                                |  |
|--------------------------------|--|
| <b>Assigned processing</b>     | A 0.2 processing unit with a single core virtual process and a weight factor of 128. |
| <b>Assigned memory</b>         | 4 GB.  |
| <b>Disk</b>                    | 10 GB.   |
| <b>Virtual adapter</b>         | At least one virtual SCSI adapter and one virtual Ethernet adapter.                  |
| <b>AIX release and version</b> | AIX 7.1. TL01 SP01.  |
| <b>Additional software</b>     | OpenSSH, OpenSSL, <code>unzip</code> , and <code>tar</code> utilities.               |



### ***Base Linux image***

An initial base Linux image might be required for your initial appliance builds. This base image has the following software requirements:

|                                  |  |
|----------------------------------|--|
| <b>Assigned processing</b>       | A 0.2 processing unit with a single core virtual process and a weight factor of 128.   |
| <b>Assigned memory</b>           | 4 GB.  |
| <b>Disk</b>                      | 10 GB.   |
| <b>Virtual adapter</b>           | At least one virtual SCSI adapter and one virtual Ethernet adapter.  |
| <b>Linux release and version</b> | SLES 10 SP3.<br>RHEL 5.4.  |
| <b>Additional software</b>       | Service and productivity tool RPMs:<br><a href="http://www14.software.ibm.com/webapp/set2/sas/f/lopdiags/home.html">http://www14.software.ibm.com/webapp/set2/sas/f/lopdiags/home.html</a> |

### ***Base IBM i image***

An initial IBM i base image might be required for your initial appliance builds. This base image has the following software requirements:

|                                  |   |
|----------------------------------|---|
| <b>Assigned processing</b>       | A 0.5 processing unit with five virtual processors and a weight factor of 128.  |
| <b>Assigned memory</b>           | 8 GB.   |
| <b>Disk</b>                      | 100 GB.   |
| <b>Virtual adapter</b>           | At least one virtual SCSI adapter and one virtual Ethernet adapter.   |
| <b>IBM i release and version</b> | IBM i 7.1 TR3.  |
| <b>Additional software</b>       | The following PTFs are required: PTF SI48604 (in cum 3037), SI45682, SI43673, SI44762, SI44642, SI43821, MF53425, SI46118, SI44195, MF53194, SI43756 and SI44754. |

**Note:** Depending on the intended use of the base image for applications, additional CPU, memory, or disk space might be required.

## **Network**

You can set up the network in your build environment in several ways. For simplicity reasons, use a single network. If you decide to have two networks, use one as your management network (private), and use the other network as your deployment network (public). All components of the build environment continue to have a single management interface, except for the VIOS and deployed instances, which have both networks.

For the Image Construction and Composition Tool to export virtual appliances, it must be able to communicate over the network with the storage subsystem, VIOS, and target system (scp) for export.

To prepare for building the environment, identify the following details of the network:

- ▶ Subnet network
- ▶ Subnet mask
- ▶ Subnet gateway
- ▶ DNS
- ▶ Available target deployment range of IP addresses

### 4.2.3 Creating the build environment

Before you create the build environment, you must ensure that the hardware and networking are in place. This section is based on the assumption that the hardware and network are already setup and working.

For information about setting up IBM Power Systems hardware, see the IBM Power Systems Hardware Information Center at:

<http://pic.dhe.ibm.com/infocenter/powersys/v3r1m5/index.jsp>

If you are using SAN Volume Controller or Storwize V7000, you can find more information at:

- ▶ IBM Storwize V7000 Information Center

<http://publib.boulder.ibm.com/infocenter/storwize/ic/index.jsp>

- ▶ IBM SAN Volume Controller Information Center

<http://publib.boulder.ibm.com/infocenter/svc/ic/index.jsp>

After setting up the hardware and network, make sure that communication between the devices is in place to perform the tasks that are described in the following sections. The following sections describe the setup using SAN storage pools and VMControl Storage Copy Services.

#### Setting up the storage

In the following storage setup steps, the correct zoning and communication between the storage device and VIOS host are already in place:

1. Create the storage pool on the Storwize V7000.

**Note:** Multiple storage pools can be created based on the function, for example, management, deployment, or repository pools. In this example, we created a single pool.

2. Create volumes for the VIOS, management, and Image Construction and Composition Tool partitions in the pool.

**Tip:** If you plan to set up NIM, create an additional volume to attach to the management LPAR, or size the single volume for management LPAR appropriately.

3. Map the volumes to the VIOS host.

#### Setting up the Virtual I/O Server

To set up the VIOS, use the following steps:

1. From the HMC or Flex System Manager interface, create the VIOS partition. If you are installing on a POWER processor-based blade server, you do not need to create a partition.
2. Create appropriate virtual SCSI and virtual Ethernet adapters. Take into account the hosted management and Image Construction and Composition Tool LPARs.
3. Activate the partition to begin the VIOS installation. For the POWER processor-based blade server, power on the blade.

**Tip:** When presented with the installation options, select the volume that you designated for the VIOS installation. If you created volumes for the management and Image Construction and Composition Tool LPAR, they are displayed in your hdisk list.

After the installation is complete, the partition restarts.

4. Log in with the padmin user when the console becomes available.
5. Configure VIOS networking. Set up a shared Ethernet adapter, and assign the networking address.
6. Tune VIOS:

- a. Configure the VIOS for optimal performance and availability. Tune the following parameters for each Fibre Channel port:

|                      |  |
|----------------------|--|
| <b>num_cmd_elems</b> | Set to the maximum value allowed. Reboot is required:<br><code>chdev -dev fcsX -attr num_cmd_elems=&lt;max&gt; -perm</code>    |
| <b>max_xfer_size</b> | Set the maximum size of the FC packets that are sent:<br><code>chdev -dev fcsX -attr max_xfer_size=0x1000000 -perm</code>      |
| <b>dyntrk</b>        | Enable dynamic tracking of the volumes:<br><code>chdev -dev fcsX -attr dyntrk=yes -perm</code>                                 |
| <b>fc_err_recov</b>  | Use to try another path if the current one is unresponsive:<br><code>chdev -dev fcsX -attr fc_err_recov=fast_fail -perm</code> |

- b. Reboot the VIOS:

```
shutdown -force -restart
```

7. Enable the Director Common Agent and on subsequent restarts:

```
startsvc DIRECTOR_agent  
cfgsvc DIRECTOR_agent -attr RESTART_ON_REBOOT=TRUE
```

8. Create a Virtual Optical Media Library to prepare for installing the management LPAR and Image Construction and Composition Tool LPAR.

**Virtual Optical Media Library:** By creating a Virtual Optical Media Library in VIOS, you can upload the ISO files of the installation media, such as AIX 7.1 DVD, and then use that media to install other partitions. For information about setting up the Virtual Optical Media Library, see the “Creating a Virtual Optical drive in an HMC-Managed VIOS Partition” topic:

<http://www.ibm.com/support/docview.wss?uid=nas15dd15b31badfa4738625755b00811805>

Installation of the VIOS is complete. Additional setup is required for a deployment and capture based on VMControl Storage Copy Services for Systems Director VMControl.

## Creating the management or NIM LPAR

To create the management or NIM LPAR, complete the following steps:

1. From the IVM or HMC interface, create an AIX LPAR:
  - a. Map the appropriate disk and networking to the partition.
  - b. When prompted to select Virtual Optical Media, select the installation media for AIX 7.1.
2. Activate the partition to begin the AIX installation.
3. After the installation is complete, follow the prompts by the installation assistant to set the root password and networking.

4. After the installation, complete these steps:
  - a. Change the paging size to 4 GB.
  - b. Install OpenSSH and OpenSSL.
  - c. Install the **unzip** and **tar** utilities.
5. Tune AIX:
  - a. Change the root user limits as shown in Example 4-1.

*Example 4-1 Changing the root user limits*

---

```
# chuser fsize=-1 root
# chuser nofiles=3000 root
# chuser data=-1 root
# chuser rss=-1 root
```

---

**Tip:** The changes do not take effect immediately. Exit from the current shell, and then log in again to apply the changes. Run the **ulimit -a** command to see whether your changes took effect.

- b. Change the file system size as shown in Example 4-2.

*Example 4-2 Changing the file system sizes*

---

```
# chfs -a size=1G /
# chfs -a size=4G /usr
# chfs -a size=2G /var
# chfs -a size=12G /tmp
# chfs -a size=1G /
# chfs -a size=16G /opt
```

---

If you created this partition as a NIM LPAR, you can skip to “Installing NIM” on page 62.

## Installing the management software stack

To install the management software stack, complete the following steps:

1. Download IBM Systems Director, plug-ins, and updates. You will be required to log in with an IBM ID and password. To download the latest installable media, go to the Systems Director download page:

<http://www.ibm.com/systems/software/director/downloads>

On the Management servers tab, scroll to and select **IBM Systems Director 6.3.x (Power)**. On the next page, download the DVD image by selecting **IBM Systems Director Server 6.3.2 for AIX**.

For the plug-ins, on the same web page, click the **Plug-ins** tab, and then select and download the **Systems Director Storage Control** plug-in.

**Tip:** To host the installation packages, have a file system size of 15 GB.

2. Install IBM Systems Director by entering the following command:  

```
./dirinstall.server -r dirserv.rsp.
```

3. Configure the agent manager by entering the following command:

```
/opt/ibm/director/bin/configAgtMgr.sh -user <agent_username> -agtmgrpass  
<agent_password> -regpass <registration_password>
```

4. Manually start IBM Systems Director by entering the following command:

```
/opt/ibm/director/bin/smstart
```

### **Systems Director VMControl plug-in**

As of IBM Systems Director 6.3, the Systems Director VMControl plug-in is installed by default. However, you are required to activate the plug-in, which provides a 90-day license for Systems Director VMControl Enterprise.

To activate the plug-in, enter the following command on the Systems Director server:

```
/opt/ibm/director/bin/smcli activatemgrs VMControl
```

You can obtain the Systems Director VMControl Enterprise license from IBM PartnerWorld® (signing in is required):

[http://www.ibm.com/partnerworld/mem/pat/pat\\_sas\\_systemsdirector\\_plugin\\_keys.html](http://www.ibm.com/partnerworld/mem/pat/pat_sas_systemsdirector_plugin_keys.html)

After obtaining the license package, extract it to a location on the Systems Director server, and run the following commands:

```
<extracted location>/vmcontrol_image_manager_key_aix.sh -i silent  
<extracted location>/vmcontrol_system_pools_key_aix.sh -i silent
```

After installing the license key, stop and start the Systems Director Server service.

### **Updating IBM Systems Director and Systems Director VMControl**

To update IBM Systems Director and Systems Director VMControl, use the following steps:

1. Install the necessary updates from a location if you downloaded the fixes:

```
/opt/ibm/director/bin/smcli installneeded -v <location of fixes>
```

Alternatively, do an update from the IBM Systems Director Home tab of the web UI (Figure 4-2).

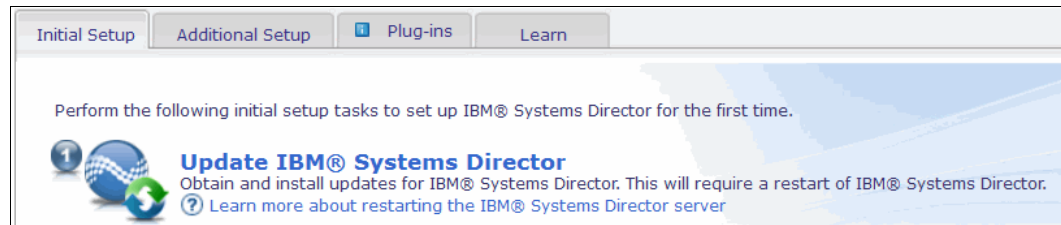


Figure 4-2 Updating System Director and Systems Director VMControl by using the web UI

2. Restart IBM Systems Director.

## Installing the Systems Director Storage Control plug-in

The VMControl Storage Copy Services implementation requires the installation of the Systems Director Storage Control plug-in.

To install the Systems Director Storage Control plug-in, complete the following steps:

1. Extract the installation file for Systems Director Storage Control.

**Important:** You must install Systems Director Storage Control when IBM Systems Director is active. To check whether IBM Systems Director is active, type the following command:

```
/opt/ibm/director/bin/smsstatus
```

2. Start the installation of the Systems Director Storage Control plug-in:

```
./StorageControlInstall.sh
```

3. To accept the license agreement, press the Spacebar key until the end of the text. Type 1, and then press Enter.

4. Restart IBM Systems Director by entering the following commands:

```
/opt/ibm/director/bin/smsstop  
/opt/ibm/director/bin/smsstart  
/opt/ibm/director/bin/smsstatus -r
```

5. Check whether the TCP instance is running (Figure 4-3).

```
netstat -an | grep 9550  
tcp        0      0  *.9550          *.*              LISTEN
```

Figure 4-3 TCP listening on the default port

6. Check that a Farm object is created in IBM Systems Director (Figure 4-4).

```
smcli lssys -t Farm  
mgr.itso.ibm.com
```

Figure 4-4 Farm object

The installation of Systems Director Storage Control is complete.

**Increasing the JVM heap size of Systems Director Storage Control:** Use the following steps to increase the Systems Director Storage Control JVM heap size:

1. Stop the IBM Systems Director server by entering the following command:

```
/opt/ibm/director/bin/smsstop
```

2. Open the `server.xml` file in a text editor. The file is in the

```
/opt/IBM/TPC/device/apps/was/profiles/deviceServer/config/cells/DefaultNode/  
nodes/DefaultNode/servers/server1/ folder.
```

3. Search for JVM arguments, and change the value of the `-Xmx512m` parameter to `-Xmx1536m`. Then save the modified `server.xml` file.

4. Start the IBM Systems Director server by entering the following command:

```
/opt/ibm/director/bin/smsstart
```

## ***Installing NIM***

A dedicated volume is expected for the NIM image repository. To install NIM:

1. Configure the storage for NIM by entering the following commands:

```
mkvg -y nimvg <hdisk>
crfs -v jfs2 -g nimvg -m /export/nim -a size=500G -A yes
mount /export/nim
```

2. Install the following required file sets for the NIM master configuration:

- dsm.core
- openssh.base.client
- openssl.base
- bos.sysmgmt.nim.master
- bos.sysmgmt.nim.spot
- bos.sysmgmt.nim.client
- bos.net.nfs.server

3. Verify that the AIX installation media is in the optical device and is attached to the management partition.

4. Create the NIM master and its initial resource:

```
/usr/sbin/nim_master_setup -a mk_resource=yes -a volume_group=nimvg -a  
device=/dev/cd0
```

The installation of the management software stack is complete.

After completing these steps, you can install a virtual server from the NIM. To create an image repository for use with Systems Director VMControl for NIM-based deployments, continue with the steps in “Configuring the management software stack” on page 62.

## **Configuring the management software stack**

You can configure the VMControl and Storage Control plug-ins in IBM Systems Director or Flex System Manager for deployments and captures based on NIM and VMControl Storage Copy Services. For an environment based on VMControl Storage Copy Services, we use Storwize V7000 to detail the configuration tasks.

### ***Setting up VMControl Storage Copy Services***

The Systems Director or Flex System Manager Storage Control plug-in is required for this implementation.

To create an environment to use VMControl Storage Copy Services, use the following steps:

1. Create an SSH key pair for adding the Storwize V7000 data source:

- a. Create a pair of SSH keys on the IBM Systems Director server so that IBM Systems Director can access Storwize V7000. Enter the following command:

```
ssh-keygen -t rsa
```

- b. Obtain the public key from the IBM Systems Director server because you need to upload it to Storwize V7000 by using the browser GUI.
- c. Log in to the Storwize V7000 browser, create a user, and upload the public key for the password file.
- d. Test the results by running **ssh** to connect to the Storwize V7000 from the IBM Systems Director server. Confirm that you are not prompted for a password.

2. Configure Systems Director Storage Control. To add the Storwize V7000 and SAN switches as data sources to IBM Systems Director Storage Control:

- a. Add the Storwize V7000 data source by running the following command:

```
smcli datasource -c svc -f /.ssh/id_rsa -v V7000 -i <IP of V7000>
```

- b. Add the SAN switch data source by running the following command:

```
smcli mkdatasource -c fabric -t <protocol> -i <IP of SMI-s provider> -p <port> -u <username> -w <password> -n "<namespace>"
```

- c. Collect inventory by running the following command:

```
smcli collectinv -p "All Inventory"
```

After collecting the inventory, Storwize V7000 and switches are displayed in IBM Systems Director or Flex System Manager. Furthermore, if you look at the inventory of the Storwize V7000, you see the storage pools that you created previously.

**Namespace for the SAN switch:** When adding the data source for the SAN switch, you specify the SMI-S provider information: protocol (HTTP or HTTPS), IP address, port, user name, and password. The namespace is specific to the SAN switch for which you are creating the fabric data source. For more information, see the Interoperability Namespace List:

<http://www.ibm.com/support/docview.wss?uid=swg21366393>

**Enabling IBM FlashCopy® in VMControl Storage Copy Services:** By enabling FlashCopy, you can decrease copy times when capturing workloads and deploying virtual appliances. To enable IBM FlashCopy, enter the following command on the IBM Systems Director server:

```
smcli mksvcsshrsap -s "<Name of Storwize V7000 device in Systems Director>" -u <Username created on Storwize V7000> -f/.ssh/id_rsa
```

This command uses the ssh private key that is generated in step 1 on page 62. To find the name of the Storwize V7000, run the **smcli lssys** command.

### ***Configuring the VMControl plug-in***

To configure the VMControl plug-in, use the following steps:

1. From the VIOS CLI, disable the Common Agent Services (CAS) agent:

```
$ oem_setup_env
# /var/opt/tivoli/ep/runtime/agent/bin/endpoint.sh stop
```

2. From the Systems Director or Flex System Manager CLI, discover the VIOS partition:

```
# smcli discover -i <IP of VIOS>
```

3. Access the VIOS partition:

```
# smcli accesssys -u padmin -p <padmin Password> <IP of VIOS>
```

4. Add a CAS connection to the Systems Director server:

```
# /opt/ibm/director/agent/runtime/agent/toolkit/bin/configure.sh -amhost <Director IPAddr> -passwd <Agent Registration password> -force
```

5. Collect inventory:

```
# smcli collectinv -p "All Inventory"
```



## Creating an image repository

You can create two types of image repositories:

- ▶ Create a NIM-based image repository:
  - a. Install the Common Agent Subagent for VMControl NIM on the NIM server.
  - b. Restart the IBM Systems Director server.
  - c. Confirm that you can see the image repository on the **Virtual appliance** tab of the VMControl Plug-in.
- ▶ Create the image repository for VMControl Storage Copy Services:
  - a. Define the correct components on Systems Director or Flex System Manager, and make sure that they are available. Run the **smcli dumpstcfg** command, and make sure that the storage configuration (switch and storage subsystem) and host accessible containers (VIOS can see the SAN storage pools) are available. Each storage pool must be listed under each VIOS.

Example 4-3 shows the SAN configuration output.

*Example 4-3 SAN configuration output from the smcli dumpstcfg command*

---

```
# smcli dumpstcfg
SAN Configuration
-----
Switches
-----
Name      OID      Provider IP      Switch IP      WWPN

IBM8Gb    12411 192.168.71.117  192.168.71.117  100000C0DD2415DF

Storage Controllers
-----
Name      OID      Provider IP      Subsystem IP      Largest
Slice
Storwize V7000-2076-V7000a-IBM 14744 172.27.34.1      { '192.168.71.120' }
2729.0

Host Accessible Containers
-----
NAME: STORAGE SUBSYSTEM/POOL
VIOS1:  Storwize V7000-2076-V7000a-IBM/mdiskgrp0
```

---

- b. Install the Common Agent Subagent for the VMControl common repository.
- c. Create a common repository from the VMControl Basics tab. Enter a name for the repository, select the VIOS partition, and select the storage pool created for the image repository.
- d. Confirm that you can see the image repository on the Virtual Appliance tab of the VMControl Plug-in.

**More information:** For more information about installing Systems Director VMControl subagents, see “Installing IBM Systems Director VMControl subagents using the installation wizard” topic in the IBM Systems Director V6.3 Information Center:

[http://pic.dhe.ibm.com/infocenter/director/pubs/topic/com.ibm.director.vim.help.s.doc/fsd0\\_vim\\_t\\_installing\\_agent\\_wizard.html](http://pic.dhe.ibm.com/infocenter/director/pubs/topic/com.ibm.director.vim.help.s.doc/fsd0_vim_t_installing_agent_wizard.html)

### **Summary**

The configuration of the management platform is now complete and you now have a functioning PowerVM provider. Next, you configure the PowerVM provider and create a base operating system appliance. For information about configuring the PowerVM provider with the Image Construction and Composition Tool, see Chapter 7, “PowerVM Express cloud provider” on page 125.

## **4.3 Setting up the build environment for KVM virtual appliances**

Creating virtual appliances is an iterative process. As your experience with the Image Construction and Composition Tool grows, you might notice a need for a basic infrastructure setup to streamline the management of appliances under construction.

To help you efficiently construct robust appliances, consider the following artifacts and components:

- ▶ OS ISO images
- ▶ Prebuilt base images
- ▶ Installable images
- ▶ Bundles

In a larger organization, these components are often shared by several cloud architects and specialists. Therefore, keep a central, well-organized, and easily accessible repository of these artifacts to eliminate redundancy and limit tedious and time-consuming management tasks.

In addition, isolate your appliance construction environment from your in-house public network. Building images involves moving around fairly large amounts of data. The ability to control the data flow can shorten image build times.

### 4.3.1 Reference architecture

Figure 4-5 illustrates the reference architecture for building KVM appliances.

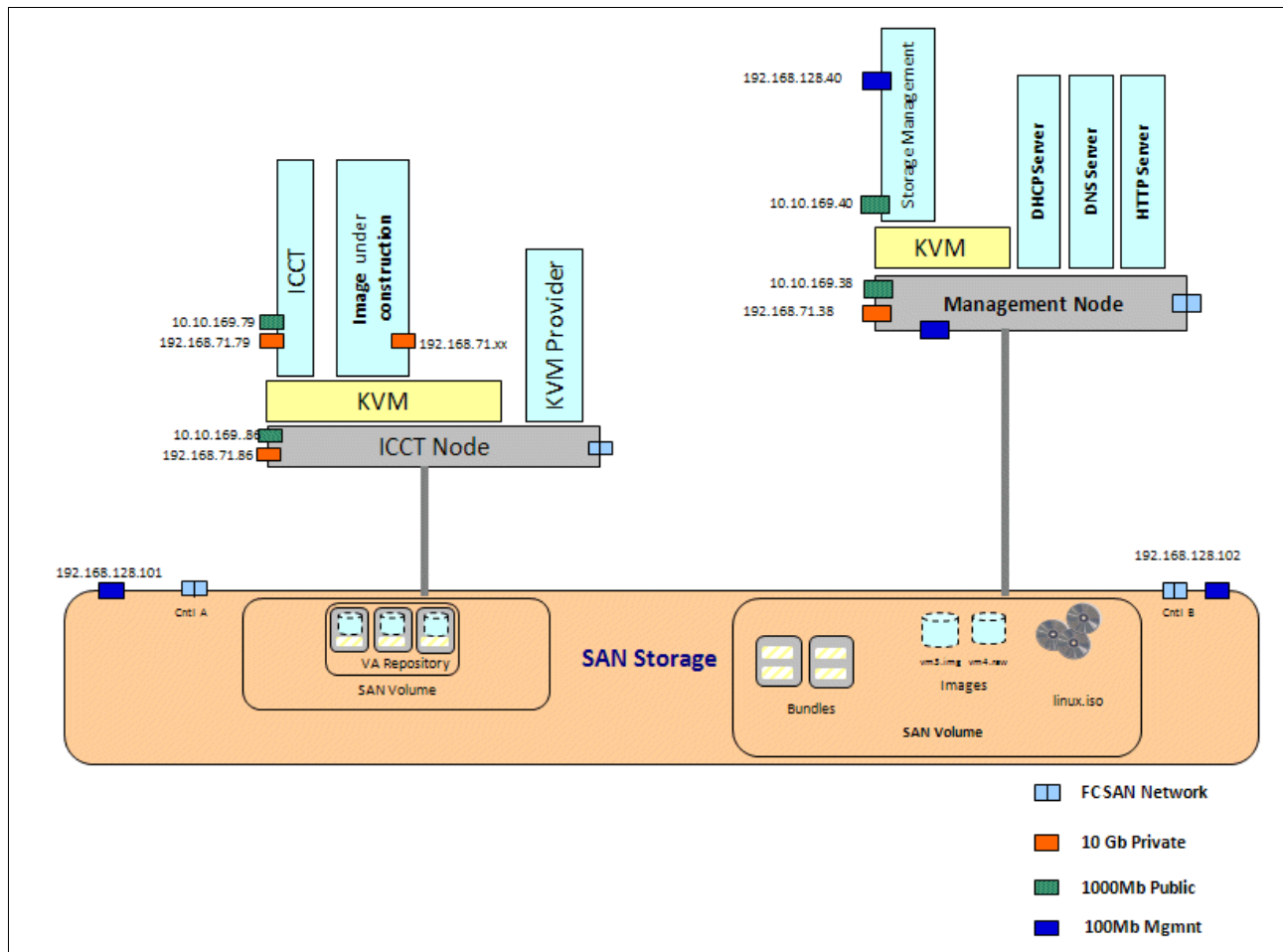


Figure 4-5 Reference architecture of the build environment for KVM

### 4.3.2 Nodes

The reference architecture has the following nodes (systems), each of which play an important role:

- ▶ *ICCT node*, which is the end-to-end virtual appliance construction process is performed.
- ▶ *Management node*, which includes several repositories that contain the artifacts that are needed in the appliance creation process. It also hosts the HTTP, DNS, and DHCP services.

#### The ICCT node

The ICCT node is a System x system that is set up as a KVM host. For a script that illustrates how to install the KVM hypervisor and the associated tools, see Appendix A, “Sample script for KVM host setup” on page 281.

The host has two network interfaces:

- |                      |  |
|----------------------|--|
| <b>10Gb Ethernet</b> | Is used in the private network. The entire HTTP and NFS traffic between the ICCT and management nodes flows over this interface. The inter-node communication among the ICCT instance, the KVM provider, and the virtual images that run the local KVM hypervisor is also routed through this interface. |
| <b>1Gb Ethernet</b>  | Is used in the public network and enables the communication with the build environment from any place in the organization.   |

The ICCT node mounts a SAN-attached volume. The ICCT node also hosts the KVM cloud provider. The KVM provider home directory (`/var/lib/va`) is on the SAN volume.

The Image Construction and Composition Tool itself runs as a KVM appliance, which has several advantages:

- ▶ You can run several instances of the Image Construction and Composition Tool, each of which are assigned to a different appliance architect.
- ▶ You can simultaneously run different versions of the Image Construction and Composition Tool, which can be useful if you need to support other users or different configurations.

**Tip:** Create a virtual machine (VM) image with at least 20 GB of disk space for the Image Construction and Composition Tool instance. The disk space is needed to import the ISO images used to construct base appliances. An average ISO file size for the commonly used Linux distributions 4 GB.

## The management node

Similarly to the ICCT node, the management node is a System x server that is set up as a KVM host. This node has three network interfaces:

- |                        |  |
|------------------------|--|
| <b>10 Gb Ethernet</b>  | Is used in the private network. The entire HTTP and NFS traffic between the ICCT and management nodes flows over this interface. |
| <b>1 Gb Ethernet</b>   | Is used in the public network and enables communication with the node from any place in the organization.                        |
| <b>100 MB Ethernet</b> | Is used to connect the management node to the SAN Storage Controller.  |

The management node mounts a SAN volume that is used to host several repositories. The home directory for the repositories is `/export/repositories`.

Each repository, which is a subdirectory of the home directory, contains a collection of objects of the same type:

- ▶ **Bundles**  
A repository contains reusable bundles for middleware products such as WebSphere Application Server, DB2, tomcat, and user applications. The middleware product specialists maintain the middleware bundles. The software architects maintain the application bundles. The cloud solution architects import the bundles into the Image Construction and Composition Tool to assemble the virtual appliances.
- ▶ **ISO**  
A repository contains ISO images for the supported distributions (RHEL and SLES). These ISO images can be imported into the Image Construction and Composition Tool to use the Create from ISO function. The images are also used to build base virtual appliances by using tools such as **virt-manager**.

- Repositories based on the yum style

Each supported distribution also has a repository based on yum and hosted on the management node. A yum repository is mounted in the guest OS of virtual appliances under construction so that the software prerequisites for a bundle or a software component can be satisfied.

- Installation images

A repository contains software installable images, such as the DB2 installation image and the installation scripts, that are needed to construct bundles or virtual appliances.

- OVAs

Virtual appliances are packaged in the industry standard Open Virtualization Format (OVF). They are the base appliances that were tested and validated and can be used to assemble final virtual appliances.

- Virtual images

Virtual images are in their hypervisor-specific format. We use the raw format for the KVM hypervisor. The OS specialists build these images in different sizes (small, medium, and large) to accommodate various application needs. The images are built in accordance to the standards of the organization, such as security, user profiles, and custom management agents.

### 4.3.3 Tips and techniques

An experienced Linux administrator should not have problems with most of the tasks for setting up the Image Construction and Composition Tool and management node. This section includes tips and techniques that are based on real-life experiences to help you avoid common traps.

**Example environment:** The examples in this section are based on an RHEL server installation. You must adjust the paths and files names if you installed the nodes on SLES.

#### HTTP server

The most practical way to share the repositories that are hosted on the management node is through the HTTP protocol. Therefore, set up a separate virtual server for that purpose. Example 4-4 shows a relevant excerpt from the `http.conf` configuration file.

*Example 4-4 Creating a virtual host configuration in the `http.conf` file*

---

```
<VirtualHost *:80>
    ServerAdmin jarek@dummy-host.example.com
    DocumentRoot /export/repositories
    ServerName cbx3650.ovafactory.ibm.com
    ErrorLog logs/cbx3650.ovafactory.ibm.com-error_log
    CustomLog logs/cbx3650.ovafactory.ibm.com-access_log common

    Alias /repositories "/export/repositories"

    <Directory "/export/repositories">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

---

In the line with the bold highlighting, an alias is defined so that the URL to access the repositories root directory is reduced to the following address:

`http://cbx3650/repositories/`

Because an internal private network is used for inter-node communication, encryption through HTTPS is not necessary.

## DHCP server

The DHCP server helps to dispense IP addresses in the private network. For example, when you create a base image from the ISO during the initial installation, the guest OS tries to obtain an IP address through DHCP.

To begin, ensure that the DHCP demon listens only on the private network interface. Otherwise, its services are broadcasted through all the interfaces, including the public network interface, which can cause chaos in the public network of your organization.

To instruct the demon to work only over the private network, set the **DHCPDARGS** parameter in the `/etc/sysconfig/dhcpd` file, for example: `DHCPDARGS="br2"`. In this case, the service listens only on the `br2` interface.

It is convenient to assign a specific IP address based on the MAC address of the guest for the network interface. For example, you might want to boot a prebuilt image that uses DHCP and to assign a specific IP to that image so that you can use `ssh` to perform administration or configuration tasks. In this case, you complete the following steps:

1. Provide the IP to MAC mapping in the `/etc/dhcpd.conf` configuration file as in the following example:

```
# we want the virtual servers to appear at a fixed addresses
host vs100 {
    option host-name "vs100.ovafactory.ibm.com";
    hardware ethernet 00:16:3e:6c:00:64;
    fixed-address 192.168.71.100;
}
```

In this case, a network interface with the MAC address of `00:16:3e:6c:00:64` is assigned a fixed IP address of `192.168.71.100`. Notice also the presence of the `host-name` option, which is used in the DNS server configuration.

2. Use the predefined MAC address in the VM domain descriptor to make sure that the VM boots with an IP address as in the following example:

```
<interface type='bridge'>
  <mac address='00:16:3e:6c:00:64' />
  <source bridge='br0' />
  <model type='virtio' />
</interface>
```

Upon boot, the guest OS IP address is set to `192.168.71.100`.

## DNS server

The DNS server helps you to access the virtual appliances under construction by using their host names rather than the IP addresses. The IP-hostname mapping for the private network is provided in the `/var/named/data/db.ovafactory.ibm.com` configuration file. For example, the following example shows the entry for the `vs100` virtual system:

|  |                 |                    |  |
|--|-----------------|--------------------|--|
| <code>vs100.ovafactory.ibm.com.</code> | <code>IN</code> | <code>A</code>     | <code>192.168.71.100</code>              |
| <code>vs100.ovafactory.ibm.com.</code> | <code>IN</code> | <code>HINFO</code> | <code>"Virtual Server" "LINUX"</code>    |
| <code>vs100.ovafactory.ibm.com.</code> | <code>IN</code> | <code>MX</code>    | <code>10 mail.ovafactory.ibm.com.</code> |
| <code>vs100.ovafactory.ibm.com.</code> | <code>IN</code> | <code>TXT</code>   | <code>""</code>                          |

Make sure that you also provide a corresponding entry in the `db.192.168.71` configuration file as in the following example:

```
100    IN    PTR vs100.rchland.ibm.com
```

This entry ensures that the DNS server properly handles the reverse lookup requests.

## Setting up a repository based on yum

During the virtual appliance construction process, you need an access from a guest OS to the software repository that contains the OS installable image. Because a repository based on yum works well for both RHEL and SLES, set up such repositories for the distributions you need in your environment.

The following example steps show how to create a repository for RHEL 6.2. In this example, the ISO images are already copied to the `/export/repositories/ISO` repository on the management node.

1. If needed, install the **createrepo** utility on the management node:

```
yum --nogpgcheck install createrepo
```

2. Create the directory for the repository:

```
mkdir -p /export/repositories/rhel/6.2/x86_64/iso/{1,2}
```

3. Mount the ISO on the newly created repository directory:

- a. Add the following command to the `/etc/fstab` configuration file:

```
/export/repositories/ISO/RHEL6.2/RHEL6.2-20111117.0-Server-x86_64-DVD1.iso  
/export/repositories/rhel/6.2/x86_64/iso/1          iso9660 ro,loop 0 0
```

- b. Mount the new mount point:

```
mount -a
```

4. Use the **createrepo** command to generate the necessary XML metadata:

```
cd /export/repositories/rhel/6.2/x86_64/iso  
createrepo .
```

5. Validate the setup:

- a. Log in to a guest that is running RHEL 6.2 and create the `rhel62-base.repo` definition file in the `/etc/yum.repos.d` directory with the following content:

```
[rhel62-base]  
name=RHEL 6.2 disk 1 image  
baseurl=http://cbx3650.ovafactory.ibm.com/repositories/rhel/6.2/x86_64/iso/1/  
gpgcheck=0  
enabled=1
```

- b. Run the following command on the guest OS:

```
yum repolist
```

The command provides output similar to the following results:

| repo id         | repo name             | status |
|-----------------|-----------------------|--------|
| rhel62-base     | RHEL 6.2 disk 1 image | 3,391  |
| repolist: 3,391 |                       |        |

To set up a repository for systems based on SLES, use the following similar approach:

1. On the management node, enter the following command:

```
mkdir -p /export/repositories/sles/11.1/x86_64/iso/{1,2}
```

- a. Add the following command to the `/etc/fstab` configuration file:

```
/export/repositories/ISO/sles/11.1/SLES-11-SP1-DVD-x86_64-GM-DVD1.iso  
/export/repositories/sles/11.1/x86_64/iso/1      iso9660 ro,loop 0 0
```

- b. Enter the following command:

```
cd /export/repositories/sles/11.1/x86_64/iso  
createrepo .
```

2. On the client system based on SLES, enter the following command:

```
zypper ar --type yast2 -f  
"http://cbx3650.ovafactory.ibm.com/repositories/sles/11.1/x86_64/iso/1" "SLES  
11 SP1 x86_64 D1"  
zypper lr
```

The following example shows the expected output:

| # | Alias                 | Name                  | Enabled | Refresh |
|---|-----------------------|-----------------------|---------|---------|
| 1 | SLES 11 SP1 x86_64 D1 | SLES 11 SP1 x86_64 D1 | Yes     | Yes     |







# Product Activator Development Kit

The Product Activator Development Kit (PADK) is an Eclipse plug-in that streamlines the process of creating, testing, and validating the IBM Virtual Solutions Activation Engine (VSAE) extensions. The PADK and the IBM Image Construction and Composition Tool (referred to as *the tool* in this book) have the following relationship:

- ▶ The Image Construction and Composition Tool is the utility that assembles various artifacts, such as installation images, installation, and configuration scripts, into a deployment-ready virtual image (appliance).
- ▶ The PADK is a software development tool that you can use to quickly code, test, and validate the configuration scripts that are required by the VSAE.

The IBM Image Construction and Composition Tool uses what the PADK produces.

This chapter describes the PADK functionality and includes the following sections:

- ▶ Overview of the Product Activator Development Kit
- ▶ Installing PADK
- ▶ A walkthrough of the development process by using PADK

**Previous knowledge:** The assumption for this chapter is that you understand and have practical experience with the following concepts:

- ▶ Virtual appliance standard as codified by the Distributed Management Task Force (DMTF) Open Virtualization Format (OVF) specification
- ▶ Virtual Solutions Activation Engine
- ▶ Eclipse integrated development environment (IDE)
- ▶ Python

## 5.1 Overview of the Product Activator Development Kit

In this document, the term *activation program* is used as a synonym of *product activator*. The product activator is a more formal term that is codified by the DMTF specification. Similarly, the IBM Image Construction and Composition Tool is the virtual appliance construction toolkit. This tool is sometimes referred to as *ICCT* in window captures or messages.

By using the PADK, you can design, implement, and test the VSAE extensions (activation programs for your software components) from the comfort of the Eclipse IDE. The Eclipse IDE was chosen because of its popularity and ease of use. The PADK is delivered as an Eclipse plug-in.

The VSAE does not dictate which programming language to use to create the activation programs. The programs can be written in any programming language that is supported on the guest operating system if the parameters can be passed and retrieved in one of the formats that are supported by the VSAE.

Python was chosen as the programming language that is directly supported by PADK for several reasons. VSAE is written in Python, is popular, offers ease of use, and is supported on all key guest operating systems. Selecting one specific language does not limit the usefulness of the PADK. A script or program written in any language can be easily started from a Python wrapper.

PADK has the following main functions:

- ▶ Uses the PyDev Eclipse plug-in for Python source-code management with color-coded keywords and automatic formatting.
- ▶ Provides a GUI-based Program Activator Extension Editor that you can use to specify the activation program parameters and their default values, in addition to OS Service Dependencies.
- ▶ Automatically generates the Python program templates for both activate and reset actions.
- ▶ Generates the VSAE artifacts that are needed for effective testing, such as the activation logic file and the environment file (`ovf-env.xml`).
- ▶ Automatically transfers the source code and other components to the target virtual machine (VM) and runs the test from the Eclipse IDE.
- ▶ Supports remote debugging on the target VM (virtual system). As of this writing, remote debugging is not enabled on AIX, IBM i, or Windows.
- ▶ Generates the bundle that is compliant with the Image Construction and Composition Tool and transfers it to the instance of the tool.
- ▶ Exports generated bundles directly from the instance of the tool.

The target VM can be any platform that is supported by the VSAE, except IBM i. As of this writing, the PADK plug-in does not support IBM i.

## 5.2 Installing PADK

To install the PADK on your development workstation, complete the following steps:

1. Install Eclipse SDK version 3.7 or newer.
2. Install the Eclipse PyDev plug-in. In Eclipse IDE, select **Help** → **Install New Software**. Then in the Install window (Figure 5-1):
  - a. For Work with, enter the following link to the software repository for the PyDev plug-in:  
pydev - <http://pydev.org/updates>
  - b. Under Name, select **PyDev** and **PyDev for Eclipse**.
  - c. Select the following check boxes:
    - Show only the latest versions of available software
    - Group items by category
    - Contact all update sites during install to find required software
  - d. Click **Next**.

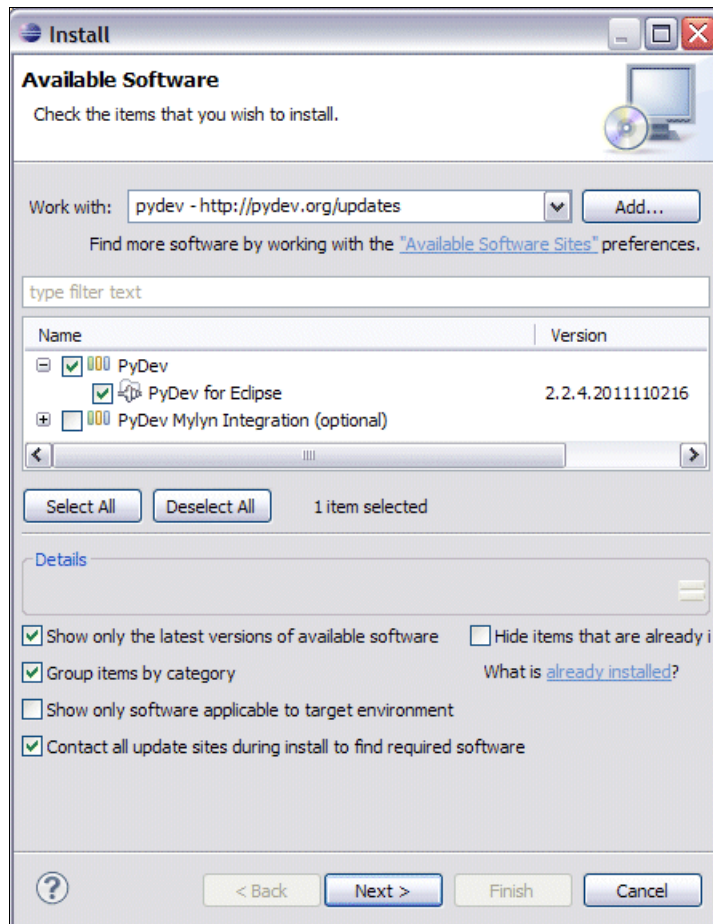


Figure 5-1 Installing the PyDev Eclipse plug-in

3. Install the PADK plug-in. In Eclipse IDE, select **Help** → **Install New Software**. Then, in the Install window (Figure 5-2):
  - a. For Work with, enter the link to the software repository for the PADK plug-in:  
PADK Plugin - <http://hmb30icn:9099/padkp2repo>
  - b. Under Name, select **IBM VADK PA Extension** and **VADK PA Extension Feature**.
  - c. Select the following check boxes:
    - Show only the latest versions of available software
    - Group items by category
    - Contact all update sites during install to find required software
  - d. Click **Next**.

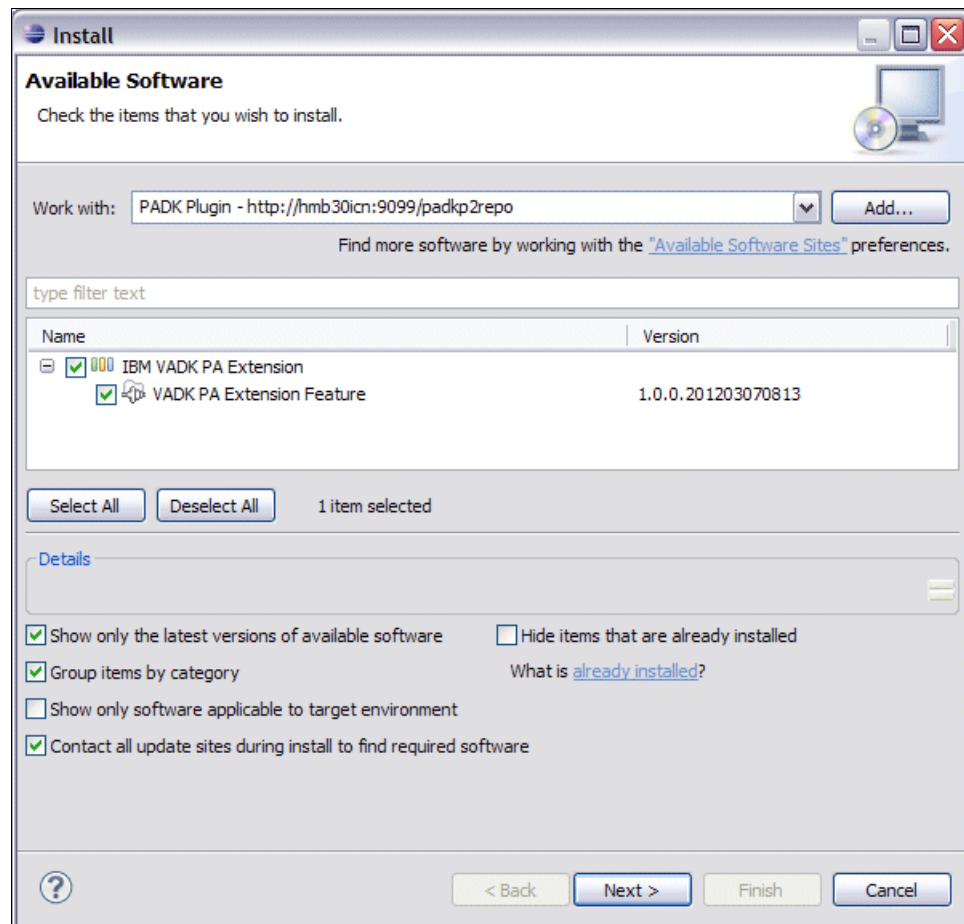


Figure 5-2 Installing the PADK

The PADK is included as part of the Image Construction and Composition Tool. Therefore, the Image Construction and Composition Tool must be installed and running for this repository to be accessible.

**Tip:** You can set up the PADK installation image on a separate HTTP server:

1. Copy the PADK installation repository from the `/drouter/padkp2repo` directory on the Image Construction and Composition Tool system to the HTTP server.
2. Make the HTTP directory accessible through the HTTP protocol. For more information about the HTTP server setup, see “HTTP server” on page 68.

**Updating the PADK:** Future releases of the Image Construction and Composition Tool will provide updated versions of the PADK. You can upgrade the PADK by following the installation procedure in this section. The installer recognizes the fact that you already have an instance of the PADK in your Eclipse environment and automatically switches to the upgrade process. You must restart Eclipse for the changes to take effect.

## 5.3 A walkthrough of the development process by using PADK

To illustrate a typical PADK use case, this section takes you through an end-to-end development process. This scenario is based on a VM image with SUSE Linux Enterprise Server (SLES) 11.1 and WebSphere Application Server 7 already installed. The task is to make sure that, when the appliance is deployed for the first time at a new location (data center), the new (different) host name is applied to the WebSphere Application Server instance.

To begin, be sure that a supported version of the VSAE is installed in the VM. Use the tool to install VSAE automatically. As explained in Chapter 6, “KVM Express cloud provider” on page 91, one of the following two processes produces an image with the installed and configured VSAE:

- ▶ Creating a base virtual appliance from the ISO (currently supported by the KVM provider)
- ▶ Importing a running VM

In most cases, you do not need to manually install the VSAE. However, if you must manually install VSAE for debug or test purposes, follow the steps in 5.3.1, “Installing the Virtual Solutions Activation Engine manually” on page 77.

### 5.3.1 Installing the Virtual Solutions Activation Engine manually

**Important:** Use the manual installation method for VSAE only for debugging and testing purposes.

The Image Construction and Composition Tool includes the VSAE installable images for the RHEL and SLES Linux distributions. To install the VSAE into your VM image, you must have access to a system that is running the tool.

To install the VSAE for SLES, use the following steps:

1. Copy the VSAE installation image into the VM:

```
cd /var/opt
scp root@iccthost:/drouter/vsaebundles/sles11/* .
```

Where *iccthost* is the system that is hosting the Image Construction and Composition Tool instance.

Example 5-1 shows the files that are copied from the system of the tool.

*Example 5-1 Copied files*

---

```
ls -la
-rw-r--r-- 1 root root 115830 Mar 15 13:01
com.ibm.icon.kvm.enablement.sles11_1.0.0.6.ras
-rw-r--r-- 1 root root 662 Mar 15 13:01 install.sh
-rw-r--r-- 1 root root 2613 Mar 15 13:01 ovf-env.xml
```

---

2. Make the **install.sh** script runnable:

```
chmod +x install.sh
```

3. Be sure that the **unzip** utility is installed in the VM. If the utility is not installed, install it.

```
zypper in unzip
```

4. Run the installer:

```
./install.sh
```

5. Validate that the activation services were successfully configured:

```
ls -la /etc/rc.d/rc5.d
```

Example 5-2 shows the list of services that are scheduled to run at run level 5 and the activation services.

*Example 5-2 List of services that are scheduled to run at run level 5 and the activation services*

---

```
lrwxrwxrwx 1 root root 6 Jun 24 2011 S01kbd -> ../kbd
lrwxrwxrwx 1 root root 22 Mar 15 13:21 S02activate.ntp-client ->
../activate.ntp-client
lrwxrwxrwx 1 root root 7 Mar 15 12:44 S02dbus -> ../dbus
lrwxrwxrwx 1 root root 14 Jun 24 2011 S02earlysyslog -> ../earlysyslog
lrwxrwxrwx 1 root root 19 Jun 24 2011 S02java.binfmt_misc ->
../java.binfmt_misc
lrwxrwxrwx 1 root root 9 Jun 24 2011 S02random -> ../random
lrwxrwxrwx 1 root root 12 Mar 15 12:44 S03haldaemon -> ../haldaemon
lrwxrwxrwx 1 root root 10 Mar 15 12:44 S03network -> ../network
lrwxrwxrwx 1 root root 29 Mar 15 13:21 S04activate.network-interface ->
../activate.network-interface
lrwxrwxrwx 1 root root 23 Mar 15 13:21 S04activate.system-user ->
../activate.system-user
lrwxrwxrwx 1 root root 19 Mar 15 12:44 S04network-remotefs ->
../network-remotefs
lrwxrwxrwx 1 root root 7 Mar 15 12:44 S04sshd -> ../sshd
lrwxrwxrwx 1 root root 9 Mar 15 12:44 S04syslog -> ../syslog
lrwxrwxrwx 1 root root 6 Mar 15 12:44 S05ntp -> ../ntp
lrwxrwxrwx 1 root root 22 Mar 15 13:21 S06activate.dns-client ->
../activate.dns-client
lrwxrwxrwx 1 root root 23 Mar 15 13:21 S06activate.system-host ->
../activate.system-host
lrwxrwxrwx 1 root root 10 Mar 15 12:44 S06postfix -> ../postfix
lrwxrwxrwx 1 root root 10 Mar 15 12:44 S06rpcbind -> ../rpcbind
lrwxrwxrwx 1 root root 10 Mar 15 12:44 S06tomcat6 -> ../tomcat6
lrwxrwxrwx 1 root root 24 Mar 15 13:21 S07activation.ConfigIcon ->
../activation.ConfigIcon
lrwxrwxrwx 1 root root 7 Mar 15 12:44 S07cron -> ../cron
lrwxrwxrwx 1 root root 6 Mar 15 13:21 S08aed -> ../aed
```

---

The new service, `activation.ConfigIcon` (highlighted in bold text in Example 5-2 on page 78), is a watchdog service; it ensures that all other activate services run successfully.

The previous output is specific to Linux. On AIX, the services are set up in the `/etc/rc.d/rc2.d` directory.

### 5.3.2 Prototyping the activation program

As mentioned previously, your task is to create an activation program that sets the WebSphere Application Server host name to a new (appropriate) value. A good practice is to prototype the required action, which is fairly simple to do for the WebSphere Application Server configuration. You can use the **wsadmin** shell script to run WebSphere Application Server administrative commands as documented in the “Scripting the application serving environment (wsadmin)” topic in the WebSphere Application Server Information Center:

<http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.express.doc/info/exp/ae/welc6topscripting.html>

Specifically, you can use the **AdminTask.changeHostName** command to change the host name. The following example shows the syntax of the command for Jython:

```
/opt/IBM/WebSphere/Profiles/DefaultAppSrv01/bin/wsadmin.sh -lang jython -conntype NONE -c \"AdminTask.changeHostName ('[-nodeName DefaultAppSrvNode01 -hostName vmcvs100 ]')\"
```

This command requires two parameters that you need to pass them to the activation program. They are called **was\_hostname** and **was\_nodename** in this scenario.

### 5.3.3 Creating a product activator project in Eclipse

Next, create a Python activation program to reconfigure the host name in the WebSphere Application Server instance that runs in the VM:

1. Start the Eclipse IDE.
2. Switch to the PyDev perspective:
  - a. Select **File** → **New** → **Other**.
  - b. In the Select Wizard, click **IBM PADK** in the list and select **IBM PADK Project**. Click **Next**.
3. In the IBM PADK Extension Project wizard, enter a project name, for example, `was-hostname-config`. Click **Next**.
4. In the IBM PADK Project window (Figure 5-3 on page 80), do the following steps:
  - a. Enter the product name, which in this scenario is `was-hostname-config`.
  - b. Enter a unique ID, which in this scenario is `com.ibm.vsa.was-hostname-config`.
  - c. Enter a product description.
  - d. Select the OS type, which is **Linux** in this scenario.
  - e. Select the OS distribution, which is **SLES** in this scenario.
  - f. Select the OS arch, which is **X86-64** in this scenario.



- g. Enter the OS version, which is 11 in this scenario.
- h. Optional: To set the user profile under which the product activator is run, use the **Run As** parameter. The default value is **root**.

For Linux, you can change the value to a non-root profile. Make sure that the user profile that you specify has the correct set of authorities that allow access to the objects that the product activator code intends to manipulate.

For AIX, do not set this value.

**OS Service Dependencies:** As of writing, the OS Service Dependencies are not transferred to the Image Construction and Composition Tool bundle. Therefore, these dependencies must be specified through the interface of the tool after the bundle is created in the tool. For more information, see 5.3.7, “PADK integration with Image Construction and Composition Tool” on page 86.

- i. Click **Finish**.

**IBM PADK Project**  
Specify Product Information.

Product name: was-hostname-config  
 Unique ID: com.ibm.vsa.was-hostname-config  
 Product Description: Program Activator to reconfigure the WebSphere Application Server hostname upon the first boot of the appliance  
 OS Type: Linux  
 OS Distribution: SUSE Linux Enterprise Server (SLES)  
 OS Arch: x86-64  
 OS Version: 11  
 Run As:

OS Service Dependencies | PA Properties

| No. | Key          | Value               | Configurable | Required | Label         | Comment                       |
|-----|--------------|---------------------|--------------|----------|---------------|-------------------------------|
| 1   | was_hostname |                     | true         | false    | Hostname      | Hostname for the WAS instance |
| 2   | was_nodename | DefaultAppSrvNode01 | false        | false    | WAS node name | WAS node name                 |

Add Property Delete Property

< Back Next > Finish Cancel

Figure 5-3 Creating a PADK project in Eclipse

The project is created including the source code templates for the *activate* and the *reset* actions. Figure 5-4 shows the directory structure for the newly created project:

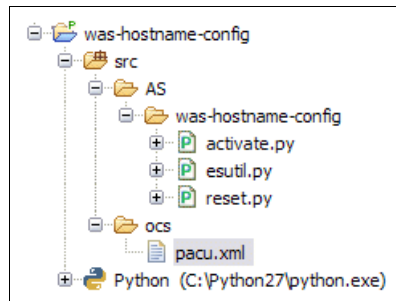


Figure 5-4 PADK project directory structure

The PADK project has the following artifacts:

- activate.py** The activation program that is started by the VSAE for the web-hostname-config configuration step.
- reset.py** The reset program that is called by the VSAE to reset the virtual appliance.
- esutil.py** A helper program that is used by **activate.py** program to parse the input parameters.
- pacu.xml** Metadata created by the Program Activator Extension Editor. It contains the activation logic metadata that is needed by the VSAE and the OVF section that is needed by the tool.

Example 5-3 shows additional excerpts from the **activate.py** source code.

Example 5-3 Excerpts from the activate.py source code

---

```
#!/usr/bin/python
...
class Activator:
    def __init__(self):
        try:
            #don't remove the lines below
            #BEGIN_KEYS_MARKER
            keys=["help", "was_hostname=", "was_nodename="]
            #END_KEYS_MARKER

            argmap = esutil.parse_cmd_args( argv, "hv", keys )
        except getopt.GetoptError, err:
            print str(err)
            self._usage()
            exit(2)

        if argmap.has_key( "-v" ):
            verbose = True
        elif argmap.has_key("-h") or argmap.has_key("--help"):
            self._usage()
            exit()

    def _usage(self):
        #don't remove the lines below
        #BEGIN_KEYVALUES_MARKER
```

```

        print '\nUSAGE:\n AS/was-hostname-config/' + os.path.basename(argv[0]) +
"""
--was_hostname --was_nodename DefaultAppSrvNode01""";
#END_KEYVALUES_MARKER

...
def main():
    #don't remove the line below
    #_____IBMDEBUGMARKER_____
    activator = Activator()
    activator.action()

if __name__ == "__main__":
    main()

```

---

In Example 5-3 on page 81, you *must not* remove or modify the sections marked as **1a**, **b**, and **c**. These sections are used to generate the correct parameter list and to enable remote debug. The markers are required only for scripts that are developed by using the PADK. Scripts that are created outside of the PADK do not need any of the markers. Line **2** in the example shows parsed and validated parameters that are passed to the activation program.

The following steps illustrate these concepts, in the Eclipse IDE:

1. Switch to the Program Activator Extension Editor.
2. Add a third parameter, for example, `was_admin`.
3. Save the changes.
4. Switch back to the **activate.py** source code view. The source code is automatically generated to handle this additional parameter.
5. Remove the parameter in the Program Activator Extension Editor.
6. Save the updates.

### 5.3.4 Coding the activation program

The activation program must do several tasks:

1. Retrieve the input parameters.
2. If a null or empty string is passed, set the parameter to a default value.
3. Construct the **wasadmin** command.
4. Run the command, and check the results.

### 5.3.5 Setting up the connection to the target virtual machine

To run and debug the activation program on the target VM, set up the connection to that system:

1. In the Eclipse IDE, select **Windows** → **Preferences**.
2. In the left pane of the Preferences window, click the **IBM PADK Extension** element. Choose **Target VSAE Hosts**.
3. In the Target VSAE Hosts window, click **New**.

4. In the Add Target VSAE Host window (Figure 5-5), do these steps:
  - a. Enter the host, port, user, password, and VSAE directory information to connect to the target VM.
  - b. Leave the SSHD Server text box empty.
  - c. Click **Test**.

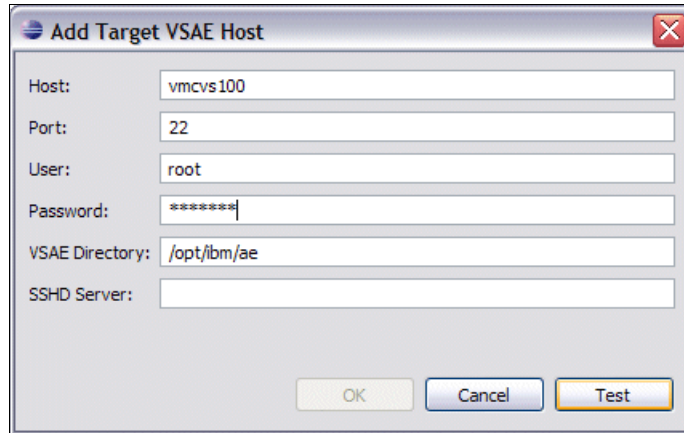


Figure 5-5 Adding a target host

The SSD Server information that is retrieved from the target VM is shown in the SSHD Server text box.

5. If you receive an error message that indicates that the connection failed, do these steps:
  - a. Enter **ssh** to connect to the VM.
  - b. Edit the sshd configuration file (/etc/ssh/sshd\_config). Change the **PasswordAuthentication** parameter to **yes**, and save the changes.
  - c. Restart the sshd service:

```
service sshd restart
```

For AIX, enter:

```
stop -s sshd  
start -s sshd
```

**Important:** If the test target system is an AIX system, check whether the path to the sftp subsystem is correct in the /etc/ssh/sshd\_config directory. In AIX 6.1, this path was changed from /usr/libexec/sftp-server to /usr/sbin/sftp-server. To validate the path, enter the following command:

```
Subsystem sftp /usr/sbin/sftp-server
```

6. Switch back to Eclipse, and retest the connection. Make sure that no errors are reported.

7. In the Target VSAE Hosts panel (Figure 5-6), make sure that the local host address shows the IP address of your workstation. Leave the port numbers at their default values. Then click **Apply**.

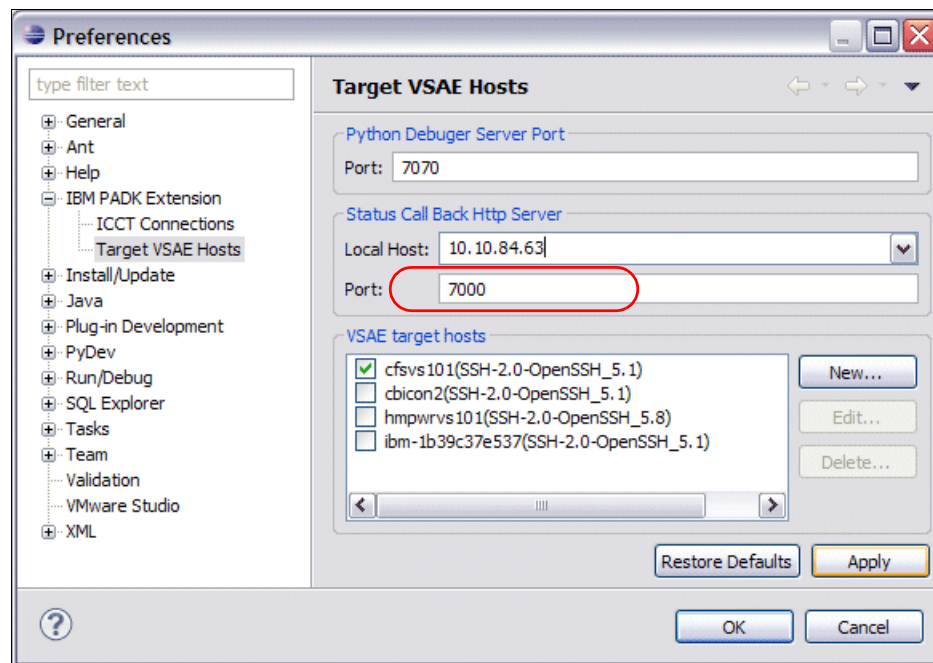


Figure 5-6 Details for the connection to the target VM

Next, run and, if needed, debug an activation program on the remote VM.

**Important:** Be sure that the corresponding default ports are open on both the target and local systems.

### 5.3.6 Running and debugging the activation program

To run the newly created activation program on the target machine, use the following steps:

1. In Eclipse, right-click the project in the left panel, and then select **Run As → Run Configurations**.
2. In the Run Configurations window (Figure 5-7 on page 85), do the following steps:
  - a. In the left pane, expand **PADK Extension Run**, and click the **New launch configuration** icon on the left side of the toolbar.
  - b. In the right pane, select the VSAE Target Host, and make sure that the local host IP address is set.
  - c. Click **Apply**, and then click **Run**.

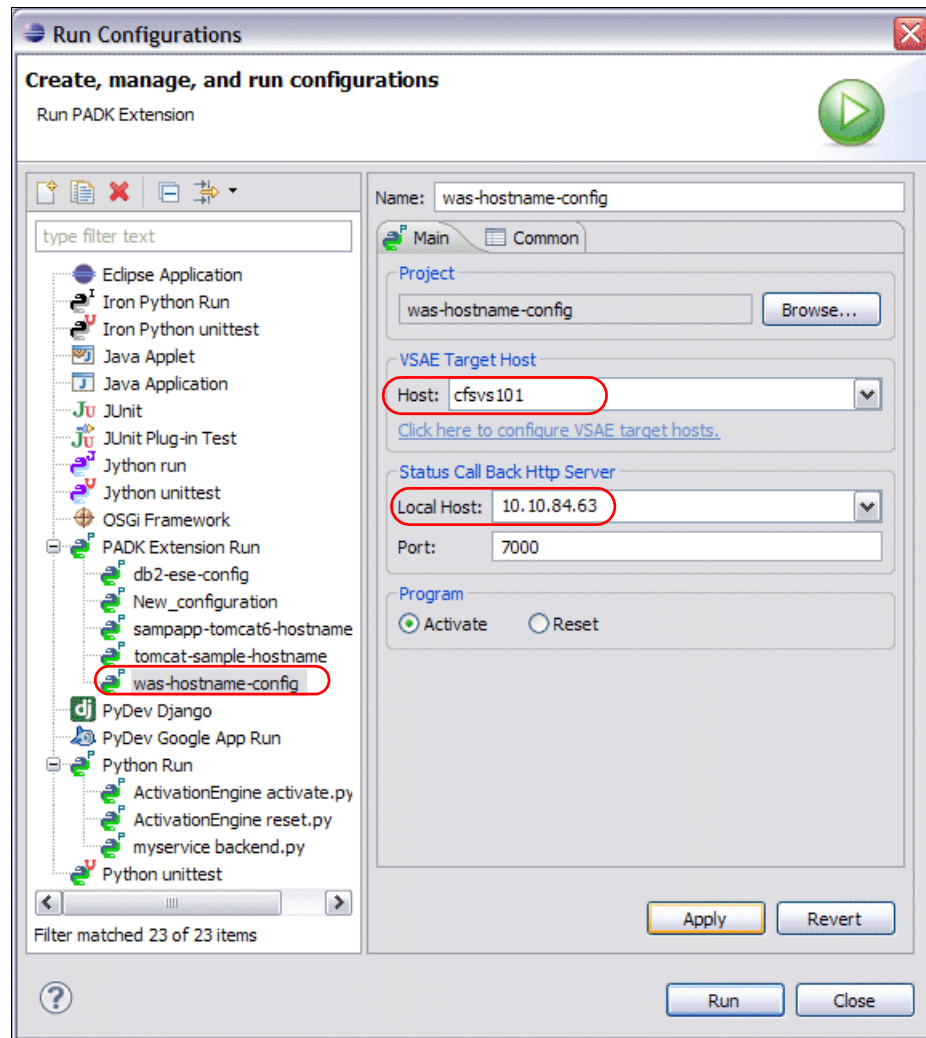


Figure 5-7 Running an activation program on a remote VM

3. Watch for any error messages in the Eclipse console.

If an error occurs, fix it by using the remote debugger:

1. To debug the activation program, right-click the project, and then select **Debug As** → **Run Configurations**.
2. In the left pane of the Run Configurations window, click **PADK Extension Run**, and click **New launch configuration**. Make sure that the values for host and local host are correct. Click **Debug**. The Eclipse perspective automatically switches to Debug, and the program execution stops at the entry point.
3. Set the breakpoints in the source code where you want the debugger to stop, and then press F8 (Resume). The execution resumes and continues until the first breakpoint is reached (see Figure 5-8 on page 86).

**Remote debug:** As of this writing, the remote debugger is not enabled for the AIX, IBM i, and Windows targets.

Figure 5-8 shows the Eclipse Debug perspective for the sample activation program. The activation program runs on a remote system (in a VM). The standard output is intercepted and redirected to the local system that is running Eclipse. The messages are displayed in the Output Console pane at the bottom. The variables can be set in the Variables pane in the right top corner.

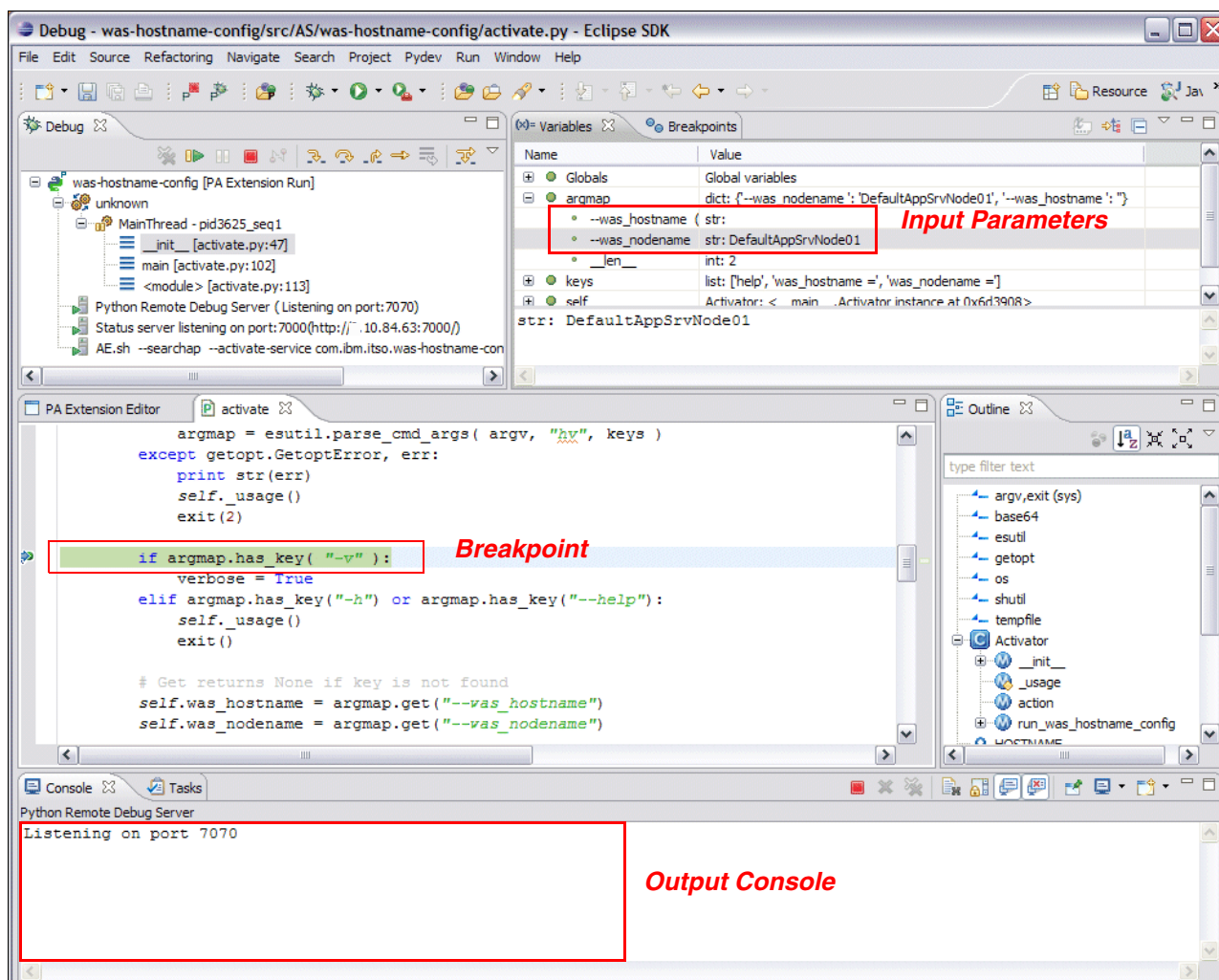


Figure 5-8 Remote debug session

### 5.3.7 PADK integration with Image Construction and Composition Tool

After successfully testing the activation program on the target VM, you can package it into a bundle that is compliant with the Image Construction and Composition Tool. Then push it into the bundle repository of your tool right from the Eclipse IDE.

#### Setting up the connection information for the tool

Before you start using PADK to manage the bundle, set up the Image Construction and Composition Tool connection information in the PADK:

1. In the Eclipse IDE, select **Windows** → **Preferences**.
2. In the Preferences window, in the left pane, click **IBM PADK Extension** element. Choose **ICCT Connections**.



3. In the Add ICCT Connection window (Figure 5-9), enter the information that pertains to the instance of your Image Construction and Composition Tool. Click **Test** to validate.

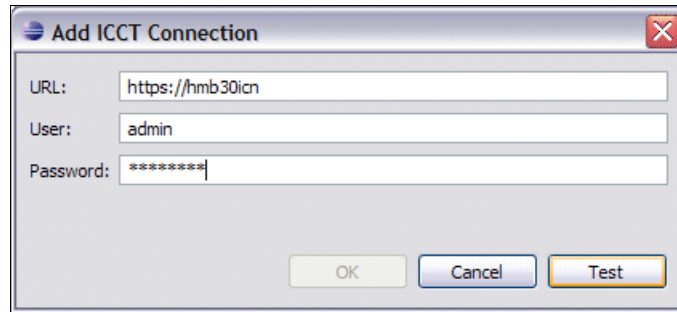


Figure 5-9 Defining a connection to the Image Construction and Composition Tool

4. Back in the Preferences window, click **Apply**.
5. After the connection is defined and successfully tested, right-click your PADK project and select **Connect to ICCT**.

## Managing the bundle

Now you are ready to use PADK to perform the following bundle management tasks directly from the Eclipse IDE:

- ▶ Create a bundle directly in the instance of the Image Construction and Composition Tool. PADK defines only the configuration step.
- ▶ After the bundle is created in the tool, retrieve the metadata of the bundle from the tool.
- ▶ Export the bundle from the tool.
- ▶ Delete the bundle from the tool.

## Creating a bundle for the current project

You create a bundle by using the Bundle Editor window (Figure 5-10 on page 88), which opens in the main Eclipse panel. The Bundle Editor has four tabs at the bottom that map to the corresponding Bundle Editor tabs in the Image Construction and Composition Tool:

- ▶ General
- ▶ Requirements
- ▶ Activation (mapped to the Configuration in the Image Construction and Composition Tool)
- ▶ Reset

The values for the parameters are filled automatically from the corresponding information in the product activator project. Therefore, no values are editable. To modify the bundle, you must go back to the Program Activator Extension Editor, make your changes, and save them. The metadata for the bundle is automatically re-created with the new information.

To create a bundle for the current project, in the Bundle Editor window, click **Create an Image Construction and Composition Tool bundle** icon in the upper-right corner of the Bundle Editor. The bundle is created in the Image Construction and Composition Tool. The three empty text fields (uri, Created on, and Updated date) on the General tab are populated with the information that is reported by the tool.

You can now switch to the GUI of the tool to inspect the content of the bundle that is generated by the PADK.



**General** [Configure ICON Connections](#)

**ICCT Connection**

ICCT Host:

**Base Information**

Name:

Repository:

Description:

Universal ID:

Version:

Publisher:

uri:

Created on:

Updated date:

**Products in the bundle**


| Product Name        | Version | Vendor   |
|---------------------|---------|----------|
| was-hostname-config | 1.0.0.0 | MyVendor |
|                     |         |          |
|                     |         |          |

**General** **Requirements** **Activation** **Reset**

Figure 5-10 Using the Bundle Editor

## Exporting the bundle

After you successfully test the deployment of the bundle to the target VM (by using the Image Construction and Composition Tool), export the bundle to be shared across tool instances and virtual appliance development teams:

1. Click the **Export** icon () in the Bundle Editor.
2. In the Export Bundle window (Figure 5-11 on page 89), enter the information that points to the location where you want to export:
  - a. Enter a host.
  - b. Enter a destination folder.
  - c. Enter a file name.
  - d. Enter a user name.
  - e. Enter a password.
  - f. Click **Export**, and watch for any messages.
  - g. When the transfer completes, click **OK**.

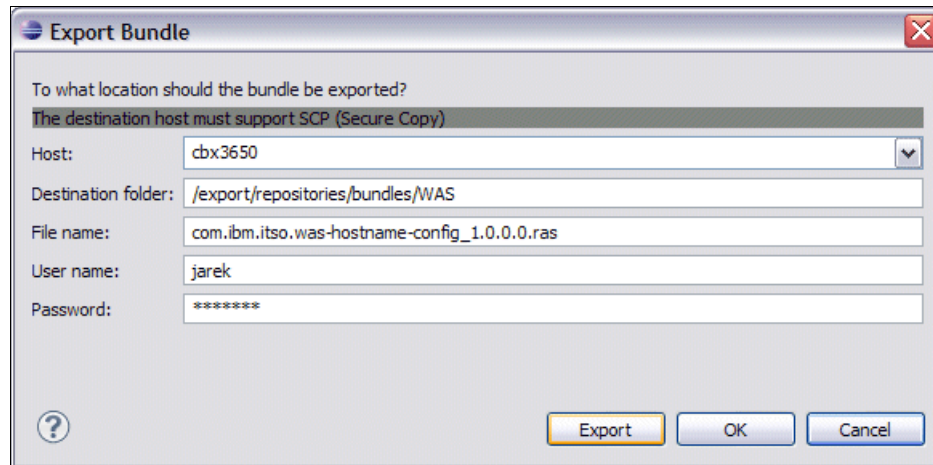


Figure 5-11 Exporting a bundle to an external file system

Keep a central repository of bundles that are accessible by all the tool instances that are running in your network. This process encourages sharing and eliminates redundancy.





## KVM Express cloud provider

The KVM Express cloud provider (referred to as *KVM provider* in this book) is a self-contained software component. It is fully compliant with the cloud provider specification of the IBM Image Construction and Composition Tool (referred to as *the tool* in this book). It supports the entire development cycle of a virtual appliance. It also targets deployments to Flex System x86 compute nodes in a PureFlex environment managed by a Flex System Manager and IBM System x nodes that are managed by IBM Systems Director VMControl system software.

This chapter provides an in-depth look into the internal workings of this cloud provider and explains how to use it to create robust, deployment-ready KVM virtual appliances. It includes the following sections:

- ▶ Overview of the KVM provider architecture
- ▶ Setting up the KVM provider
- ▶ Creating a base image from ISO
- ▶ Importing a running virtual machine
- ▶ Importing virtual images (appliances) from the KVM cloud provider
- ▶ Removing an image (appliance) from the virtual appliance repository
- ▶ Working with bundles
- ▶ Validating virtual images by using the OVA Runtime
- ▶ Tips to override default behavior of the KVM provider
- ▶ Troubleshooting

**Previous knowledge:** The assumption in this chapter is that you understand and have practical experience with the following concepts:

- ▶ KVM hypervisor and `libvirt`
- ▶ Virtual appliance standard as codified by Distributed Management Task Force (DMTF) Open Virtualization Format (OVF) specification
- ▶ Shell script programming

## 6.1 Overview of the KVM provider architecture

The KVM provider is a back-end node that implements the following functions that are required by the Image Construction and Composition Tool:

- ▶ Creation of a base virtual image from ISO
- ▶ Capture of a running KVM virtual machine (VM)
- ▶ Import of a base virtual image
- ▶ Installation of bundles in a virtual image
- ▶ Capture of an extended virtual image
- ▶ Export of a virtual image

You can deploy the virtual images (virtual appliances) that the Image Construction and Composition Tool creates with the KVM provider into an IBM Flex System x86 compute node that is configured as a KVM host in a PureFlex environment managed by the IBM Flex System Manager. You can also deploy the virtual appliances into an IBM System x node that is managed by IBM Systems Director VMControl.

**Important:** IBM Systems Director and Systems Director VMControl are not required during the virtual appliance build and test processes. The KVM provider builds appliances that are fully compliant with the Systems Director VMControl format. These appliances are built even if an appliance architect does not have access to the Systems Director VMControl Management stack at the appliance construction time. Therefore, the creation and deployment phases can be fully separated.

The KVM provider has two major software components:

- ▶ Open Virtual Appliance (OVA) run time  
This run time is a set of Python scripts that implement the basic lifecycle management of a virtual appliance. It uses the `libvirt` library to communicate with the underlying hypervisor to request its services such as to create a domain, start a domain, and destroy a domain. The OVA Runtime makes its services available through a REST API. These APIs are used by an Image Construction and Composition Tool instance to communicate with the KVM provider.
- ▶ VARepo  
VARepo is a small footprint virtual appliance repository that is used by the OVA Runtime to manage its appliances. It performs such tasks as precloning of a master appliance, releasing the clones, and storage cleanup.

The KVM provider home directory is `/var/lib/va`. Do not change this directory; changing it can cause unpredictable behavior. The KVM home directory includes the following subdirectories:

|              |   |
|--------------|---|
| <b>vafes</b> | Contains the Python scripts that implement the OVA Runtime functions.   |
| <b>ovas</b>  | Is the repository of the virtual appliances in the OVA format. This directory contains the master copies of the virtual appliances that are visible to the KVM provider. These copies are never deployed or changed. The virtual appliance name <i>must</i> have a prefix that uniquely identifies this appliance in your network.  |
| <b>state</b> | Contains a corresponding directory (for each master virtual appliance in the <code>ovas</code> directory) that includes the current state of each clone (workload) created by VARepo for a master virtual appliance. The virtual appliance directory name comes from the prefix of the virtual appliance name, simplifying the pairing of the appliance in the <code>ovas</code> directory with the corresponding directory in the <code>state</code> directory. Besides the clones states, the |

directory contains the `policy.xml` file, which regulates the number of clones to be created and maintained by the VAREpo service.

**workloads**

Contains the clones that were created by VAREpo for the virtual appliances in the **ovas** directory. Each clone is in its own subdirectory. VAREpo generates a Global Unique Identifier (GUID) for each clone, which becomes the name of the clone subdirectory under **workloads**. The clones are used by the KVM provider to provision the VMs for a base appliance. A clone that is provisioned as a VM can then be extended and captured in the Image Construction and Composition Tool. A successful capture creates a master virtual appliance, which is copied into the **ovas** directory.

A high-level overview of the components and their interaction is in 3.2, “Component view” on page 25.

## 6.2 Setting up the KVM provider

To configure and start the KVM provider, you must have the following systems requirements:

- ▶ Supported operating system: Red Hat Enterprise Server 6.1 or later (64-bit)
- ▶ IBM BladeCenter HS22V blade or equivalent IBM System x server
- ▶ Two cores
- ▶ 8 GB RAM
- ▶ 500 GB disk space
- ▶ One network interface card (NIC), 1 Gb, or more

The KVM provider runs on a KVM host, which is a Red Hat Enterprise Linux (RHEL) 6.1 or newer Linux system with the KVM hypervisor and other virtualization tools installed.

For a sample script that automates to a large degree the process of installing and setting up the components required by the KVM provider, see Appendix A, “Sample script for KVM host setup” on page 281. You might need to edit the script to better fit your environment.

**Important:** Make sure that the prerequisites are installed successfully. Specifically, verify that KVM kernel modules, **libvirt**, Perl-XML, and **web.py** installation did not return any error messages.

After the KVM host is set up, test it by creating a sample VM from scratch. For example, you can use the **virt-manager** tool to create a VM image.

### 6.2.1 Installing the KVM provider

To install the KVM provider on a KVM host, use the following steps:

1. Download or copy the `vafes.tar.gz` installation image from your Image Construction and Composition Tool instance. The `vafes.tar.gz` file is in the `/drouter/vafes` directory.
2. Extract the archive:

```
tar -xzf vafes.tar.gz
```
3. Run the installer:

```
cd /root/vafes/vafes
./install.sh
```

Be sure no error messages are displayed.

## 6.2.2 Managing the KVM provider

The provider runs as two Linux services that map directly to the major components previously described:

**vafesd**        Implements OVA runtime functions and is displayed as a REST web service.

**varepod**      Implements the lightweight virtual appliance repository functions.

You can use each service to start or stop the provider as shown in the following examples:

- ▶ Start the provider:  

```
service varepod start
service vafesd start
```
- ▶ Stop the provider:  

```
service varepod stop
service vafesd stop
```

To update the provider, use the following steps:

1. Download and extract the new version of the provider.
2. Stop the provider services.
3. Run the following command from the directory where you extracted the provider:  

```
./updatevafes.sh
```
4. Restart the provider services.

## 6.2.3 Creating the KVM cloud provider

You can now create the KVM provider definition by using the user interface of the Image Construction and Composition Tool. When you open the tool for the first time, the “Create a new cloud provider” window is displayed. If a cloud provider is already configured, this window is not displayed by the tool.

If the “Create a new cloud provider” window does not open automatically, select **Administer** → **Manage cloud providers**, and then configure the KVM provider.

To configure the KVM cloud provider, use the following steps:

1. In the “Create a new cloud provider Welcome” window, click **Next**.
2. In the General window (Figure 6-1 on page 95), complete the following actions:
  - a. In the Name field, enter a name for the cloud provider, such as KVM provider on local.
  - b. Enter a brief description.
  - c. For Cloud Provider Type, select the **KVM Express** provider.
  - d. Click **Next**.

**Create cloud provider**

Welcome > General > Credentials > Cloud details > Summary

**Specify a name and description of the connection**

Name: KVM Provider on hmb30 private network

Description: KVM Provider on hmb30 over the private network using eth2 interface.

Cloud Provider Type: KVM Express

Previous Next Cancel

Figure 6-1 Creating a KVM cloud provider

3. In the Credentials window, enter your user name, password, and location for the KVM provider server. You can enter the host name or the IP address for the KVM provider server location.
4. In the Cloud details window (Figure 6-2 on page 96), enter the network information for the network to build the VMs:
  - a. In the Netmask field, enter the netmask value for the configuration, such as 255.255.255.0.
  - b. In the Gateway address field, enter the gateway address for configuration, such as 192.168.71.38.
  - c. Enter the primary DNS, for example, 192.168.71.38.
  - d. Enter the secondary DNS, for example, 192.168.71.38.
  - e. Provide one or more IP addresses for the Image Construction and Composition Tool to use when it creates VMs.  
 You can enter deployment IP addresses or IP address ranges. Click the **plus sign** (+) to enter the IP address ranges. Add the available IP addresses that can be resolved with a reverse DNS lookup. Press the Enter key after you enter each IP address to save it in the table. If necessary, you can modify this information later.
  - f. Click **Next**.



**Create cloud provider**

Welcome > General > Credentials > Cloud details > Summary

Select or enter the details for deploying and building images.

Netmask:

Gateway Address:

Primary DNS:

Secondary DNS:

Deployment IP Addresses: Enter a number of IP addresses that are valid and available. Press Enter after each IP address to save it to the table

| IP Address Range Begin | IP Address Range End |
|------------------------|----------------------|
| 192.168.71.135         | 192.168.71.139       |

Previous Next Cancel

Figure 6-2 Create cloud provider – Cloud details window

5. In the Summary window, review the details. If you need to make changes, click **Previous**, and make the required changes. When you are satisfied that all of the details are correct, click **Done**.

## 6.3 Creating a base image from ISO

You can create a base operating system image, starting from an installable ISO image of a supported Linux distribution, by using the Image Construction and Composition Tool.

### 6.3.1 Preliminary setup

Before you can successfully build the base OS image, verify the following requirements:

1. Verify that the base operating system is one of the following options:
  - Red Hat Enterprise Linux 5.6 or later
  - Red Hat Enterprise Linux 6.0 or later
  - SUSE Linux Enterprise Server (SLES) 11.1
2. Edit the `/drouter/ramdisk2/mnt/raid-volume/raid0/config/httpurlroot.properties` file of the server of your tool to point to your tool server. For example, if the IP address of the server of your tool is 192.168.71.79, you edit the first line as follows (remember to uncomment):

```
httpurlroot=http://192.168.71.79:80/isos
```

**Important:** The address that you provide must be visible to the VM that is being provisioned. Use the IP for the network interface that will be used to communicate with the KVM provider.

3. Install and setup an HTTP server (Apache) on the server of your tool:

- a. Make sure that the HTTP service is automatically started:

```
chkconfig httpd on
```

Create a virtual host to serve the ISO images to the VMs being built. Modify the `httpd.conf` file by adding the virtual host definition (Example 6-1). We use the `/drouter/isos` directory to host the installation images. Also, verify that the port on which the HTTP server listens matches the port setting in the `httpurlroot` entry as set in Example 6-1.

*Example 6-1 Adding the virtual host definition to the `httpd.conf` file*

---

```
<VirtualHost *:80>
    ServerAdmin jarek@dummy-host.example.com
    DocumentRoot /drouter/isos
    ServerName 192.168.71.79
    ErrorLog logs/192.168.71.79-error_log
    CustomLog logs/192.168.71.79-access_log common

    Alias /isos "/drouter/isos"

    <Directory "/drouter/isos">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

---

- b. Verify that the Apache server can access the `/drouter/isos` directory. Run the following commands:

```
chmod 755 /drouter
chmod 755 /drouter/isos
```

- c. Start the `httpd` server:

```
service httpd start
```

If you receive an error message that indicates that the Apache server was unable to bind to address `0.0.0.0:443`, comment out the following line in the `httpd.conf` file:

```
Include conf.d/*.conf
```

## 6.3.2 Importing an ISO image

The feature to import ISO images into IBM Image Construction and Composition Tool is disabled by default so you must enable it as follows:

1. Open the following Image Construction and Composition Tool file:

```
/opt/IBM/icon/icn.app/config/configuration.config.
```

2. Edit the file by removing or commenting out the line that contains the following text.

```
"manageIsos" : "disable",
```

Example 6-2 shows the result.

*Example 6-2 Enabling the feature to import ISO images*


---

```
"action": {  
  # "personalities": "disable",  
  # "manageIsos": "disable",  
  "patterns": "disable"  
}
```

---

If the Image Construction and Composition Tool is running, you must stop and restart it for this change to take effect. To do so, first run the `/opt/IBM/icon/stop.sh` script and then the `/opt/IBM/icon/start.sh` script.

After you verify the prerequisites and enabled the ISO management, import the ISO image:

1. From the Welcome page of the Image Construction and Composition Tool, select **Images and bundles** → **Manage ISO Resources**.
2. Click the **Import** icon () and define the location of your ISO file.
3. In the Import an ISO window (Figure 6-3 on page 99), complete the following actions:
  - a. For OS Type, select Linux because only Linux is currently supported.
  - b. For OS Distribution, select **RHEL** or **SLES**.
  - c. For Version, enter the version of the distribution.
  - d. For Path, enter the address of the ISO file. This address can be a remote path, over the HTTP and HTTPS protocols, or a local path on the system on which the tool is running.
  - e. For User name, enter a user profile name for the newly created OS image.
  - f. For Password, enter a user password for the newly created OS image.
  - g. Click **OK**.

**Import an ISO**

OS Type:

OS Distribution:

Version:

Path:

*either http, https, or local VM file (for ESX only)*

User name:

Password:

Verify password:

Figure 6-3 Importing an ISO installable image

The import process now begins.

### 6.3.3 Building a base operating system appliance

After the import process completes, you can use the ISO to build your own base operating system appliance:

1. Select **Images and bundles** → **Build and manage images**.
2. On the Images page, click the **Add** icon (+).
3. In the Create a new image window, select **Create Image from an ISO image**, and then click **Proceed**.
4. In the Create New Image window, complete the following information:
  - a. For the Name field, enter a name for the VM that you are creating. Currently, no white spaces are allowed in the Name field.
  - b. For the Universal ID field (required), enter a universal ID for the VM that you are creating.

**Universal ID and Version field values:** The value that you enter for Universal ID must be unique when combined with the value that you enter in the Version field. The Version and Universal ID fields are validated according to OSGi conventions. For the Universal ID field, you can use alphanumeric characters, an underscore, a dash, or a period. A period is not allowed as the first character.

- c. For the Version field, enter a version number for the new VM that you are creating. Use the format <major>.<minor>.<micro>.<qualifier>, where *major*, *minor*, and *micro* are all numeric values and are required. However, *qualifier* is optional and can contain the same type of characters as the universal ID.
  - d. For the Description field, enter a description for the new VM that you are creating.
5. In the next Create New Image window (Figure 6-4 on page 101), select the ISO image that you want to use to build the virtual image. You must first import the ISO as explained in 6.3.2, “Importing an ISO image” on page 98.

In this window, you can modify the default Kickstart or AutoYaST files that are included with the tool:

- When you create the RHEL image, click **Download the default Kickstart file**.
- When you create an SLES image, click **Download the default AutoYaST Configure file**.

**Notes:** When you install from ISO images under a KVM cloud provider, use the default files and modify them as needed. Consider the following information:

- ▶ The `#iconnetwork` and `#iconurl` lines must be present and not changed.
- ▶ SSH must be allowed through the firewall.
- ▶ The password for `rootpw` can be changed.
- ▶ Keyboard and Language values can be modified.
- ▶ Additional packages can be added to the packages section.
- ▶ Can add other statements like create users.
- ▶ Not recommended to modify any of the other lines in the `ks.cfg` file.
- ▶ Recommended to get a minimal installation working before trying additional changes.

The `yast.cfg` file used for SLES has similar requirements:

- ▶ The `<!--iconnetwork-->` must be present and unchanged.
- ▶ The root password must match the password used when the first synchronize is done.

**Default kickstart for RHEL:** The default kickstart is compatible with RHEL 6.x. However, a few changes are required to make it work with RHEL 5.x images:

- a. Remove `--plaintext` from `rootpw` command.
- b. Add the **key** `--skip` command *before* the `%packages` directive.
- c. Add `--ignoremissing` to the `%packages` directive.
- d. Change `@network-tools` to `@net-tools`.
- e. Remove the `%end` line.

The key change in step b and the `%end` change in step e are incompatible with RHEL 6.

After you modify the configuration file, click **Upload** to upload the file to the tool. Click **Next**.

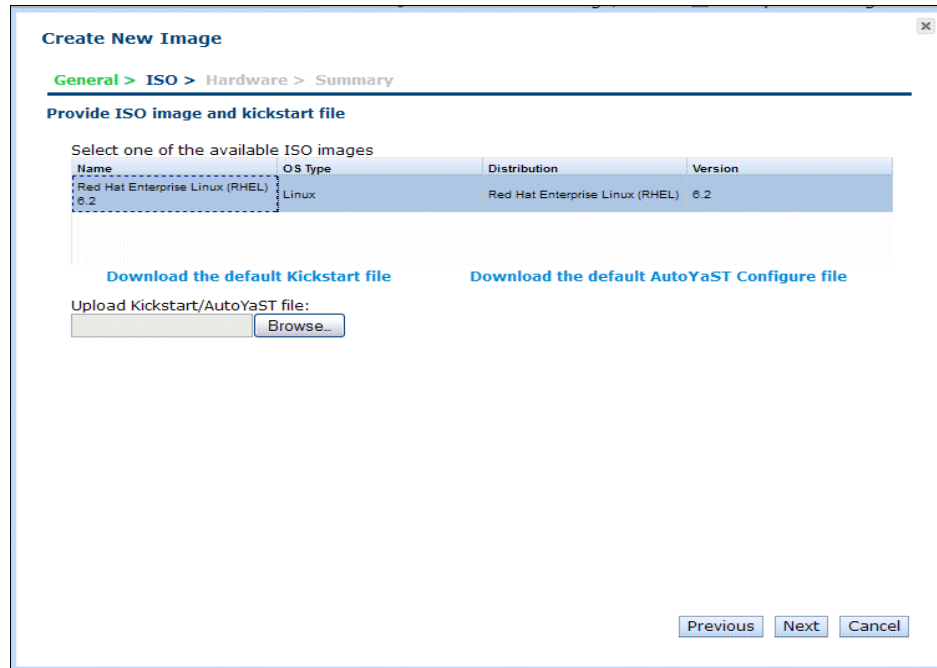



Figure 6-4 Creating base image from ISO

6. In the next window, specify the basic attributes of the virtual image to be created:
  - a. In the Virtual processors field, enter the number of virtual processors for the VM.
  - b. In the Memory field, enter the amount of memory in MB for the VM.
  - c. In the CD/DVD Drives field, specify the number of CD/DVD devices to be attached to the VM.
  - d. In the Virtual hard disk field, enter the attributes for the virtual hard disk drive. Only the Size attribute is editable.

**Disk size:** Make sure that you specify a disk size that is large enough to fit the OS and all other components that you plan to install. A size of 4 GB seems to be a minimum in most cases.

- e. Click **Next**.
7. In the Summary window, review the information, and then click **Done**. A new virtual image is created in the tool. This image is initially out of sync.
8. Click the **Synchronize** icon () , and provide the parameters that are required for the deployment. After the synchronization is completed, click **capture**.

The Image Construction and Composition Tool now creates a VM and uses the ISO and kickstart or autoyast configuration file to install the operating system that is in the ISO.

## 6.4 Importing a running virtual machine

You can import a VM running on a KVM hypervisor into the Image Construction and Composition Tool to package it for deployment through IBM Systems Director VMControl or to extend it further with bundles. The process consists of capturing the VM as a base virtual image and importing it into the virtual appliance repository. The VM to be captured must be deployed on the same KVM host on which you installed the KVM provider.

### 6.4.1 Requirements


The VM that you are importing must meet the following requirements:

- ▶ The base operating system on the VM must be one of the following options:
  - Red Hat Enterprise Linux v5.6 or later
  - Red Hat Enterprise Linux v6.1 or later
  - SUSE Linux Enterprise Server v11.1
- ▶ Network Manager must be disabled or uninstalled.
- ▶ SELINUX must be set to permissive or disabled.
- ▶ SSH must be running and available.
- ▶ The VM can have one or more hard disks.
- ▶ The VM must have only one network interface. Multiple network interfaces are not supported.
- ▶ If present, the **HWADDR** parameter in the corresponding network interface script (typically **ifcfg-eth0**) must be removed.

**Attention:** Do not use a critical or important VM that you cannot afford to disrupt or damage. Depending on the contents of the VM, the Image Construction and Composition Tool might install and reset the VM and change it to a state that you do not want, which can include resetting network settings. After you import a VM into the Image Construction and Composition Tool, do not restart and import the VM again.

### 6.4.2 Importing a running VM from the host system of the KVM provider

To import a running VM from the system that is hosting the KVM provider, use the following steps:

1. From the Image Construction and Composition Tool Welcome page, select **Images and bundles** → **Build images**.
2. Click the plus icon () , and select the **Create image from running VM** option. Then, click **Proceed**.
3. In the Import Images from Running VM window (Figure 6-5 on page 103), complete the following actions for the new VM that you are creating:
  - a. Enter a name. Currently, no white spaces are allowed in the name.
  - b. Enter a universal ID field (required).

**Universal ID and Version field values:** The value that you enter for Universal ID must be unique when combined with the value that you enter in the Version field. The Version and Universal ID fields are validated according to OSGi conventions. For the Universal ID field, you can use alphanumeric characters, an underscore, a dash, or a period. A period is not allowed as the first character.

- c. Enter a version number.
- d. Enter a description.
- e. In the Virtual machine to Capture field, select a running VM that you want to capture.  
The KVM provider queries the KVM hypervisor to retrieve the list of VMs running on that host. The VMs that are associated with the virtual appliances that are managed by the KVM provider are then filtered out. Only the VMs that were not deployed as virtual appliances are listed.
- f. Enter the IP address of the VM.
- g. Enter the user name of the user to connect to the VM.
- h. Enter the password for the user.
- i. Click **Create**.

**Import Images from Running VM**

Enter required fields

Name: RHEL6.2-2G-KVM

Universal ID: com.ibm.itso.rhel6.2.2G.kvm

Version: 1.0.0

Description: Base OS, 2GB image, imported from a running VM.

Virtual machine to capture: rhel6-base-2G

IP Address: 192.168.71.211

User ID: root

Password: .....

Verify Password: .....

Create Cancel

Figure 6-5 Importing an image from a running VM



A new image is displayed in the user interface, which has an *Importing* status. The Image Construction and Composition Tool connects to the VM and creates or edits the required properties. It installs IBM Virtual Solutions Activation Engine (VSAE) 2.1.1 if required and then resets and shuts down the VM. The tool copies the VM disk files into the virtual appliance repository and creates a corresponding OVF file based on the new VM. When this process is finished, the status of the new image changes to *Completed*. Then you can extend the image. Upon completion of the import process, the VM remains shut down. You can restart the VM and use it independently of the tool.

## 6.5 Importing virtual images (appliances) from the KVM cloud provider

You can import existing virtual images (appliances) from the KVM provider. As mentioned previously, the VARepo component manages the virtual images that are visible to the KVM provider. The existing virtual images, such as those images that are created by another Image Construction and Composition Tool instance or that are created manually, can be easily imported into VARepo.

### 6.5.1 Importing virtual images

To import the virtual images from the KVM cloud provider, use the following steps:

1. Make sure that the KVM provider is running.
2. Copy the virtual appliance binary (a tape archive (tar) file with a standard name extension of ova) into the `/var/lib/va/ovas` directory on the KVM provider host.

A GUID is used as a prefix in the appliance name to avoid same-named appliances in the virtual appliance repository. Name the virtual appliances by using the following pattern:

```
GUID_UserFriendlyName.ova
```

As mentioned previously in this chapter, the Global Unique Identifier (GUID) ensures that virtual appliances always have a unique name to avoid name conflicts and so that they can be easily located. A GUID can be generated in many ways. For example, you can run the following command on a KVM host:

```
echo 'import virtinst.util ; print  
virtinst.util.uuidToString(virtinst.util.randomUUID())' | python
```

3. To create the clones for the virtual appliance, run the following command:

```
vaset policy -ova de7e1fa8-6b2b-3f58-a565-4a9d1f4a467f -available 1 -total 3
```

In this command, the **-ova** parameter points to the name of the appliance to be imported. The **available** parameter is used to instruct VARepo the number of appliance clones that should be available for deployment. In addition, the **total** parameter sets the maximum number of clones for an appliance. Depending on the number of clones to be created (in this case, just one), the throughput of the I/O subsystem, and the size of the appliance, this process can take some time to complete.

You can check the progress of the cloning process by inspecting the content of the `/var/lib/va/state/de7e1fa8-6b2b-3f58-a565-4a9d1f4a467f` directory. The name of the last directory in the path matches the prefix in the name of the virtual appliance. After the cloning process completes, the content of the corresponding state directory is similar to the following example:

```
-rw-r--r-- 1 vareporun varepo    0 Dec 17 10:31  
BCA5F4D7-28D6-4ABD-8133-870872D49FFC.free
```

The file extension of `.free` indicates that a workload (appliance clone) is available for deployment. Running the **vaset** command shown in step 3 on page 104 resulted in a clone that was created by the VARepo service. The bootable image for a clone is in the corresponding workload subdirectory, as in Example 6-3.

*Example 6-3 Workload directory of the bootable image for a clone*

```
ls -la /var/lib/va/workloads/BCA5F4D7-28D6-4ABD-8133-870872D49FFC
total 28
drwxr-xr-x  3 vareporun varepo  4096 Dec 17 10:31 .
drwxrwxr-x 27 root          varepo  4096 Dec 17 12:02 ..
-rw-r--r--  1 vareporun varepo 12476 Nov 29 07:55
de7e1fa8-6b2b-3f58-a565-4a9d1f4a467f_SmallTomcatVMC24Comp.ovf
drwxr-xr-x  2 vareporun varepo  4096 Nov 29 07:55 vs0
```

In Example 6-3, the bootable raw image can be found in the **vs0** directory (Example 6-4).


*Example 6-4 Bootable raw image in the vs0 directory*

```
ls -la vs0
total 955412
drwxr-xr-x  2 vareporun varepo      4096 Nov 29 07:55 .
drwxr-xr-x  3 vareporun varepo      4096 Dec 17 10:31 ..
-rw-r--r--  1 vareporun varepo      2013 Jun 24 12:34 al.xml
-rw-r--r--  1 vareporun varepo        709 Jun 24 12:34 configUnitDescriptor.xml
-rw-r--r--  1 vareporun varepo 978321408 Nov  3 14:16 disk0.raw
```

**Important:** Before you import the newly created OVA into Image Construction and Composition Tool, wait until the cloning process finishes so that at least one appliance is in the corresponding state subdirectory with a status of *free*.

## 6.5.2 Importing a virtual appliance

After a clone is in the workloads directory and has a *free* status, import the virtual appliance into the instance of your tool:

1. Select the KVM provider (list in the upper-right corner).
2. From the main menu, select **Images and bundles** → **Build Images**.
3. Click the **Import from the Cloud Provider** icon (.
4. Select the image that you want to import from the list of images shown in the left pane of the Images page. You can search for images by name or keyword.

**Important:** All virtual appliances in the `/var/lib/va/ovas` directory in the KVM provider are listed, except the virtual appliances that are already listed in the Image Construction and Composition Tool under Images.

5. Select the images that you want to add, and click **Add**. Provide a user ID and password to use to access the appliance. The actual image content does not transfer to the tool; only a copy of the metadata is saved to the tool.
6. Click **Import**. You can safely ignore any warning messages that are displayed for all “base” appliance images that are imported and cloned in the tool.

## 6.6 Removing an image (appliance) from the virtual appliance repository

Occasionally, you might need to free the disk space that is occupied by the virtual appliances and the corresponding workloads that are no longer needed.

**Important:** The `ovas` directory contains the `GUID_isotemplate.ova` appliances, which *cannot* be deleted. If they are deleted, the Create from ISO function will not work.

To free disk space, use the following steps:

1. Log in to the KVM provider host and navigate to the `/var/lib/va/vafes` directory.
2. To delete unnecessary appliances, run the following command:  

```
python ovaDelete.py
```
3. Select the virtual appliance that you want to delete from the list that is displayed (Figure 6-6).

```
[root@hmb30 vafes]# python ovaDelete.py
OVA Selection List:
0 - 5a8eb52a-2174-4771-9e8c-27c34de095a3_rhel6ext.ova
1 - 0a16c0e0-9827-4140-b330-fdd4c30cb791_visionext6.ova
2 - 9cbe3643-67d1-4cfb-88e7-ca24aaf23a9b_DavKidnap.ova
3 - 7da211d7-0186-44ae-9a18-3dc1cdc1c166_TomcatBuiltLocal.ova
4 - ef8f20b9-7c3c-4069-b676-5bae205b5780_SmallTomcat.ova
5 - 24283a7f-4b0c-4e8b-a8d6-1d4041b3b66b_TomcatExtVMCComp.ova
6 - ff9a31ca-8d4d-517a-c787-6cbf316c6891_SmallTomcatWithVSAE.ova
7 - Exit
Select ova (7):
```

Figure 6-6 List of virtual appliances to delete

In this example, we select option 6, which results in deleting the virtual appliances with all associated workloads. Figure 6-7 shows the results of this action.

```
>6
You Selected ff9a31ca-8d4d-517a-c787-6cbf316c6891_SmallTomcatWithVSAE.ova
Deleting /var/lib/va/workloads/FB6D32D2-955A-4FAB-8C38-855E9BB290D1/vs0...
Deleting /var/lib/va/workloads/7C46B4C6-08AC-49C4-83FB-DF0CA5124184/vs0...
Deleting /var/lib/va/workloads/F9125A9C-F0DB-4ABA-8C1F-B315D1AC9B4E/vs0...
```

Figure 6-7 Results of selecting option 6

In this example, a virtual appliance and three corresponding workloads were deleted from the provider system.

## 6.7 Working with bundles

The concept of software bundles and the way that they can be created and maintained in the Image Construction and Composition Tool is not specific to the KVM provider. For information about how to create and manage bundles in the tool, see *Tips and Best Practices for Software Bundle Creation for the IBM Image Construction and Composition Tool version 1.1.1*:

[https://www.ibm.com/developerworks/mydeveloperworks/blogs/9e696bfa-94af-4f5a-ab50-c955cca76fd0/entry/icct\\_software\\_bundle\\_creation\\_whitepaper6?lang=en](https://www.ibm.com/developerworks/mydeveloperworks/blogs/9e696bfa-94af-4f5a-ab50-c955cca76fd0/entry/icct_software_bundle_creation_whitepaper6?lang=en)

This book addresses the topics that are not covered in this referenced document.

### 6.7.1 Passing parameters to the reset operation

Typically, a bundle has both the configuration and the reset steps defined. The configuration step maps to the activation operation that is handled by the VSAE, which is installed in your image by the Image Construction and Composition Tool. Similarly, the reset step maps to the reset operation that is handled by the activation engine.

In some cases, the reset operation requires the same set of input parameters that were passed to the activate operation. You can achieve this behavior of your bundle by using one of the following two approaches:

- Create a bundle in tool by using the Product Activator Development Kit (PADK) as explained in Chapter 5, “Product Activator Development Kit” on page 73.

Currently by using PADK, you can create just one activate (Configuration) operation and one reset operation per bundle. Furthermore, the PADK automatically generates the activation logic files that are required by the activation engine with the same class name for both the activate and reset operations. The resulting `master.al` file that drives the behavior of the activation engine has just one (merged) product activation section for both operations. This merged section, in turn, ensures that the parameters passed to the activation program at the virtual appliance deployment time are also passed to the reset program if the reset operation is started.

- When you use the GUI for the Image Construction and Composition Tool to create a bundle, be sure that the bundle has only one configuration operation (Figure 6-8) and that the operation name matches the bundle name. Similar to the other approach, the names must match so that the activate and reset operations can be merged into a single resulting activation logic file.

**was-hostname-config** Draft

General Requirements Install **Configuration** Firewall Reset

**Deploy-time configuration**  
Define how to configure the bundle in a new instance of a virtual machine

**Configuration Operations** +

was-hostname-config

Operation name: was-hostname-config

Service name: was-hostname-config

**Files to Copy**

Files to be copied to the target machine: +

| Source (URI or file name) | Executable                          |  |  |  |
|---------------------------|-------------------------------------|--|--|--|
| reset.py                  | <input type="checkbox"/>            |  |  |  |
| esutil.py                 | <input type="checkbox"/>            |  |  |  |
| activate.py               | <input checked="" type="checkbox"/> |  |  |  |

**Command**

Run Command: activate.py Hide Preview  
Select an executable script to run

```
activate.py --was_hostname ${Hostname} --was_nodename ${WAS instance node}
```

Run As: root

Parameter Style: Long Space Style

**Arguments:** +

| Name         | Label             | Value             | Hide Value               |   |   |   |
|--------------|-------------------|-------------------|--------------------------|---|---|---|
| was_hostname | Hostname          |                   | <input type="checkbox"/> | ↑ | ↓ | × |
| was_nodename | WAS instance node | DefaultAppSrvNode | <input type="checkbox"/> | ↑ | ↓ | × |

Figure 6-8 Configuring software bundle

In Figure 6-8, the Configuration tab has only one operation, and the operation name matches the bundle name. As a result, the corresponding activation logic files have the same class name for both operations, and the resulting master .a1 file has only one product activation section for this bundle.

Consider the excerpt in Example 6-5 from the `master.al` file.

*Example 6-5 Excerpt from the master.al file*

---

```
<ProductActivation class="com.ibm.vsaе.was-hostname-config">
  <Properties>
    <Property key="was_hostname"/>
    <Property key="was_nodename"/>
  </Properties>
  <Program cmdLine="true" cmdOptionStyle="longSpace" envVarPrefix="AE_"
envVars="false"
href="/opt/ibm/ae/AS/com.ibm.vsaе.was-hostname-config_1.0.0.0/activation/com.ibm.vsaе.was-hostname-config/activate.py"/>
  <ResetProgram
href="/opt/ibm/ae/AS/com.ibm.vsaе.was-hostname-config_1.0.0.0/reset/com.ibm.vsaе.was-hostname-config_1.0.0.0_resetOperation_1/reset.py"/>
    <Service name="com.ibm.vsaе.was-hostname-config">
      <RunLevel value="3"/>
      <RunLevel value="5"/>
    </Service>
  </ProductActivation>
```

---

In the example, only one Product Activation section is shown. This section instructs the activation engine to pass the two parameters to the reset program when the reset action is started.

The necessary activation logic files are automatically generated by the tool. Typically you do not need to manually edit them. If want to validate their content, you can find the files in the `/opt/ibm/ae/AL` directory in the VM image.

## 6.8 Validating virtual images by using the OVA Runtime

The final step in the virtual appliance construction process is to use the Export function in the Image Construction and Composition Tool to export the appliance. By using this function, the appliance can be exported and deployed to the target platform, which is an infrastructure managed by IBM Flex System Manager or IBM Systems Director VMControl.

To validate the images, deploy the final version of a virtual appliance to the target platform, which in this case is to the Systems Director VMControl managed infrastructure. However, in some cases, the target platform might not be readily available at the virtual appliance construction facility. Under such circumstances, the virtual appliance validation tests can be performed directly on the KVM provider. The successful completion of such validation tests gives the virtual appliance architect a high level of confidence that the appliance will successfully deploy to Systems Director VMControl.

The KVM provider is delivered with a small, self-contained virtual appliance run time, which displays its services through a set of Python programs. You can use these programs to test the newly created appliance.

## 6.8.1 Deploying a virtual image (appliance) with OVA Runtime

To deploy a virtual image (appliance) that is created in the Image Construction and Composition Tool through the services that are displayed by the KVM provider, use the following steps:

1. After the virtual appliance construction process is finalized, capture the appliance, and then export it:
  - a. In the Image Construction and Composition Tool, select **Images and bundles** → **Build Images**.
  - b. From the list of available images (appliances), select the image you want to validate.
  - c. From the image details that are displayed in the right pane, click the **Export** icon in the upper-right corner.
  - d. When you see the Export image as OVA message box (Figure 6-9), review the message and click **OK**.

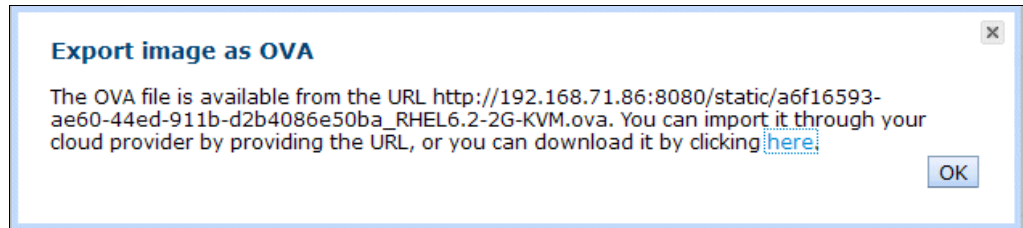


Figure 6-9 Exporting an image as an OVA

2. Copy the fully qualified name for your appliance. In this example, the name is as follows:  
a6f16593-ae60-44ed-911b-d2b4086e50ba\_RHEL6.2-2G-KVM.ova
3. On the KVM provider system, go to the `/var/lib/va/vafes` directory, and enter the following command:  

```
python ovaRun.py
```
4. From the list of all the appliances that are managed by the KVM provider, including the newly created appliance (Figure 6-10), select the option that corresponds to the appliance that you want to deploy. In this case, we select option 0.

```
OVA Selection List:
0 - a6f16593-ae60-44ed-911b-d2b4086e50ba_RHEL6.2-2G-KVM.ova
1 - 81713dcf-3644-427f-b435-a228a388d5ee_RHEL6.2-4G-KVM.ova
2 - Exit
Select ova (2):
```

Figure 6-10 List of appliances managed by KVM provider

The OVA Runtime locates one of the available workloads (clones) that were automatically created by the VARepo service. You are now presented with a series of prompts. The OVA Runtime parses the OVF that is associated with the appliance and shows the key hardware abstraction layer (HAL) and Product Section attributes.

5. Accept the default values, or provide values that correspond to your environment. In Figure 6-11 on page 111, the Hardware Section and the Product Sections that allow for user configurable parameters are highlighted in a bold font. This figure also shows that a workload was allocated and a new VM was created and started. The parameters are

passed to the activation engine installed in the image by using the ISO transport. The VM was successfully started and is now running under the IP address you requested.

**Hardware Section:**

Enter New Value for Number Of CPUs (2):

>1

Enter New Value for Memory in MBytes (2048):

>4096

**Product (Software) Section com.ibm.vsaes.2\_1.system-user:**

Enter New Value for User name (root):

>

Enter New Value for User password (s3cur3!):

>Passw0rd4u

Enter New Value for Gecos (Super User (root@localhost)):

>

**Product (Software) Section com.ibm.vsaes.2\_1.network-interface:**

Enter New Value for Use DHCP (false):

>

Enter New Value for IP address (192.168.254.10):

>192.168.71.127

Enter New Value for IP netmask (255.255.255.0):

>

Enter New Value for IP gateway (192.168.254.1):

>192.168.71.38

Enter New Value for Is default route (true):

>

**Product (Software) Section com.ibm.vsaes.2\_1.system-host:**

Enter New Value for Hostname (myhost):

>vs127

Enter New Value for Domain Name (mydomain.com):

>ovafactory.rochester.ibm.com

**Product (Software) Section com.ibm.vsaes.2\_1.dns-client:**

Enter New Value for Primary DNS server (192.168.1.100):

>192.168.71.38

Enter New Value for Secondary DNS server (192.168.1.200):

>192.168.71.38

Enter New Value for Domain search list (ibm.com):

>ovafactory.rochester.ibm.com

**Product (Software) Section com.ibm.vsaes.2\_1.ntp-client:**

Enter New Value for Ntp server (0.pool.ntp.org):

>

/var/lib/va/workloads/14FCE7FF-D6C3-48D0-8370-1D93C03D8AAB/14FCE7FF-D6C3-48D0-8370-1D93C03D8AAB\_domain\_vs0.xml

Creating ISO Transport

Defining Domain 14FCE7FF-D6C3-48D0-8370-1D93C03D8AAB\_vs0:

Starting Domain: 14FCE7FF-D6C3-48D0-8370-1D93C03D8AAB\_vs0

Domain 14FCE7FF-D6C3-48D0-8370-1D93C03D8AAB\_vs0:

Domain ID: 46

VNC Port :5907

State: Running

Figure 6-11 Deploying an appliance by using the command-line ovaRun.py utility



6. Use `ssh` or `vnc` to connect to the running VM to validate that the activation was successful.
7. Run the necessary validation tests.

## 6.8.2 Stopping and releasing a running VM (workload)

After you complete the validation tests, you can clean up the environment by stopping the workload and releasing it. By releasing the workload, you instruct the OVA runtime to remove the files that are associated with it, including the disk images.

To stop and release a workload, use the following steps:

- ▶ To stop a running workload for an appliance that you previously deployed, use these steps:
  - a. Enter the following command:

```
python ovaStop.py
```
  - b. In the list that is displayed, select the name of the appliance that you previously deployed. In this case, we select the following file:

```
a6f16593-ae60-44ed-911b-d2b4086e50ba_RHEL6.2-2G-KVM.ova
```
  - c. From the list of workloads that are running for the selected appliance, select the workload to stop.
- ▶ To release a workload after it stops, release the resources associated with it:

```
python ovaRelease.py
```

The workload is released, and the resources are freed.

## 6.9 Tips to override default behavior of the KVM provider

If you need to override the default behavior of the KVM provider, you can use the tips and techniques as explained in this section.

### 6.9.1 Modifying the template OVF file

Creating production-ready appliances is an iterative process. Therefore, you must go through the build process several times before you are satisfied with the results. The final step is to validate the virtual appliance. The KVM provider ships with a template OVF file that gets packaged with the virtual image each time that a virtual appliance is created. This template OVF has generic values for such items as the DNS IP addresses, domain name, and root password. The OVF template, named `generic.ovf`, is in the `/var/lib/va/vafes/templates` directory on the KVM provider system.

You can modify this file to adjust the values to your current environment so that you do not need to change these values every time you deploy an appliance. Before you modify the file, you *must* make a copy of the original template file. The XML parser is sensitive to typographical errors or syntax error, and you can end up with an unusable KVM provider installation.

## 6.9.2 Changing the default number of clones

The number of clones that are created by the VARepo service is governed by the settings in the `policy.xml` file in the `/var/lib/va/state/default` directory on the KVM host provider host.

Example 6-6 shows the default values.

*Example 6-6 Default values in the policy.xml file*

---

```
<!-- define the clone factory strategy -->
  <clone>
    <!-- define how many free clones should be available -->
    <available>10</available>
    <!-- define how many clones should be max available -->
    <total>20</total>
  </clone>
```

---

The number in the `available` element specifies the number of clones that are available for deployment. The `total` element defines the total number of clones in all possible states. The default numbers are appropriate for most multi-user environments. However, if the KVM provider is used by a single Image Construction and Composition Tool instance, these values are too high and consequently use up a lot of disk space. In a single user environment, a setting of 3 for `available` and 5 for `total` seems to be more appropriate.

Each virtual appliance has a local copy of the policy file, which is in the corresponding virtual appliance state directory.

You must restart the `varepod` service for the changes to take effect.

## 6.10 Troubleshooting

To diagnose and eliminate issues that you might encounter when using the Image Construction and Composition Tool, follow the techniques provided in this section.

### 6.10.1 Checking the Image Construction and Composition Tool and the KVM provider versions

When you report issues to IBM technical support, include the version information for the Image Construction and Composition Tool and the KVM provider. To look up the required information, two options are available:

- For the Image Construction and Composition Tool, select **About** from the menu in the upper-right corner. A message box is opened that shows the version information.
- For the KVM provider, run the following command on the system that hosts the provider:  

```
cat /var/lib/va/vafes/Version
```

## 6.10.2 Image Construction and Composition Tool logs

Check the host system log files of the Image Construction and Composition Tool to determine whether any problems occurred. To access log files of the tool, use the following steps:

1. In the Welcome window, click the **Download logs** link.
2. Save the compressed file.
3. Examine the log files. The compressed file contains the following server log files:
  - The `trace.log` file is the most recent trace file.
  - The `error.log` file is the most recent error file.

You can also access the log files by selecting **Administer** → **Download logs**.

Alternatively, you can check the logs directly on the system that hosts the tool. The log files are in the `/drouter/ramdisk2/mnt/raid-volume/raid0/logs` directory.

## 6.10.3 KVM provider logs

Each of the two KVM provider components creates logs that you can use to analyze and fix issues.

Two OVA Runtime logs are available in the `/var/log/cfes` directory:

|                    |  |
|--------------------|--|
| <b>cfes.log</b>    | Records the actions that are performed by the OVA Runtime. Such actions include deploying virtual appliance, creating a domain, starting a domain, capturing a VM, and creating a VM from ISO. |
| <b>console.log</b> | Records the flow of the REST API between the Image Construction and Composition Tool and the KVM provider.   |

The VAREpo component has two logs in the `/var/log/varepo` directory:

|                    |   |
|--------------------|---|
| <b>varepod.log</b> | Records the errors that are encountered by the virtual appliance repository when managing virtual appliances, workloads (clones), and workloads state (status).   |
| <b>vaset.log</b>   | Provides the history of the OVA cloning process. It records the current settings for the cloning such as the number of clones that need to be available and the total maximum number of clones for a virtual appliance. |

## 6.10.4 Validating that the KVM provider is active

Many issues can be eliminated by ensuring that your tool instance can communicate with the KVM provider. First make sure that the necessary services are running on the system that hosts the KVM provider (Example 6-7).

*Example 6-7 Services running on the system*

---

```
# service varepod status
Checking for service varepo varepod (pid 23913) is running...
# service vafesd status
VA Factory Express Server is running as [23966].
```

---

As mentioned earlier, the Image Construction and Composition Tool interacts with the KVM provider by using the REST APIs. Because information about the supported APIs goes beyond the scope of this book, use Poster, which is a Firefox plug-in, to verify that the REST web services of the KVM provider are operational and accessible by the tool.

To verify that the REST web services are operational and accessible, use these steps:

1. Download and install the Poster Firefox add-on.
2. Shut down and then restart Firefox.
3. From the Firefox main menu, select **Tools** → **Poster** to start Poster.
4. Enter in the REST URL in the following format, as shown in Figure 6-12, so that it points to the system that hosts the KVM provider, and then click **GET**:

`http://server.domain:8080/cfes/rest/virtualAppliances`

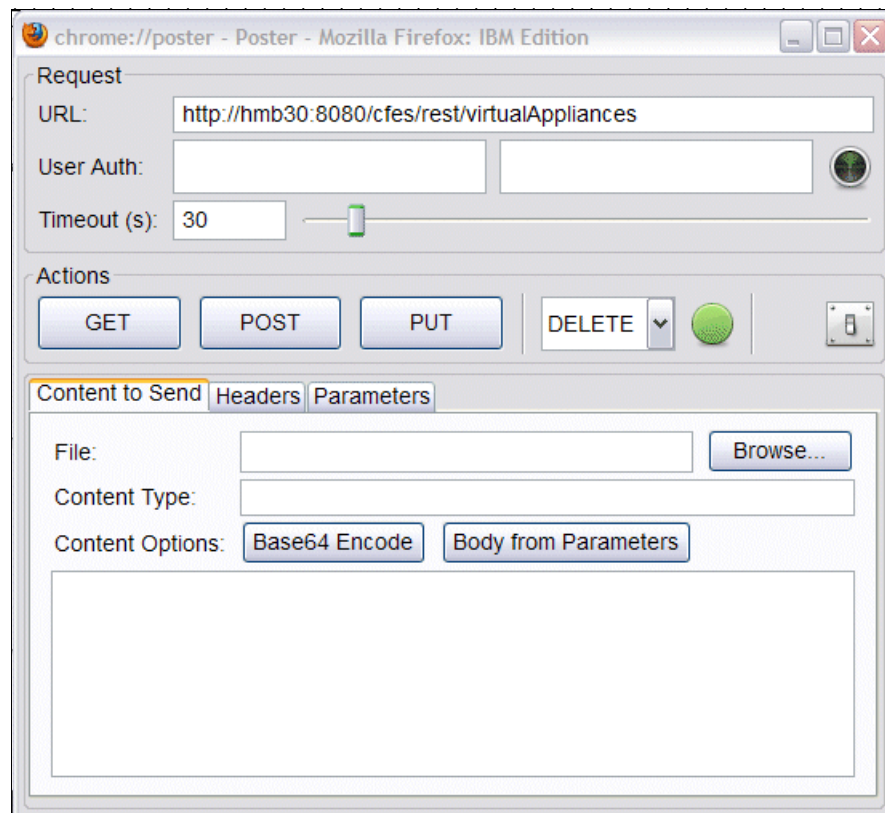


Figure 6-12 Test the REST services of the KVM provider by using Poster

Poster sends the request to the KVM provider. The response (Figure 6-13) contains the list of all virtual appliances that are currently managed by the provider.

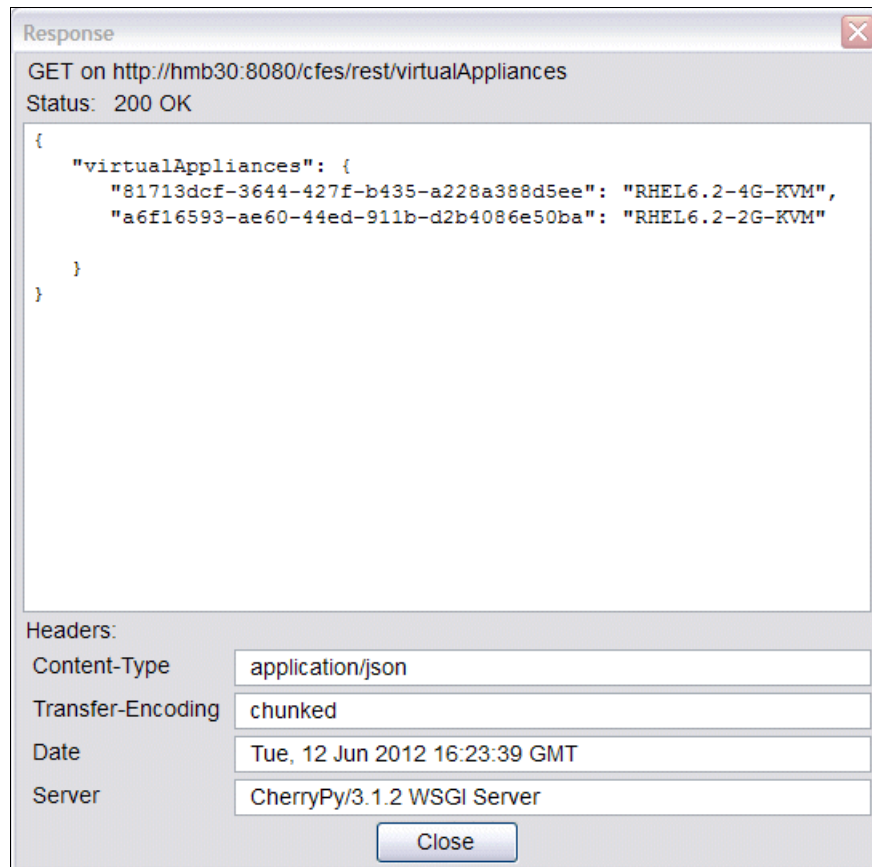


Figure 6-13 Validating the KVM provider response in Poster

### 6.10.5 Considerations for restarting the KVM provider services

Sometimes you might need to restart the KVM provider services. To ensure that the VARepo service is not in the middle of the clone creation, run the Linux **top** command, and watch for the **tar** processes that are running under the varepo user ID. Wait until these processes complete before restarting the provider. Otherwise, you might end up with several “zombie” clones that need to be cleaned up manually.

### 6.10.6 Resolving common issues

This section provides guidance for how to resolve the most common issues. The symptom, cause, and a solution are described for each issue.

#### Importing of a running VM function fails

First, be sure the REST web server on the KVM provider system does not return errors.

##### **Symptom**

Your virtual image capture fails. The console.log file contains the following error message:

```
"HTTP/1.1 POST /cfes/rest/workloads" - 500 Internal Server Error
```

### **Cause**

The REST web server on the KVM system that hosts the provider encountered an unrecoverable issue.

### **Solution**

Restart the web server by entering the following command:

```
service vafesd restart
```

## **Synchronization of a previously imported image fails**

If present, the **HWADDR** parameter in the corresponding network interface script must be removed. Otherwise, the captured virtual appliance is not usable. Specifically, you cannot extend and synchronize a new appliance that is based on the original one.

### **Symptom**

The extended virtual image does not synchronize.

### **Cause**

The **HWADDR** parameter in the corresponding network interface script (typically **ifcfg-eth0**) was not removed before the base virtual image was captured.

### **Solution**

Recapture the base virtual image from a running VM. Make sure that the **HWADDR** parameter is not present in the network interface script on the VM to be captured.

## **Synchronization fails after an image was extended with a bundle**

If errors are encountered when installing the bundle artifacts, the synchronization fails. The failure occurs if the installation scripts were created on a Windows workstation and then uploaded into a bundle.

### **Symptom**

The extended virtual image synchronization fails with a message similar to the following example:

```
Synchronization Failed
Unable to initiate add of execution bundle to VM
```

### **Solution**

To diagnose and debug errors that are found when bundle artifacts are installed in a target VM, use the following steps:

1. Find the IP address of the VM that the Image Construction and Composition Tool provisioned for the synchronization process:
  - a. In the tool UI, select **Images and bundles** → **Build and manage images**.
  - b. Select the image, for which the synchronization failed.

- c. In the right-side pane, click **Virtual System** to expand the section (Figure 6-14). Record the IP address or the host name.

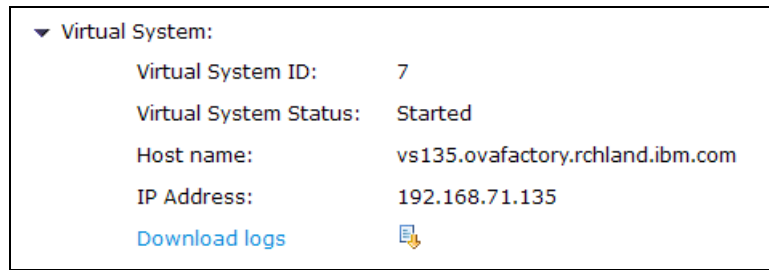


Figure 6-14 Retrieving the virtual system information

2. Download the Image Construction and Composition Tool logs and analyze the error messages:
  - a. In the Virtual System section, click the **Download logs** link.
  - b. Extract the archive on your workstation.
  - c. Open the error log that is retrieved from the target VM image, for example, the following log:

```
C:\temp\iconlogs\drouter\ramdisk2\mnt\raid-volume\raid0\local\images\2ae1f2d
d-9a6e-46c0-a6cd-945fea53d3e1\logs\error.log
```

You might see an error message, such as the following example, which means that the installation script, **install-whs.sh** in this case, did not start:

```
/bin/sh: ./install-whs.sh: /bin/sh^M: bad interpreter: No such file or
directory
```

3. Enter **ssh** to connect to the target VM by using the information retrieved in step 1 on page 117.
  - a. Go to the temporary directory where the tool copied the artifact to be installed in the VM. Look for a directory in the format `/tmp/{id}`, where *id* is a number generated by the tool, for example `/tmp/6`.
  - b. List the content of this directory (Example 6-8).

Example 6-8 Contents of the `/tmp/{id}` directory

---

```
-rw-r--r-- 1 root root 27289 Jun 22 20:41 IconImageSynchronizer.sh
drwxr-xr-x 6 root root 4096 Jun 22 20:41
com.ibm.icon.kvm.enablement.sles11_1.0.0.6
-rw-r--r-- 1 root root 797 Jun 22 20:41
com.ibm.itso.was-hostname-config-vs0.a1
drwxr-xr-x 6 root root 4096 Jun 22 20:42
com.ibm.itso.was-hostname-config_1.0.0.0
-rw-r--r-- 1 root root 434 Jun 22 20:41
com.ibm.itso.was-hostname-config_1.0.0.0_resetOperation_1-vs0.a1
-rw-r--r-- 1 root root 958 Jun 22 20:41 ovf-env.xml
-rw-r--r-- 1 root root 2577 Jun 22 20:41 resetFix.py
-rw-r--r-- 1 root root 463 Jun 22 20:41 rule_template.xml
```

---

You can now identify a directory that contains your bundle scripts, which in this case is `com.ibm.itso.was-hostname-config_1.0.0.0`. The **install-whs.sh** is in the `com.ibm.itso.was-hostname-config_1.0.0.0/execution` subdirectory.

4. To debug the issue, go to the execution subdirectory (previous step), and run the command that failed:

```
./install-whs.sh
```

You now see the error message.

5. Open the script in an editor and do the following steps:

- a. Fix the errors.
- b. Save the script.
- c. Rerun the script to make sure that it runs successfully.

In this case, the problem is caused by the unnecessary special characters that were added by an editor based on Windows.

6. Update your bundle with the corrected version of the script.
7. In the Image Construction and Composition Tool, remove the image for which the synchronization failed.
8. Extend the base image with the fixed bundle. Now the synchronization should be successful.

## Synchronization of an extended image fails

Be sure that the workload directory for the image that is being extended has at least one workload in the free state.

### Symptom

The synchronization fails with the following error message:

Virtual machine failed to start

### Cause

In the synchronization process, the Image Construction and Composition Tool needs to boot a clone of the image that is being extended. Therefore, at a minimum, the workload directory for the base image must contain at least one workload that is available for deployment. If no workload is found, the KVM provider returns an error, and the synchronization fails.

### Solution

To fix this problem, use the following steps:

1. Check the error log entries for both the Image Construction and Composition Tool and the KVM provider. The error log of the tool might have the following error message:

```
[2012-07-11 11:44:01:524 CDT] 00000468 KVMDeployThre I
com.ibm.cloud.icn.cloudprovider.kvm.KVMDeployThread run Failed deploying new vm
com.ibm.cloud.icn.cloudprovider.exceptions.CloudProviderImplFailedException:
CYOKV0010E: Cannot create workload: 500 [text/html]: Internal Error Creating
Workload : Consult Server Administrator
```

The corresponding error message in the cfes.log file on the KVM provider system contains a similar message:

```
2012-07-11 11:44:01 POST:CfesException: Failed to deploy the OVA
ac726af4-c4fc-4bfa-8b7a-66743792e0ce
```

2. Using the appliance GUID from the error message in step 1, navigate to the workload state directory, which is the /var/lib/va/state/ac726af4-c4fc-4bfa-8b7a-66743792e0ce directory in this case.

Check the workload status:

```
ls -la
```



Example 6-9 shows the output that you might see. This output validates that no workloads are available for deployment.

*Example 6-9 Output of the workload status command*

---

```
-rw-r--r-- 1 vareporun varepo 0 Jul 11 11:30
3EC90C9A-F0D6-4A7C-859B-F2E6925B9284.inUse
-rw-r--r-- 1 vareporun varepo 0 Jul 11 11:30
52F53148-91BC-491B-8F97-FACE971C00EC.inUse
-rw-r--r-- 1 vareporun varepo 0 Jul 11 11:30
6F3D36F8-0A6D-4291-8981-FD534C06DA92.inUse
-rw-r--r-- 1 vareporun varepo 0 Jul 11 11:30
72EEA811-B82A-4C0F-888C-3608DEBF76C2.inUse
-rw-r--r-- 1 vareporun varepo 0 Jul 11 11:30
75739554-DB29-41BA-8304-E87DB3134FAF.inUse
-rw-rw-r-- 1 vareporun varepo 559 Jul 11 11:35 policy.xml
```

---

3. Check the content of the `policy.xml` file in this directory. Verify that the total number of clones for an appliance was reached. Example 6-10 shows the relevant excerpt from `policy.xml` file.

*Example 6-10 Excerpt from the policy.xml file*

---

```
<clone>
    <!-- define how many free clones should be available -->
    <available>2</available>
    <!-- define how many clones should be max available -->
    <total>5</total>
</clone>
```

---

4. To increase the number of the available workloads, edit the `policy.xml` file, and set the element `total` to a larger number, for example, 10. Save the file.  
Watch the content of the state directory. After a while, the `varepod` service creates two (as per policy) free workloads. Be patient. Depending on the number of free workloads to create and the size of the appliance image, this process can take several minutes.
5. If no free workloads are displayed after a significant amount of time, try to force the cloning process by running the following command:  

```
vaset policy -ova ac726af4-c4fc-4bfa-8b7a-66743792e0ce --force
```

Where the `-ova` parameter is used to specify the GUID of the appliance.  
Again, watch the content of the corresponding state subdirectory to make sure that the new clones are created.
6. Return to the GUI of the Image Construction and Composition Tool, and try the image extension and synchronization process again.

## When deploying, reconfiguration of a component fails

Be sure that you understand the concepts that are described in 2.4, “IBM Virtual Solutions Activation Engine” on page 16, before you try to resolve the activation issues.

### Symptom

The appliance is deployed, but one or more of the software components in the stack were not properly reconfigured.

### Cause

The activation program for the component did not successfully complete, because incorrect parameters were passed.

### Solution

This problem is encountered when the activation programs were *not* created by using the PADK plug-in. PADK makes it easy to test and debug the activation programs before they are integrated in a bundle and deployed. However, you might have configuration scripts that you created outside of the PADK that you want to reuse.

To diagnose and fix problems with custom activation programs, use the following steps:

1. Check the results of the most recent activation:
  - a. Log in to the deployed VM.
  - b. Navigate to the `/opt/ibm/ae` VSAE home directory.
  - c. Examine the contents of the AR subdirectory (Example 6-11).

*Example 6-11 Contents of the directory*

---

```
-rw-r--r-- 1 root root 61 Jun 21 19:22 ConfigIcon.err
-rw-r--r-- 1 root root 1174 Jun 21 19:22 ConfigIcon.out
-rw-r--r-- 1 root root 0 Jun 21 19:22 com.ibm.vsaе.2_1.dns-client.err
-rw-r--r-- 1 root root 209 Jun 21 19:22 com.ibm.vsaе.2_1.dns-client.out
-rw-r--r-- 1 root root 186 Jun 21 19:22
com.ibm.vsaе.2_1.network-interface.err
-rw-r--r-- 1 root root 0 Jun 21 19:22
com.ibm.vsaе.2_1.network-interface.out
-rw-r--r-- 1 root root 0 Jun 21 19:22 com.ibm.vsaе.2_1.ntp-client.err
-rw-r--r-- 1 root root 35 Jun 21 19:22 com.ibm.vsaе.2_1.ntp-client.out
-rw-r--r-- 1 root root 0 Jun 21 19:22 com.ibm.vsaе.2_1.system-host.err
-rw-r--r-- 1 root root 53 Jun 21 19:22 com.ibm.vsaе.2_1.system-host.out
-rw-r--r-- 1 root root 70 Jun 21 19:22 com.ibm.vsaе.2_1.system-user.err
-rw-r--r-- 1 root root 189 Jun 21 19:22 com.ibm.vsaе.2_1.system-user.out
-rw-r--r-- 1 root root 0 Jun 21 19:22 jarek.tomcat-sample-hostname.err
-rw-r--r-- 1 root root 21 Jun 21 19:22 jarek.tomcat-sample-hostname.out
-rw-r--r-- 1 root root 2124 Jun 21 19:22 ovf-env.ar
-rw-r--r-- 1 root root 0 Jun 21 19:22 ovf-env.done
```

---

The `ovf-env.done` marker file indicates that the activation was successful. If activation issues exist, this marker file is missing. Each activation program (mapped into an activation step) has a standard output and error file associated with it, such as the following example files:

- `jarek.tomcat-sample-hostname.err`
- `jarek.tomcat-sample-hostname.out`

2. Validate the content of the `ovf-env.ar` file. Each activation program has a section that contains the status of the activation and the name of the associated log files (Example 6-12).

*Example 6-12 Status of activation and associated log files in the activation program*

---

```
<ProductActivation class="jarek.tomcat-sample-hostname">
    <Execution kind="self" status="Successful"/>
    <Log
file="/opt/ibm/ae/AR/jarek.tomcat-sample-hostname.out"/>
    <Log
file="/opt/ibm/ae/AR/jarek.tomcat-sample-hostname.err"/>
    <Properties/>
</ProductActivation>
```

---

See if any activation programs have a status other than *successful*. If any program has this status, consult the corresponding output and error logs, and fix the issues reported there.

3. If necessary, make the required changes in the bundle definition. Extend and capture the new appliance version.

In some cases, a custom activation program might not properly handle runtime errors. Consequently, the status for the activation program is successful, but yet the software component was not reconfigured. In such situations, manually debug the offending activation program:

1. Test that the activation program performs the required action. You can start the program directly from the shell. For example, run the following command from the AE home directory (`/opt/ibm/ae`):

```
AS/jarek.tomcat-sample-hostname_1.0.0.0/activate.py --sample_home_dir
/srv/tomcat6/webapps/sample --filename index.html --hostname vs140
```

Use the correct parameter style. In this case, the long-space parameter style is used (double dash in front of the parameter name and a space between parameter and its value).

2. Validate the results of the activation. Fix the issues if necessary.

Finally, simulate the deployment by using the VSAE in the interactive mode:

1. Remove the VSAE services that were previously scheduled by running the following command:

```
./AE.sh -r AL/master.al
```

2. By using the corresponding activation logic XML file in the AL subdirectory, configure the execution of the activation program that is being tested:

```
./AE.sh -a AL/jarek.tomcat-sample-hostname-vs0.al
```

Make sure that the activation program is correctly set up as the OS services. Also make sure that the invocation sequence is correct. In a Linux VM, you can run the following command:

```
ls -la /etc/rc.d/rc5.d
```

3. Create a dummy the `ovf-env.xml` file in the `/tmp/ovf` directory on the VM. Add the three parameters that are required by your activation program. See Example 6-13.

*Example 6-13 Dummy ovf-env.xml file*

---

```
<?xml version="1.0" encoding="UTF-8"?>
<Environment xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://schemas.dmtf.org/ovf/environment/1"
  xmlns:ovfenv="http://schemas.dmtf.org/ovf/environment/1"
  xsi:schemaLocation="http://schemas.dmtf.org/ovf/environment/1
  ../ovf-environment.xsd"
  ovfenv:id="vs0">
  <PlatformSection>
    <Locale>en_US</Locale>
  </PlatformSection>
  <PropertySection>
    <Property
ovfenv:key="jarek.tomcat-sample-hostname.sample_home_dir"
ovfenv:value="/srv/tomcat6/webapps/sample"/>
    <Property ovfenv:key="jarek.tomcat-sample-hostname.filename"
ovfenv:value="index.html"/>
    <Property ovfenv:key="jarek.tomcat-sample-hostname.hostname"
ovfenv:value="vs140"/>
    </PropertySection>
</Environment>
```

---

**The `ovfenv:id` attribute:** The `ovfenv:id` attribute in the `Environment` element must match the `id` attribute in the `VirtualSystem` element in the corresponding activation logic file (the `jarek.tomcat-sample-hostname-vs0.al` file in this case).

4. Run the following command to test your activation program in the AE interactive mode by using the following command:

```
./AE.sh -i /tmp/ovf/ovf-env.xml
```

5. Verify that the program performed the desired action.

If required, re-create the appliance with the corrected version of the activation program. This process might require updating a bundle that contains this activation program and reapplying the bundle to the base image.





## PowerVM Express cloud provider

This chapter explains how to configure and use the IBM PowerVM Express cloud provider (referred to as *PowerVM provider* in this book) with the IBM Image Construction and Composition Tool (referred to as *the tool* in this book). The PowerVM provider supports the entire development cycle of a virtual appliance that targets deployments to the IBM PureFlex System and IBM Power Systems that are managed by IBM Systems Director and Systems Director VMControl system software.

This chapter includes the following sections:

- ▶ Overview of the PowerVM provider
- ▶ Requirements for PowerVM provider
- ▶ Configuring a PowerVM provider
- ▶ Creating a virtual appliance
- ▶ Troubleshooting

## 7.1 Overview of the PowerVM provider

The PowerVM provider implements the functionality that is required by the Image Construction and Composition Tool to create AIX, Linux, and IBM i virtual appliances. The PowerVM provider calls IBM Systems Director VMControl to accomplish the following tasks:

- ▶ Import a base virtual image.
- ▶ Extend a base image, and add software bundles.
- ▶ Install bundles on a virtual image.
- ▶ Capture an extended virtual image.
- ▶ Export of a virtual appliance.


The virtual images (virtual appliances) that are created by the Image Construction and Composition Tool with the PowerVM provider can be deployed to IBM PureFlex System managed by the IBM Flex System Manager. They can also be deployed on IBM Power Systems that are managed by IBM Systems Director and Systems Director VMControl system software.

**Important:** IBM Systems Director with Systems Director VMControl or IBM Flex System Manager is required during the virtual appliance build and test processes.

## 7.2 Requirements for PowerVM provider

To configure a PowerVM provider in the Image Construction and Composition Tool, you must set up and configure a build environment that consists of the PowerVM provider. For information about the requirements to set up this environment, see 4.1, “Installing the Image Construction and Composition Tool” on page 46.

## 7.3 Configuring a PowerVM provider

This section explains how to configure a PowerVM provider in the Image Construction and Composition Tool. When the Image Construction and Composition Tool is started for the first time, it starts the “Create a new cloud provider” wizard to create a cloud provider. This wizard starts automatically every time you start the Image Construction and Composition Tool until you create a cloud provider. If you need to create additional cloud providers, you can click **Administer** → **Manage cloud providers**. Then click the **New cloud provider** icon (.

To configure a PowerVM provider, use the following steps:

1. In the Welcome window of the wizard (Figure 7-1), click **Next**.

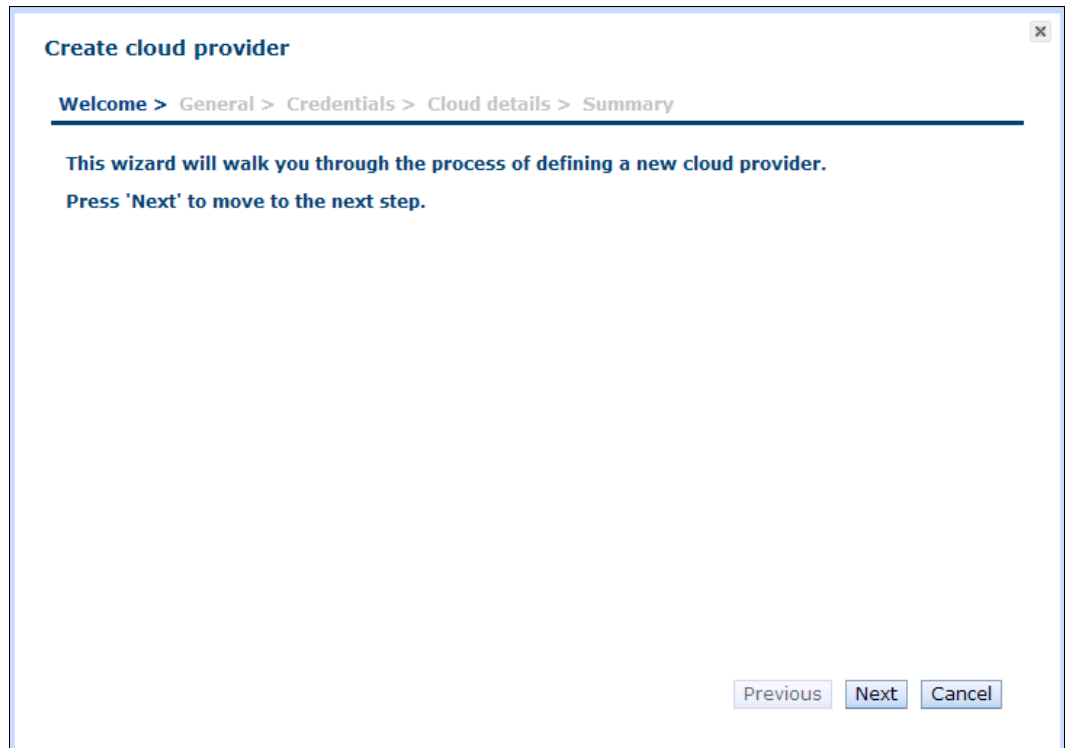


Figure 7-1 Create cloud provider Welcome window



2. In the General window (Figure 7-2), do the following steps:
  - a. Enter a unique name for the cloud provider. In this example, we enter PowerVM Provider on helium55.
  - b. Optional: Enter a brief description.
  - c. For Cloud Provider Type, select **PowerVM Express**.
  - d. Click **Next**.

**Create cloud provider**

[Welcome](#) > [General](#) > [Credentials](#) > [Cloud details](#) > [Summary](#)

---

**Specify a name and description of the connection**

Name:

Description:

Cloud Provider Type:

Figure 7-2 Create cloud provider General window

3. In the Credentials window (Figure 7-3), enter the user name, password, and host name for the PowerVM provider management server, which is either the IBM Systems Director VMControl management server or IBM Flex System Manager. You can enter the host name or the IP address for the location of the PowerVM provider management server. Click **Next**.

The screenshot shows a window titled "Create cloud provider" with a close button in the top right corner. Below the title is a breadcrumb navigation bar: "Welcome > General > Credentials > Cloud details > Summary". The "Credentials" tab is currently selected. Below the breadcrumb is a sub-header: "Enter the credentials and the host name of an IBM System Director system running VMC 2.4". There are three input fields: "User Name:" with the value "root", "Password:" with masked characters (dots), and "Host Name:" with the value "helium55". At the bottom right, there are three buttons: "Previous", "Next", and "Cancel".

Figure 7-3 Create cloud provider Credentials window

4. In the Cloud details window (Figure 7-4), enter the cloud provider target environment and network information. In addition, provide one or more IP addresses for the Image Construction and Composition Tool to use when it creates virtual machines (VMs):
  - a. For Target Hypervisor, select the Systems Management Server or Virtual I/O Server (VIOS) as defined by the Hardware Management Console (HMC), Integrated Virtualization Manager (IVM), or IBM Flex System Manager to use for your virtual appliance build environment. From a Systems Director VMControl perspective, this location is where you want to deploy the virtual appliances or the deployment target.
  - b. Enter the domain name for your build network (for example, `itso.ibm.com`).
  - c. Enter the netmask value for the configuration (for example, `255.255.255.0`).
  - d. Enter the gateway address for the configuration (for example, `10.5.169.1`).
  - e. Enter the primary DNS (for example, `10.10.244.100`).
  - f. Enter the secondary DNS (for example, `10.10.244.200`).
  - g. You can enter deployment IP addresses or IP address ranges. Click the **green plus** icon (+), and then enter the IP address ranges. Ensure that you add available IP addresses that can be resolved with a reverse DNS lookup. Also, press Enter after you enter each IP address to save it in the table.
  - h. Click **Next**.

**Create cloud provider**

Welcome > General > Credentials > **Cloud details** > Summary

Select or enter the details for deploying and building images.

Target Hypervisor:

Domain Name:

Netmask:

Gateway Address:

Primary DNS:

Secondary DNS:

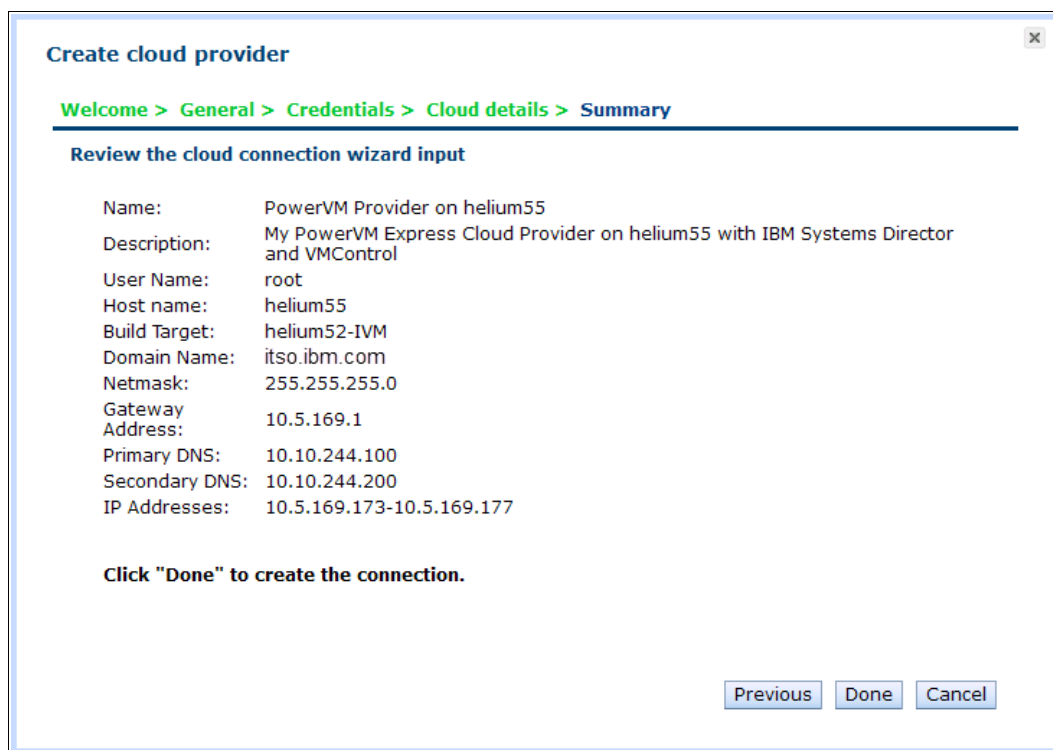
Deployment IP Addresses: Enter a number of IP addresses that are valid and available. Press Enter after each IP address to save it to the table

| IP Address Range Begin | IP Address Range End |
|------------------------|----------------------|
| 10.5.169.173           | 10.5.169.177         |

Previous Next Cancel

Figure 7-4 Create cloud provider Cloud details window

5. Review the details in the Summary window (Figure 7-5). To make changes, click **Previous**, and make the required changes. When you are satisfied that all of the details are correct, click **Done**.



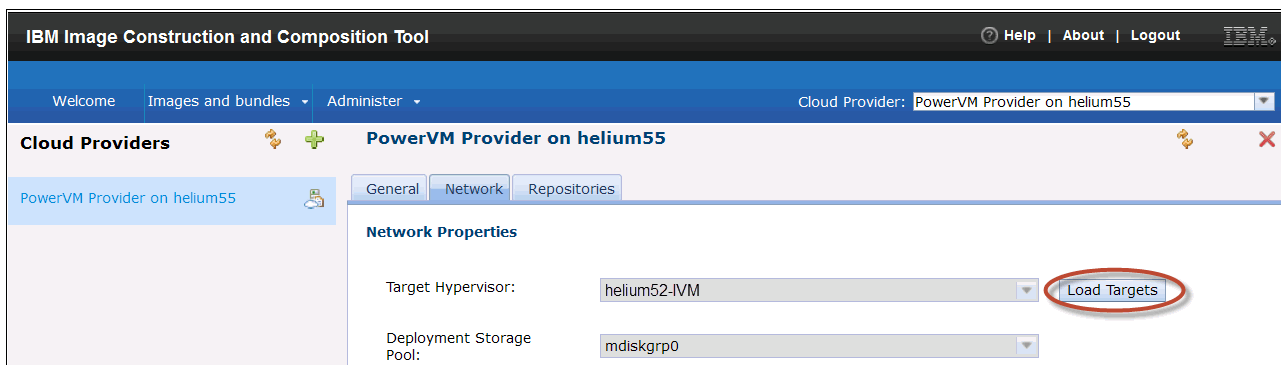
The image shows a 'Create cloud provider' window with a breadcrumb trail: Welcome > General > Credentials > Cloud details > Summary. The title is 'Create cloud provider'. Below the breadcrumb is a section 'Review the cloud connection wizard input' containing a list of configuration details:

|                  |   |
|------------------|---|
| Name:            | PowerVM Provider on helium55  |
| Description:     | My PowerVM Express Cloud Provider on helium55 with IBM Systems Director and VMControl |
| User Name:       | root  |
| Host name:       | helium55  |
| Build Target:    | helium52-IVM  |
| Domain Name:     | its0.ibm.com  |
| Netmask:         | 255.255.255.0   |
| Gateway Address: | 10.5.169.1  |
| Primary DNS:     | 10.10.244.100   |
| Secondary DNS:   | 10.10.244.200   |
| IP Addresses:    | 10.5.169.173-10.5.169.177   |

Below the details, it says 'Click "Done" to create the connection.' At the bottom right are three buttons: 'Previous', 'Done', and 'Cancel'.

Figure 7-5 Create cloud provider Summary window

6. After you configure PowerVM Express as your cloud provider, configure the default Deployment Storage Pool:
  - a. Select **Administer** → **Manage cloud providers**.
  - b. In the left pane, select the **PowerVM Provider on helium55** cloud provider that you just created.
  - c. In the right pane, on the Network tab (Figure 7-6), click **Load Targets**.



The image shows the 'IBM Image Construction and Composition Tool' interface. The top bar has 'Help', 'About', and 'Logout' links. Below the bar, there's a navigation menu with 'Welcome', 'Images and bundles', and 'Administer'. The 'Cloud Provider' dropdown is set to 'PowerVM Provider on helium55'. The left pane shows 'PowerVM Provider on helium55' selected. The right pane has tabs for 'General', 'Network', and 'Repositories'. The 'Network' tab is active, showing 'Network Properties' with two dropdowns: 'Target Hypervisor' (set to 'helium52-IVM') and 'Deployment Storage Pool' (set to 'mdiskgrp0'). A 'Load Targets' button is circled in red next to the 'Target Hypervisor' dropdown.

Figure 7-6 Cloud provider network: Load Targets

- d. After the pane refreshes, select the appropriate deployment storage pool (Figure 7-7) from the **deployment storage pool** drop-down list.

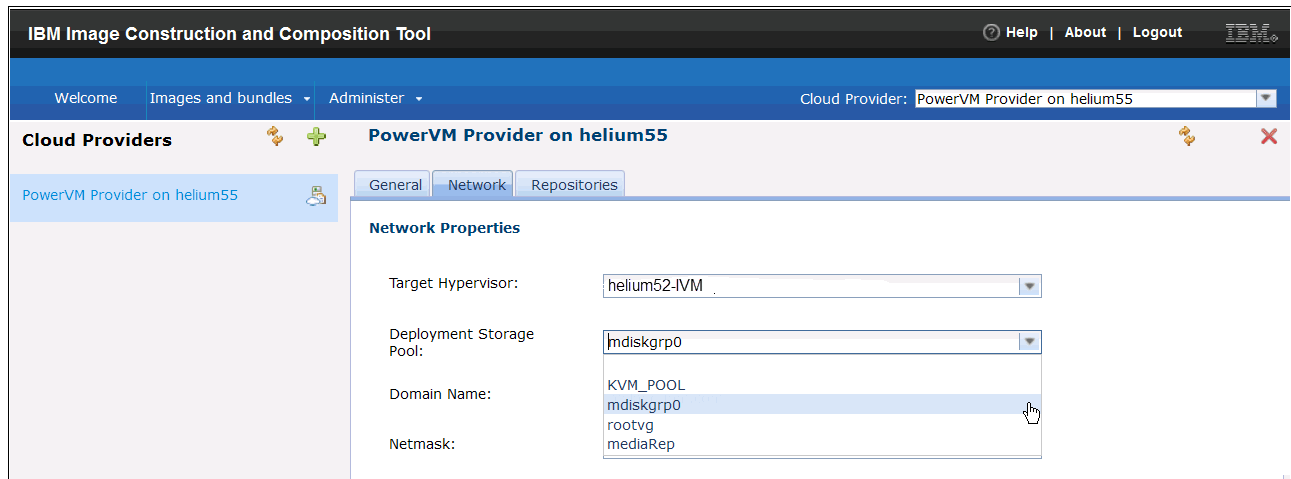


Figure 7-7 Cloud provider deployment storage pool

- e. Click the **Save** (floppy disk icon) icon.
7. After setting the Deployment Storage pool, configure the Cloud Provider Repository Credentials to enable the Export as OVA function:
  - a. In the right pane, on the Repositories tab (Figure 7-8), click the **Edit Credentials** icon (pencil icon).

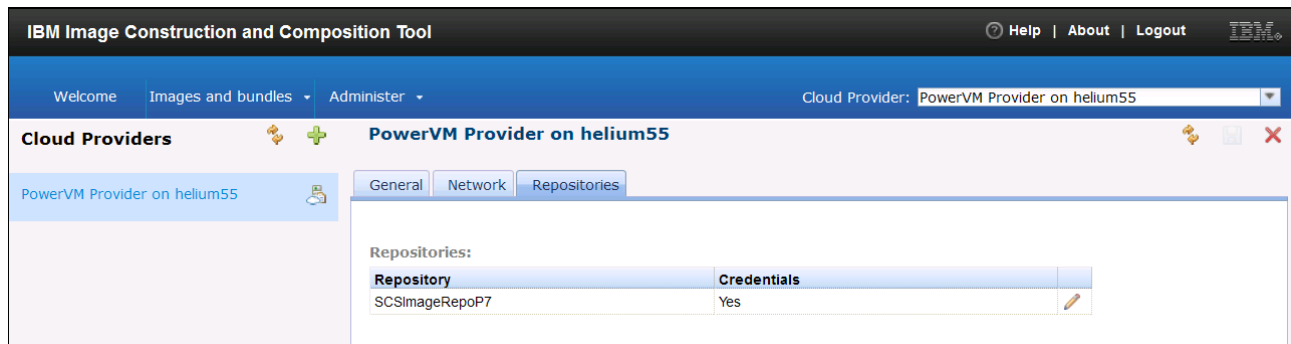
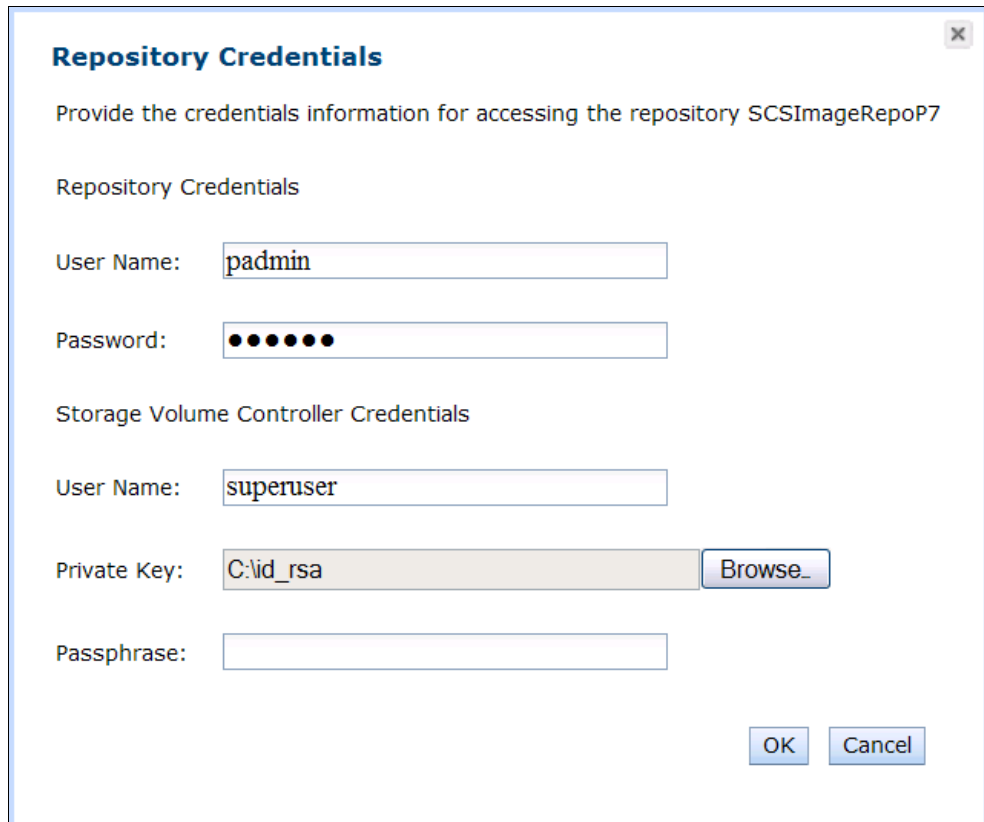


Figure 7-8 Cloud provider repositories

**Repository Credentials window:** This window can differ depending on your environment setup for Network Installation Management (NIM) or Systems Director VMControl Storage Copy Services (SCS). The following step highlights the actions first for an SCS setup and then for a NIM setup.

- b. In the Repository Credentials window (Figure 7-9 on page 133), for an SCS repository, complete these steps:
  - i. Under Repository Credentials, enter user name and password of the Systems Management Server that is acting as the Storage Copy Services image repository. In this example, we use padmin in both fields.
  - ii. Under Storage Volume Controller Credentials, enter a user name for the System Volume Controller or IBM Storwize V7000. In this example, we enter superuser.

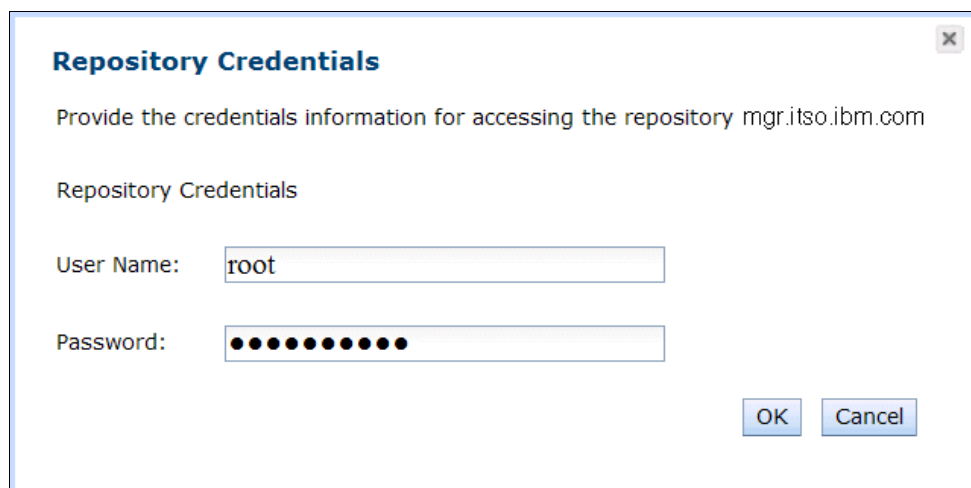
- iii. Enter a private ssh key for the IBM Systems Director or IBM Flex System Manager management server. In this example, we enter `id_rsa`. For information about obtaining the private key, see “Configuring the management software stack” on page 62.
- iv. Click **OK**.



The screenshot shows a window titled "Repository Credentials" with a close button in the top right corner. Below the title bar, it says "Provide the credentials information for accessing the repository SCSImageRepoP7". Under the heading "Repository Credentials", there are two input fields: "User Name:" with the value "padmin" and "Password:" with masked characters. Below this, under the heading "Storage Volume Controller Credentials", there are three input fields: "User Name:" with the value "superuser", "Private Key:" with the value "C:\id\_rsa" and a "Browse..." button to its right, and "Passphrase:" which is empty. At the bottom right, there are "OK" and "Cancel" buttons.

Figure 7-9 SCS Repository Credentials window

For a NIM repository, in Repository Credentials window (Figure 7-10), enter the user name and password of the Management Server that is acting as the NIM server. Then click **OK**.



The screenshot shows a window titled "Repository Credentials" with a close button in the top right corner. Below the title bar, it says "Provide the credentials information for accessing the repository mgr.itso.ibm.com". Under the heading "Repository Credentials", there are two input fields: "User Name:" with the value "root" and "Password:" with masked characters. At the bottom right, there are "OK" and "Cancel" buttons.

Figure 7-10 NIM Repository Credentials window

## 7.4 Creating a virtual appliance

This section explains how to create a virtual appliance and includes the following sections:

- ▶ Creating a base image
- ▶ Importing a base image
- ▶ Extending, synchronizing, and capturing an image
- ▶ Exporting an image as an OVA archive

Figure 7-11 illustrates the end-to-end process of creating a virtual appliance.

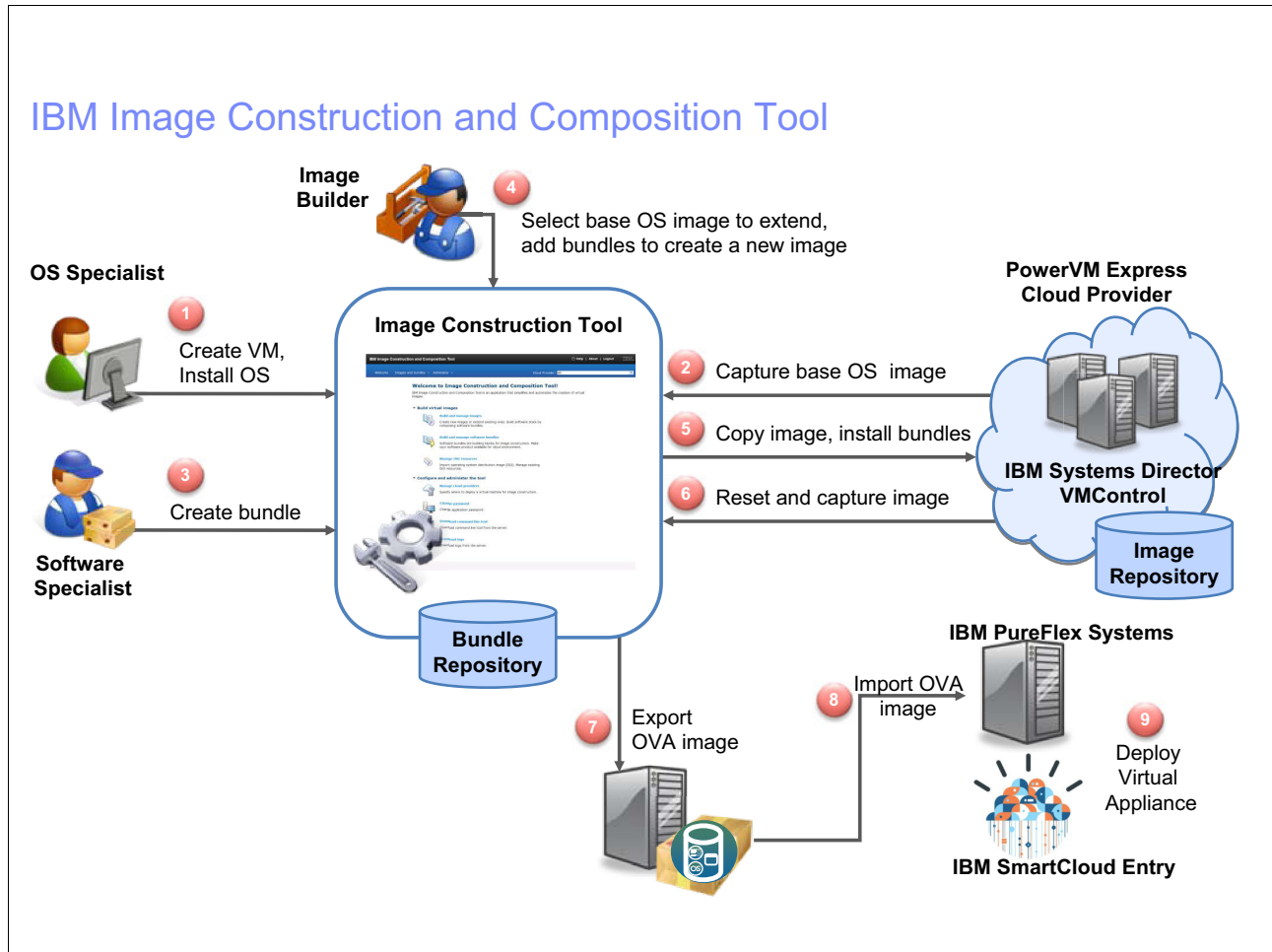


Figure 7-11 End-to-end creation process for a virtual appliance

### 7.4.1 Creating a base image

To create a virtual appliance, you must first create a base operating system image. The base operating system image must meet the requirements for successfully importing it into the Image Construction and Composition Tool. For more information about the base image requirements, see "Software" on page 54.

### ***AIX base image***

To create an AIX base image, use the following steps:

1. From the IVM, HMC, or Flex System Manager interface, create an AIX LPAR:
  - a. Map the appropriate disk and networking to the partition.
  - b. When prompted to select Virtual Optical Media, select the installation media for AIX.
2. Activate the partition to begin the AIX installation.
3. After the installation is complete, run through the Installation assistant, and set the root password and networking.
4. Complete the following post-installation steps:
  - a. Change the paging size.
  - b. Install OpenSSH and OpenSSL.
  - c. Install the GNU **unzip** and **tar** utilities.
  - d. Change the file system size.
  - e. Remove the virtual optical media.

#### **Important:**

- Make sure that there is sufficient space in the /opt filesystem. The operations assets of the IBM Virtual Solutions Activation Engine (VSAE) and Software Bundles configuration use this file system. Also, increase the /tmp file system because it is used to stage assets for the software bundle installation operations.
- As of AIX 7.1 TL1 and AIX6.1 TL6, the bos.ae file set is installed by default with AIX. The ae file is an older version of the VSAE and can cause issues with importing from a running virtual machine. Remove this file set, and remove the /opt/ibm/ae directory and its contents from the base image.

### ***Linux base image***

To create a Linux base image, use the following steps:

1. From the IVM, HMC, or Flex System Manager interface, create a Linux LPAR:
  - a. Map the appropriate disk and networking to the partition.
  - b. When prompted to select Virtual Optical Media, select the installation media for Linux.
2. Activate the partition to begin the Linux installation and follow the installation wizard.
3. After the installation is complete, you must reboot the server.
4. Complete the following post-installation steps:
  - a. Set SELinux to Disabled/Permissive.
  - b. Set networking.
  - c. Install **openssh-clients (RHEL)**.
  - d. Install Service and Productivity Tools RPMs. See service and productivity tools for IBM PowerLinux™ servers:

<http://www14.software.ibm.com/webapp/set2/sas/f/lopdiags/home.html>

**Note:** IBM makes available the IBM Installation Toolkit for PowerLinux. If you performed the install of Linux using this toolkit, the Service and Productivity Tools RPMs are already present. If installing the Toolkit RPMs during post-installation, you might be required to satisfy RPM dependencies located on the Linux distribution installation media.



- e. Remove the virtual optical media.
- f. Install the Common Agent.

### **IBM i base image**

To create an IBM i base image, use the following steps:

1. From the IVM, HMC, or Flex System Manager interface, create an IBM i LPAR:
  - a. Map the appropriate disk and networking to the partition.
  - b. When prompted to select Virtual Optical Media, select the installation media for IBM i.
2. Activate the partition to begin the IBM i installation.
3. Proceed with normal IBM i install process:
  - a. Set the QSECOFR password.
  - b. Install the platform agent product 5770UME (B\_GROUPx\_04).
  - c. Install SSH server components 5733SC1 \*BASE and option 1 (B\_GROUPx\_05).
  - d. Configure networking.
  - e. Install cumulative fix packages.
4. Complete the following post-installation steps:
  - a. Use the **UPDDTA FILE(QUSRSYS/QATOCSTART)** command and verify that the following TCP/IP servers are set to auto-start:
    - \*SSHD
    - \*CIMON
    - \*SLP

Use page down to locate the server and make the necessary changes.

- b. Verify that CIM is running:
  - i. Enter the following commands:
 

```
QSH
cd /qibm/proddata/os/slp/bin
./slp_query --type=service:* --address=127.0.0.1
```
  - ii. Verify the following output:
 

```
URL: service:directory-agent://<system IP>
URL: service:wbem:https://<system IP>:5989
URL: service:management-software.IBM:platform-agent://<system IP>
```
  - iii. If you do not see all three entries, exit from QSH and enter the following commands:
 

```
ENDTCPSVR *CIMOM
STRTCPSVR *CIMOM
```
- c. Remove the virtual media from the partition.

**Important:** To capture an IBM i OS resource manually using IBM System Director VMControl, you also need to do the following steps:

1. Order and install the program temporary fix SI48604 (in cum 3037).
2. Enable the activation engine by running the following command:
 

```
CALL PGM(QSYS/QAENGCHG) PARM(*ENABLE)
```

## 7.4.2 Importing a base image

You can choose from various methods to import a base operating system image into the Image Construction and Composition Tool. For a PowerVM provider, the following methods are available to import AIX, Linux and IBM i base images:

- ▶ Importing from a running virtual machine
- ▶ Importing from a cloud provider

### Importing from a running virtual machine

Before you import a virtual image into the Image Construction and Composition Tool from a running VM, see which VMs are available to import from IBM Systems Director:

1. Open the IBM Systems Director management console.
2. In the left pane (Figure 7-12), expand **Inventory** → **Views**, and select **Virtual Servers and Hosts** to see all the virtual servers and hosts (right pane).

| Select                   | Name         | State   | Access  | Problems    | Compliance | OS Name                  | OS Type an... |
|--------------------------|--------------|---------|---------|-------------|------------|--------------------------|---------------|
| <input type="checkbox"/> | helium51-IVM | Started | OK      | Minor       | OK         |                          |               |
| <input type="checkbox"/> | helium51     | Started | OK      | Minor       | OK         | helium51.rchland.ibm.com | VIOS 2.2.1.3  |
| <input type="checkbox"/> | helium57     | Stopped | OK      | OK          | OK         |                          |               |
| <input type="checkbox"/> | helium60     | Stopped | Offline | Information | OK         | helium60.rchland.ibm.com | AIX 7.1       |
| <input type="checkbox"/> | helium65     | Started | OK      | OK          | OK         |                          |               |
| <input type="checkbox"/> | helium68     | Started | OK      | OK          | OK         |                          |               |
| <input type="checkbox"/> | helium52-IVM | Started | OK      | Minor       | OK         |                          |               |
| <input type="checkbox"/> | helium52     | Started | OK      | Minor       | OK         | helium52.rchland.ibm.com | VIOS 2.2.1.3  |
| <input type="checkbox"/> | helium80     | Started | OK      | Minor       | OK         |                          |               |
| <input type="checkbox"/> | helium84     | Stopped | OK      | OK          | OK         |                          |               |
| <input type="checkbox"/> | helium85     | Started | OK      | OK          | OK         |                          |               |
| <input type="checkbox"/> | helium53-IVM | Started | Offline | Minor       | OK         |                          |               |
| <input type="checkbox"/> | helium53     | Unknown | OK      | OK          | OK         | helium53.rchland.ibm.com | VIOS 2.2.1.3  |
| <input type="checkbox"/> | helium54-IVM | Started | OK      | OK          | OK         |                          |               |
| <input type="checkbox"/> | helium54     | Started | OK      | Minor       | OK         | helium54.rchland.ibm.com | VIOS 2.2.1.3  |

Figure 7-12 IBM Systems Director VMControl Virtual Servers and Hosts pane

When presenting the list of VMs to capture, the Image Construction and Composition Tool filters the complete list of virtual servers from IBM Systems Director based on this criteria:

- ▶ Virtual servers that are in a *Started* state
- ▶ Virtual servers that are *not* part of a workload
- ▶ Virtual servers that have AIX, Linux, or IBM i as their base operating system

To import images from running VM, use the following steps:

1. From the Image Construction and Composition Tool Welcome window, click **Build and manage images**.
2. Click the **plus** icon (+) to create an image.

3. In the Import Images from Running VM window (Figure 7-13), complete the following actions for the virtual image that you are creating:
  - a. Enter a name.
  - b. Enter a universal ID.
  - c. Enter a version number.
  - d. Enter a brief description.
  - e. For Virtual machine to capture, select the VM to capture.
  - f. For Repository for Capture, select in which repository to place the imported image.
  - g. Enter the IP address of the VM to import.
  - h. Enter the user ID to log in to the VM that you are importing.
  - i. Enter the password for the user ID of the VM that you are importing.
  - j. Enter the password again for the user ID of the VM you are importing.
  - k. Click **Create**.

**Import Images from Running VM**

Enter required fields

Name:

Universal ID:

Version:

Description:

Virtual machine to capture:

Repository for Capture:

IP Address:

User ID:

Password:

Verify Password:

Figure 7-13 Import Images from Running VM window

During the import process, the Image Construction and Composition Tool injects the VSAE into the image automatically to make it available to the software bundle configuration scripts.

## Importing from a cloud provider

You can import previously captured virtual images into the Image Construction and Composition Tool from the PowerVM provider.

**Important:** Before you capture images with Systems Director VMControl:

1. Use the **reset\_diragent\_keys** command to reset the identification keys that are related to the IBM Systems Director Common Agent on AIX.
2. Manually install the IBM Virtual Solutions Activation Engine and reset the image as explained in the “Capture support and requirements in an SCS-based Power Systems virtualization environment” topic in the IBM Systems Director information center:

[http://publib.boulder.ibm.com/infocenter/director/pubs/index.jsp?topic=%2Fcom.ibm.director.vim.helps.doc%2Ffsd0\\_vim\\_r\\_sb\\_aix\\_on\\_power\\_capture\\_reqs.html](http://publib.boulder.ibm.com/infocenter/director/pubs/index.jsp?topic=%2Fcom.ibm.director.vim.helps.doc%2Ffsd0_vim_r_sb_aix_on_power_capture_reqs.html)

Before you import from a cloud provider, see which images are available to import from IBM Systems Director:

1. Open the IBM Systems Director management console (Figure 7-14).
2. In the left pane, expand **System Configuration** and select **VMControl**.
3. In the right pane, click the **Virtual Appliances** tab.

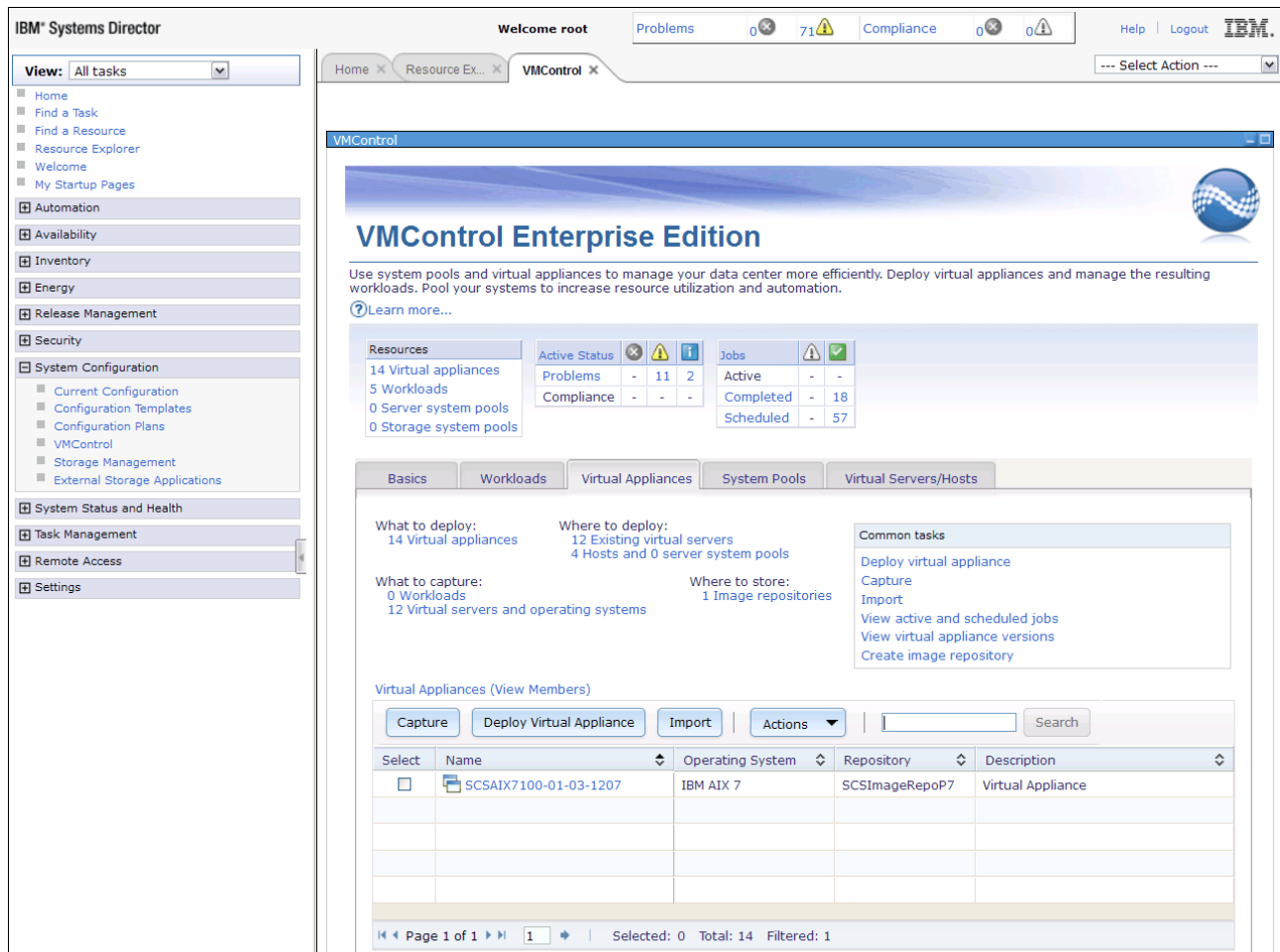



Figure 7-14 IBM Systems Director VMControl Virtual Appliances window

To import previously captured images from the cloud provider, use the following steps:

1. From the Image Construction and Composition Tool Welcome window, click **Build and manage images**.
2. Click the **Import from Cloud Provider** icon (.
3. In the Import Images from Cloud Provider window (Figure 7-15), do the following steps:
  - a. From the list of images that are available to import (left side), search for an image that you want to import by name or keyword. Select the image, and then click **Add**. Repeat this step to add up to five images to import.
  - b. Scroll to the right side of the window, and enter the user name and password for each image to import.
  - c. Click **Import**.

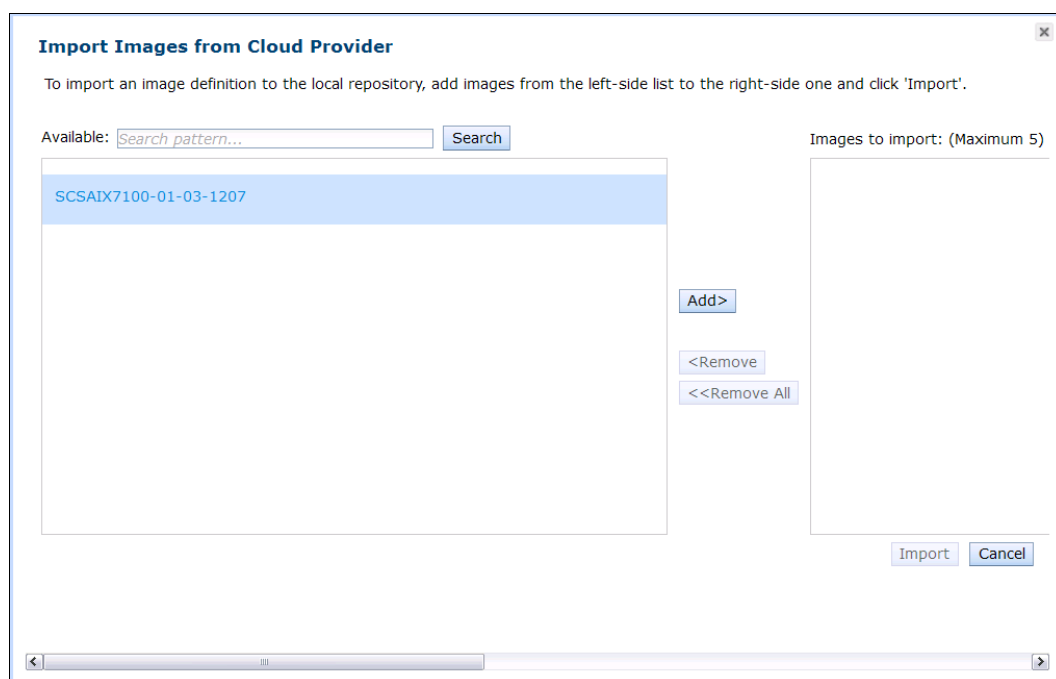



Figure 7-15 Import Images from Cloud Provider window

### 7.4.3 Extending, synchronizing, and capturing an image

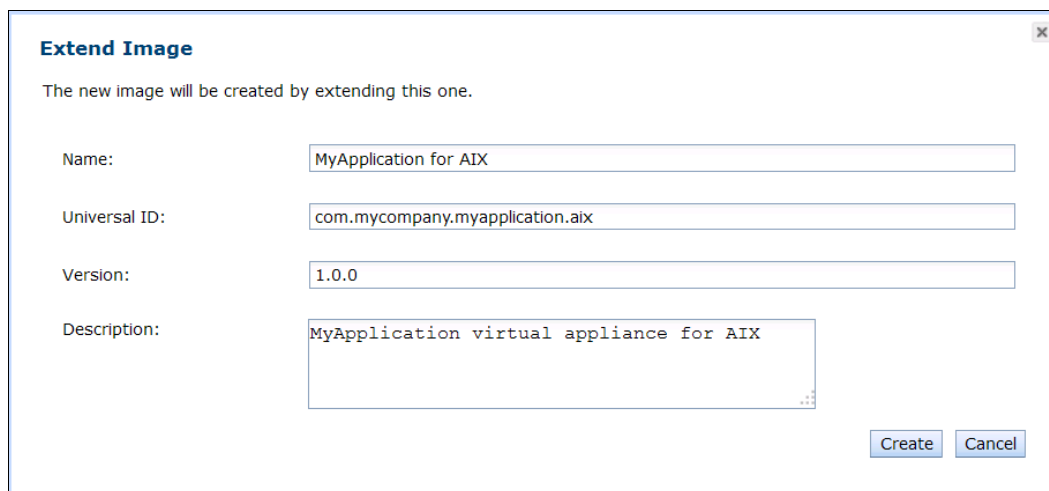
You can extend an image, add bundles, synchronize the image, and capture the virtual appliance.

#### Extending a virtual image

When you extend a virtual image, it is cloned so that you can reuse the original image without re-creating it from scratch each time. To create an image by extending an imported image, use the following steps:

1. From the Image Construction and Composition Tool Welcome window, click **Build and manage images**.
2. In the left pane, select the image that you want to extend, and click the **Extend** icon (.

3. In the Extend Image window (Figure 7-16), complete the following steps for the new virtual image:
  - a. Enter a name. In this example, we enter `MyApplication for AIX`.
  - b. Enter a unique universal ID. In this example, we enter `com.mycompany.myapplication.aix`.
  - c. Enter a unique version number. In this example, we enter `1.0.0`.
  - d. Enter a description. In this example, we enter `MyApplication virtual appliance for AIX`.
  - e. Click **Create** to extend the selected image to create an image.



**Extend Image**

The new image will be created by extending this one.

Name:

Universal ID:

Version:


Description:

Figure 7-16 Extend Image window

The new image is now displayed in the list on the left. You can use the same steps to import Linux and IBM i images.

## Adding software bundles to a virtual image


You can use software bundles to include additional software content to a virtual image. With the new image selected on the left side:

1. Click the **Edit** icon () to start configuring the new extended virtual image. You can change the name and description of the new image in this window.
2. To add software bundles to this image, click the **Software Bundles** section, and click **Add bundle**.
3. In the Add bundle to image window, select one or more bundles to include, and click **Add**.

After you add more than one bundle to your virtual image, you can specify the order of installation by using the up and down arrows. You can change only the *order* of the bundles that are added to this virtual image, and not bundles that are part of the base or previously extended image.

4. Click a software bundle to see its parameters.


If the software bundle creator provided installation options for the software bundle, the parameters are displayed in the *Install Parameters* section. The second set of parameters defines the configuration options that are made available during the deployment of the virtual image.

- a. Click **Install Parameters**, and review the default settings. Decide whether you want to modify any of the default values. The installation parameters are used during the virtual image synchronization to install the product in the software bundle.
  - b. Click **Deploy Parameters**, and review the default settings. Decide whether you want to modify any of the default values. The deploy parameters define the default parameter values that are used during virtual image deployment. The deployment parameters are used to customize the product for the environment for which they are being created.
5. When you are done, click the **Done editing** icon, and then click the **Save** icon () to save the configuration.

## Synchronizing a virtual image

*Synchronization* is the process of creating a VM by deploying the virtual image, deploying all the software bundles, and then in order, running the defined installation script of each bundle. The software bundles installation script is run according to the order specified in the virtual image.

With your image selected on the left side, synchronize the virtual image:

1. Click the **Synchronize** icon (.
2. In the Synchronize the image window (Figure 7-17), do the following steps:
  - a. Enter the user ID of the base image that was extended that you want to use.
  - b. Enter the password of the base image that was extended that you want to use.
  - c. Click **Done** to begin the synchronization process.

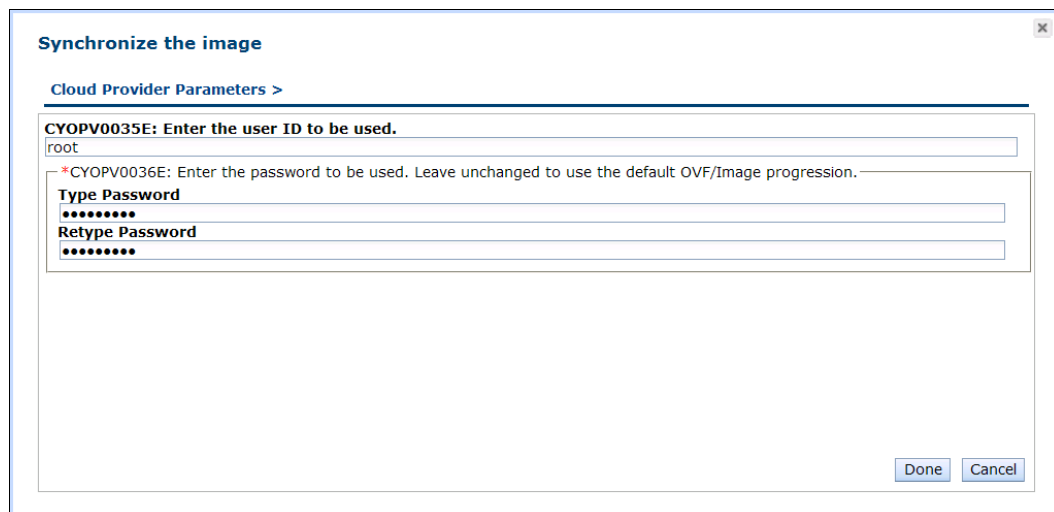



Figure 7-17 Synchronize the image window

You can click the **Refresh** icon () to view the progress of the synchronization process.

## Validating a virtual image

After the image is in *Synchronized* status, your bundles are installed on the image. If the bundle contained an installation operation (**Install** tab), the software defined by that operation is installed. If the bundle contained deploy-time operations (**Configuration** tab), the configuration defined by that operation is installed by the VSAE.

To further verify that these actions occurred, with your synchronized image selected, use the following steps:

**Important:** For some of the steps in this task, you must be familiar with the VSAE.

1. Expand the **Virtual System** section of the synchronized image that you want to validate. Write down the IP address and the host name that was assigned to the image.
2. Enter **ssh** to connect to the system by using the IP address or host name from step 1.
3. When prompted for a password, enter the password that you entered for the image during the synchronize process (Figure 7-17 on page 142).
4. After you log in to the system, verify that all the software bundles were installed and configured. The steps to verify each bundle depend on the specifics of the software that is part of the bundle. The basic steps are as follows:
  - a. Verify that the installation operations of the bundle completed successfully.
  - b. Verify that the deploy-time operations were successfully installed.

For AIX, use these steps:

- i. Verify your bundle name <Universal ID>\_<version> in the /opt/ibm/ae/AS directory.
- ii. Verify your configuration operations in the /opt/ibm/ae/AL/master.al file.
- iii. Verify that your activation program is correctly set up as an OS service:

```
ls -al /etc/rc.d/rc2.d
```

For Linux, use these steps:

- i. Verify your bundle name <Universal ID>\_<version> in the /opt/ibm/ae/AS directory.
- ii. Verify your configuration operations in the /opt/ibm/ae/AL/master.al file.
- iii. Verify that your activation program is correctly set up as an OS service:

```
ls -al /etc/rc.d/rc3.d
ls -al /etc/rc.d/rc5.d
```

For IBM i, use these steps:

- i. Verify your bundle name <Universal ID>\_<version> in the following directory:  
/QOpenSys/opt/ibm/ae/AS
- ii. Verify your configuration operations in the following file:  
/QOpenSys/opt/ibm/ae/AL/master.al
- iii. Verify that your activation program is correctly set up as an OS service:

```
ls -al /QOpenSys/opt/ibm/aescripts
```


The validation of the synchronized image is now completed. From here, you can capture the virtual image as described in the next section.




## Capturing a virtual image

After the virtual image is synchronized, the capture process takes a snapshot of the VM and saves it as a new image in the IBM Systems Director VMControl repository.

To capture a virtual image, use the following steps:

1. Do the clean-up tasks:
  - Delete any files or directories that are not needed.
  - Unmount any disks that are used during the installation of the virtual image that will not be available after the capture is complete.
  - Clean any history or log files.
2. Select the image on the left side, and then click the **Capture** icon () to start the capture process.


You can click the **Refresh** icon () to view the progress of the capture process.

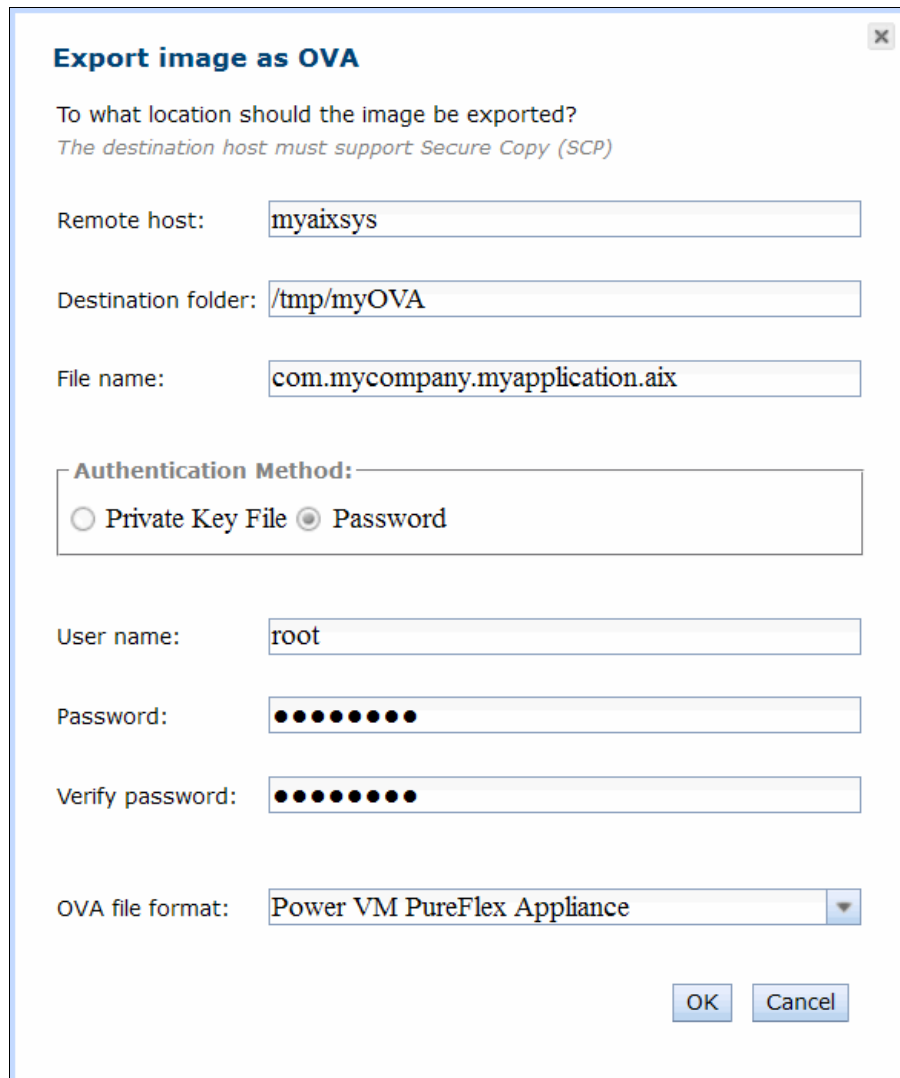
### 7.4.4 Exporting an image as an OVA archive

Exporting an image as an Open Virtual Appliance (OVA) archive entails capturing an image from the IBM Systems Director VMControl repository and transferring a compressed version of the image to a target system.

**Important:** Additional disk space is required on the VIOS to support export. For more information, see “Software” on page 54.

To export the image, use the following steps:

1. Select the image on the left, and click the **Export as OVA** icon () to start the export process.
2. In the Export image as OVA window (Figure 7-18 on page 145), do these steps:
  - a. In the Remote host field, enter the host name or IP address of the remote system to receive the OVA file.
  - b. In the Destination folder field, enter the folder name on the remote host system.
  - c. Keep the File name file, which is populated with the Universal ID for the image.
  - d. Select the authentication method.
  - e. Enter the user name and password of the user to access the remote host system.
  - f. Enter the password again to verify it.
  - g. For OVA file format, select **Power VM PureFlex Appliance**.
  - h. Click **OK** to begin the export process.



**Export image as OVA**

To what location should the image be exported?  
*The destination host must support Secure Copy (SCP)*

Remote host:

Destination folder:

File name:

**Authendication Method:**

☐ Private Key File ☒ Password

User name:

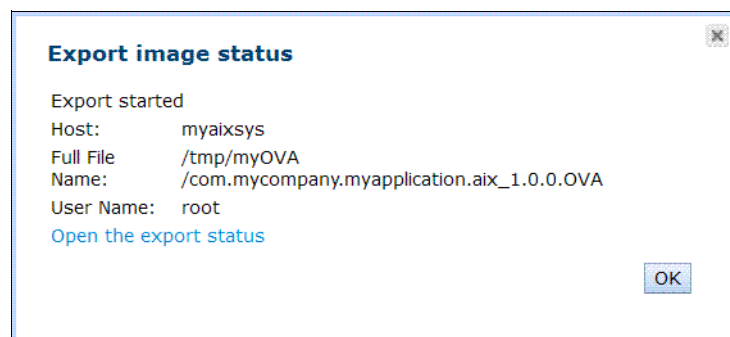
Password:

Verify password:

OVA file format:

Figure 7-18 Export image as OVA window

3. In the Export image status window (Figure 7-19), click the **Open the export status** link to display the status. Review the status and then click **OK** to close this window.



**Export image status**

Export started

Host: myaixsys

Full File Name: /tmp/myOVA/com.mycompany.myapplication.aix\_1.0.0.OVA

User Name: root

[Open the export status](#)

Figure 7-19 Export image status

You can verify that the OVA was exported by going to the specified directory on the remote host.

## 7.5 Troubleshooting

If you encounter problems while using the Image Construction and Composition Tool with the PowerVM provider, you can use the techniques in this section to help diagnose and eliminate them.

### 7.5.1 Checking the Image Construction and Composition Tool and the PowerVM provider versions

When reporting issues to IBM support, include the version information for both the Image Construction and Composition Tool and the PowerVM provider. To look up this required information:

- ▶ For the Image Construction and Composition Tool, in the upper-right corner of the window, click **About**. A message window opens that shows the version information.
- ▶ For PowerVM provider, click the **Home** tab of the Management Server web interface to see the version and release levels for IBM Systems Director, Systems Director VMControl, and Storage Management.

### 7.5.2 Checking the PowerVM provider logs

Check the PowerVM provider, IBM Systems Director, and Systems Director VMControl log files to determine where problems occurred. To access the log files and other information to provide when reporting a problem, go to the IBM Systems Director Information Center at the following address and search for Information to provide to the IBM Technical Support Center:

<http://pic.dhe.ibm.com/infocenter/director/pubs/index.jsp>

### 7.5.3 Checking the Image Construction and Composition Tool logs

Check the host system log files of the Image Construction and Composition Tool to determine where problems occurred. To access these log files, in the Welcome window, click the **Download logs** link, save the compressed file, and then examine the files.

The compressed file contains the following server log files:

- ▶ The `trace.log` file is the most recent trace file.
- ▶ The `error.log` file is the most recent error file.

You can also access the log files by clicking **Administer** → **Download logs**.

Alternatively, you can check the log files directly on the system that hosts the Image Construction and Composition Tool. These files are in the following directory:

`/drouter/ramdisk2/mnt/raid-volume/raid0/logs`

### 7.5.4 Importing from a running virtual machine

To begin, you see a list of all virtual servers that are visible on IBM Systems Director VMControl in the Virtual Servers and Hosts panel. The list is filtered based on the following criteria:

- ▶ Virtual servers that are in a *Started* state
- ▶ Virtual servers that are not part of a workload
- ▶ Virtual servers that have AIX, Linux or IBM i as their base operating system

After you select a running virtual server, the Image Construction and Composition Tool performs several actions to prepare the virtual server for capture. Such actions include system discovery, inventory collection, and request access. These operations might take several minutes depending on the various parameters.

While you are waiting for the capture process to complete, monitor the Image Construction and Composition Tool logs. Look for success messages and any exceptions. The Image Construction and Composition Tool then checks whether the virtual server has an activation engine installed. If an activation engine is installed, the Image Construction and Composition Tool runs a reset script, which is indicated by the following statement in the log:

```
"Detected AE 2.1.1 and going to run /opt/ibm/ae/AE.sh --reset"
```

After a successful reset operation, the contents of the `/opt/ibm/ae/AL/master.al` and `/etc/init.d` files are printed in the log. For IBM i virtual machines, use the `/QOpenSys/opt/ibm/ae` directory to look at the log files.

If the activation engine is not installed, the Image Construction and Composition Tool installs it and prints a success message in the log. If the virtual server is in the SCS repository, as required by the capture process, the virtual server is powered down. For other repositories, the virtual server is not powered down. For the SCS repository, the Image Construction and Composition Tool collects inventory every 10 minutes while waiting for the server to power down. A message, such as the following example, is entered in the log:

```
CloudProvider I com.ibm.cloud.icn.cloudprovider.powervm.PowerVMCloudProviderImpl  
waitForVirtualServerPowerOffCompletion Waiting for VirtualServer state of Stopped;  
status = Started
```

When collecting the inventory, the Image Construction and Composition Tool ignores any errors and continues to wait until the virtual server is stopped or it times out.

## Common errors

The common errors that are highlighted in the following sections can occur.

### Running VM is not available in the Import list

You might not see the VM in the Import list.

#### **Symptom**

The virtual server is not part of a workload, it is an AIX, Linux or IBM i system, and it is running. However, it is not available in the capture list of the Image Construction and Composition Tool.

#### **Cause**

Sometimes, Systems Director VMControl incorrectly shows the status of a VM as *Starting* or *Stop* when the VM is actually running. You can verify the status by entering `ssh` to connect to that system.

#### **Solution**

Check the Status column of the Virtual Servers and Hosts panel in the VM in Systems Director VMControl. If the status does not show a *Started* status, it will not be available in the capture list for the Image Construction and Composition Tool. In this case, start and stop the VM from Systems Director VMControl, and make sure that the state is *Started*.

## **Capture failed with a reason code greater than or equal to 300**

If the capture fails, you might see a reason code greater than or equal to 300 in the log file.

### ***Symptom***

The capture failed, and you see a message similar to the following example in the log:

"Capture of workload failed: 305"

The number can be any value greater than or equal to 300.

### ***Cause***

This failure occurs for many reasons, but is not caused by the tool. For example, the problem might exist in the capture repository or be from a corrupted image. If images are in the SCS repository, the server might not be stopped but the status in Systems Director VMControl is displayed as *Stopped*.

### ***Solution***

Look at the log files in Systems Director VMControl. Also try to reproduce this problem by capturing the virtual server from Systems Director VMControl. By reproducing the problem, you can see the error in the Systems Director VMControl logs. For the SCS repository, make sure that the server is stopped and that Systems Director VMControl is showing the correct status.

## **Import Failed**

If the import fails, check the log for messages indicating a problem finding the VM or accessing the VM.

### ***Symptom***

The capture failed and you see a message similar to either of the following examples in the log:

CY0PV0017E: Cannot find VM for capture.

CY0PV0018E: Denied access to VM for capture.

### ***Cause***

This failure can occur for many different reasons, all related to the interaction of the steps which must be done to prepare the VM for capture by Systems Director VMControl. Problems with these interactions can be more common when the Systems Director VMControl system is very busy.

One common cause is that the System Discovery step did not complete in the expected amount of time.

### ***Solution***

Because of the order in which the preparation steps are done, nothing has been done to the running VM at this point. Attempting the import again is safe and is likely to be successful.

If this problem is repeated several times, it is likely that the IP address or credentials entered on the Import dialog are incorrect and must be verified.

## 7.5.5 Importing from a cloud provider

The Image Construction and Composition Tool shows a list of images to choose from. After the user selects an image to import, and the selected image is already imported to the Image Construction and Composition Tool, the tool ignores the request and displays the existing image.

After importing the remote image into the Image Construction and Composition Tool host, the Image Construction and Composition Tool gets the OVF from the virtual appliance. Some topology-related information is updated on the local image.

### Import failed

A problem occurs when the status of the import is displayed as *Failed*.

#### Symptom

You see a failure message, such as the following example, in the log file:

```
"Unable to find specified image for import to ICON."
```

#### Cause

The image that is selected for import might be corrupted.

#### Solution

View the IBM Systems Director VMControl log files. Try to verify whether the image is corrupted by deploying it with Systems Director VMControl. If the image is not corrupted, restart the Image Construction and Composition Tool instance, and try the import process again.

### Unable to find the specified image

When the Image Construction and Composition Tool is unable to locate the image specified.

#### Symptom

An error message, such as the following example, is displayed upon import and in the log file, and is followed by the exception stack output:

```
"Unable to find specified image for import to ICON"
```

#### Cause

The image was deleted from the repository before the user selected an image and clicked **Import**.

#### Solution

Close the Import window in the Image Construction and Composition Tool, and restart the import process.

## 7.5.6 Extending an image

An imported image must be extended so that you can change it. After you click the **Extend** button, a copy of the original image is made. You can edit the copy of the image so that the original version of the image is protected if any mistakes are made.

### Failed to extend

Extending an existing image failed.

#### *Symptom*

An error message is displayed when you click the **Extend** button.

#### *Cause*

The image that is being extended might be corrupted.

#### *Solution*

Refresh the browser. Try to import other images, and extend them to determine whether the problem is with that specific image. Delete the image and import it again. Then deploy the image by using Systems Director VMControl to ensure that the image is not corrupted.

## 7.5.7 Synchronizing an image

After an image is extended, edited, and saved, it becomes unsynchronized. A synchronization operation, at a higher level, entails several actions, including creating a workload and a new VM in it, and then deploying the image.

You can monitor the progress by watching the log file of the Image Construction and Composition Tool and inspecting the web UI of Systems Director VMControl, where a new VM is deployed. This process might take a while because the Image Construction and Composition Tool waits for the VM to be fully deployed and started.

Several failure issues, and the symptom, cause, and solution of each, are described in this section.

### Systems Director VMControl deployment failed

The synchronization process fails after a few minutes.

#### *Symptom*

The status of the image being synchronized is *synchronization failed* and the log contains a message similar to the following example:

```
com.ibm.cloud.icn.cloudprovider.powervm.PVMDeployThread run DeployResponse:201
[application/octet-stream; charset=UTF-8]: DNZEMW350I Workload[32793] creation
started.
```

Another message is also issued:

```
com.ibm.cloud.icn.cloudprovider.powervm.PVMDeployThread checkForDelete The
workload is in Failed state
```

### **Cause**

The deployment of the image in Systems Director VMControl failed. Some of the parameters for the new VM were incorrect. Retrieving the Systems Director VMControl logs to pinpoint the problem might be necessary.

The following causes are likely:

- ▶ A domain name mismatch

At the time the PowerVM cloud provider is configured, the IP addresses in the address pool are resolved to a name, using the DNS configuration of the system on which the tool is running. If an IP address cannot be resolved to a name, a host name and domain are generated for it. The host name and domain name associated with that IP address are used to configure the deployed VM when an image is synchronized. Systems Director VMControl also resolves the assigned IP address to a name, and if the domain resolved by Systems Director VMControl does not match the domain name being used by the tool, Systems Director VMControl can fail the deployment.
- ▶ A problem with the storage pool that was used

If no storage pool is configured in the PowerVM cloud provider configuration, the tool will use the first one in the list of storage pools obtained from Systems Director VMControl. If the storage pool used does not have enough free storage to allocate the disks for the VM being deployed, the deployment will fail.
- ▶ A problem with the network adapter mappings

If VLAN mappings have been configured in the tool's `configuration.config` file, and the network ID specified for the mapping is invalid for the image being synchronized, the deployment will fail.

### **Solution**

The following list provides solutions for common problems:

- ▶ A domain name mismatch

Verify that the DNS configuration is correct on the system where the tool is running. In the PowerVM cloud provider configuration, remove each IP address and save the change, and then add the IP address and save the change. This way causes the tool to resolve each IP address again.
- ▶ A problem with the storage pool that was used

Configure the PowerVM cloud provider to use the correct storage pool.
- ▶ A problem with the network adapter mappings

Remove the VLAN mappings from the tool's `configuration.config` file and restart the tool. If VLAN mappings are required, enter values that are correct for the image being synchronized.

## **Synchronization failed**

The synchronization process fails.

### **Symptom**

A failed message is displayed.

### **Cause**

The following typical causes result in synchronization failing:

- ▶ The user name and password do not match the credentials in the image.
- ▶ The activation engine fails to reconfigure the image.



- The network interface does not come up.
- The network interface was mapped to a VLAN that is not bridged so that it is accessible to the system on which the tool is running.

### ***Solution***

From the tool's host system, if you can ping the IP address that the Image Construction and Composition Tool shows for the Virtual System of the image being synchronized, then the network interface activated successfully. Try to connect to the IP address using SSH and logon using the credentials that were entered in the synchronization dialog. If you can log on, check the `/opt/ibm/ae/AR/ovf-env.ar` file to determine whether there were activation failures.

If pinging the IP address assigned by the tool is not possible, use the Systems Director VMControl UI to determine whether the workload and virtual server started successfully. You might need to connect to the deployed virtual server using the Systems Director VMControl console function to check the `/opt/ibm/ae/AR/ovf-env.ar` file to determine whether there were activation failures.

To determine whether a VLAN was assigned that is not bridged, go through the virtual appliance deployment panels in the Systems Director VMControl user interface to deploy the same virtual appliance to the same hypervisor as the PowerVM cloud provider is using. On the Network Mapping panel, if the virtual network that is assigned by default says it is not bridged, you must specify overrides for the VLAN mappings in the `configuration.config` file.

On the Systems Director VMControl Network Mapping panel, determine the Network ID to be mapped and the virtual network it should be mapped to, and then update the tool's `configuration.config` file by using the following steps:

1. Locate the tool's `configuration.config` file. This file is usually in the following location, although the actual location can vary depending on how the tool was installed:  
`/opt/IBM/icct/icn.app/config/configuration.config`,
2. Edit the `configuration.config` file with a text editor of your choice, such as `vi`. When you substitute the actual PowerVM cloud provider name in the following steps, replace any spaces with underscores. Edit the file as follows:
  - a. Add a line to specify the Network ID to be mapped, in the following form:
 

```
"PowerVM_<PowerVMCloudProviderName>_Deploy_NetworkMappingNetworkId" :  
"<Network Mapping ID>",'
```

 Replace `"<PowerVMCloudProviderName>"` and `"<Network Mapping ID>"` with the actual values.
  - b. Add a line to specify the virtual network it will map to, in the following form:
 

```
"PowerVM_<PowerVMCloudProviderName>_Deploy_NetworkMappingHostVnet" : "<Host  
Vnet>",'
```

 Replace `"<PowerVMCloudProviderName>"` and `"<HostVnet>"` with the actual values.
  - c. If one of the lines added in these steps is the last value in the file, remove the comma at the end.
  - d. Save the updated `configuration.config` file.
3. Restart the tool so that the updated configuration can take effect:
  - a. Run the `/opt/IBM/icct/stop.sh` command.
  - b. Run the `/opt/IBM/icct/start.sh` command.
4. Try to synchronize the image again.

If, with these actions, you are not able to find the cause of the failure, check the status of the VM in Systems Director VMControl. Look at the trace log in the Image Construction and Composition Tool for the following statement:

"Location from the header of deploy response:"

This message contains a URI with the workload ID as the last part of the URI. Write down the workload ID, and check the log file of Systems Director VMControl.

## 7.5.8 Capturing an image

The Image Construction and Composition Tool can capture a deployed image, which is part of a workload. The tool performs several actions to prepare the virtual server for capture. Such actions include system discovery, inventory collection, and request access. These operations might take several minutes depending on the various parameters. While you are waiting for the capture process to complete, monitor the trace and error log files of the tool.

The following message means that the synchronization process detected VSAE version 2.1.1 and will perform the reset of the activation engine and the virtual machine:

"Detected AE 2.1.1 and going to run /opt/ibm/ae/AE.sh --reset"

After a successful reset operation, the contents of the /opt/ibm/ae/AL/master.al and /etc/init.d files are printed in the trace log of the tool. For IBM i virtual machines, use the /QOpenSys/opt/ibm/ae directory to look at the log files.

If the activation engine is not installed, the tool installs it with a success message in the log file. If the activation engine is in SCS repository, as required by the capture process, the virtual server is then powered down. For other repositories, the virtual server is not powered down. For the SCS repository, the tool collects an inventory every 10 minutes while waiting for the server to power down.

A message, similar to the following example, is displayed in the trace log of the tool:

```
CloudProvider I com.ibm.cloud.icn.cloudprovider.powervm.PowerVMCloudProviderImpl
waitForVirtualServerPowerOffCompletion Waiting for VirtualServer state of Stopped;
status = Started
```

When collecting inventory, the tool ignores any error and continues to wait until the virtual server is stopped or times out.

### **Capture failed with a reason code that is greater than or equal to 300**

If the capture fails, you might see a reason code greater than or equal to 300 in the log file.

#### ***Symptom***

The capture failed, and you see a message similar to the following example in the log:

"Capture of workload failed: 305"

The number can be any value greater than or equal to 300.

#### ***Cause***

This failure occurs for many reasons, but is not caused by the tool. For example, the problem might exist in the capture repository or be from a corrupt image. If images are in the SCS repository, the server might not be stopped but the status in Systems Director VMControl is displayed as *Stopped*.

### ***Solution***

Look at the log files in Systems Director VMControl. Also try to reproduce this problem by capturing the virtual server from Systems Director VMControl. For the SCS repository, verify that the server is stopped and that Systems Director VMControl is showing the correct status.

### **Capture failed**

If the capture fails, check the log for messages that indicate a problem finding the VM or accessing the VM.

### ***Symptom***

The capture failed and you see a message similar to either of the following examples in the log:

CY0PV0017E: Cannot find VM for capture.  
CY0PV0018E: Denied access to VM for capture.

### ***Cause***

This failure can occur for many reasons, all related to the interaction of the steps that must be done to prepare the VM for capture by Systems Director VMControl. Problems with these interactions can be more common when the Systems Director VMControl system is very busy.

One common cause is that the System Discovery step did not complete in the expected amount of time.

### ***Solution***

Because of the order in which the preparation steps are completed, nothing has been done to the running VM at this point. Attempting the import again is safe and is likely to be successful.

If this problem is repeated several times, the IP address or credentials entered on the Import dialog are likely incorrect and must be verified.

## **7.5.9 Exporting an image**

Depending on the size of the image, export can take a long time to finish. At a higher level, during an export operation, the Image Construction and Composition Tool gains access to a copy of the image, prepares the rest of the OVA contents for export and merges the image with the rest of the OVA contents on the target system.

**Disk space:** Because the whole process might take several minutes or hours to complete, make sure that enough disk space is available on the repository and on the target system. The tool logs whether the target system has enough disk space, but continues processing. If you see that available space is less than the space needed in the log, allocate more space in the target system.

The tool differentiates images that are in the SCS and NIM repositories. For exports from the SCS repository, the tool performs several actions to gain access to a copy of the image; this can take a very long time, depending on the image size. Monitor the log file to see the details of some of the steps.

You can monitor the target directory of the target system to verify that the OVA file is being copied there (note that this will not happen for exports from SCS repositories until the copy of the image has been accessed); the image file and OVA will be merged together in a temporary subdirectory of the target directory.

The export process contains multiple, complicated steps. Therefore, monitoring the log files and the systems as described in this section is the best way to understand why it might fail.

### **Image Export status is “Auth fail”**

This status is an indication that the tool's export process is unable to connect to the system specified as the target of the export.

#### ***Symptom***

The status of the export is *Auth fail*. The log files log an exception similar to this example:

```
com.jcraft.jsch.JSchException: Auth fail
```

#### ***Cause***

The tool's export process connects to the target system to execute the necessary commands to build the export artifact. This exception can happen if the system's SSHD configuration does not allow the type of connection being made. It can also indicate that the credentials entered for the system in the export dialog are incorrect.

#### ***Solution***

Verify the target system's credentials are entered correctly in the export dialog.

Verify that the target system's SSHD configuration allows password authentication. The SSHD configuration file must have the value “yes” set for the `PasswordAuthentication` parameter in one of the following locations:

- AIX or Linux

`/etc/ssh/sshd_config`

- IBM i

`/QOpenSys/QIBM/UserData/SC1/OpenSSH/openssh-<openssh version>/etc/sshd_config`

After changing this value, restart the SSHD service.





## ESX cloud provider

This chapter explains how to configure and use a VMware ESX server as a cloud provider target in the Image Construction and Composition Tool (referred to as *the tool* in this book). It includes the following sections:

- ▶ Overview of an ESX cloud provider
- ▶ Requirements for a VMware ESX cloud provider
- ▶ Configuring an ESX cloud provider
- ▶ Creating a virtual appliance
- ▶ Troubleshooting

## 8.1 Overview of an ESX cloud provider

The ESX cloud provider (referred to as the *ESX provider* in this book) is a back-end node that implements the functions that are required by Image Construction and Composition Tool. The ESX provider calls the VMware APIs to accomplish the following tasks:

- ▶ Create a base virtual image from a running virtual machine.
- ▶ Import a base virtual image.
- ▶ Capture an ESX virtual image.
- ▶ Install bundles on a virtual image.
- ▶ Capture an extended virtual image.
- ▶ Export a virtual appliance.


## 8.2 Requirements for a VMware ESX cloud provider

To configure an ESX server as a cloud provider for the Image Construction and Composition Tool, you must install the following software:

- ▶ VMware ESX version 4.x or ESXi version 5 (Update 2 for ESX 4 and Update 1 for ESXi version 5 is required for Windows images)
- ▶ VMware vCenter version 4.x or 5 (to deploy virtual images)

**Important:** ESXi version 4 is not supported as a cloud provider platform. However, you can deploy the appliances that are built with the provider to the ESXi hypervisor.

## 8.3 Configuring an ESX cloud provider

You create an ESX provider in Image Construction and Composition Tool. When the tool is started for the first time, it starts the “Create a new cloud provider” wizard to create a cloud provider. This wizard starts automatically every time you start the tool until you create a cloud provider. If you need to create additional cloud providers, select **Administer** → **Manage cloud providers**. Then, click the **New cloud provider** icon (.

**Important:** After the cloud provider is created, you are not able to edit it.

To configure an ESX provider, complete the following steps:

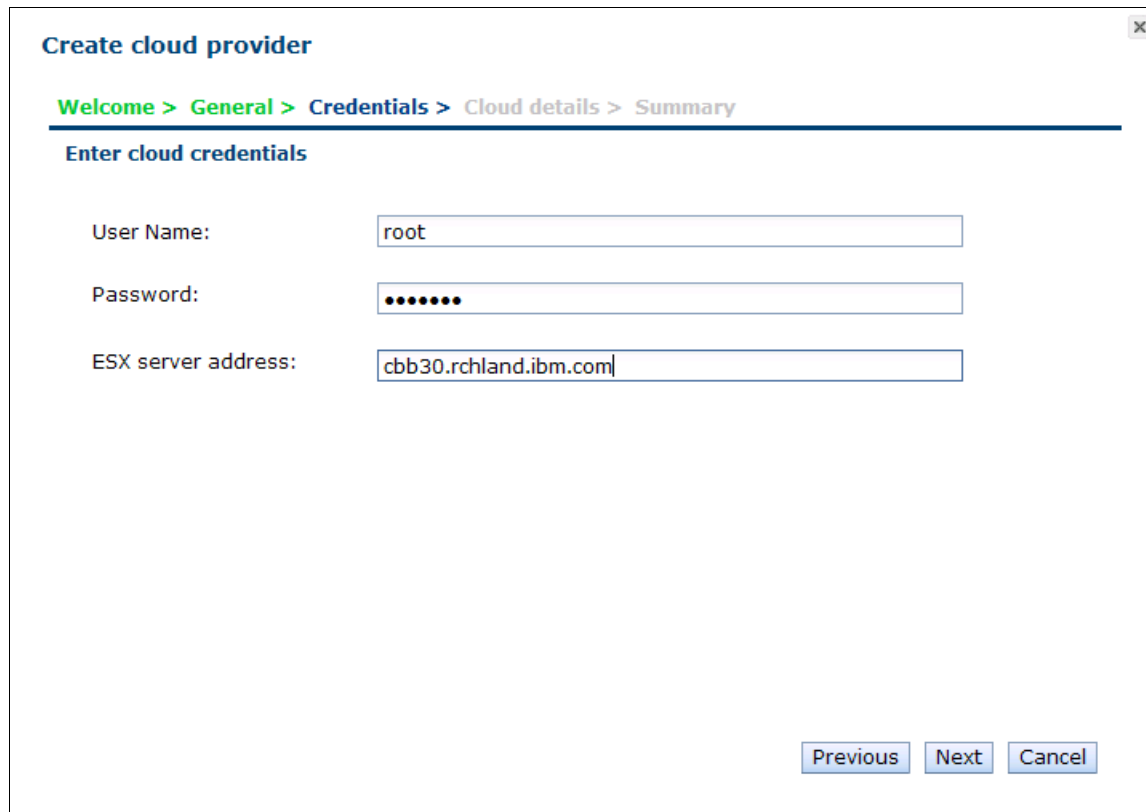
1. When the wizard starts, in the Welcome window, click **Next**.
2. In the General window (Figure 8-1), do the following tasks:
  - a. In the Name field, enter a unique name for the cloud provider.
  - b. Optional: Enter a brief description.
  - c. For Cloud Provider Type, select **VMware ESX**.
  - d. Click **Next**.

The screenshot shows a window titled "Create cloud provider" with a close button in the top right corner. Below the title is a breadcrumb navigation bar: "Welcome > General > Credentials > Cloud details > Summary". The "General" tab is selected. Below the breadcrumb is a section header "Specify a name and description of the connection". There are three input fields: "Name:" with the value "ESX Cloud Provider", "Description:" with the value "ESX Cloud Provider", and "Cloud Provider Type:" with a dropdown menu showing "VMware ESX". At the bottom right are three buttons: "Previous", "Next", and "Cancel".

Figure 8-1 General cloud provider information



3. In the Credentials window (Figure 8-2), enter the credentials to log in to the ESX server:
  - a. Enter a user name and password for the ESX server.
  - b. In the ESX Server Address field, enter the IP address or host name of ESX server.
  - c. Click **Next**.



**Create cloud provider**

Welcome > General > **Credentials** > Cloud details > Summary

---

**Enter cloud credentials**

User Name:

Password:

ESX server address:

Previous Next Cancel

Figure 8-2 Credentials for cloud provider

4. In the Cloud details window (Figure 8-3 on page 161), complete the following actions for the configuration. Notice that the Deployment Network name field and the Data store field are automatically populated by Image Construction and Composition Tool. For the Data store field, if more than one data store is available, select one from the list.
  - a. Enter the subnet network address.
  - b. In the Netmask field, enter the subnet mask.
  - c. Enter the gateway address.
  - d. In the Primary DNS field, enter the primary DNS address.
  - e. In the Secondary DNS field, enter the secondary DNS address.
  - f. Enter an IP address or a range of IP addresses to be used when extending an image in the tool. To enter an IP address or a range of addresses, do the following tasks and then click **Next**:
    - i. Click the plus icon (+).
    - ii. Enter the IP address or range of addresses.
    - iii. After you enter each IP address, press Enter to save the address in the table.

**Important:** Ensure that the IP address that you enter for your deployment addresses can be resolved by the DNS server. You can verify the address by doing a reverse DNS lookup.

**Create cloud provider**

Welcome > General > Credentials > Cloud details > Summary

Data store: datastore1 - https://cbb30.rchland.ibm.com/sdk#Datastore1

Subnet address: 192.168.10.0

Netmask: 255.255.255.0

Gateway Address: 192.168.10.1

Primary DNS: 192.168.10.200

Secondary DNS: 192.168.10.100

Deployment IP Addresses: Enter a number of IP addresses that are valid and available. Press Enter after each IP address to save it to the table

| IP Address Range Begin | IP Address Range End |
|------------------------|----------------------|
| 192.168.10.5           | 192.168.10.50        |

Previous Next Cancel

Figure 8-3 Cloud detail in the Create cloud provider wizard

5. Review the cloud provider Summary page and verify that the information is correct. Click **Done**.

## 8.4 Creating a virtual appliance

This section explains how to create a virtual appliance and includes the following sections:

- ▶ Creating a base OS image
- ▶ Importing a base image
- ▶ Working with bundles
- ▶ Extending, synchronizing, and capturing a virtual image
- ▶ Exporting an image as an OVA archive

Figure 8-4 illustrates the end-to-end process of creating a virtual appliance.

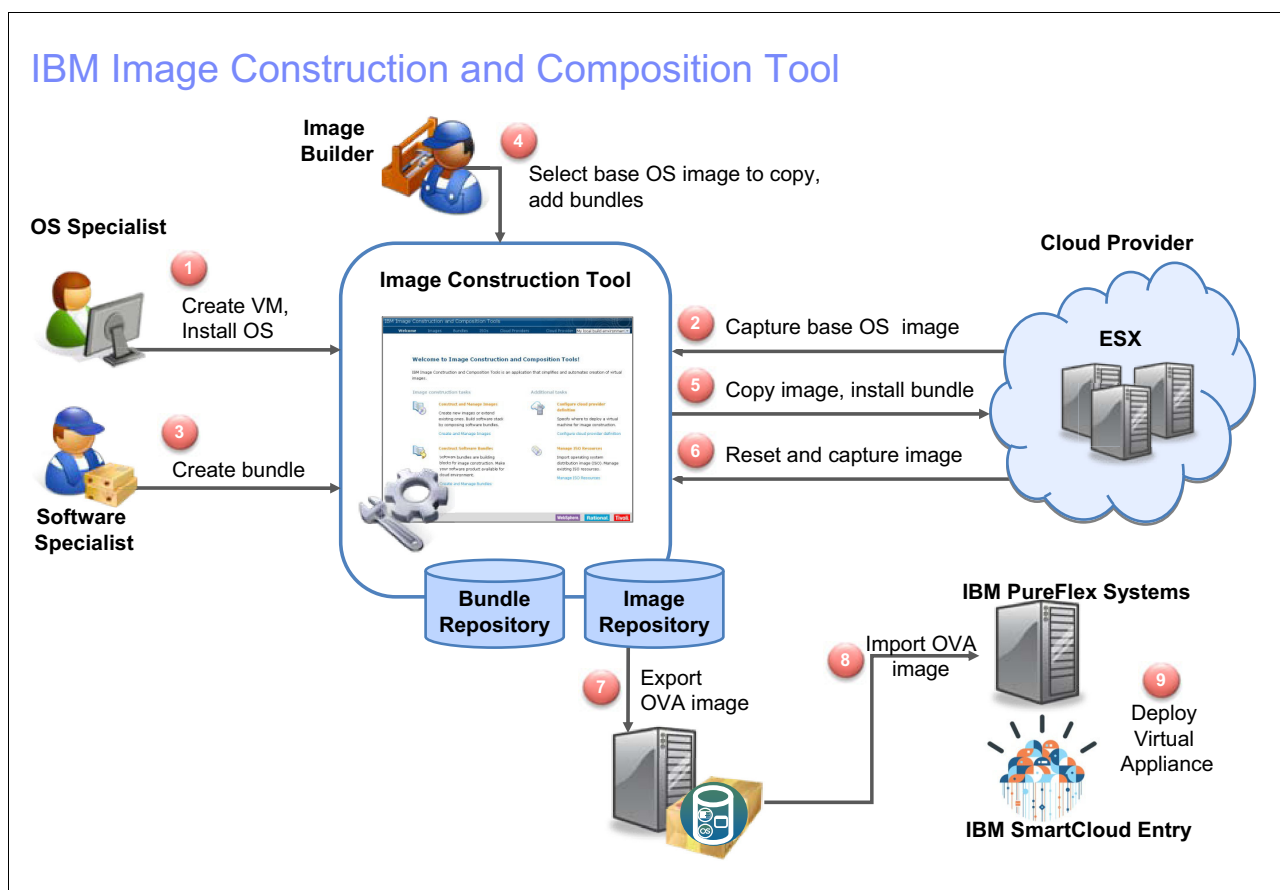


Figure 8-4 End-to-end creation process of a virtual appliance

### 8.4.1 Creating a base OS image

To create a virtual appliance, you must first create a base operating system image, which entails a two-phase process. First, you need to create a base virtual machine, and then you install a base OS. The steps in this section use VMware vCenter to create this image. However, you might choose to use a different tool, such as VMware Studio, to accomplish this task.

The base operating system image that you are creating must meet the following requirements to successfully import it into the Image Construction and Composition Tool:

- ▶ Must be one of the following supported operating systems:
  - Red Hat Enterprise Linux (RHEL) 6.x
  - SUSE Linux Enterprise Server (SLES) V11 SP 1
  - Windows 7 (64 bit only)
  - Windows 2008 R2 (64 bit only)
- ▶ Must have VMware tools installed.
- ▶ Must have the hard disk type set to SCSI. IDE disk type is not supported.
- ▶ Must have the hard disk defined as thick provisioned or Virtual Machine File System (VMFS) flat. Thin provisioning or VMFS sparse is not supported.
- ▶ Must have one or more hard disks.
- ▶ Delta disks are not supported.

- ▶ Snapshots are not supported.
- ▶ Must have only one network interface. Multiple network interfaces are not supported.
- ▶ On Windows, the Remote Registry service must be running to enable Remote Execution and Access (RXA).
- ▶ On Linux:
  - Must have Network Manager disabled or uninstalled.
  - Must have SELINUX set to permissive or disabled.
  - Must have SSH running and available.

If you are going export the image for IBM Workload Deployer, do not use fourth extended (ext4) file systems types. They are not supported with IBM Workload Deployer.

## Creating a Linux base virtual machine

You can use several tools to create a VMware virtual machine. However in this chapter, the base operating system image is created on the ESX server by using VMware vCenter. You need the vSphere client to complete the process. You can download the client directly from your VCenter installation by pointing your browser to the vCenter system:

`http://yourVCenterSystem`

Click **Download vSphere client** to download and install the client.

To create a base virtual machine, complete the following steps:

1. From the VMware vSphere client, log in to your ESX server.
2. Right-click the IP or host name of the ESX server on which you want to create the image and select **File** → **New** → **Virtual Machine**.
3. In the window that opens, select **Typical** for the most common configuration, and then click **Next**.
4. In the next window, enter a unique name for the virtual machine, and then click **Next**.

5. In the Host/Cluster panel (Figure 8-5), select the ESX server on which you want the virtual machine to be installed. Then click **Next**.

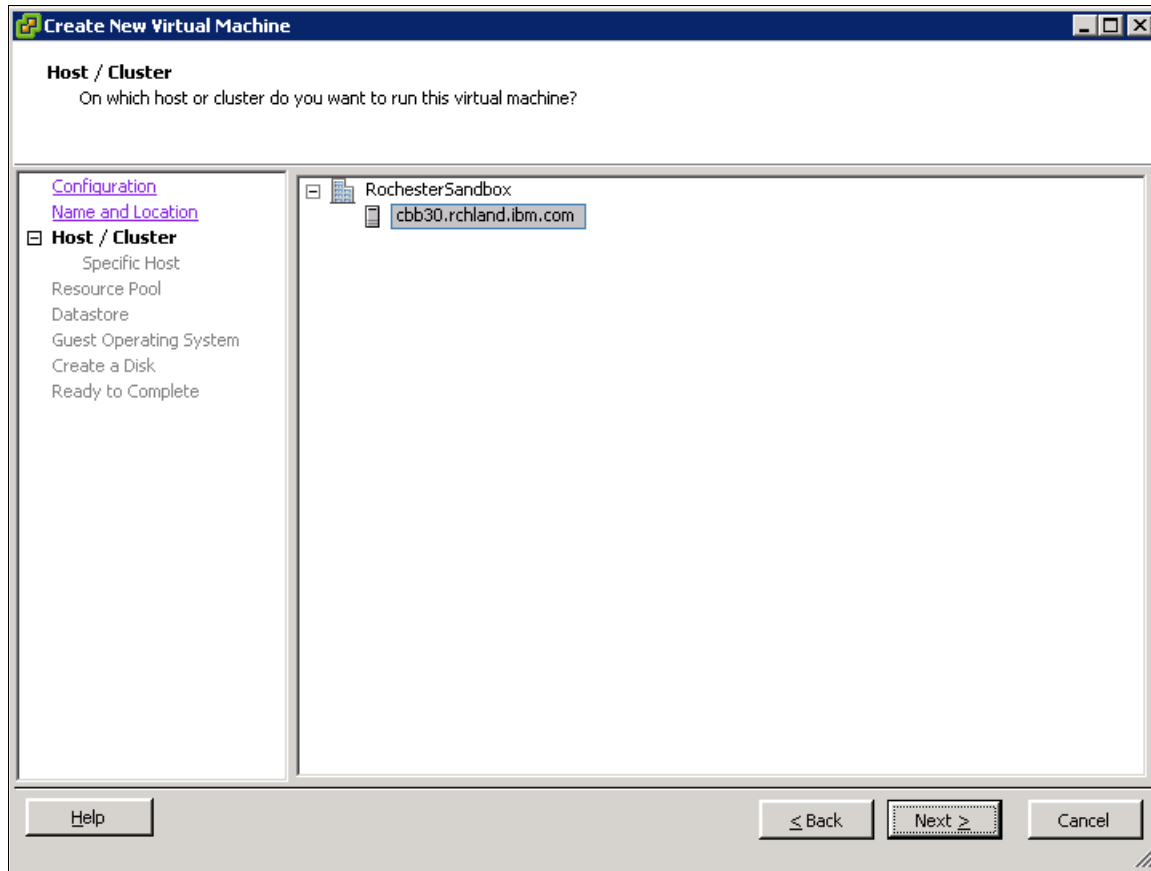


Figure 8-5 Selecting the ESX server

6. In the Datastore panel (Figure 8-6), select the data store where the virtual image will be stored. In this example, only one data store is available to choose from. Click **Next**.

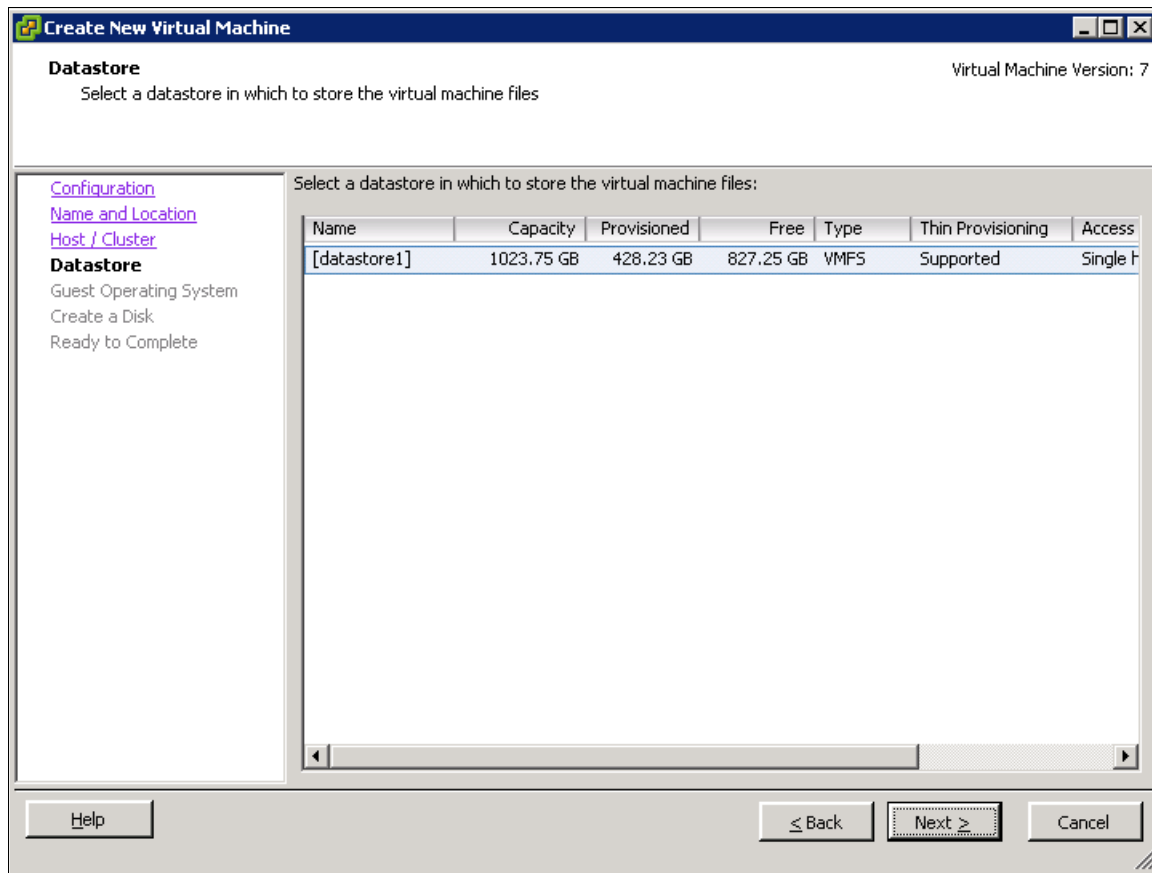


Figure 8-6 Selecting the data store to store the virtual image

7. Select the type of operating system that you are installing. For this example, Red Hat Enterprise 6.2 (64-bit) is installed. Therefore, we select **Linux** for the operating system. Then from the version list, select **Red Hat Enterprise Linux 6 (64-bit)**. Click **Next**.

8. In the Create a Disk panel (Figure 8-7), specify the required size of your virtual disk. For this example, the size is fairly small. Therefore, in the Virtual disk size field, we selected **12 GB**.

**Unsupported:** Do not select the following option because thin provisioning is not supported with the Image Construction and Composition Tool:

Allocate and commit space on demand (Thin Provisioning)

Then, click **Next**.

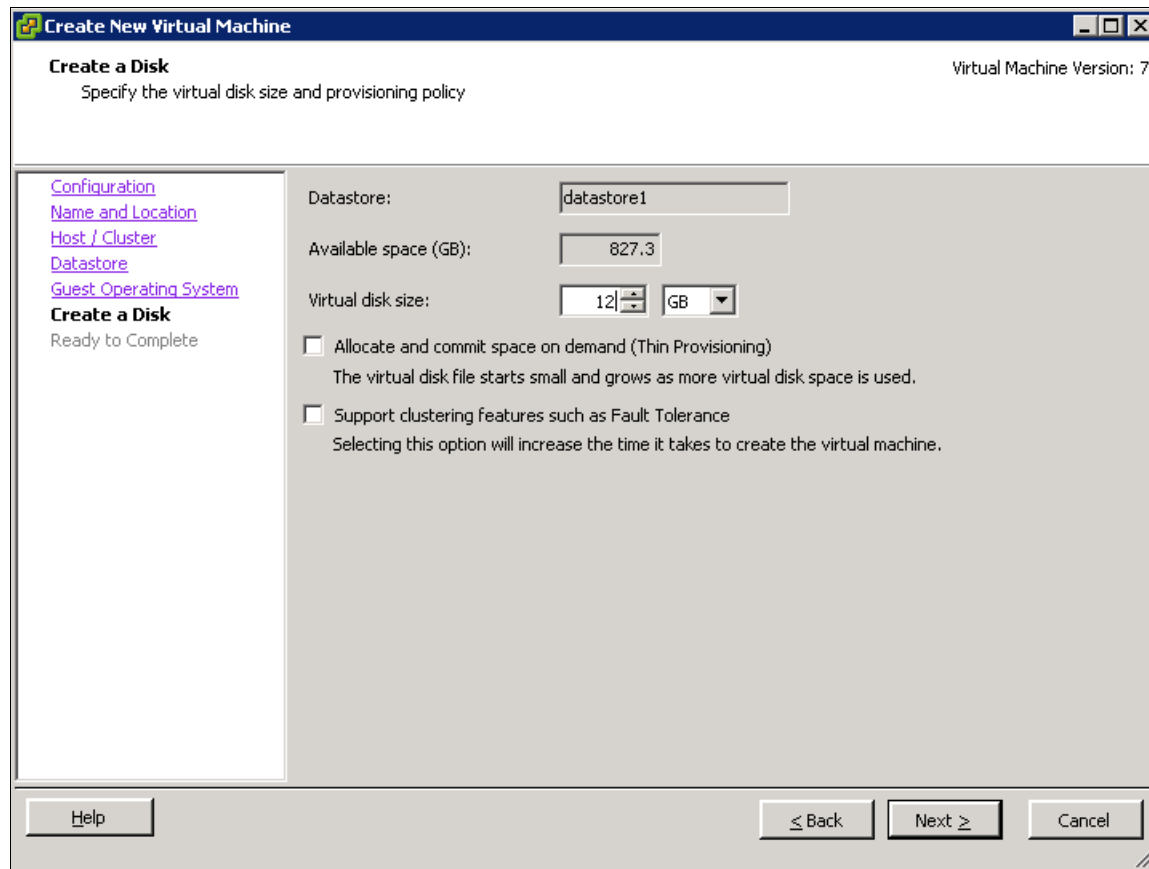


Figure 8-7 Specifying the disk size for the virtual image

9. In the Ready to Complete panel (Figure 8-8), review the virtual machine settings. Select the **Edit the virtual machine settings before completion** check box, and then click **Continue**.

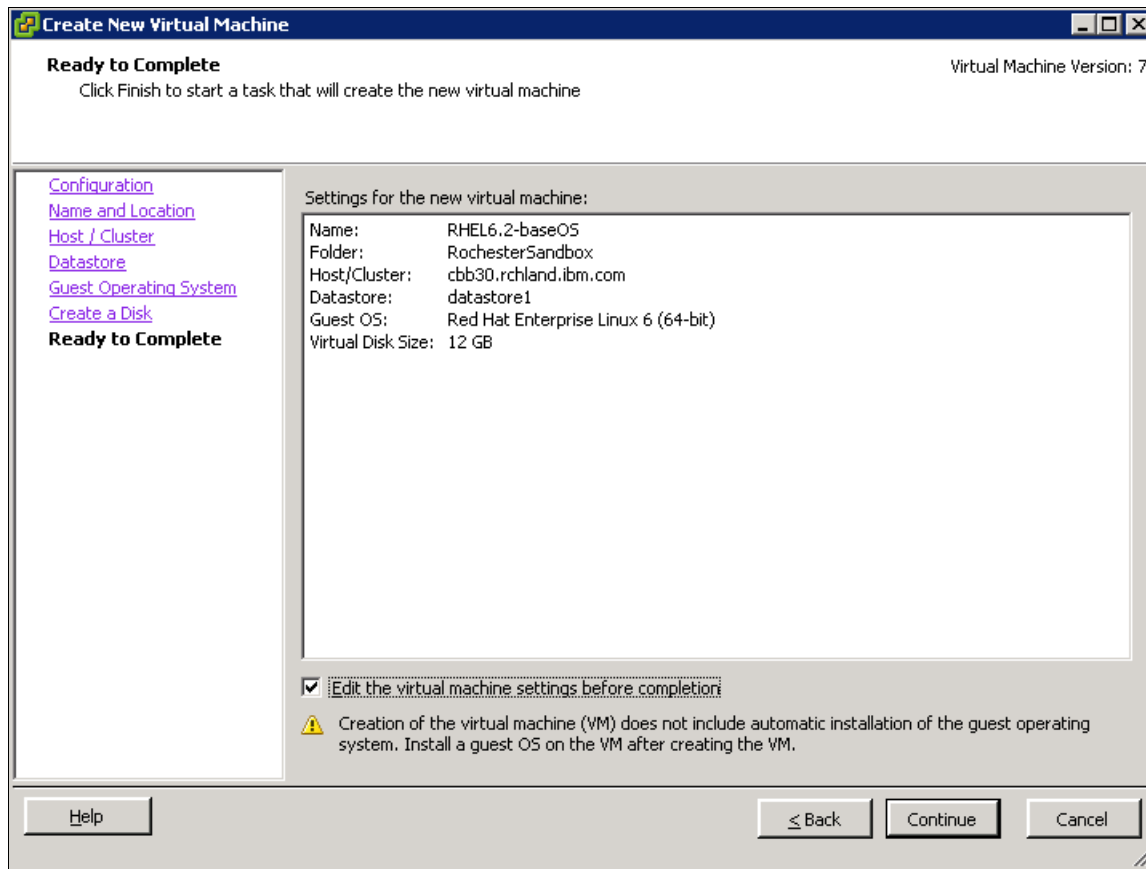


Figure 8-8 Ready to Complete panel

10. Edit the amount of memory that is required for your base operating system image. For this example, we use 2 GB. Therefore, change the Memory size field to 2.



11. In the Virtual Machine Properties window (Figure 8-9), on the **Hardware** tab, do the following actions:
- Select **New CD/DVD (adding)**. For this example, the operating system is installed from an ISO.
  - In the right side of the window, select **Datastore ISO File**.
  - Click the **Browse**, and locate your ISO file.
  - Select the **Connect at power on** check box.
  - Click **Finish**.

**Tip:** Copy all operating system ISOs that you want available when creating images to a directory on the data store, to make them available for selection.

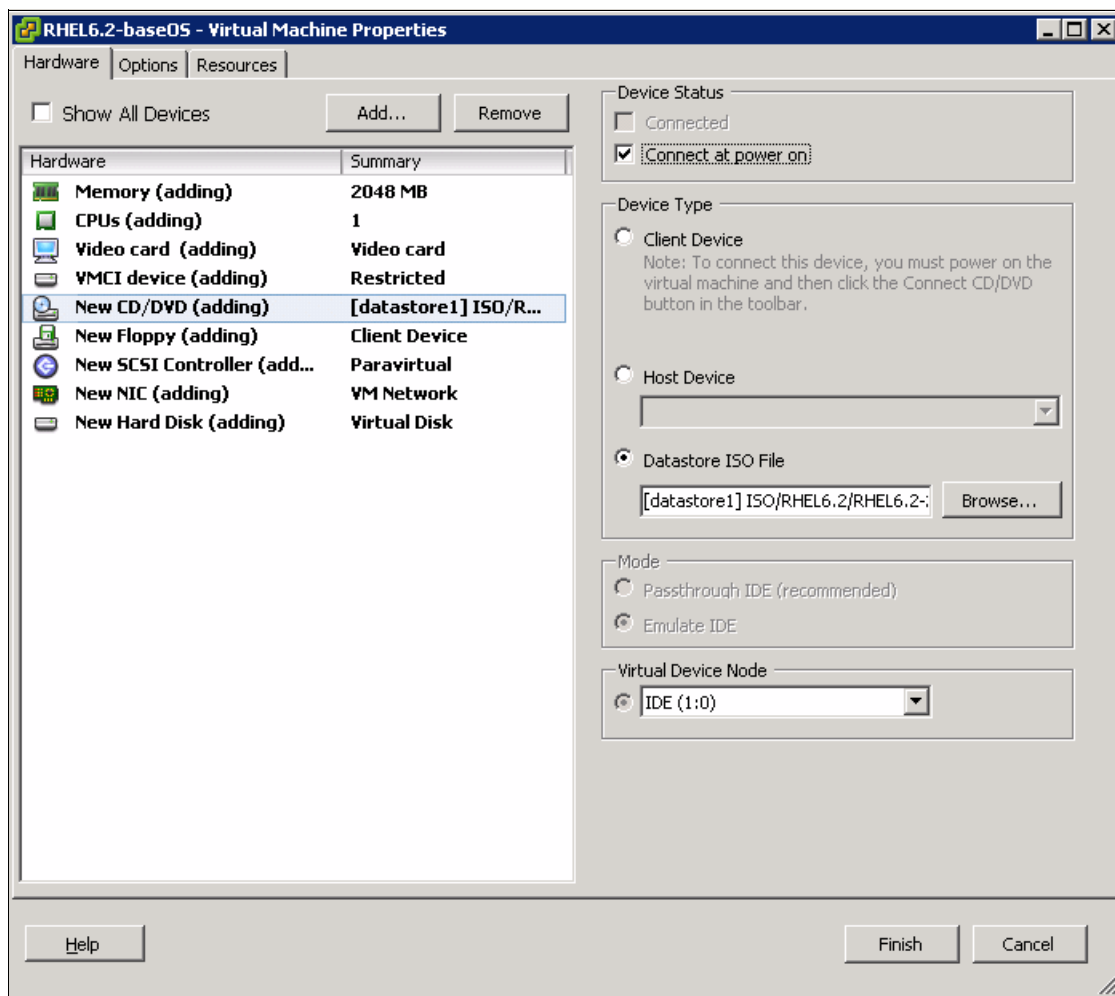


Figure 8-9 Configuring the virtual machine

12. After the virtual machine is created, install the operating system:

- a. In the vCenter, select the ESX server on which you created the virtual machine. A list of virtual machines is displayed in the right panel. Right-click the virtual machine you just created, and select **Power** → **Power On the virtual machine**.
- b. When you see the green triangle next to the image, right-click the image, and select **Open Console** to complete the installation of the operating system. Make sure that the operating system meets the requirements listed previously.

13. Install the VMware tools, and configure them in the image. For more information, see the VMware Workstation 5 documentation:

[http://www.vmware.com/support/ws5/doc/new\\_guest\\_tools\\_ws.html](http://www.vmware.com/support/ws5/doc/new_guest_tools_ws.html)

## Creating a Windows base virtual machine

The process for creating a Windows base virtual machine is similar to creating a Linux base image. After the operating system is installed, a few operating system configuration differences must be configured. This section explains the settings for the Windows 7 and Windows Server 2008 R2 base images. The requirements in 8.4.1, “Creating a base OS image” on page 162 still apply for Windows images.

### Windows 7 (64 bit)

To create a Windows and base virtual machine, complete the following steps:

1. Verify that port 445 is not blocked by the firewall.
2. Verify that the Remote Registry service is running to enable RXA. This service is set to manually start by default. To verify that the service is enabled and running:
  - a. Click **Start** → **Run** and type `services.msc`. Then press Enter.
  - b. After the Microsoft Management Console starts, make sure that the Remote Registry service is started.

If this service is not started, do the following tasks:

    - i. Right-click the service, and click **start**.
    - ii. To set the service to start automatically, click the **Remote Registry service** again, but this time, select **Properties**.
    - iii. For Startup type, select **Automatic**.

3. If you are a member of a local administrator group and you use a local user account, to perform administration tasks on the target machine, disable the User Account Control function:

- a. Click **Start** → **Run**.
- b. Type `regedit`, and press Enter.

You must perform this step when you administer a workstation with a Security Account Manager local user account. If you do not disable this option, you do not connect with full administrator rights and are unable to perform administrator tasks.

4. Find the `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\system` subkey. If the `LocalAccountTokenFilterPolicy` registry entry does not exist, create it:
  - a. Right-click the right panel, click **New** → **DWORD value**.
  - b. Enter `LocalAccountTokenFilterPolicy` and then press Enter.
  - c. Right-click **LocalAccountTokenFilterPolicy**, and click **Modify**.
  - d. In the Value data field, type 1, and click **OK**.
5. Restart the system.

### **Windows server 2008 R2 (64 bit)**

To create a base virtual machine on Windows server 2008 R2, complete the following steps:

1. Verify that port 445 is not blocked by the firewall.
2. Disable User Account Control if your account is not a domain user account. If you have a domain user account, ensure that the local and the target machines are both members of a Windows domain.
3. If you are a member of a local administrators group and you use a local user account, do the following tasks:
  - a. Enable the built-in Administrator account, and use it to connect:
    - i. Open the Windows Control Panel.
    - ii. Select **Administrative Tools** → **Local Security Policy** → **Security Settings** → **Local Policies** → **Security Options**.
  - b. Disable User Account Control when you administer with a Security Account Management local user account as you did in the Windows 7 image.

Alternatively, run the following command:

```
cmd /c reg add  
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\system /v  
LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
```


## **8.4.2 Importing a base image**

You can import a base OS image into Image Construction and Composition Tool by using various methods. For an ESX provider, the following methods are possible:

- ▶ From a running VM
- ▶ From a template
- ▶ From a cloud provider

### **Importing from a running virtual machine**

To import a virtual image into the Image Construction and Composition Tool from a running virtual image, complete the following steps:

1. Log in to vCenter to see what all virtual images are running, and obtain the IP address of the image.
2. From Image Construction and Composition Tool Welcome page, click **Build and manage images**.
3. Click the plus icon () to create an image in the tool.
4. Select **Create image from running virtual machine**, and then click **Proceed**.
5. In the Import Images from Running VM window (Figure 8-10 on page 171), do the following tasks:
  - a. Enter a name for the new virtual image.
  - b. Enter a universal ID for the new virtual image. This field is required.
  - c. Enter a version number for the new virtual image.

**Universal ID and Version field values:** The value that you enter for Universal ID must be unique when combined with the value that you enter in the Version field. The Version and Universal ID fields are validated according to OSGi conventions. For the Universal ID field, you can use alphanumeric characters, an underscore, a dash, or a period. A period is not allowed as the first character.

- d. Enter a brief description of the new virtual image.
- e. Enter the IP address of the virtual image that you are going to import into the tool.
- f. Enter the user ID to log in to the virtual image that you are importing.
- g. Enter the password for the user of the virtual image you are importing.
- h. Enter the password for the user again.
- i. Click **Create**.

**Import Images from Running VM**

Enter required fields

Name: RHEL62- 12G- ESX

Universal ID: com.ibm.itso.baseOS.rhel62.esx

Version: 1.0.0

Description: RHEL 6.2 BaseOS 64bit

IP Address: 192.168.10.75

User ID: root

Password: .....

Verify Password: .....

Create Cancel


Figure 8-10 Importing Image from Running VM

A new image is displayed in the user interface, with an *Importing* status. The tool connects to the virtual image and creates or edits the required properties, installs the IBM Virtual Solutions Activation Engine (VSAE) if required, and then resets and shuts down the virtual machine. The tool copies the virtual machine disk files into the /drouter directory on the Image Construction and Composition Tool system. When this process is finished, the status of the new image changes to *Completed*, at which time you can then extend the image. Upon completion of the import process, the virtual machine on the ESX hypervisor remains shut down. You can restart the virtual machine and use it independently of the Image Construction and Composition Tool.

## Importing from a template image

You can use an existing image from the Image Construction and Composition Tool as a template to create another image. You can select a virtual image as the template. It creates a virtual image with the same bundles as the template does. You then select the base virtual image to apply this template to. You can edit the bundles that are added before you synchronize.

To import a virtual image from a template, complete the following steps:


1. Make sure that your ESX provider is selected.
2. From the Welcome window, click **Build and manage images**.
3. Click the plus icon () to create an image.
4. Select **Create image from a template image**.
5. In the General details window, do the following tasks:
  - a. Enter a name for the virtual image that you are creating.
  - b. Enter a universal ID for the virtual image that you are creating. This field is required.
  - c. Enter a version number for the new virtual image that you are creating.
  - d. Enter a brief description of the new virtual image.
  - e. In the Template section of the wizard, select the image that you want to use as your template. To choose a template, select a virtual image that has the software products (bundles) that you want to include in the virtual image that you are creating.
  - f. Click **Next**.
6. Optional: Select a base virtual image from the completed virtual image on the cloud provider. If you select a base virtual image, it is the base image for the new virtual image that you are creating.



If you do not select a base virtual image, the tool checks all base virtual images of the target cloud provider that are compatible with the template virtual image.

If you do not select a base image, the tool searches for solutions that match the template virtual image that you selected. A solution consists of a base virtual image, plus any software bundles. After the search is completed, you see up to 10 of the closest matching solutions. The top ranking solution is selected by default. You can browse to any of the solution and select the one that best fits your needs.

Click **Next**.

7. Review the summary information, and then click **Done**.

The image is displayed in the user interface of the tool. The image status is out of sync.
8. Edit this virtual machine to add or remove bundles by clicking the **Edit** icon ()
9. Expand the **Software Bundles** section of the virtual image, and then click **Add bundle** to add additional bundles.


Alternatively, expand one of the bundles that is assigned to this image. Then click the **Delete** icon () to delete the bundle. After you finish editing, click the **Done editing** icon () to exit the edit mode.

10. Click the **Synchronize** icon () to import of the image into the tool.

## Importing from a cloud provider

You can import existing virtual images (appliances) from the ESX provider. Existing virtual images, such as those images that are created by another instance of the tool, can be easily imported into the Image Construction and Composition Tool.

To import an image from a cloud provider:

1. Select the ESX provider (upper right corner).
2. From the main menu of the Image Construction and Composition Tool, select **Images and bundles** → **Build and manage images**.
3. Click the **Import from a cloud provider** icon ().
4. In the Import Images from Cloud Provider window, from the list of images available to import on the left side, search for the images by name or keyword. Select the images that you want to add, and click **Add**.
5. Click **Import**.

### 8.4.3 Working with bundles

The concept of software bundles and the way that they can be created and maintained in the Image Construction and Composition Tool is not specific to the ESX provider. For information about how to create and manage bundles in the Image Construction and Composition Tool, see Chapter 5, “Product Activator Development Kit” on page 73. See also the “Working with IBM Image Construction and Composition Tool” topic in the IBM Workload Deployer Information Center at:

[http://pic.dhe.ibm.com/infocenter/worlodep/v3r1m0/index.jsp?topic=%2Fcom.ibm.worlodep.doc%2FICON%2Ftopics%2Fwd\\_cicn\\_overview.html](http://pic.dhe.ibm.com/infocenter/worlodep/v3r1m0/index.jsp?topic=%2Fcom.ibm.worlodep.doc%2FICON%2Ftopics%2Fwd_cicn_overview.html)


### 8.4.4 Extending, synchronizing, and capturing a virtual image

You can extend the base image by adding bundles, synchronize to install and configure the bundles, and then capture the image, which resets the VSAE.

#### Extending a virtual image

When you extend a virtual image, it is cloned so that you can reuse the original image without re-creating it from scratch each time.

To extend a base image, complete the following steps:

1. From the Welcome window, click **Build and manage images**.
2. On the left side, select the image you want to extend, and then click the **Extend** icon (.

3. In Extend Image window (Figure 8-11), notice that the fields are populated based on the base image that you selected. Change the following fields as needed:
  - a. Enter a name for the new virtual image. The name cannot exceed 50 characters.
  - b. Enter a unique identifier for the new virtual image.
  - c. Enter a unique version number for the new virtual image.
  - d. Optional: Enter a description for the new virtual image.
  - e. Click **Create** to extend the image.

**Extend Image**

The new image will be created by extending this one.

Name:

Universal ID:

Version:

Description:

Figure 8-11 Extending the base image

The image is displayed in the left pane with an *out-of-sync* status. You can now configure the extended image by adding some bundles.

## Adding bundles to a virtual image

To use a software bundle to add or configure a software component, use the following steps:

1. Click the **Edit** icon (✎) to start configuring the extended virtual image. You can change the name and description for the new virtual image in this window.
2. Click the **Software Bundles** section, and then click the **Add bundle** icon (+).
3. Select the software bundles that you want to add to this virtual image, and then click **add**. After you add more than one bundle to your virtual image, you can specify the order of installation by using the up (↑) and down (↓) arrows. You can change the order of only the bundles that are scheduled, but not of the bundles that are already installed.
4. Click a software bundle to see its parameters.

If the software bundle creator provided configuration options for the software bundle, the first set of parameters is displayed in the Install Parameters section. The second set of parameters defines the configuration options that are made available during the deployment of the virtual image.

5. Click **Install Parameters**, and review the default settings.

Decide whether you want to modify any of the default parameters. The installation parameters are used during virtual image synchronization to install the product in the software bundle. For example, if you are installing a DB2 product in the software bundle, you might specify parameters values for DB2 binary files or the location of a DB2 response file. If your parameters do not match your specific environment or requirements, errors might be in the virtual image and occur during run time.

6. Click **Deploy Parameters**, and review the default settings.

Decide whether you want to modify any of the default parameters. The deployment parameters define the default parameters values that are used during virtual image deployment when the select personality is deployed. These parameters are also used to customize the product for the environment for which they are being created. For example, an LDAP server host name and port number can be changed at deployment time. You can decide whether to make the deployment parameters configurable by clicking the corresponding **Lock** icon for each parameter.


7. When you are done with your changes, click the **Save** icon () to save the configuration.

As you edit the virtual image, missing or invalid input is highlighted and any validation messages are displayed in the validation status section at the bottom of the user interface. Click **Expand to** open the view. When you save the virtual image, semantic validation is performed to ensure that the virtual image definition is consistent. For example, if the specified software bundle order satisfies any dependency requirements. This validation is performed by the server and is done only when you save your changes. The validation report entries have three severities: information, warning, or error. If your virtual image has any error severity validation entries, any synchronize or capture action for the virtual image is likely to fail. Even if the synchronize or capture action succeeds, the resulting virtual image might still be unusable.


## Synchronizing a virtual image

Synchronizing is the process that runs the selected software bundles in the environment, by copying files and running scripts as defined in each software bundle that is assigned to the base image. The software bundles are run according to the order specified in the virtual image definition. To synchronize a virtual image, a bundle must be in a planned state. Any bundles that already run as part of a synchronization process are not run again.

To synchronize a virtual image, complete the following steps:

1. From the Welcome window, click **Build and manage images**.
2. Select the extended virtual image that you configured with the planned software bundles that you want to synchronize.
3. Click the **Synchronize** icon ()

**Software bundles:** If you installed software bundles in the virtual image, the Deployment Parameters wizard starts. The wizard prompts you to enter values for the deployment parameters, such as passwords. Default deployment values that were specified during software bundle creation are displayed. Change the deployment parameters as necessary.

4. Click the **Refresh** icon () to view the progress of the synchronization process.

When the synchronization process is complete, the image status changes to *Synchronized*.

## Validating a virtual appliance

To log in to the virtual image that you created and to validate that the image was created correctly, complete the following steps:

1. When the image has a *Synchronized* status, access the image to verify that all the bundles were installed and configured correctly.
2. Expand the **Virtual System** section of the image that you want to validate. The IP address and host name that were assigned to the image are displayed.



3. Use SSH to connect to the system:  


```
ssh user@ipaddress
```
4. When prompted, enter the password that you entered for the image during the synchronization process (see “Synchronizing a virtual image” on page 175).
5. After you log in to the system, verify that all the bundles were installed and configured. The steps to verify each software component and configuration depend on the specific application.

### Capturing a virtual image

You can capture a virtual image after it is in a *Synchronized* state. The capture process captures the updated disk from the deployed virtual image and creates a new, updated, physical copy of the virtual disk with the corresponding metadata. The Image Construction and Composition Tool stores this copy of the virtual image in the local repository.

To capture a virtual image, complete the following steps:


1. Clean the disk image before you make the capture. Whatever is on the hard disk when the capture is performed will be on the final virtual appliance.
  - a. Delete any files or directory that are not needed.
  - b. Unmount any disk that is used during the installation and configuration of the virtual image that will not be available after the capture is performed.
  - c. To help save space, the disks that are copied during a capture are compressed. To ensure efficient compression, fill the disk space with bytes that contain zeros as shown in the following example for Linux:

```
# cat /dev/zero > /tmp/zero.tmp
# sleep 5
# rm -rf /tmp/zero.tmp
```
2. Click the **Capture** icon () to start the capture process.

**Hint:** The Image Construction and Composition Tool might not capture images correctly for **vmrk** file names that have multilingual characters in them.

## 8.4.5 Exporting an image as an OVA archive

To export the virtual image as an OVA archive that then can be imported into a cloud, complete the following steps:

1. Click the **Export as OVA** icon () .
2. In the Export image as OVA window (Figure 8-12 on page 177), do the following tasks:
  - a. In the Remote host field, enter the IP address or host name of the remote host to copy the OVA to.
  - b. Enter the destination folder on the remote host.
  - c. In the File name field, leave the UUID name for the image.
  - d. Select the authentication method.
  - e. Enter a user name and password to access the remote host.
  - f. For OVA file format, select the correct OVA file format.
  - g. Click **OK**.

**Export image as OVA**

To what location should the image be exported?  
*The destination host must support Secure Copy (SCP)*

Remote host:

Destination folder:

File name:

**Authentication Method:**

☐ Private Key File ☒ Password

User name:

Password:

Verify password:

OVA file format:

OK Cancel

Figure 8-12 Export image as OVA

3. When you receive the export images status, click **Open the export status** to view the status. Click **OK**.
4. To verify that the OVA was exported, go to the directory on the remote host, and validate that the OVA is in the directory you specified.

## 8.5 Troubleshooting

If you encounter issues when using the Image Construction and Composition Tool, you can use several techniques to diagnose and eliminate them.

### 8.5.1 The Image Construction and Composition Tool version

When reporting issues to the IBM support, include the version of the Image Construction and Composition Tool. To find the version of the tool, in the upper-right corner, select **About**. A message box opens; it indicates the version information.

## 8.5.2 Image Construction and Composition Tool logs

Check the Image Construction and Composition Tool host system log files to determine whether problems occurred.

To access the Image Construction and Composition Tool log files, use the following steps:

1. From the Welcome window, click the **Download logs** link.
2. Save the compressed file.
3. Examine the log files.

The compressed file contains the following server log files:

- ▶ The `trace.log` file is the most recent trace file.
- ▶ The `error.log` file is the most recent error file.

You can also access the log files by clicking **Administer** → **Download logs**.

Alternately, you can check the logs directly on the system that is hosting the Image Construction and Composition Tool. They are in the following directory:

`/drouter/ramdisk2/mnt/raid-volume/raid0/logs`

## 8.5.3 Installation logs

For installation problems with the Image Construction and Composition Tool, check the following Installation Manager log files:

- ▶ `/var/ibm/InstallationManager/logs/<date-time>.xml`
- ▶ `/var/ibm/IntallationManager/logs/ant/<date-time>.xml`

The `date-time` value is the date and time in a format similar to 2012024\_1201.

## 8.5.4 Image synchronization logs

If the synchronization of virtual image fails you can check the following log files:

- ▶ `/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/<image uuid>/logs/err.out`
- ▶ `/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/<image uuid>/logs/out.out`

You can also access these logs by clicking **Select the image** → **Virtual System section** → **Download logs**.

## 8.5.5 Synchronization fails message: vmPath is null

Check the log files in the `/drouter/ramdisk2/mnt/raid-volume/raid0/logs` directory.

### **Symptom**

When you attempt to synchronize an image in the tool, the synchronization fails. The following error message is in the log file of the tool:

`Java.lang.IllegalArgumentException: Argument "vmPath" cannot be null`

### **Cause**

This issue is caused by the data store of the ESX provider running out of space.

### ***Solution***

To resolve the problem, clean up the cache on the ESX server:

1. Shut down the tool.
2. Log in to the ESX server.
3. Go to the data store and look for the `/vmfs/volumes/<datastore name>/cloudburst_cache/virtualimages` directory.
4. Delete directories that are more than a few days old.
5. Start the tool.
6. Restart the synchronization of the image that was failing.

If these steps do not solve your problem, another reason for this error is that the **mkisofs** tool is not installed. To install **mkisofs** tool on the Image Construction and Composition Tool server, enter the following command:

```
yum install mkisofs
```

## **8.5.6 Resolving common issues**

For information about how to resolve common issues with the Image Construction and Composition Tool and the ESX provider, see the “Working with IBM Image Construction and Composition Tool” topic in the IBM Workload Deployer information center:

[http://pic.dhe.ibm.com/infocenter/worlodep/v3r1m0/index.jsp?topic=%2Fcom.ibm.worlodep.doc%2FICON%2Ftopics%2Ftrcn\\_troubleshooting.html](http://pic.dhe.ibm.com/infocenter/worlodep/v3r1m0/index.jsp?topic=%2Fcom.ibm.worlodep.doc%2FICON%2Ftopics%2Ftrcn_troubleshooting.html)





## Constructing simple virtual appliances

This chapter explains how to create a simple virtual appliance. The example provided in this chapter shows how to create a base image, implement activation scripts, construct software bundles, and construct a virtual appliance.

This example includes usage of the following technologies:

- ▶ IBM Image Construction and Composition Tool (referred to as *the tool* in this book)
- ▶ IBM Virtual Solutions Activation Engine (VSAE)
- ▶ IBM Systems Director and Systems Director VMControl
- ▶ IBM WebSphere Application Server Community Edition

**Argument names:** This chapter uses the following abbreviations in the argument names:

- ▶ VMC for IBM Systems Director VMControl
- ▶ WASCE for WebSphere Application Server Community Edition

This chapter includes the following sections:

- ▶ Scenario overview
- ▶ Implementing the installation and configuration scripts
- ▶ Creating a base image
- ▶ Creating software bundles for the Image Construction and Composition Tool
- ▶ Building a virtual appliance
- ▶ Log file of the Image Construction and Composition Tool

## 9.1 Scenario overview

The scenario in this chapter simulates the production environments of most small and medium businesses (SMB) that consist of a one-tier web application server topology. The single virtual machine includes the operating system, application server middleware, and the business solution. This extends the base AIX 7.1 image with WebSphere Application Server Community Edition 3.0 and the PlantsByWebSphere application. WebSphere Application Server Community Edition is based on Apache Geronimo 3.0. Apache Geronimo integrates Apache Karaf 2.2.1, Apache Aries 0.3, Apache Tomcat 7.0.19.1, Apache OpenEJB 4.0, Apache OpenWebBeans 1.1.1, and other open source components.

This example demonstrates the following tasks:

- ▶ Implementing the installation and configuration scripts, which includes the following tasks:
  - Analyzing the installation and configuration operations for WebSphere Application Server Community Edition and PlantsByWebSphere application.
  - Creating the installation and configuration scripts for WebSphere Application Server Community Edition and the creating the configuration script for the PlantsByWebSphere application.
- ▶ Creating a base image (AIX 7.1)
- ▶ Creating software bundles for the Image Construction and Composition Tool
- ▶ Building a virtual appliance, which includes the following tasks:
  - Extending the base image to create the sample image
  - Adding software bundles to sample image
  - Synchronizing the sample image
  - Validating the sample virtual image
  - Capturing the sample virtual appliance
  - Exporting the sample virtual appliance
  - Verifying the results

## 9.2 Implementing the installation and configuration scripts

When creating a new virtual appliance, you need to analyze which steps can be performed when the appliance is being built and which steps must be performed on the customer system. If such information as the host name, IP address, and port numbers are needed, that customization must be done when the virtual appliance is deployed on the customer system. If products can be extracted, installed, or configured without awareness by the customer system, the customization can be done during build time.

The installation scripts run during the synchronization process in the example build environment. The configuration scripts run at deployment time on the customer system. Both the installation and configuration scripts can accept parameters, of which some or all can have default values.

In this example, you develop the installation and configuration scripts for IBM WebSphere Application Server Community Edition and the sample web application.

## 9.2.1 Developing the installation script for IBM WebSphere Application Server Community Edition

First, you develop a script to install WebSphere Application Server Community Edition. Although this example does not explain how to install WebSphere Application Server Community Edition, it shows the approach for creating the script to install it.

For a list of system requirements, see the “Detailed system requirements for WebSphere Application Server Community Edition” topic:

<http://www.ibm.com/support/docview.wss?rs=2359&uid=swg27006834>

When determining the approach to create the installation script, you must determine the steps to do the installation in a repeatable, automated (without manual intervention) manner or rather, as a silent installation. One way to determine the steps is to install manually and make note of the tasks so that they can be scripted. When approaching installation script creation in this manner, consider the following key items:

- ▶ Dependencies on other software; for example, are any other third-party software components or utilities required for the installation?
- ▶ How software and supporting files are included; for example, are they part of the software bundle or are they available on a remote server?
- ▶ Parameters for the installation; for example, are any arguments required for installation and how are they passed?

Other considerations might exist for installation scripts that you develop in the future. They are typically uncovered when you go through the manual installation process.

For the WebSphere Application Server Community Edition installation script, the main assets are as follows:

- ▶ WebSphere Application Server Community Edition installation package
- ▶ The **sed** and **unzip** utilities
- ▶ Installation script

To develop the installation script, complete the following steps:

1. Download the installation package for WebSphere Application Server Community Edition from IBM developerWorks:

<https://www.ibm.com/developerworks/downloads/ws/wasce/>

You can also download the **sed** and **unzip** utilities can from the IBM AIX Toolbox for Linux Applications:

<http://www.ibm.com/systems/power/software/aix/linux/toolbox/download.html>

2. Determine how to include the assets, which you can do in several ways, depending on your environment setup. You can take advantage of an existing repository if you can access it as a remote share. In this example, the packages and utilities are included in the software bundle, which provides them locally when the synchronization of the image occurs. Including the packages and utilities in the software bundle eliminates the need for creating additional scripting to connect and obtain the assets from a remote source.



3. Define the arguments for the installation. Based on the requirements of the WebSphere Application Server Community Edition installation, the example defines the following arguments:

|                            |   |
|----------------------------|---|
| <b>WASCE_INSTALL_PATH</b>  | The location to install WebSphere Application Server Community Edition.         |
| <b>INSTALLED_JAVA_HOME</b> | The location of JAVA_HOME.  |
| <b>WASCE_INSTALL_EXEC</b>  | The installation executable for WebSphere Application Server Community Edition. |
| <b>WASCE_ZIP_NAME</b>      | The installation package for WebSphere Application Server Community Edition.    |

When the script runs, it allows the passing of the following arguments:

```
install.sh -wasce_install_path <WASCE_INSTALL_PATH> -installed_java_home  
<INSTALLED_JAVA_HOME> -wasce_install_exec <WASCE_INSTALL_EXEC> -wasce_zip_name  
<WASCE_ZIP_NAME>
```

4. When you create the software bundle in the Image Construction and Composition Tool, choose a parameter style. Example 9-1 shows one way that you can achieve parsing arguments by using the Short Space parameter style input. The style that you choose in the tool must correspond with how you implement the script.

*Example 9-1 Installation script arguments in short-space style*

---

```
#Read arguments  
while [ $# -ne 0 ]  
do  
    case $1 in  
        -wasce_install_path*)  
            WASCE_INSTALL_PATH=$2  
            ;;  
        -installed_java_home*)  
            INSTALLED_JAVA_HOME=$2  
            ;;  
        -wasce_install_exec*)  
            WASCE_INSTALL_EXEC=$2  
            ;;  
        -wasce_zip_name*)  
            WASCE_ZIP_NAME=$2  
            ;;  
        *)  
            ;;  
    esac  
    shift 1  
done
```

---

5. Create a temporary directory. When working with archive or compress files, extract the files to a temporary location. For example, run the following command to create a directory:

```
mkdir -p /tmp/wasce
```

The directory is used to extract the installation package for WebSphere Application Server Community Edition and run the installation.

6. Extract the installation package of WebSphere Application Server Community Edition with the appropriate utility. In this example, the **unzip** and **sed** utilities were installed. The **unzip** utility is not installed by default in AIX, but it is required. The **sed** utility is available in AIX. However, we decided to install the GNU **sed** package to take advantage of additional capability that is used for the WebSphere Application Server Community Edition configuration as shown in the following example:

```
rpm -Uvh unzip-5.51-1.aix5.1.ppc.rpm
rpm -Uvh sed-4.1.1-1.aix5.1.ppc.rpm
```

Extract the installation package of WebSphere Application Server Community Edition by using the parameter name `$WASCE_ZIP_NAME` as shown in the following example. The parameter and value of `$WASCE_ZIP_NAME` are set in the software bundle of the tool and are passed during the synchronization phase.

```
unzip $WASCE_ZIP_NAME -d /tmp/wasce
```

7. Install WebSphere Application Server Community Edition as shown in the following example. Run the installation silently, specifying the installation location parameter of WebSphere Application Server Community Edition. These parameter values are set in the software bundle of Image Construction and Composition Tool and passed during the synchronization phase.

```
./$WASCE_INSTALL_EXEC -i silent -DUSER_INSTALL_DIR=$WASCE_INSTALL_PATH
```

You now have a script that silently installs WebSphere Application Server Community Edition. Example B-1 on page 284 shows a sample of this script.

## 9.2.2 Developing the configuration script for IBM WebSphere Application Server Community Edition

In this section, you develop a script to configure WebSphere Application Server Community Edition. This configuration occurs when the image is deployed. Because WebSphere Application Server Community Edition is already installed, we address the reconfiguration of the software when it is deployed on a new virtual server. Such items as the host name, user name, and passwords change from deployment to deployment. This section addresses these items in the development of this script. As in the previous section, the approach for script development favors a method in which it can run and complete without manual intervention.

To develop the configuration script, complete the following steps:

1. Determine the configurable resources to reconfigure WebSphere Application Server Community Edition. They are not the same as the parameters that were used for installing WebSphere Application Server Community Edition. Configurable resources focus on the activation of the software in a new environment. For the configurable resources that require user input, pass the following parameters to the script:

|                             |  |
|-----------------------------|--|
| <b>num_servers</b>          | The number of instances you want to deploy.  |
| <b>HOSTNAME</b>             | The new host name.   |
| <b>WASCE_HOME</b>           | The location of where WebSphere Application Server Community Edition is installed. |
| <b>WASCE_ADMIN_USER</b>     | The administrative user name for WebSphere Application Server Community Edition.   |
| <b>WASCE_ADMIN_PASSWORD</b> | The administrative password for WebSphere Application Server Community Edition.    |

This example uses the short-space style to accept the arguments in the script (see Example 9-2).

---

*Example 9-2 Script arguments in short-space style*

---

```
while [ $# -ne 0 ]
do
    case $1 in
        -num_servers*)
            num_servers=$2
            ;;
        -WASCE_HOME*)
            WASCE_HOME=$2
            ;;
        -WASCE_ADMIN_USER*)
            WASCE_ADMIN_USER=$2
            ;;
        -WASCE_ADMIN_PASSWORD*)
            WASCE_ADMIN_PASSWORD=$2
            ;;
        *)
            ;;
    esac
    shift 1; shift 1
done
```

---

Notice that we do not include the host name as a script argument in Example 9-2. The reason is that we define the host name based on the new host name of the new virtual server instance.

2. Using the **hostname** command, define the **HOSTNAME** variable as follows:

```
HOSTNAME='hostname'
```

3. In preparation for additional applications to install on WebSphere Application Server Community Edition, make the **WASCE\_HOME**, **WASCE\_ADMIN\_USER**, and **WASCE\_ADMIN\_PASSWORD** parameters and values available (see Example 9-3). By design, all instances that are created with the Image Construction and Composition Tool contain a file with environment variables. This file is in the `/etc/virtualimage.properties` directory.

---

*Example 9-3 Defining environment variables based on parameter values*

---

```
# provide variables to be consumed by other applications
echo "WASCE_HOME=$WASCE_HOME" > /etc/virtualimage.properties
echo "WASCE_ADMIN_USER=$WASCE_ADMIN_USER" >> /etc/virtualimage.properties
echo "WASCE_ADMIN_PASSWORD=$WASCE_ADMIN_PASSWORD" >>
/etc/virtualimage.properties
```

---

4. After you have the values of arguments, customize the instance with them. We use the GNU **sed** utility to search and replace current values in a file with the new argument values (Example 9-4 on page 187):
  - a. Edit the `$WASCE_HOME/var/config/config-substitutions.properties` file. When WebSphere Application Server Community Edition starts, it reads the `$WASCE_HOME/var/config/config.xml` configuration file and applies the `$WASCE_HOME/var/config/config-substitutions.properties` file to it. We use this approach to apply the new **\$HOSTNAME** to the configuration during the deployment.

- b. Edit the `WASCE_HOME/var/security/users.properties` file. This file is used to define the administration user and password by replacing system and master with parameter values `$WASCE_ADMIN_USER` and `$WASCE_ADMIN_PASSWORD`.

The `$WASCE_HOME/var/security/groups.properties` file defines which user is the administrator. In this example, we replace the system with user `$WASCE_ADMIN_USER`.

---

*Example 9-4 Search and replace to provide new parameter values for reconfiguration*

---

```
/opt/freeware/bin/sed -i s/"ServerHostname = 0.0.0.0"/"ServerHostname = $HOSTNAME"/g
$WASCE_HOME/var/config/config-substitutions.properties
/opt/freeware/bin/sed -i s/"RemoteDeployHostname = localhost"/"RemoteDeployHostname =
$HOSTNAME"/g $WASCE_HOME/var/config/config-substitutions.properties
/opt/freeware/bin/sed -i s/"system=master"/"$WASCE_ADMIN_USER=$WASCE_ADMIN_PASSWORD"/g
$WASCE_HOME/var/security/users.properties
/opt/freeware/bin/sed -i s/"admin=system"/"admin=$WASCE_ADMIN_USER"/g
$WASCE_HOME/var/security/groups.properties
```

---

5. Start WebSphere Application Server Community Edition:

`$WASCE_HOME/bin/startup.sh`

If additional instances (`num_servers`) are defined as part of the deployment, the script creates them. The code in Example 9-5 checks how many instances were defined. For each instance, it creates a `$WASCE_HOME/$instName` directory and copies the `$WASCE_HOME/var` configuration directory to the directory of the new instance.

The script defines a unique port for the instance accomplished by the `PortOffset` definition in the `$WASCE_HOME/$instName/var/config/config-substitutions.properties` file. The actual value is generated by `y=$svr_ct-1`.

6. Before starting the server, ensure that the `org.apache.geronimo.server.name` system property is set before you start the server. Use the following syntax for an instance named `$instName` in the `$WASCE_HOME/$instName` directory:

`-Dorg.apache.geronimo.server.name=$instName`

Adding this property to the `GERONIMO_OPTS` environment variable makes it immediately available for the start of the new server instance (Example 9-5).

---

*Example 9-5 Creating a WebSphere Application Server Community Edition instance*

---

```
svr_ct=1
while [ $svr_ct -lt $num_servers ]
do
    let svr_ct++
    instName="instance$svr_ct"
    echo "Creating $instName instance" >> $logfile
    mkdir $WASCE_HOME/$instName
    cp -r $WASCE_HOME/var $WASCE_HOME/$instName
    let y=$svr_ct-1
    /opt/freeware/bin/sed -i s/"PortOffset = 0"/"PortOffset = $y"/g
    $WASCE_HOME/$instName/var/config/config-substitutions.properties
    GERONIMO_OPTS=-Dorg.apache.geronimo.server.name=$instName
    export GERONIMO_OPTS=$GERONIMO_OPTS
    $WASCE_HOME/bin/geronimo.sh start
    echo "Started $instName instance" >> $logfile
done
```

---

The script to configure WebSphere Application Server Community Edition is now developed. For an example of this script, see Example B-2 on page 286.

## 9.2.3 Developing a configuration script for the web application

**Web application installation script:** The example in this chapter does not include an installation script for the web application. The PlantsByWebSphere application is provided as an enterprise archive (EAR) file and is packaged as part of the configuration operation of the software bundle defined by the Image Construction and Composition Tool. When defining software bundles, you are required to have at least one operation.

In this section, you develop a script to configure the PlantsByWebSphere sample application. This configuration occurs when the image is deployed after the configuration of WebSphere Application Server Community Edition. The tasks involve creating the data source object and deploying the EAR file.

To develop the configuration script, complete the following steps:

1. Obtain the assets for this sample application by going to the WebSphere Application Server Community Edition download site:

<http://publib.boulder.ibm.com/wasce/V3.0.0/en/samples.html>

2. After you download the assets, extract the wasce\_samples\_3.0.0.3.zip file.

This compressed file contains that the WASCE-sample-datasource and PlantsByWebSphere samples that you work with. The assets that you need are two XML deployment plans: one for the WASCE-sample-datasource, and one for PlantsByWebSphere and PlantsByWebSphere EAR file. These assets are in the following files:

- <extracted\_directory>/applications/javaee5/WASCE-sample-datasource/plan.xml
- <extracted\_directory>/applications/javaee5/PlantsByWebSphere/plans/pbw-plan.xml
- <extracted\_directory>/applications/javaee5/PlantsByWebSphere/PlantsByWebSphereEAR/target/PlantsByWebSphereEAR-3.0.0.3.ear

Now that you have these assets, you can begin work on the script.

3. Enter the following command to source these environment variables in the /etc/virtualimage.properties file (done previously) to use them in the script:  

```
. /etc/virtualimage.properties
```
4. Create the data source to the embedded Derby database that is included with WebSphere Application Server Community Edition to reference the deployment plan, the plan.xml file, from WASCE-datasource-sample:

```
${WASCE_HOME}/bin/deploy.sh --host $HOSTNAME --user $WASCE_ADMIN_USER  
--password $WASCE_ADMIN_PASSWORD deploy  
${WASCE_HOME}/repository/org/tranql/tranql-connector-derby-embed-xa/1.7/tranql-  
connector-derby-embed-xa-1.7.rar plan.xml
```

5. Deploy the PlantsByWebSphere EAR file to reference the EAR file and deployment plan, pbw-plan.xml assets:

```
${WASCE_HOME}/bin/deploy.sh --host $HOSTNAME --user $WASCE_ADMIN_USER  
--password $WASCE_ADMIN_PASSWORD deploy PlantsByWebSphereEAR-3.0.0.3.ear  
pbw-plan.xml
```

When this script is activated at deployment, you have a configured data source and the PlantsByWebSphere application is running.


The script to configure the web application is now developed. For an example of this script, see Example B-3 on page 287.

You are now ready to package all your scripts into a software bundle.

## 9.3 Creating a base image

The first step in the process of creating a virtual image is to create a base operating system image. This example creates an AIX 7.1 logical partition (for example, helium85) and allocates 4 GB of memory and 20 GB of disk to it. Also, the virtual machine remains up and running to use the Import Running Virtual Machine function of the Image Construction and Composition Tool to add support for the IBM Virtual Solutions Activation Engine (VSAE).

This example follows the process to import from a running virtual machine as explained in “Importing from a running virtual machine” on page 137 to create the base image. To import an image from a running virtual machine, use the following steps:

1. In the Welcome window of the Image Construction and Composition Tool, click **Build and manage images**.
2. Click the **plus sign** icon () to create an image.
3. In the Import Images from Running VM window, enter the following information:
  - a. Enter a name. In this example, we enter baseOS-aix7-20GB-SCS.
  - b. Enter the universal ID. In this example, we enter com.ibm.itso.baseOS.aix7.
  - c. Enter the version, which is 1.0.0 in this example.
  - d. Enter a description. In this example, we enter Base AIX virtual image.
  - e. Select the virtual machine to capture, which is helium85 in this example.
  - f. In the Repository for capture field, select in which repository to place the imported image.
  - g. Enter the IP address, which is 10.5.169.178 in this example.
  - h. Enter the user ID, which is root in this example.
  - i. Enter the password, which is passw0rd in this example.
  - j. Verify the password.


During the import process, the Image Construction and Composition Tool injects the VSAE into the image automatically so that it is available to software bundle configuration scripts.

## 9.4 Creating software bundles for the Image Construction and Composition Tool

Now you create a software bundle for IBM WebSphere Application Server Community Edition and a second software bundle for the web application.

### 9.4.1 Creating a software bundle for IBM WebSphere Application Server Community Edition

To create a software bundle, complete the following steps:

1. In the Welcome window for the Image Construction and Composition Tool, click **Build and manage software bundles**.
2. Click the **plus** icon () to create a bundle.

3. In the Create Bundle window, do the following actions:
  - a. Enter a name, which is IBM WebSphere Application Server Community Edition in this example.
  - b. Enter the universal ID, which is com.ibm.itso.wasce.aix.ppc in this example.
  - c. Enter the version, which is 1.0.0 in this example.
  - d. Enter a description. In this example, we enter: This bundle installs WebSphere Application Server Community Edition 3.0.0.3 or later versions that follow the same installation process.
  - e. For Storage Location, select **Local**.
  - f. Leave the Community field blank.
  - g. Leave the **Uses IBM Installation Manager** check box cleared.
4. In the left pane, select the newly created bundle. Then in the General Properties window (Figure 9-1), on the **General** tab for the software bundle, at the bottom, for Products in the bundle:
  - a. Enter the product name, which is WebSphere Application Server Community Edition in this example.
  - b. Enter the version, which is 3.0.0.3 in this example.
  - c. Enter the vendor, which is IBM.

The screenshot shows the IBM Image Construction and Composition Tool interface. The main window is titled "IBM Image Construction and Composition Tool" and includes a navigation bar with "Welcome", "Images and bundles", and "Administer". The "Cloud Provider" is set to "PowerVM Provider on helium55". The left pane shows a list of bundles, with "IBM WebSphere Application ..." selected. The right pane displays the "General Properties" tab for this bundle. The fields are filled with the following information:

- Name:** IBM WebSphere Application Server Community Edition
- Repository:** Local
- Description:** This bundle installs Web Sphere Application Server Community Edition 3.0.0.1 or higher versions that follow the same install process.
- Universal ID:** com.ibm.itso.wasce.aix.ppc
- Version:** 1.0.0
- Publisher:** IBM
- Created on:** March 8, 2012 6:43:50 PM Central Standard Time
- Updated date:** June 29, 2012 9:31:08 AM Central Daylight Time

At the bottom, the "Products in the bundle:" section shows a table with one product entry:

| Product Name                                    | Version | Vendor |
|---|---------|--------|
| Web Sphere Application Server Community Edition | 3.0.0.3 | IBM    |

Figure 9-1 General tab for the IBM WebSphere Application Server Community Edition software bundle

5. On the **Requirements** tab, under Supported Operating Systems (Figure 9-2), supply the following information:
  - a. For Type, select **AIX**.
  - b. For Distribution, select **AIX Server**.
  - c. For Architecture, select **POWER System**.
  - d. In the Version field, to support all levels of AIX, leave this field blank.

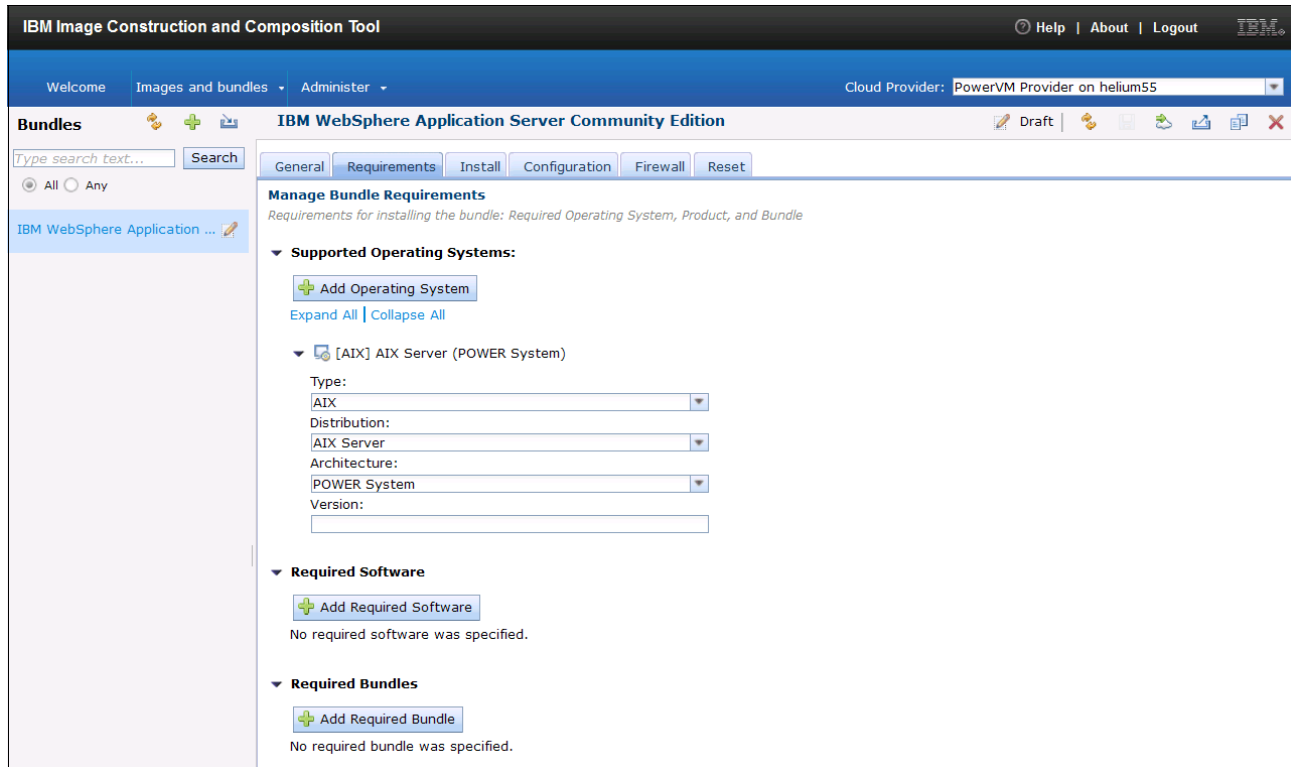


Figure 9-2 Requirements tab for the IBM WebSphere Application Server Community Edition software bundle

6. On the **Install** tab (Figure 9-3 on page 192), do the following tasks:
  - a. Under Files to Copy, click the plus icon (+) to add each of the following files. Select the **Executable** check box for each one:
    - install.sh
    - wasce\_ibm60sdk\_setup-3.0.0.3-ppc64aix.zip
    - sed-4.1.1-1.aix5.1.ppc.rpm
    - unzip-5.51-1.aix5.1.ppc.rpm
  - b. Under Command, complete the following steps:
    - i. For Run Command, select **install.sh**.
    - ii. For Run As, enter root.
    - iii. For Parameter Style, select **Short Space Style**.
    - iv. For Arguments, click the plus icon (+) for each of the arguments, as listed in Table 9-1 on page 192.



Table 9-1 WebSphere Application Server Community Edition Install Operation Arguments

| Name                | Label                    | Value  |
|---------------------|--------------------------|--|
| wasce_install_path  | WASCE Install Path       | /opt/IBM/WebSphere/AppServerCommunityEdition |
| installed_java_home | JAVA_HOME                | /usr/java6                                   |
| wasce_install_exec  | WASCE Install Executable | wasce_setup-3.0.0.3-unix.bin                 |
| wasce_zip_name      | WASCE Install Zip        | wasce_ibm60sdk_setup-3.0.0.3-ppc64aix.zip    |

**IBM Image Construction and Composition Tool**

Welcome | Images and bundles | Administrator | Cloud Provider: PowerVM Provider on helium55

**Bundles**

Type search text... Search

All Any

IBM WebSphere Application ...

**IBM WebSphere Application Server Community Edition**

Draft

General Requirements Install Configuration Firewall Reset

**Install operation configuration**

Define how to install the bundle

**Files to Copy**

Files to be copied to the target machine:

| Source (URI or file name)                 | Executable                          |
|---|-------------------------------------|
| install.sh                                | <input checked="" type="checkbox"/> |
| wasce_ibm60sdk_setup-3.0.0.1-ppc64aix.zip | <input checked="" type="checkbox"/> |
| sed-4.1.1-1.aix5.1.ppc.rpm                | <input checked="" type="checkbox"/> |
| unzip-5.51-1.aix5.1.ppc.rpm               | <input checked="" type="checkbox"/> |

**Command**

Run Command: install.sh

Select an executable script to run

Hide Preview

```
install.sh -wasce_install_path ${WASCE Install Path} -installed_java_home ${JAVA_HOME} -wasce_install_exec ${WASCE Install Executable} -wasce_zip_name ${WASCE Installation Package}
```

Run As: root

Parameter Style: Short Space Style

**Arguments:**

| Name                | Label                    | Value  | Hide Value               |
|---------------------|--------------------------|--|--------------------------|
| wasce_install_path  | WASCE Install Path       | /opt/IBM/WebSphere/AppServerCommunityEdition | <input type="checkbox"/> |
| installed_java_home | JAVA_HOME                | /usr/java6                                   | <input type="checkbox"/> |
| wasce_install_exec  | WASCE Install Executable | wasce_setup-3.0.0.1-unix.bin                 | <input type="checkbox"/> |

Figure 9-3 IBM WebSphere Application Server Community Edition software bundle Install tab

7. On the Configuration tab (Figure 9-4 on page 194), click the plus icon (+) to add an operation, and then enter the following information:
  - a. For Operation name, enter ConfigWASCE.
  - b. Under Files to Copy, click the plus icon (+) to add the configuration file, and then select the **Executable** check box for the ConfigWASCE.sh file.
  - c. Under Command, do the following steps:
    - i. For Run Command, select **ConfigWASCE.sh**.
    - ii. For Run As, enter root.
    - iii. For Parameter Style, select **Short Space Style**.
    - iv. For Arguments, click the plus icon (+) for each of the arguments in Table 9-2.

Table 9-2 WebSphere Application Server Community Edition Install operation arguments

| Name                 | Label                | Value  |
|----------------------|----------------------|--|
| num_servers          | Number of Servers    | 1  |
| WASCE_HOME           | WAS CE Home          | /opt/IBM/WebSphere/AppServerCommunityEdition |
| WASCE_ADMIN_USER     | WASCE Admin User     | system                                       |
| WASCE_ADMIN_PASSWORD | WASCE Admin Password | manager                                      |

- d. Keep the Dependencies section as it is, because no dependencies exist for this software.

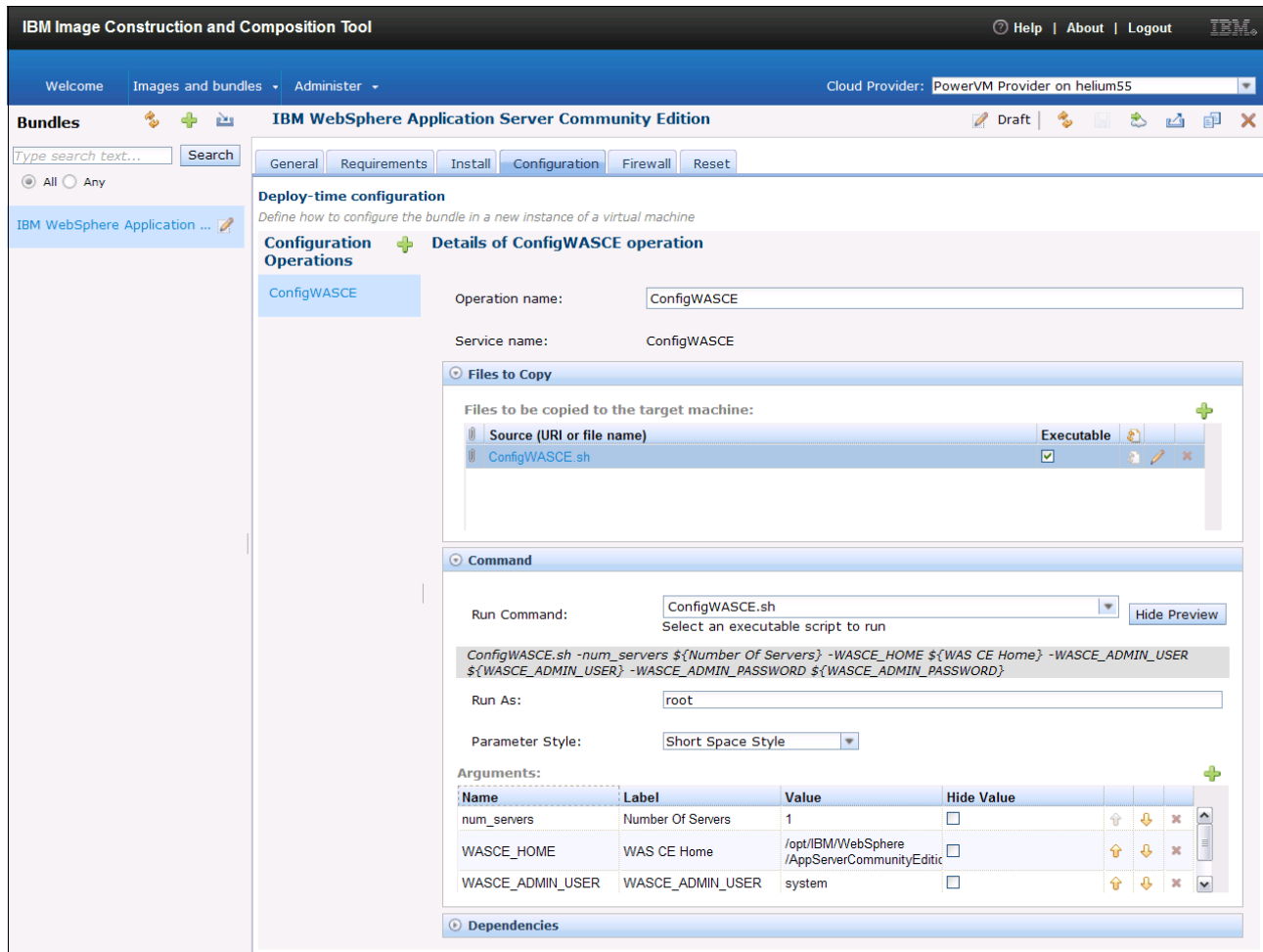


Figure 9-4 Configuration tab of the IBM WebSphere Application Server Community Edition software bundle

In this example, no special firewall ports need to be modified nor reset scripts that need to be called. Therefore, you do not need to modify any of the fields on the Firewall or Reset tabs.

8. Click the **Save** icon (💾).

## 9.4.2 Creating a software bundle for a web application

To create a software bundle for a web application, complete the following steps:

1. In the Welcome window of the Image Construction and Composition Tool, click **Build and manage software bundles**.
2. Click the icon (⊕) to create a bundle.
3. In the Create Bundle window, supply the following information:
  - a. For Name, enter Plants By WebSphere Application.
  - b. For Universal ID, enter com.ibm.itso.plants.wasce.bundle.
  - c. For Version, enter 1.0.0.

- d. For Description, enter Creates the data source and deployed the PlantsByWebSphere EAR file.
  - e. For Storage Location, select **Local**.
  - f. Leave the Community field blank.
  - g. Do not select the **Uses IBM Installation Manager** check box.
4. In the left pane, select the newly created bundle. In the General Properties window (Figure 9-5) for the software bundle, under Products in the bundle:
    - a. For Product Name, enter PlantsByWebSphere.
    - b. For Version, enter 1.0.
    - c. For Vendor, select **IBM**.

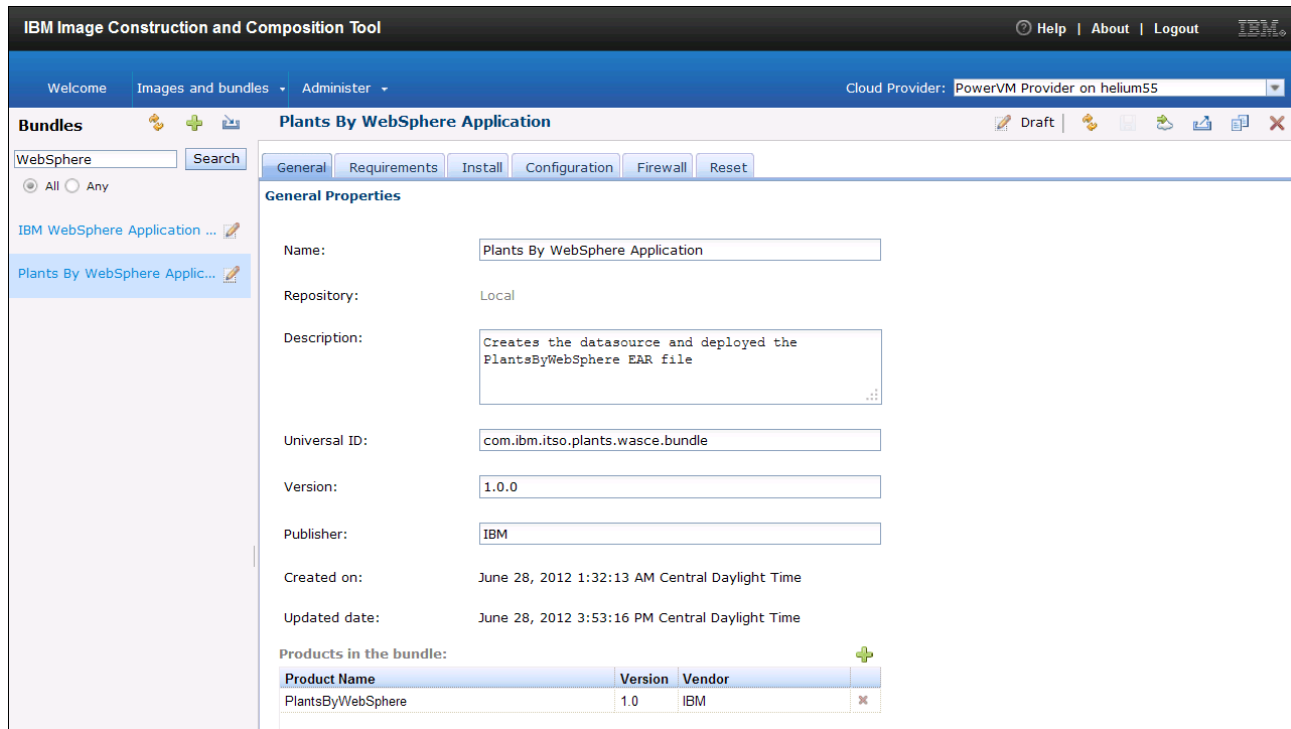


Figure 9-5 General tab of the Plants By WebSphere Application software bundle

5. On the **Requirements** tab, under Supported Operating Systems, enter the following information:
  - a. For Type, select **AIX**.
  - b. For Distribution, select **AIX Server**.
  - c. For Architecture, select **POWER System**.
  - d. For Version, to support all levels of AIX, leave this field blank.
6. Under Required Bundles, select **Add Required Bundle**.
7. In the Add requirement to bundle window (Figure 9-6 on page 196), select **IBM WebSphere Application Server Community Edition**, and then click **Add** to include this bundle as a requirement.

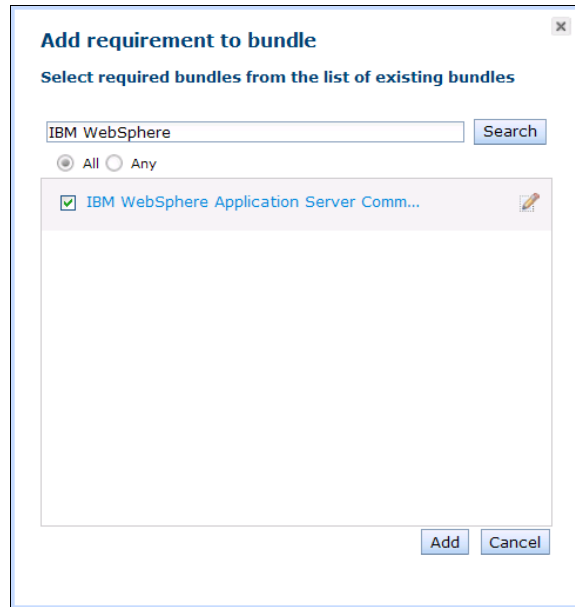


Figure 9-6 Add requirement to bundle window

Figure 9-7 shows the bundle as an added requirement under Required Bundles.

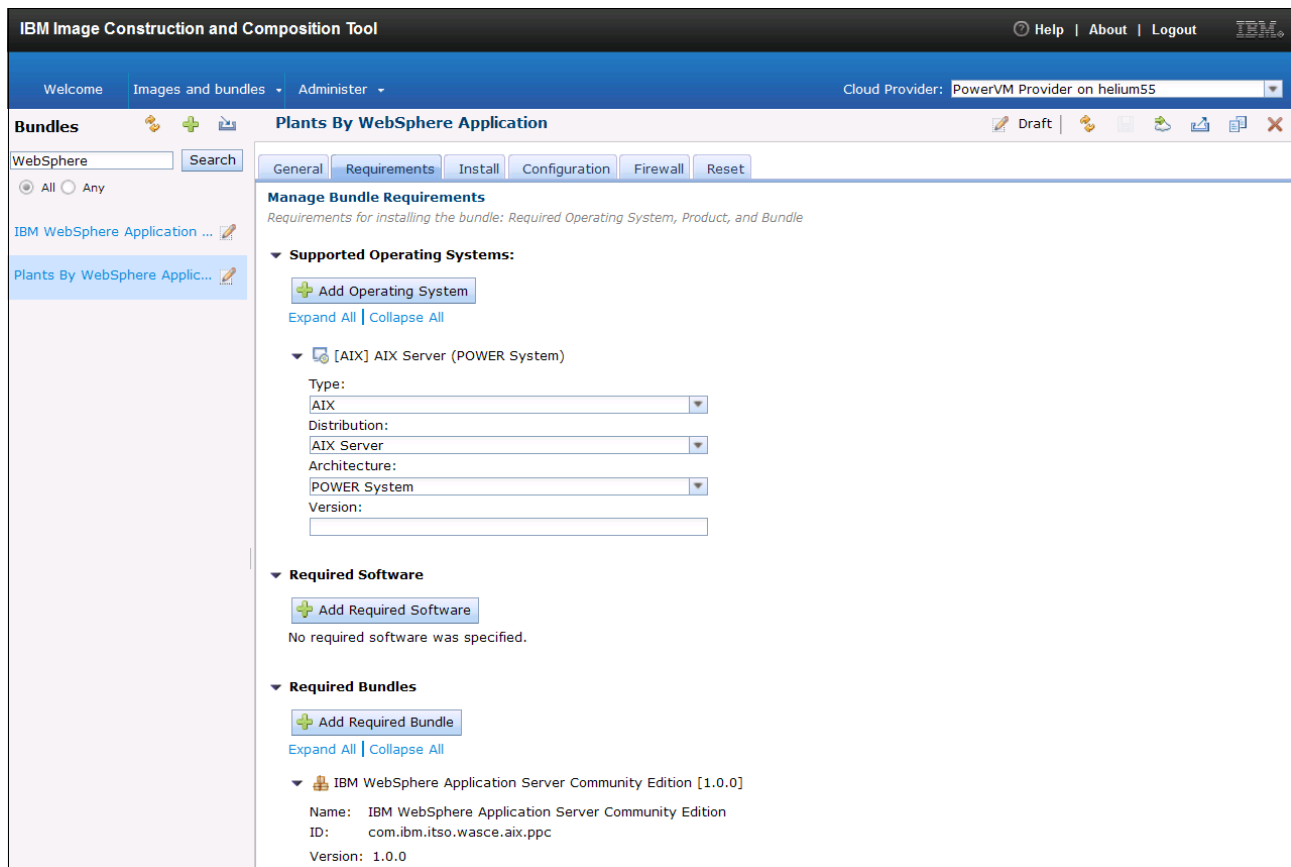


Figure 9-7 Requirements tab of the Plants By WebSphere Application software bundle

Because the application installation is done in the configuration script, you do not need to add anything on the **Install** tab.

8. On the **Configuration** tab (Figure 9-8), click the plus icon (+) to add an operation:
  - a. For Operation name, enter ConfigApp.
  - b. Under Files to Copy, click the plus icon (+) to add the following configuration files. Select the **Executable** check box for the activate.sh file:
    - PlantsByWebSphereEAR-3.0.0.3.ear
    - pbw-plan.xml
    - plan.xml
    - activate.sh
  - c. Under Command, do the following steps:
    - i. For Run Command, select **activate.sh**.
    - ii. For Run As, enter root.
    - iii. For Parameter Style, select **Short Space Style**.

No arguments exist for this application.
  - d. Under Dependencies, click the plus icon (+) to add each service.
    - ConfigWASCE

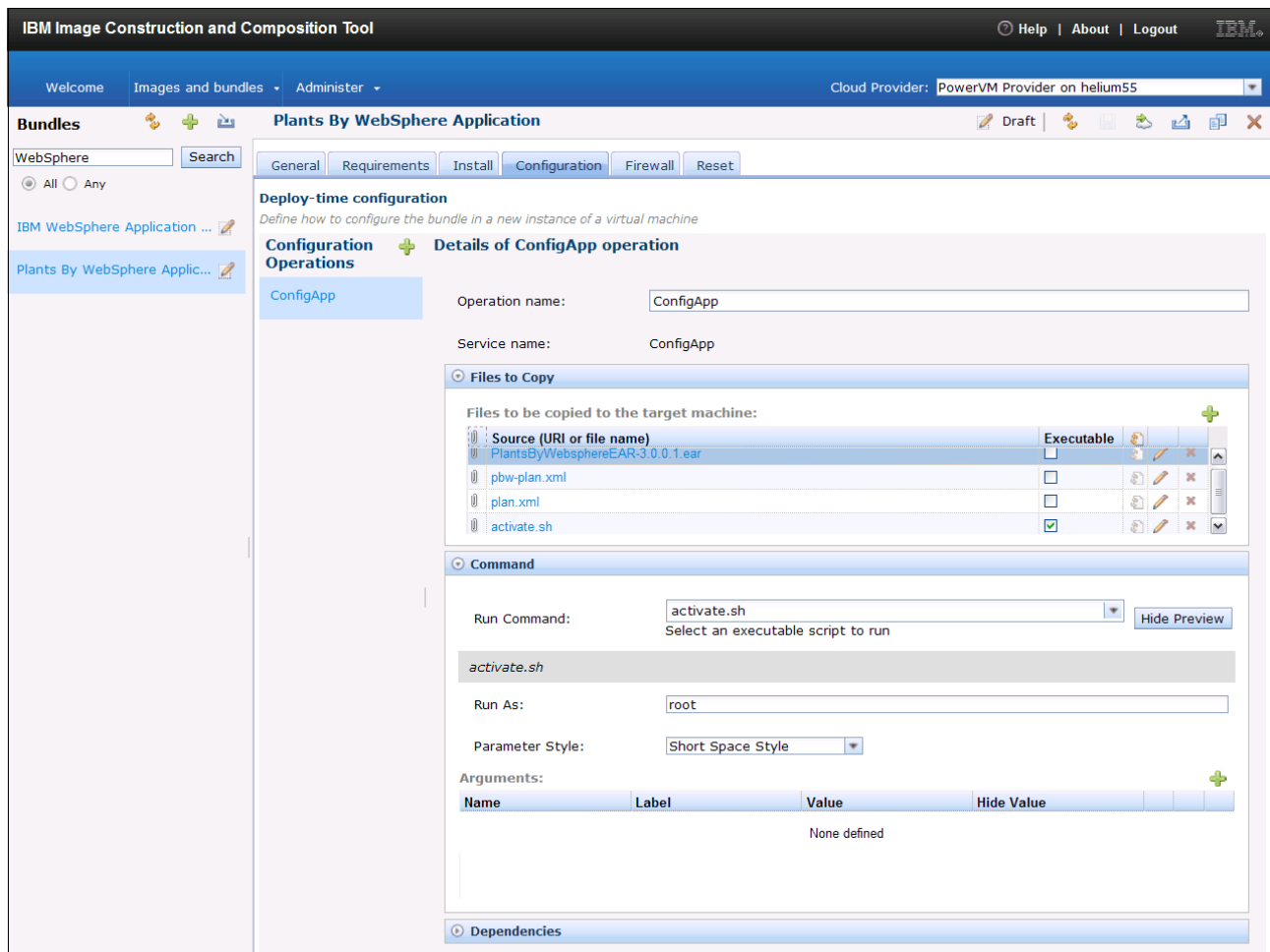


Figure 9-8 Configuration tab for the Plants By WebSphere Application software bundle

In this example, you do not need to modify any special firewall ports or to reset scripts to call. Therefore, no updates are necessary to any of the fields on the **Firewall** or **Reset** tabs.

9. Click the **Save** icon (💾).

## 9.5 Building a virtual appliance

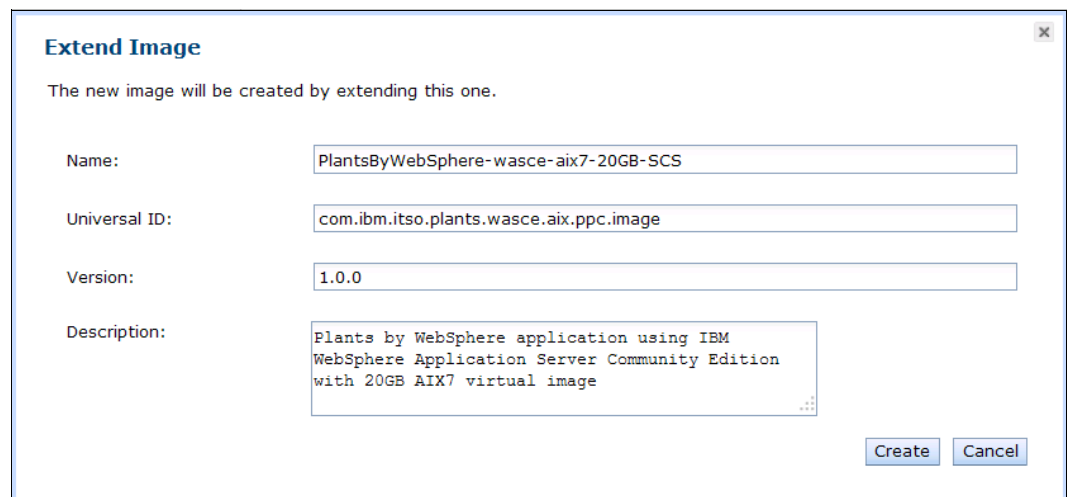
Continuing with the previous example, you now extend the base image, add the two software bundles, synchronize the image, and capture the virtual appliance.

### 9.5.1 Extending the base image to create the sample image

When you extend a virtual image, it is cloned so that you can reuse the original image without having to creating it all over again each time.

To create an image by extending an imported image, complete the following steps:

1. In the Welcome window of the Image Construction and Composition Tool, click **Build and manage images**.
2. On the left side, select the **baseOS-aix7-20GB-SCS** image, and then click the **Extend** icon (🔗).
3. In the Extend Image window (Figure 9-9), do the following tasks:
  - a. For Name, enter `PlantsByWebSphere-wasce-aix7-20GB-SCS`.
  - b. For Universal ID, enter `com.ibm.itso.plants.wasce.aix.ppc.image`.
  - c. For Version, enter `1.0.0`.
  - d. For Description, enter `Plants by WebSphere application using IBM WebSphere Application Server Community Edition with 20GB AIX7 virtual image`.
  - e. Click **Create** to extend the selected image to create an image.



**Extend Image**

The new image will be created by extending this one.

Name:

Universal ID:

Version:

Description:

Figure 9-9 Extend Image window

The new image is displayed in the list on the left side as shown in Figure 9-10.

The screenshot shows the IBM Image Construction and Composition Tool interface. The top navigation bar includes 'Welcome', 'Images and bundles', and 'Administer'. The 'Cloud Provider' is set to 'PowerVM Provider on helium55'. The left sidebar shows a list of images, with 'PlantsByWebSphere-wasce-aix7-20GB-SCS' selected. The main panel displays the details for this image.

|                   |   |
|-------------------|---|
| Description:      | Plants by WebSphere application using IBM WebSphere Application Server Community Edition with 20GB AIX7 virtual image   |
| Universal ID:     | com.ibm.itso.plants.wasce.aix.ppc.image   |
| Version:          | 1.0.0   |
| Extends Image:    | baseOS-aix7-20GB-SCS [1.0.0.0]  |
| Image Status:     | Out of sync   |
| Created on:       | July 11, 2012 4:23:32 PM Central Daylight Time  |
| Updated date:     | July 11, 2012 4:23:33 PM Central Daylight Time  |
| Operating System: | Type: AIX<br>Distribution: AIX Server<br>Version: 7<br>Activation Framework: Enablement Bundle AIX6 for PowerVM Express |
| Cloud Provider:   | PowerVM Provider on helium55  |
| Software Bundles: |   |
| Products:         |   |
| Hardware:         |   |
| Virtual System:   |   |
| Credentials:      |   |
| License:          |   |

Figure 9-10 PlantsByWebSphere new image



## 9.5.2 Adding software bundles to sample image

Now you add the two software bundles to the new virtual image:

1. Select the new **PlantsByWebSphere-wasce-aix7-20GB-SCS** image on the left.
2. Click the **Edit** icon (✎) to start configuring the new extended virtual image.
3. To add the two software bundles to this image, do the following steps:
  - a. Click **Software Bundles** section, and select **Add bundle**.
  - b. In the Add bundle to image window (Figure 9-11), select the two bundles to include, and then click **Add**.

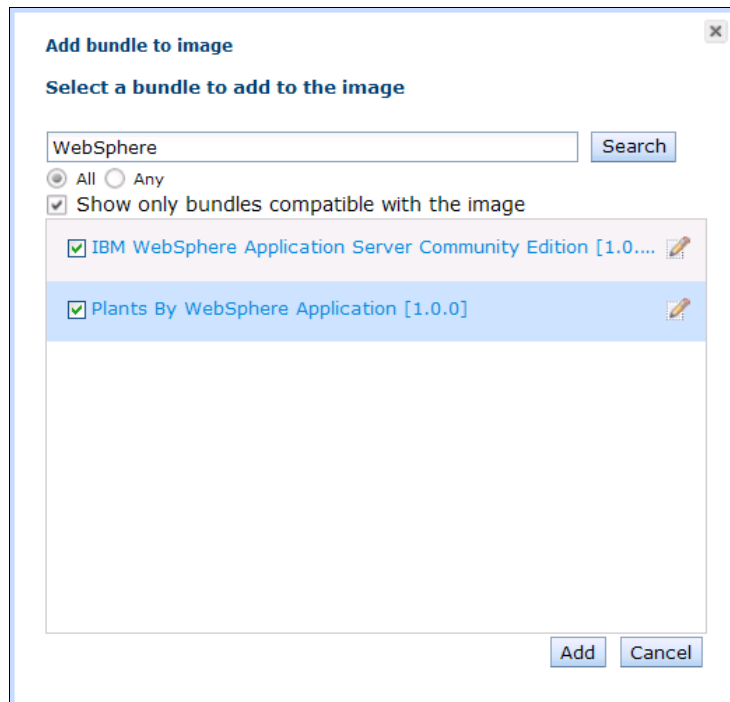




Figure 9-11 Add bundles to image window

4. In the next window (Figure 9-12), where you now see the two software bundles added to the image, click the **Done editing** icon () and then click the **Save** icon ()

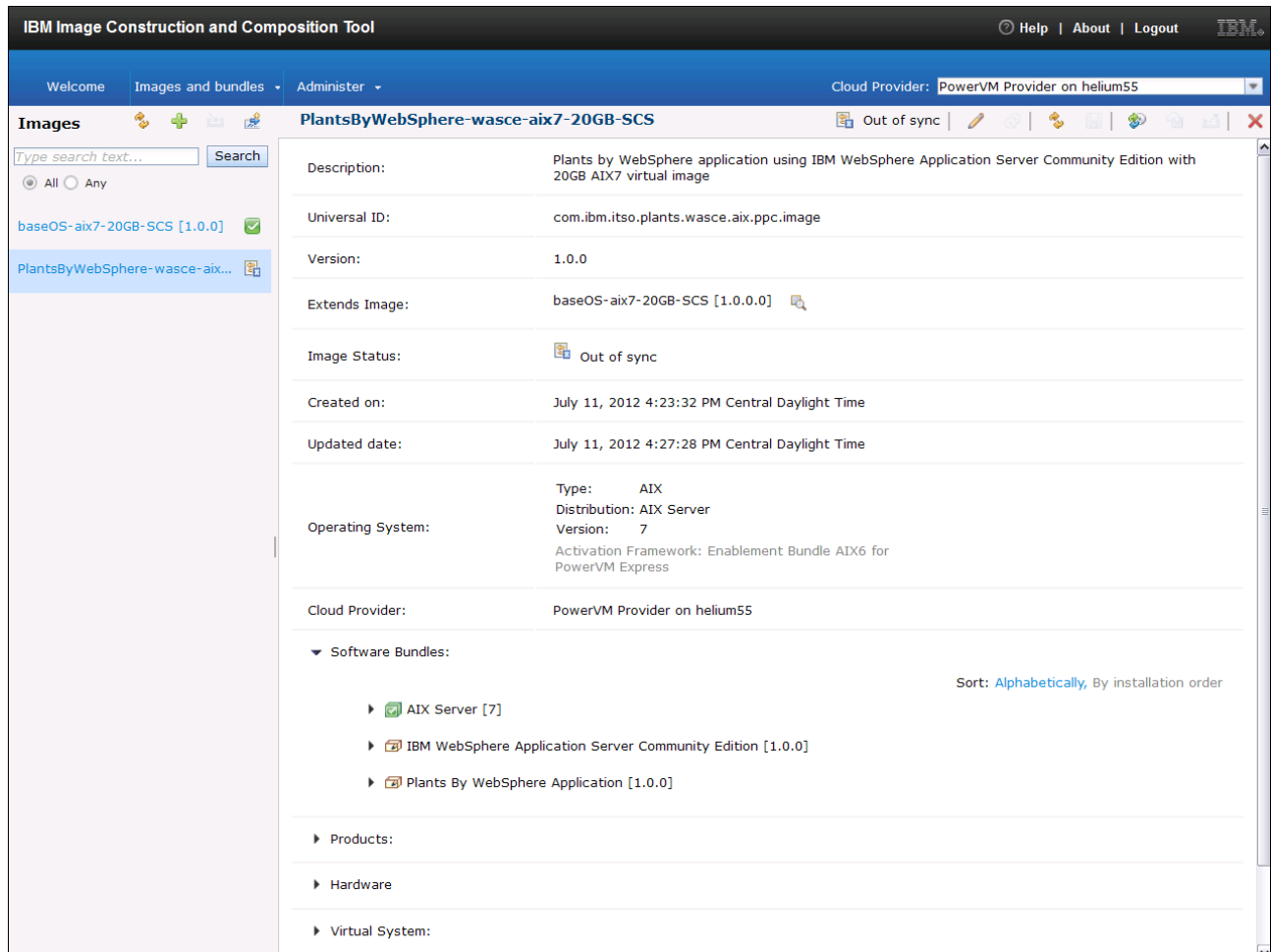




Figure 9-12 PlantsByWebSphere image out of sync

**Note:** The bundle order is important for the proper creation of the virtual appliance. If bundles are added out of order, you can click the **Edit** icon () which will give you the option to re-order () bundles under Software Bundles as shown in Figure 9-13.

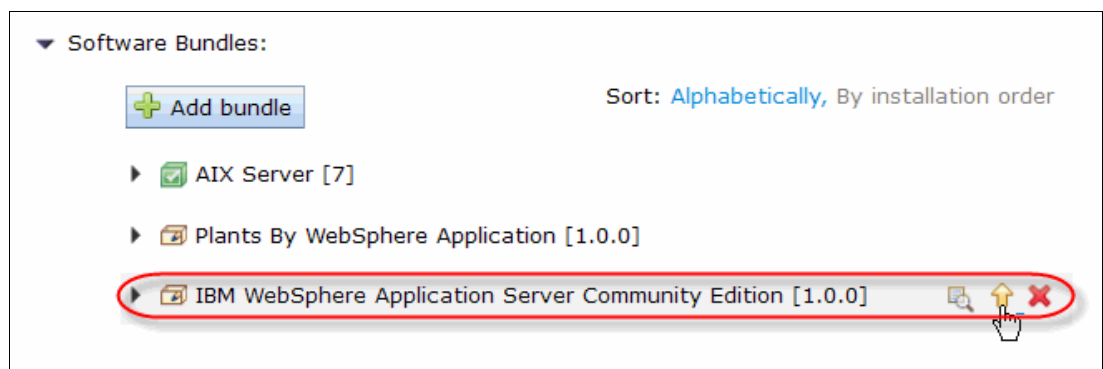




Figure 9-13 Re-order Software Bundles

## 9.5.3 Synchronizing the sample image

*Synchronization* is the process of creating a new virtual machine by deploying the virtual image, deploying the two software bundles, and then in order, running the defined installation scripts of each bundle.

1. Select the **PlantsByWebSphere-wasce-aix7-20GB-SCS** image on the left side.
2. Click the **Synchronize** icon () to start the synchronization process.
3. In the Synchronize the image window, enter root for the user ID and the password (for example, passw0rd) for the image.
4. Click **Done** to begin the synchronization process.

You can click the **Refresh** icon () to view the progress of the synchronization process. When completed, the Image Status is now *Synchronized* as shown in Figure 9-14.

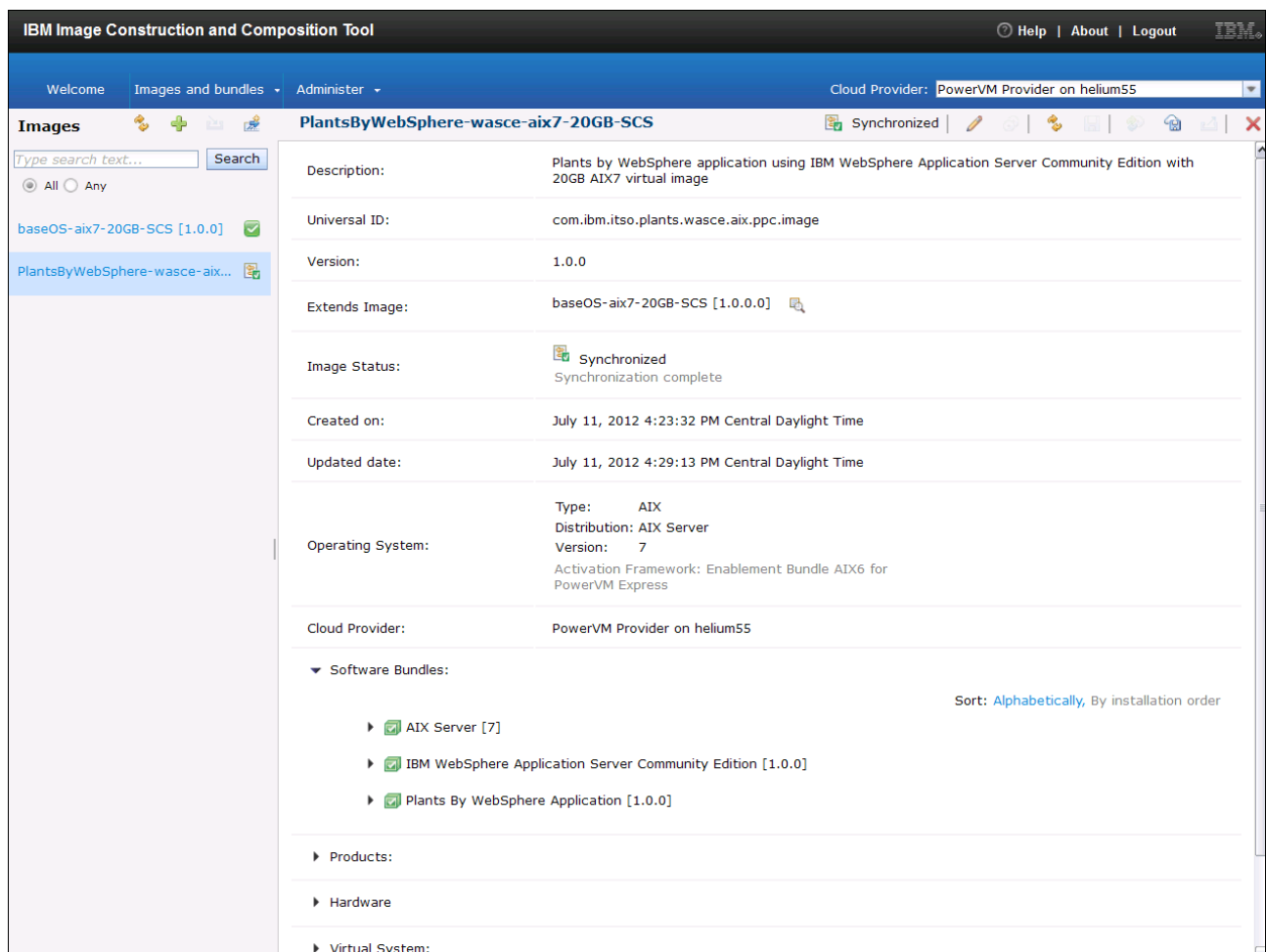


Figure 9-14 PlantsByWebSphere image synchronized

## 9.5.4 Validating the sample virtual image

After the image is synchronized, the WebSphere Application Server Community Edition and Plants by WebSphere bundles are installed on the image. For the WebSphere Application Server Community Edition bundle, the installation operation (**Install** tab) is complete. For the WebSphere Application Server Community Edition and Plants By WebSphere bundles, deploy-time operations (**Configuration** tab) are installed by the VSAE.

In continuing the example, you now access the synchronized image and verify the bundle installation. Before completing these steps, you must be familiar with the VSAE. You can also review the log of Image Construction and Composition Tool. For more information, see 9.6.1, “Synchronizing a simple virtual appliance image” on page 209.

To validate the sample virtual image, complete the following steps:

1. Select the synchronized **PlantsByWebSphere-wasce-aix7-20GB-SCS** image.
2. Expand the **Virtual System** section of the synchronized image that you want to validate. The IP address and host name that were assigned to the image are displayed.
3. Use SSH to connect to the system that is using the IP address or host name from step 2.
4. When prompted for a password, enter the password you entered for the image during the synchronization process. You are now logged in to the system.
5. Verify that the software bundles were installed:
  - a. Verify that the installation operation of the WebSphere Application Server Community Edition bundle completed successfully:

```
ls -al $WASCE_INSTALL_PATH
```
  - b. Verify that the deploy-time operations of the bundles were successfully installed:
    - i. Verify the bundle in the `/opt/ibm/ae/AS` directory (Example 9-6).

*Example 9-6 Bundles in the /opt/ibm/ae/AS directory*

---

```
ls -al /opt/ibm/ae/AS | grep com.ibm.itso
drwxr-xr-x  3 root    system      256 Jun 26 12:32
com.ibm.itso.plants.wasce.bundle_1.0.0
drwxr-xr-x  3 root    system      256 Jun 26 12:32
com.ibm.itso.wasce.aix.ppc_1.0.0
```

---

- ii. Verify the configuration operations in the `/opt/ibm/ae/AL/master.al` directory (Example 9-7).

*Example 9-7 Excerpt from the master.al file*

---

```
<ProductActivation class="ConfigWASCE">
  <Properties>
    <Property key="num_servers"/>
    <Property key="WASCE_HOME"/>
    <Property key="WASCE_ADMIN_USER"/>
    <Property key="WASCE_ADMIN_PASSWORD"/>
  </Properties>
  <Program cmdLine="true" envVarPrefix="AE_" envVars="false"
href="/opt/ibm/ae/AS/com.ibm.itso.wasce.aix.ppc_1.0.0/activation/ConfigWASCE/ConfigWASCE.sh"/>
  <Service name="ConfigWASCE">
    <RunLevel value="2"/>
  </Service>
```

---

```

</ProductActivation>
<ProductActivation class="ConfigApp">
  <Properties/>
  <Program cmdLine="true" envVarPrefix="AE_" envVars="false"
href="/opt/ibm/ae/AS/com.ibm.itso.plants.wasce.bundle_1.0.0/activation/Co
nfigApp/activate.sh"/>
  <Service name="ConfigApp">
    <RunLevel value="2"/>
  </Service>
</ProductActivation>

```

---

- iii. Verify that your activation program is correctly set up as an OS service (Example 9-8).

*Example 9-8 Activation program services in the /etc/rc.d/rc2.d file*

---

```

# ls /etc/rc.d/rc2.d
K50activate.vmc.network-interface  S02activate.vmc.network-interface
K50activate.vmc.system             S02activate.vmc.system
K50activate.vmc.system-networking  S02activate.vmc.system-networking
K50aed                             S02aed
K53ConfigWASCE                    S03ConfigWASCE
K54ConfigApp                      S04ConfigApp
K71itcaIBMTivoliCommonAgent0      S71itcaIBMTivoliCommonAgent0
Kradiusd                          Sradiusd
Ksshd                             Ssshd
Kwpars

```

---

The validation of the synchronized image is now complete. From here, you can capture the virtual image as described in the next section.

## 9.5.5 Capturing the sample virtual appliance



After the PlantsByWebSphere-wasce-aix7-20GB-SCS virtual image is synchronized, the capture process takes a snapshot of the virtual machine and saves it as a new image in the IBM Systems Director VMControl repository.

To capture the virtual appliance, complete the following steps:

1. Clean up before capturing:
  - a. Delete any files or directories that are no longer needed.
  - b. Unmount any disk that is used during the installation of the virtual image that will not be available after the capture is performed.
  - c. Clean any history or log files:

```

# > ~/.ssh/known_hosts
# > ~/.vi_history
# > ~/.sh_history
# errclear 0

```
2. Select the image on the left side, and click the **Capture** icon () to start the capture process.
3. Click the **Refresh** icon () to view the progress of the capture process.

When completed, the Image Status is Completed - Capture complete as shown in Figure 9-15.

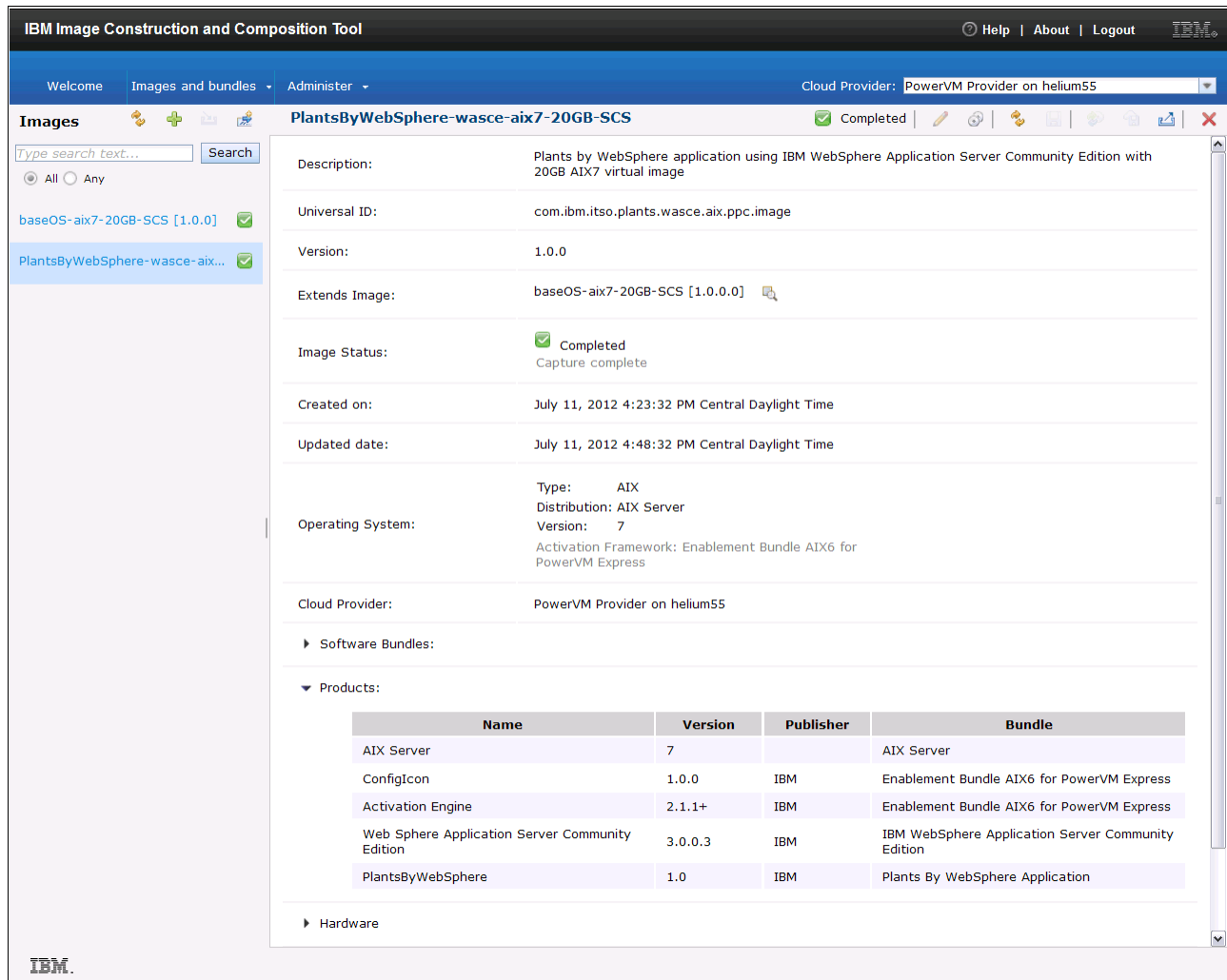



Figure 9-15 PlantsByWebSphere image capture complete

## 9.5.6 Exporting the sample virtual appliance

Exporting the image as an Open Virtual Appliance (OVA) archive entails taking the captured image out of the IBM Systems Director VMControl repository and sending a compressed version of the image to a target system. In this example, you send it directly to the IBM Systems Director VMControl management server.

To export the sample virtual appliance, complete the following steps:

1. Select the **PlantsByWebSphere-wasce-aix7-20GB-SCS** image on the left.
2. Click the **Export as OVA** icon () to start the export process.

3. In the Export image as OVA window (Figure 9-16), do the following steps:
  - a. For Remote host, enter helium55.
  - b. For Destination folder, enter /tmp/gsh/plants.
  - c. Skip the File name field, because it is populated with the Universal ID for the image.
  - d. Select the Authentication Method, which is **Password** in this example.
  - e. Enter a user name of root.
  - f. Enter the password for the user to access the remote host system.
  - g. Enter the password again to verify it.
  - h. For OVA file format, select **Power VM PureFlex Appliance**.
  - i. Click **OK** to begin the export process.

**Export image as OVA**

To what location should the image be exported?  
*The destination host must support Secure Copy (SCP)*

Remote host:

Destination folder:

File name:

**Authentication Method:**

☐ Private Key File ☒ Password

User name:

Password:

Verify password:

OVA file format:

Figure 9-16 Export image as OVA window

4. In the Export image status window (Figure 9-17), do the following tasks:
  - a. Click the **Open the export status** link to view the status.

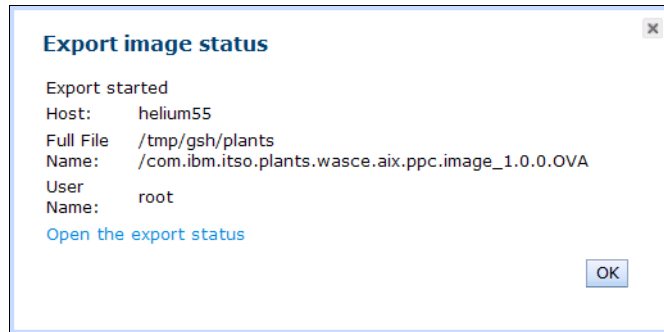


Figure 9-17 Export image status window

- b. Click **Refresh** to view the latest status. When finished, the Status changes to Ready (Figure 9-18). Close the browser window when finished.

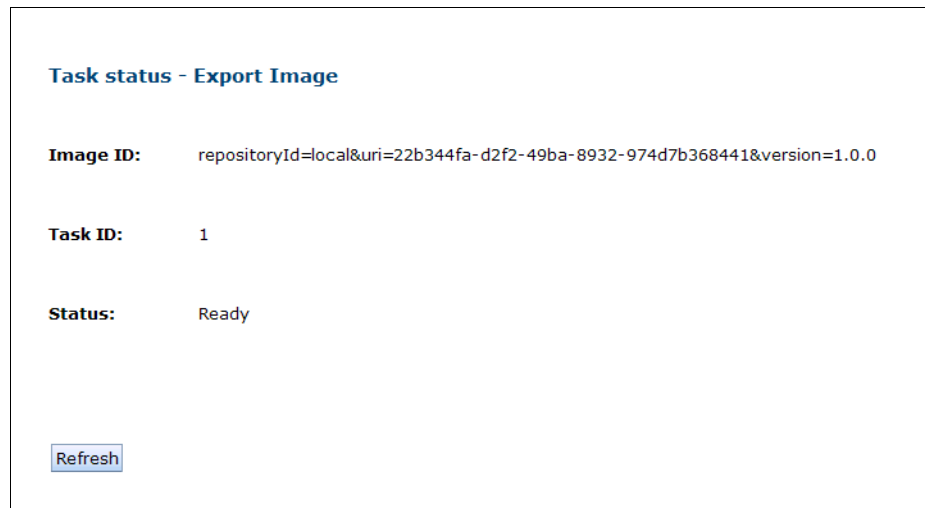


Figure 9-18 Task status - Export image - Ready

- c. In the Export image status window, click **OK**.
5. Go to the specified directory on the remote host to verify that the OVA was exported (Figure 9-19).

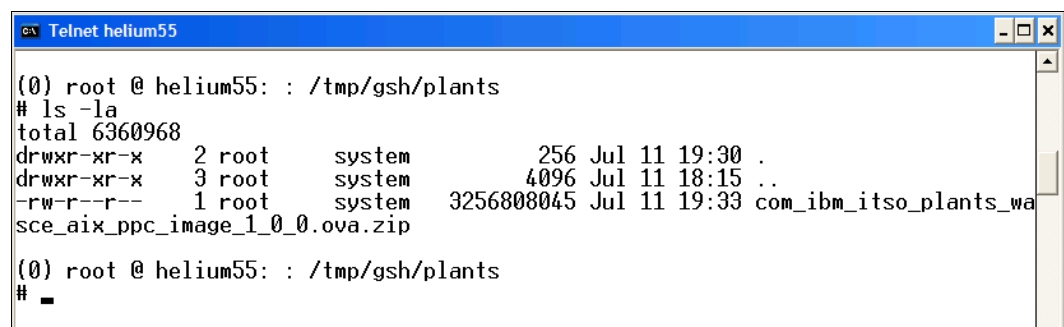


Figure 9-19 Target system with a compressed OVA file



## 9.5.7 Verifying the results

After the export is completed, you have a virtual appliance that consists of WebSphere Application Server Community Edition and the web application, PlantsByWebSphere. You can now use the Systems Director VMControl plug-in of IBM Flex System Manager or IBM Systems Director to import and deploy the virtual appliance easily and quickly from the web GUI.

When deploying the virtual appliance, all of the changeable parameters are displayed in the Systems Director VMControl wizard.

Table 9-3 shows the details of the configurable parameters and their default values for the Plants By WebSphere virtual appliance.

Table 9-3 Configurable parameters

| Activation program | Description          | Input parameters  | Default values                                 |
|--------------------|----------------------|---|--|
| ConfigWASCE        | num_servers          | Configures the number of WebSphere Application Server Community Edition instances | '1'  |
|                    | WASCE_HOME           | Defines WebSphere Application Server Community Edition home                       | '/opt/IBM/WebSphere/AppServerCommunityEdition' |
|                    | WASCE_ADMIN_USER     | Defines the administrative user   | 'system'                                       |
|                    | WASCE_ADMIN_PASSWORD | Defines the password of the administrative user                                   | 'manager'                                      |

## 9.6 Log file of the Image Construction and Composition Tool

For problem determination or to monitor the progress of an operation, you can view the Image Construction and Composition Tool error.log file in the /drouter/ramdisk2/mnt/raid-volume/raid0/logs/error/ directory. During the synchronization, capture, and export steps, the logs files were examined for specific messages.

**Log messages in this example:** In the examples in the following sections, many log messages were removed, and each message was condensed for easier readability.

## 9.6.1 Synchronizing a simple virtual appliance image

During the synchronization step, a new virtual machine is created, the base image is deployed, and the software bundle installation scripts are run.

Example 9-9 shows a synchronization log file.

*Example 9-9 Synchronization log file for the Image Construction and Composition Tool*

---

```
==> Start Synchronization Process
[2012-07-11 16:29:13:606 CDT] started synching

==> Create new virtual machine
[2012-07-11 16:29:13:871 CDT] create new vm
[2012-07-11 16:29:13:933 CDT] allocateIpAddress Using address: {CLOUDPROVIDERID=1,
HOSTNAME=helium80, DOMAIN=rchland.ibm.com, ADDRESS=10.5.169.173, CREATED=1339086952586,
UPDATED=1342041491175, CURRENTSTATUS=inactive, ID=4}
[2012-07-11 16:29:20:525 CDT] run DeployResponse: 201 [application/octet-stream; charset=UTF-8]:
DNZEMW350I Workload [55392] creation started.

==> Waiting for virtual machine to start and connect
[2012-07-11 16:29:20:535 CDT] Starting to try to connect to 10.5.169.173
[2012-07-11 16:29:43:968 CDT] waiting for vm
[2012-07-11 16:39:04:181 CDT] Completed the remote access connection to host: 10.5.169.173 Time
taken: 583646 Attempts taken: 138

==> Prepare Software Bundles into a single script
[2012-07-11 16:39:14:164 CDT] IconImageSynchronizer run creating execution package
[2012-07-11 16:39:19:966 CDT] createExecutionPackage Task sourceBundleUrl is
com.ibm.icon.powervm.enablement.aix6_1.0.0.1
[2012-07-11 16:39:19:967 CDT] createExecutionPackage Task sourceBundleUrl is
com.ibm.itso.wasce.aix.ppc_1.0.0

==> Transfer Software Bundles to virtual server
[2012-07-11 16:39:42:206 CDT] Transferring bundle file to vm:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/IconImageSynchronizer.zip

==> Run Software Bundle Install scripts on virtual server
[2012-07-11 16:40:07:147 CDT] Running bundle install command: /bin/ksh IconImageSynchronizer.sh
[2012-07-11 16:41:19:953 CDT] Execution package out: BEGIN Roll Your Own customization:
Automation
./install.sh: === Install IBM WAS CE ===
./install.sh: Arguments are: -wasce_install_path /opt/IBM/WebSphere/AppServerCommunityEdition
-installed_java_home /usr/java6 -wasce_install_exec wasce_setup-3.0.0.1-unix.bin -wasce_zip_name
wasce_ibm60sdk_setup-3.0.0.1-ppc64aix.zip

==> Set up Activation Engine services
[2012-07-11 16:41:21,760] INFO: Creating system services for activation. Input AL file is
/opt/ibm/ae/AL/ConfigWASCE-vs0.al.
[2012-07-11 16:41:30,042] INFO: Creating system services for activation. Input AL file is
/opt/ibm/ae/AL/ConfigApp-vs0.al.
[2012-07-11 16:41:33,065] INFO: Created system services for activation.

==> Synchronization Completed
[2012-07-11 16:41:19:969 CDT] Bundle installation succeeded
```

---

## 9.6.2 Capturing a simple virtual appliance image

When capturing an image, the Image Construction and Composition Tool performs an IBM Systems Director Collect Inventory on the virtual machine, runs the software bundle reset scripts, and shuts down. Then, by using Systems Director VMControl, the tool captures the stopped virtual machine.

Example 9-10 shows a log file from a capture process.

*Example 9-10 Capture log file for the Image Construction and Composition Tool*

---

```
==> Start Capture Process
[2012-07-11 16:48:33:047 CDT] Set captured status to Capturing

==> Collect Inventory on virtual machine
[2012-07-11 16:50:04:133 CDT] Waiting 10 minute(s) for inventory completion...
[2012-07-11 16:53:44:570 CDT] Inventory completed

==> Prepare virtual machine to be shutdown
[2012-07-11 16:53:45:996 CDT] Calling reset on VM
[2012-07-11 16:53:46:469 CDT] getAEVersion return version=2.1.1+
[2012-07-11 16:53:46:546 CDT] resetVM Running AE.sh --reset

==> Wait for virtual machine to shutdown
[2012-07-11 16:55:57:237 CDT] Waiting for workload state of 5; status = Started (8)
[2012-07-11 16:56:57:396 CDT] Waiting for workload state of 5; status = Stopped (5)

==> Waiting for capture to complete
[2012-07-11 16:56:59:373 CDT] Waiting for capture to complete: 404
[2012-07-11 16:57:19:425 CDT] Waiting for capture to complete: 200
<ovf:Envelope ...
...
</ovf:Envelope>

==> Update OVF file
[2012-07-11 16:57:19:586 CDT] doCapture Generated OVF:<?xml version="1.0" encoding="UTF-8"?>
<ovf:Envelope ...
...
</ovf:Envelope>
[2012-07-11 16:57:20:032 CDT] Updated appliance OVF reponse:200

==> Delete virtual machine
[2012-07-11 16:58:35:298 CDT] Deleted operating system resource 55420: 204 [null]: null

==> Capture Process Complete
[2012-07-11 16:58:35:300 CDT] Set captured status to Captured

==> Clean up
[2012-07-11 16:59:05:355 CDT] running the completer
[2012-07-11 16:59:05:364 CDT] updating cloud base image

==> Clean up, Shutdown and Delete workload if exists
[2012-07-11 16:59:05:450 CDT] powerOffWorkload Workload already powered off: 55392
[2012-07-11 16:59:48:207 CDT] deleteVM Delete Response:500
[2012-07-11 16:59:48:207 CDT] deleteVM Delete of workload failed: 500
```

---

### 9.6.3 Exporting a simple virtual appliance image

The export process for the PowerVM provider is a complex process that requires additional disk space on the Virtual I/O Server (VIOS) and on the Image Construction and Composition Tool server. During export, a new disk must be created and assigned to the VIOS so that the captured image can be copied to the new disk. Then, the disk is compressed and sent to the Image Construction and Composition Tool server where it is extracted, and the Open Virtualization Format (OVF) is modified and then repacked into an OVA. Finally the OVA is compressed and sent to the target system.

Example 9-11 shows a log taken during an export process.

*Example 9-11 Log file from the Image Construction and Composition Tool during export*

---

```
==> Start Export Process
[2013-05-28 12:07:28:313 EDT] Will export OVF/OVA in WCA format
[2013-05-28 12:07:28:438 EDT] run USING NEW UDPATE CODE!!!!!!!!!!!!!!!!!!!!!!
[2013-05-28 12:07:28:438 EDT] run updating message bundles
[2013-05-28 12:07:28:442 EDT] addToOvaContents Adding to OVA: Semantic.topology ->
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/Semantic.to
pology
[2013-05-28 12:07:28:442 EDT] addToOvaContents finished putting ova contents into map
[2013-05-28 12:07:28:445 EDT] addToOvaContents Adding to OVA: Automation.topology ->
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/Automation.
topology
[2013-05-28 12:07:28:445 EDT] addToOvaContents finished putting ova contents into map
[2013-05-28 12:07:28:446 EDT] run building ova contents
[2013-05-28 12:07:28:446 EDT] buildOVAContentsMap Failed read of artifact:
sanvolume://00000200A0A15FC4/@/600507680282857F10000000000000B4

==> Get Image and Repository information
[2013-05-28 12:07:28:453 EDT] run Envelope for export:
[2013-05-28 12:07:28:458 EDT] getExportDir Image to export has remote asset id of 114033
[2013-05-28 12:07:28:480 EDT] getExportDir Virtual appliance 114033 is in repository 114014
[2013-05-28 12:07:28:486 EDT] getExportDir Repository 114014:
[2013-05-28 12:07:28:487 EDT] getExportDir Repository 114014 is of type SCS
[2013-05-28 12:07:28:567 EDT] getSvcHost Resource (StoragePool[14758]) :
[2013-05-28 12:07:28:609 EDT] getSvcHost Resource (StorageSubsystem[14744]) :
[2013-05-28 12:07:28:609 EDT] getJSchAccess Connecting to hosts: [192.168.71.120]
[2013-05-28 12:07:28:609 EDT] getJSchAccess Host name 192.168.71.120 parsed into: [192.168.71.120]
[2013-05-28 12:07:28:609 EDT] getJSchAccess Connecting to: 192.168.71.120
[2013-05-28 12:07:28:900 EDT] getScsHost Resource (System[113774]) :
[2013-05-28 12:07:28:900 EDT] getHost IPv4Address = [192.168.71.203]
[2013-05-28 12:07:28:900 EDT] getJSchAccess Connecting to hosts: [192.168.71.203]
[2013-05-28 12:07:28:900 EDT] getJSchAccess Connecting to: 192.168.71.203
[2013-05-28 12:07:29:304 EDT] getHostName HostName = [wflx1node3.flx1.bow.iic.ihost.com]
[2013-05-28 12:07:29:304 EDT] getRepositoryFcPortDeviceId Resource (FCPort[115639]) SourceTokens:
[IBM-DPSM,
root/cimv2:IBMAIX_FCPort.CreationClassName="IBMAIX_FCPort",DeviceID="21000024ff20553a",SystemCreationClassNa
ame="AIX_ComputerSystem",SystemName="wflx1node3.flx1.bow.iic.ihost.com"]
[2013-05-28 12:07:29:305 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@59ed59ed (any: null, chunkSize: <unset>, compression:
<unset>, href: sanvolume://00000200A0A15FC4/@/600507680282857F10000000000000B4, id: file1, localRef: null,
size: 21474836480, anyAttribute: [])
[2013-05-28 12:07:29:305 EDT] getSanVolumeName Found SAN volume name = 600507680282857F10000000000000B4

==> Create new disk on V7000 and map to VIOS
[2013-05-28 12:07:29:305 EDT] Running command on 192.168.71.120: svcinfo lsvdisk -filtervalue
vdisk_UID=600507680282857F10000000000000B4 -delim :
[2013-05-28 12:07:29:379 EDT] Property not found <PVM_JSCH_PAUSE>: using default value <1000>
```

```

[2013-05-28 12:07:29:379 EDT] logValue Value of PVM_JSCH_PAUSE is 1000
[2013-05-28 12:07:30:384 EDT] Volume properties: {fc_map_count=0, capacity=20.00GB, se_copy_count=0,
fast_write_state=empty, vdisk_UID=600507680282857F1000000000000B4, FC_name=, type=striped, FC_id=,
mdisk_grp_name=mdiskgrp0, copy_count=1, IO_group_id=0, id=9, status=online, compressed_copy_count=0,
IO_group_name=io_grp0, RC_id=, RC_change=no, RC_name=, mdisk_grp_id=0,
name=PlantsByWebSphere-wasce-aix7-20GB-SCS1}
[2013-05-28 12:07:30:384 EDT] Running command on 192.168.71.120: svctask mkvdisk -iogrp io_grp0 -mdiskgrp
mdiskgrp0 -size 21474836480 -unit b
[2013-05-28 12:07:31:440 EDT] Running command on 192.168.71.120: flashCopySvcVolume Parsed disk id:42
[2013-05-28 12:07:31:440 EDT] Running command on 192.168.71.120: svctask mkfcmap -source
PlantsByWebSphere-wasce-aix7-20GB-SCS1 -target 42 -copyrate 100 -autodelete -cleanrate 50
[2013-05-28 12:07:32:501 EDT] Running command on 192.168.71.120: svctask startfcmap -prep 0
[2013-05-28 12:07:33:562 EDT] Running command on 192.168.71.120: svcinfo lsvdisk -filtervalue vdisk_id=42
-delim :
[2013-05-28 12:07:34:614 EDT] Running command on 192.168.71.120: getSvcVolumeInfo Volume properties:
{fc_map_count=1, capacity=20.00GB, se_copy_count=0, fast_write_state=empty,
vdisk_UID=600507680282857F1000000000000178, FC_name=fcmap1, type=striped, FC_id=0,
mdisk_grp_name=mdiskgrp0, copy_count=1, IO_group_id=0, id=42, status=online, compressed_copy_count=0,
IO_group_name=io_grp0, RC_id=, RC_change=no, RC_name=, mdisk_grp_id=0, name=vdisk2}
[2013-05-28 12:07:34:614 EDT] Running command on 192.168.71.120: svcinfo lsfabric -wwpn=21000024ff20553a
-delim :
[2013-05-28 12:07:35:675 EDT] Running command on 192.168.71.120: mapDriveToHost Fabric properties:
{name=SN10AAAAA_HBA1, state=active, id=1, local_wwpn=500507680210D264, local_port=1, remote_nportid=010300,
type=host, local_nportid=010000, remote_wwpn=21000024FF20553A, node_name=node1, cluster_name=}
[2013-05-28 12:07:35:675 EDT] Running command on 192.168.71.120: svctask mkvdiskhostmap -host
SN10AAAAA_HBA1 -force 42
[2013-05-28 12:07:40:799 EDT] Running command on 192.168.71.203: cfgdev
[2013-05-28 12:07:41:800 EDT] Property not found <PVM_JSCH_WAIT>: using default value <10000>
[2013-05-28 12:07:41:801 EDT] logValue Value of PVM_JSCH_WAIT is 10000
[2013-05-28 12:09:01:813 EDT] Running command on 192.168.71.203: for dev in $(lsdev -type disk -field name
-fmt :); do echo "$dev:~lsdev -dev $dev -attr unique_id | awk '{if (NR==3) print $0}'"; done | grep
600507680282857F1000000000000178
[2013-05-28 12:09:03:815 EDT] getMappedHdisk Parsed mapped hdisk: hdisk6
[2013-05-28 12:09:07:879 EDT] Running command on 192.168.71.203: oem_setup_env
[2013-05-28 12:09:08:881 EDT] Running command on 192.168.71.203: dd if=/dev/hdisk6 bs=512 | bzip2 >
/home/padmin/PlantsByWebSphere-wasce-aix7-20GB-SCS1.1369757248488.raw.bz2
[2013-05-28 12:29:41:972 EDT] getRawImageFileToIcon Wrote properties in image proxy file:
{STORAGECONTROLLERUSERNAME=USERID, REPOSITORYTYPE=SCS, REPOSITORYHOSTUSERPASSWORD=PASSWORD,
REPOSITORYHOSTUSERNAME=padmin,
STORAGECONTROLLERUSERPRIVATEKEY=/resources/files/2c2d35fc-d550-46ca-bala-abd8dfa0278f/id_rsa,
imageHost=192.168.71.203, REPOSITORYNAME=POWER_SCS_REPO, uniqueId=1369757248488, imageFileSize=21474836480,
imageFile=/home/padmin/PlantsByWebSphere-wasce-aix7-20GB-SCS1.1369757248488.raw.bz2}
[2013-05-28 12:29:46:035 EDT] Running command on 192.168.71.203: rmdev -dev hdisk6
[2013-05-28 12:29:58:037 EDT] Running command on 192.168.71.120: svctask rmvdisk -force 42

==> Generate Message and Parts Metadata
[2013-05-28 12:29:59:099 EDT] syncOvfAndIwdParts parttype: osNode1369758599099
[2013-05-28 12:29:59:277 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/de-DE-bundle.msg
[2013-05-28 12:29:59:304 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/en-US-bundle.msg
[2013-05-28 12:29:59:313 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/es-ES-bundle.msg
[2013-05-28 12:29:59:323 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/fr-FR-bundle.msg

```

```

[2013-05-28 12:29:59:327 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/it-IT-bundle.msg
[2013-05-28 12:29:59:336 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/ja-JP-bundle.msg
[2013-05-28 12:29:59:340 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/ko-KR-bundle.msg
[2013-05-28 12:29:59:343 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/pt-BR-bundle.msg
[2013-05-28 12:29:59:346 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/zh-CN-bundle.msg
[2013-05-28 12:29:59:349 EDT] Creating message bundle file:
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/zh-TW-bundle.msg
[2013-05-28 12:29:59:354 EDT] Adding to OVA: PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw
[2013-05-28 12:29:59:354 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:354 EDT] Adding to OVA: de-DE-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/de-DE-bundle.msg
[2013-05-28 12:29:59:354 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:354 EDT] Adding to OVA: en-US-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/en-US-bundle.msg
[2013-05-28 12:29:59:354 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:355 EDT] Adding to OVA: es-ES-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/es-ES-bundle.msg
[2013-05-28 12:29:59:355 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:355 EDT] Adding to OVA: fr-FR-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/fr-FR-bundle.msg
[2013-05-28 12:29:59:355 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:355 EDT] Adding to OVA: it-IT-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/it-IT-bundle.msg
[2013-05-28 12:29:59:355 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:355 EDT] Adding to OVA: ja-JP-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/ja-JP-bundle.msg
[2013-05-28 12:29:59:355 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:356 EDT] Adding to OVA: ko-KR-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/ko-KR-bundle.msg
[2013-05-28 12:29:59:356 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:356 EDT] Adding to OVA: osNode1369758599099.xml ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/osNode1369758599099.xml
[2013-05-28 12:29:59:356 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:356 EDT] Adding to OVA: osNode1369758599099C.xml ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/osNode1369758599099C.xml
[2013-05-28 12:29:59:356 EDT] finished putting ova contents into map

```

```

[2013-05-28 12:29:59:356 EDT] Adding to OVA: pt-BR-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/pt-BR-bundle.msg
[2013-05-28 12:29:59:356 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:357 EDT] Adding to OVA: zh-CN-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/zh-CN-bundle.msg
[2013-05-28 12:29:59:357 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:357 EDT] Adding to OVA: zh-TW-bundle.msg ->
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/zh-TW-bundle.msg
[2013-05-28 12:29:59:357 EDT] finished putting ova contents into map
[2013-05-28 12:29:59:360 EDT] 00000174 IconImageExpo I com.ibm.cloud.icn.execution.engine.IconImageExporter
run <?xml version="1.0" encoding="UTF-8"?>
[2013-05-28 12:29:59:361 EDT] updateOvfForExport Found raw files:
[PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw]
[2013-05-28 12:29:59:362 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@57695769 (any: null, chunkSize: <unset>, compression:
<unset>, href: Automation.topology, id: Automation.topology, localRef: null, size: 100, anyAttribute: [])
[2013-05-28 12:29:59:362 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@59725972 (any: null, chunkSize: <unset>, compression:
<unset>, href: Semantic.topology, id: Semantic.topology, localRef: null, size: 100, anyAttribute: [])
[2013-05-28 12:29:59:362 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@59ed59ed (any: null, chunkSize: <unset>, compression:
<unset>, href: sanvolume://00000200A0A15FC4/e/600507680282857F10000000000000B4, id: file1, localRef: null,
size: 21474836480, anyAttribute: [])
[2013-05-28 12:29:59:362 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@1c101c1 (any: null, chunkSize: <unset>, compression:
<unset>, href: osNode1369758599099.xml, id: osNode1369758599099.xml, localRef: null, size: 15025,
anyAttribute: [rainmaker_2009_3:part2Definition=true])
[2013-05-28 12:29:59:362 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@4d804d8 (any: null, chunkSize: <unset>, compression:
<unset>, href: osNode1369758599099C.xml, id: osNode1369758599099C.xml, localRef: null, size: 14851,
anyAttribute: [rainmaker_2009_3:part2Definition=true])
[2013-05-28 12:29:59:362 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@6f0c6f0c (any: null, chunkSize: <unset>, compression:
<unset>, href: de-DE-bundle.msg, id: de-DE-bundle.msg, localRef: null, size: 57344, anyAttribute: null)
[2013-05-28 12:29:59:362 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@c780c78 (any: null, chunkSize: <unset>, compression:
<unset>, href: en-US-bundle.msg, id: en-US-bundle.msg, localRef: null, size: 65536, anyAttribute: null)
[2013-05-28 12:29:59:363 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@2d742d74 (any: null, chunkSize: <unset>, compression:
<unset>, href: es-ES-bundle.msg, id: es-ES-bundle.msg, localRef: null, size: 49152, anyAttribute: null)
[2013-05-28 12:29:59:363 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@365e365e (any: null, chunkSize: <unset>, compression:
<unset>, href: fr-FR-bundle.msg, id: fr-FR-bundle.msg, localRef: null, size: 65536, anyAttribute: null)
[2013-05-28 12:29:59:363 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@46a846a8 (any: null, chunkSize: <unset>, compression:
<unset>, href: it-IT-bundle.msg, id: it-IT-bundle.msg, localRef: null, size: 49152, anyAttribute: null)
[2013-05-28 12:29:59:363 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@46ca46ca (any: null, chunkSize: <unset>, compression:
<unset>, href: ja-JP-bundle.msg, id: ja-JP-bundle.msg, localRef: null, size: 49148, anyAttribute: null)
[2013-05-28 12:29:59:363 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@44a844a8 (any: null, chunkSize: <unset>, compression:
<unset>, href: ko-KR-bundle.msg, id: ko-KR-bundle.msg, localRef: null, size: 40957, anyAttribute: null)
[2013-05-28 12:29:59:364 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@62b962b9 (any: null, chunkSize: <unset>, compression:
<unset>, href: pt-BR-bundle.msg, id: pt-BR-bundle.msg, localRef: null, size: 49152, anyAttribute: null)

```

```

[2013-05-28 12:29:59:364 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@46054605 (any: null, chunkSize: <unset>, compression:
<unset>, href: zh-CN-bundle.msg, id: zh-CN-bundle.msg, localRef: null, size: 32766, anyAttribute: null)
[2013-05-28 12:29:59:364 EDT] getSanVolumeRefs Examining fileRef:
org.dmtf.schemas.ovf.envelope.impl.FileTypeImpl@3df53df5 (any: null, chunkSize: <unset>, compression:
<unset>, href: zh-TW-bundle.msg, id: zh-TW-bundle.msg, localRef: null, size: 32767, anyAttribute: null)
[2013-05-28 12:29:59:388 EDT] getVirtualSystem Parsed OVF and found virtual system with ID: vs0

==> Create OVA on Image Construction and Composition Tool server with placeholder disk
[2013-05-28 12:29:59:392 EDT] Starting OVA creation.
[2013-05-28 12:29:59:392 EDT] run Updated OVF:
[2013-05-28 12:29:59:442 EDT] buildOva Processing Automation.topology compression = null
[2013-05-28 12:29:59:445 EDT] buildOva Automation.topology will be bundled inside ova
[2013-05-28 12:29:59:446 EDT] buildOva Automation.topology available in destination
[2013-05-28 12:29:59:446 EDT] buildOva Processing Semantic.topology compression = null
[2013-05-28 12:29:59:446 EDT] buildOva Semantic.topology will be bundled inside ova
[2013-05-28 12:29:59:446 EDT] buildOva Semantic.topology available in destination
[2013-05-28 12:29:59:446 EDT] buildOva Processing PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw compression =
null
[2013-05-28 12:29:59:446 EDT] buildOva PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw will be bundled inside
ova
[2013-05-28 12:29:59:458 EDT] buildOva Processing osNode1369758599099.xml compression = null
[2013-05-28 12:29:59:458 EDT] buildOva osNode1369758599099.xml will be bundled inside ova
[2013-05-28 12:29:59:458 EDT] buildOva Processing osNode1369758599099C.xml compression = null
[2013-05-28 12:29:59:458 EDT] buildOva osNode1369758599099C.xml will be bundled inside ova
[2013-05-28 12:29:59:459 EDT] buildOva Processing de-DE-bundle.msg compression = null
[2013-05-28 12:29:59:459 EDT] buildOva de-DE-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:460 EDT] buildOva Processing en-US-bundle.msg compression = null
[2013-05-28 12:29:59:460 EDT] buildOva en-US-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:461 EDT] buildOva Processing es-ES-bundle.msg compression = null
[2013-05-28 12:29:59:461 EDT] buildOva es-ES-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:462 EDT] buildOva Processing fr-FR-bundle.msg compression = null
[2013-05-28 12:29:59:462 EDT] buildOva fr-FR-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:463 EDT] buildOva Processing it-IT-bundle.msg compression = null
[2013-05-28 12:29:59:463 EDT] buildOva it-IT-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:464 EDT] buildOva Processing ja-JP-bundle.msg compression = null
[2013-05-28 12:29:59:464 EDT] buildOva ja-JP-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:464 EDT] buildOva Processing ko-KR-bundle.msg compression = null
[2013-05-28 12:29:59:464 EDT] buildOva ko-KR-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:465 EDT] buildOva Processing pt-BR-bundle.msg compression = null
[2013-05-28 12:29:59:465 EDT] buildOva pt-BR-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:466 EDT] buildOva Processing zh-CN-bundle.msg compression = null
[2013-05-28 12:29:59:466 EDT] buildOva zh-CN-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:467 EDT] buildOva Processing zh-TW-bundle.msg compression = null
[2013-05-28 12:29:59:467 EDT] buildOva zh-TW-bundle.msg will be bundled inside ova
[2013-05-28 12:29:59:471 EDT] buildOva tar command tokens: [tar, -cf,
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/icon_image_
plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova,
icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ovf, Automation.topology, Semantic.topology,
PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw, osNode1369758599099.xml, osNode1369758599099C.xml,
de-DE-bundle.msg, en-US-bundle.msg, es-ES-bundle.msg, fr-FR-bundle.msg, it-IT-bundle.msg, ja-JP-bundle.msg,
ko-KR-bundle.msg, pt-BR-bundle.msg, zh-CN-bundle.msg, zh-TW-bundle.msg]
[2013-05-28 12:29:59:472 EDT] buildOva Archiving OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/icon_image_
plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova
[2013-05-28 12:29:59:473 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/icon_image_
plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ovf

```



```

[2013-05-28 12:29:59:476 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/Automation.
topology
[2013-05-28 12:29:59:476 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/Semantic.to
pology
[2013-05-28 12:29:59:476 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/PlantsByWeb
Sphere-wasce-aix7-20GB-SCS1.raw
[2013-05-28 12:29:59:477 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/osNode13697
58599099.xml
[2013-05-28 12:29:59:477 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/osNode13697
58599099C.xml
[2013-05-28 12:29:59:478 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/de-DE-bundl
e.msg
[2013-05-28 12:29:59:478 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/en-US-bundl
e.msg
[2013-05-28 12:29:59:480 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/es-ES-bundl
e.msg
[2013-05-28 12:29:59:481 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/fr-FR-bundl
e.msg
[2013-05-28 12:29:59:482 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/it-IT-bundl
e.msg
[2013-05-28 12:29:59:483 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/ja-JP-bundl
e.msg
[2013-05-28 12:29:59:484 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/ko-KR-bundl
e.msg
[2013-05-28 12:29:59:485 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/pt-BR-bundl
e.msg
[2013-05-28 12:29:59:487 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/zh-CN-bundl
e.msg
[2013-05-28 12:29:59:487 EDT] addFileToArchive Adding File to OVA:
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/zh-TW-bundl
e.msg
[2013-05-28 12:29:59:492 EDT] Completed OVA creation.

==> Secure Copy (scp) OVA and disk (raw.bz2) disk files to Target System
[2013-05-28 12:29:59:492 EDT] run Start copying OVA to 192.168.71.206
[2013-05-28 12:30:02:333 EDT] sendFile Successfully exported file
/drouter/ramdisk2/mnt/raid-volume/raid0/tmp/exportedImages/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/icon_image_
plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.oVa.bz2 to root@192.168.71.206:/export/OVA/isv2
[2013-05-28 12:30:02:843 EDT] exportOva Found image files:
[/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_136975724
8488/PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw]
[2013-05-28 12:30:02:843 EDT] Connecting to hosts: [192.168.71.206]
[2013-05-28 12:30:02:843 EDT] Connecting to: 192.168.71.206
[2013-05-28 12:30:05:954 EDT] Running command on 192.168.71.206: uname -a
[2013-05-28 12:30:11:019 EDT] Running command on 192.168.71.206: cd /export/OVA/isv2

```

```

[2013-05-28 12:30:12:021 EDT] Running command on 192.168.71.206: bunzip2
icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova.bz2
[2013-05-28 12:30:13:022 EDT] exportOva Loaded properties from image proxy file:
{STORAGECONTROLLERUSERNAME=USERID, REPOSITORYTYPE=SCS, REPOSITORYHOSTUSERPASSWORD=PASSWORD=,
REPOSITORYHOSTUSERNAME=padmin, imageHost=192.168.71.203,
STORAGECONTROLLERPRIVATEKEY=/resources/files/2c2d35fc-d550-46ca-bala-abd8dfa0278f/id_rsa,
REPOSITORYNAME=POWER_SCS_REPO, uniqueId=1369757248488, imageFileSize=21474836480,
imageFile=/home/padmin/PlantsByWebSphere-wasce-aix7-20GB-SCS1.1369757248488.raw.bz2}
[2013-05-28 12:30:16:085 EDT] Running command on 192.168.71.206: cd /export/OVA/ismv2
[2013-05-28 12:30:17:086 EDT] Running command on 192.168.71.206: mkdir icon_1369758602843
[2013-05-28 12:30:18:087 EDT] Running command on 192.168.71.206: cd icon_1369758602843
[2013-05-28 12:30:19:088 EDT] Running command on 192.168.71.206: scp -q -o "StrictHostKeyChecking no"
padmin@192.168.71.203:/home/padmin/PlantsByWebSphere-wasce-aix7-20GB-SCS1.1369757248488.raw.bz2 [2013-05-28
12:30:20:089 EDT] Entered password on 192.168.71.206: *****
[2013-05-28 12:31:14:155 EDT] Running command on 192.168.71.206: cd /export/OVA/ismv2
[2013-05-28 12:31:15:157 EDT] Running command on 192.168.71.206: cd icon_1369758602843
[2013-05-28 12:31:16:158 EDT] Running command on 192.168.71.206: bunzip2
PlantsByWebSphere-wasce-aix7-20GB-SCS1.1369757248488.raw.bz2
[2013-05-28 12:35:40:258 EDT] Running command on 192.168.71.206: cd /export/OVA/ismv2
[2013-05-28 12:35:41:260 EDT] Running command on 192.168.71.206: cd icon_1369758602843
[2013-05-28 12:35:42:261 EDT] Running command on 192.168.71.206: mv
PlantsByWebSphere-wasce-aix7-20GB-SCS1.1369757248488.raw PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw
[2013-05-28 12:36:17:328 EDT] Running command on 192.168.71.206: PS1="# "
[2013-05-28 12:36:18:330 EDT] Running command on 192.168.71.206: cd /export/OVA/ismv2
[2013-05-28 12:36:19:331 EDT] Running command on 192.168.71.206: cd icon_1369758602843
[2013-05-28 12:36:20:332 EDT] Running command on 192.168.71.206: ls -l --color=never
PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw
[2013-05-28 12:36:25:397 EDT] Running command on 192.168.71.206: PS1="# "
[2013-05-28 12:36:26:398 EDT] Running command on 192.168.71.206: cd /export/OVA/ismv2
[2013-05-28 12:36:27:399 EDT] Running command on 192.168.71.206: cd icon_1369758602843
[2013-05-28 12:36:28:400 EDT] Running command on 192.168.71.206: tar -tvf
../icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova
[2013-05-28 12:36:45:419 EDT] Command output:
-rw-r--r-- root/0      21514 2013-05-28 12:29 icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ovf
-rw-r--r-- root/0      12377 2013-05-28 12:07 Automation.topology
-rw-r--r-- root/0       2754 2013-05-28 12:07 Semantic.topology
-rw-r--r-- root/0        465 2013-05-28 12:29 PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw
-rw-r--r-- root/0     15025 2013-05-28 12:29 osNode1369758599099.xml
-rw-r--r-- root/0     14851 2013-05-28 12:29 osNode1369758599099C.xml
-rw-r--r-- root/0     59105 2013-05-28 12:29 de-DE-bundle.msg
-rw-r--r-- root/0     77316 2013-05-28 12:29 en-US-bundle.msg
-rw-r--r-- root/0     57302 2013-05-28 12:29 es-ES-bundle.msg
-rw-r--r-- root/0     69920 2013-05-28 12:29 fr-FR-bundle.msg
-rw-r--r-- root/0     54175 2013-05-28 12:29 it-IT-bundle.msg
-rw-r--r-- root/0     54543 2013-05-28 12:29 ja-JP-bundle.msg
-rw-r--r-- root/0     49062 2013-05-28 12:29 ko-KR-bundle.msg
-rw-r--r-- root/0     53035 2013-05-28 12:29 pt-BR-bundle.msg
-rw-r--r-- root/0     40532 2013-05-28 12:29 zh-CN-bundle.msg
-rw-r--r-- root/0     41556 2013-05-28 12:29 zh-TW-bundle.msg
[2013-05-28 12:36:48:482 EDT] Running command on 192.168.71.206: PS1="# "
[2013-05-28 12:36:49:483 EDT] Running command on 192.168.71.206: cd /export/OVA/ismv2
[2013-05-28 12:36:50:484 EDT] Running command on 192.168.71.206: cd icon_1369758602843
[2013-05-28 12:36:51:486 EDT] Running command on 192.168.71.206: tar --delete -vf
../icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw
[2013-05-28 12:36:52:487 EDT] Running command on 192.168.71.206: tar --append -vf
../icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw
[2013-05-28 12:36:55:490 EDT] Running command on 192.168.71.206: tar -tvf
../icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova
[2013-05-28 12:36:55:491 EDT] Command output:
-rw-r--r-- root/0      21514 2013-05-28 12:29 icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ovf

```

```

-rw-r--r-- root/0      12377 2013-05-28 12:07 Automation.topology
-rw-r--r-- root/0      2754 2013-05-28 12:07 Semantic.topology
-rw-r--r-- root/0     15025 2013-05-28 12:29 osNode1369758599099.xml
-rw-r--r-- root/0     14851 2013-05-28 12:29 osNode1369758599099C.xml
-rw-r--r-- root/0     59105 2013-05-28 12:29 de-DE-bundle.msg
-rw-r--r-- root/0     77316 2013-05-28 12:29 en-US-bundle.msg
-rw-r--r-- root/0     57302 2013-05-28 12:29 es-ES-bundle.msg
-rw-r--r-- root/0     69920 2013-05-28 12:29 fr-FR-bundle.msg
-rw-r--r-- root/0     54175 2013-05-28 12:29 it-IT-bundle.msg
-rw-r--r-- root/0     54543 2013-05-28 12:29 ja-JP-bundle.msg
-rw-r--r-- root/0     49062 2013-05-28 12:29 ko-KR-bundle.msg
-rw-r--r-- root/0     53035 2013-05-28 12:29 pt-BR-bundle.msg
-rw-r--r-- root/0     40532 2013-05-28 12:29 zh-CN-bundle.msg
-rw-r--r-- root/0     41556 2013-05-28 12:29 zh-TW-bundle.msg
[2013-05-28 12:37:14:620 EDT] Running command on 192.168.71.206: PS1="# "
[2013-05-28 12:37:15:621 EDT] Running command on 192.168.71.206: cd /export/OVA/isv2
[2013-05-28 12:37:16:622 EDT] Running command on 192.168.71.206: cd icon_1369758602843
[2013-05-28 12:37:17:623 EDT] Running command on 192.168.71.206: tar -xvf
../icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova
--exclude=PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw
[2013-05-28 12:37:17:624 EDT] Command output:
tar: icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ovf: time stamp 2013-05-28 12:29:59 is
30943.425106174 s in the future
tar: Automation.topology: time stamp 2013-05-28 12:07:28 is 29592.424783957 s in the future
tar: Semantic.topology: time stamp 2013-05-28 12:07:28 is 29592.424698551 s in the future
tar: osNode1369758599099.xml: time stamp 2013-05-28 12:29:59 is 30943.424587061 s in the future
tar: osNode1369758599099C.xml: time stamp 2013-05-28 12:29:59 is 30943.424401575 s in the future
tar: de-DE-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.424195753 s in the future
tar: en-US-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.423977501 s in the future
tar: es-ES-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.423790179 s in the future
tar: fr-FR-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.423591926 s in the future
tar: it-IT-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.423341528 s in the future
tar: ja-JP-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.423159619 s in the future
tar: ko-KR-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.422988122 s in the future
tar: pt-BR-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.422812629 s in the future
tar: zh-CN-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.422646989 s in the future
tar: zh-TW-bundle.msg: time stamp 2013-05-28 12:29:59 is 30943.422416057 s in the future
[2013-05-28 12:37:48:657 EDT] Running command on 192.168.71.206: zip
../icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova.zip
icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ovf Automation.topology Semantic.topology
PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw osNode1369758599099.xml osNode1369758599099C.xml
de-DE-bundle.msg en-US-bundle.msg es-ES-bundle.msg fr-FR-bundle.msg it-IT-bundle.msg ja-JP-bundle.msg
ko-KR-bundle.msg pt-BR-bundle.msg zh-CN-bundle.msg zh-TW-bundle.msg
[2013-05-28 12:42:35:696 EDT] Running command on 192.168.71.206: rm -f
../icon_image_plantsbywebsphere_wasce_aix7_20gb_scs_1_0_0.ova
[2013-05-28 12:42:36:697 EDT] Running command on 192.168.71.206: cd /export/OVA/isv2
[2013-05-28 12:42:37:698 EDT] Running command on 192.168.71.206: ls -l --color=never *.zip
[2013-05-28 12:42:42:764 EDT] Running command on 192.168.71.206: cd /export/OVA/isv2
[2013-05-28 12:42:43:765 EDT] Running command on 192.168.71.206: cd icon_1369758602843
[2013-05-28 12:42:44:766 EDT] Running command on 192.168.71.206: rm -f *
[2013-05-28 12:42:55:768 EDT] Running command on 192.168.71.206: cd /export/OVA/isv2
[2013-05-28 12:42:56:769 EDT] Running command on 192.168.71.206: rmdir icon_1369758602843

==> Export Completed
[2013-05-28 12:42:57:770 EDT] Completed image OVA export.

==> Clean up files on Image Construction and Composition Tool server
[2013-05-28 12:42:57:772 EDT] cleanupExportFiles Found image files:
[/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaea3/export_136975724
8488/PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw]

```

```

[2013-05-28 12:42:57:772 EDT] cleanupExportFiles Loaded properties from image proxy file:
{STORAGECONTROLLERUSERNAME=USERID, REPOSITORYTYPE=SCS, REPOSITORYHOSTUSERPASSWORD=PASSWORD,
REPOSITORYHOSTUSERNAME=padmin, imageHost=192.168.71.203,
STORAGECONTROLLERUSERPRIVATEKEY=/resources/files/2c2d35fc-d550-46ca-bala-abd8dfa0278f/id_rsa,
REPOSITORYNAME=POWER_SCS_REPO, uniqueId=1369757248488, imageFileSize=21474836480,
imageFile=/home/padmin/PlantsByWebSphere-wasce-aix7-20GB-SCS1.1369757248488.raw.bz2}
[2013-05-28 12:42:57:773 EDT] Connecting to hosts: [192.168.71.203]
[2013-05-28 12:43:01:874 EDT] Running command on 192.168.71.203: rm -f
/home/padmin/PlantsByWebSphere-wasce-aix7-20GB-SCS1.1369757248488.raw.bz2
[2013-05-28 12:43:02:876 EDT] cleanupExportFiles Found files:
[/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_136975724
8488/PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/de-DE-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/en-US-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/es-ES-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/fr-FR-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/it-IT-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/ja-JP-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/ko-KR-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/osNode1369758599099.xml,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/osNode1369758599099C.xml,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/pt-BR-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/zh-CN-bundle.msg,
/drouter/ramdisk2/mnt/raid-volume/raid0/local/images/b79cf5f4-d650-4185-a859-6a6dcdcaaaa3/export_1369757248
488/zh-TW-bundle.msg]
[2013-05-28 12:43:02:876 EDT] cleanupExportFiles Deleted PlantsByWebSphere-wasce-aix7-20GB-SCS1.raw;
success = true
[2013-05-28 12:43:02:877 EDT] cleanupExportFiles Deleted de-DE-bundle.msg; success = true
[2013-05-28 12:43:02:878 EDT] cleanupExportFiles Deleted en-US-bundle.msg; success = true
[2013-05-28 12:43:02:878 EDT] cleanupExportFiles Deleted es-ES-bundle.msg; success = true
[2013-05-28 12:43:02:878 EDT] cleanupExportFiles Deleted fr-FR-bundle.msg; success = true
[2013-05-28 12:43:02:879 EDT] cleanupExportFiles Deleted it-IT-bundle.msg; success = true
[2013-05-28 12:43:02:879 EDT] cleanupExportFiles Deleted ja-JP-bundle.msg; success = true
[2013-05-28 12:43:02:879 EDT] cleanupExportFiles Deleted ko-KR-bundle.msg; success = true
[2013-05-28 12:43:02:880 EDT] cleanupExportFiles Deleted osNode1369758599099.xml; success = true
[2013-05-28 12:43:02:880 EDT] cleanupExportFiles Deleted osNode1369758599099C.xml; success = true
[2013-05-28 12:43:02:881 EDT] cleanupExportFiles Deleted pt-BR-bundle.msg; success = true
[2013-05-28 12:43:02:881 EDT] cleanupExportFiles Deleted zh-CN-bundle.msg; success = true
[2013-05-28 12:43:02:882 EDT] cleanupExportFiles Deleted zh-TW-bundle.msg; success = true
[2013-05-28 12:43:02:882 EDT] cleanupExportFiles Deleted export_1369757248488; success = true

```

---





## Constructing complex virtual appliances

As described in Chapter 1, “Introduction to virtual appliance construction” on page 1, the preferred solution for pattern deployment is to use IBM Workload Deployer to create, manage, and deploy sophisticated patterns. However, in some situations, you might not have access to the IBM Workload Deployer instance, but yet you want to deploy several appliances that are interconnected in a simple manner. For example, you might want to deploy a three-tier solution to IBM PureFlex System with IBM Flex System Manager. IBM Flex System Manager uses the IBM Systems Director VMControl plug-in for appliance management. With Systems Director VMControl, you can deploy several appliances individually, group them as a workload, and manage them as a unit.

This chapter explains how to construct several individual virtual appliances in the tool and how to deploy them through Systems Director VMControl. It provides an example of how to design and implement activation programs in the existing base appliances to manage middleware software self-activation. These base appliances, which contain the middleware software components, run on a KVM hypervisor. The appliances already have the operating system (OS)-level standard activation programs to handle, for example, the network interface, host name, and IP address, at the time the appliances are deployed.

The example shows you how to design, implement, and inject the required activation programs into their associated images to ensure that the middleware can be properly activated at deployment time. The technologies applied in the example take advantage of the features that are provided by the IBM Cloud-Computing architecture.

**Previous knowledge:** Before reading this chapter, you must be familiar with the following topics, which are explained in previous chapters of this book:

- ▶ IBM Image Construction and Composition Tool (referred to as *the tool* in this book)
- ▶ IBM Product Activator Development Kit (PADK)
- ▶ IBM Virtual Solutions Activation Engine (VSAE)
- ▶ IBM Systems Director VMControl

This chapter includes the following sections:

- ▶ Scenario overview
- ▶ Importing running virtual machines
- ▶ Preparing for activation programs
- ▶ Implementing the activation programs
- ▶ Creating software bundles
- ▶ Building virtual appliances
- ▶ Validating the results
- ▶ Summary

## 10.1 Scenario overview

The scenario in this chapter simulates production environments that consist of a three-tier topology (an edge web server, an application server, and a database server). All servers in this example run SUSE Linux Enterprise Server (SLES) 11 SP1.

Each tier has the following details:

- ▶ Web server: IBM HTTP Server v7.0.0.21
- ▶ Application server: IBM WebSphere Application Server Express v7.0.0.21
- ▶ Database server: IBM DB2 Server v9.0.21

At the start of the project, the three-tier virtual workload runs as three separate virtual machines (VMs). The first step is to import these running VMs into the virtual appliance repository of the Image Construction and Composition Tool.

**More information:** For information about situations where the use case in this chapter applies to your environment, see 12.3, “Preinstalling software or creating a bundle with the installation image” on page 269.

To ensure that each software component can be successfully activated in a new environment, their configurable resources are analyzed. For example, the host names are required to be customized when deploying the three servers in a new environment. After the required configurable resources are identified, the PADK is used to create, test, and validate their corresponding activation programs. These activation programs automatically manage the parameter customization at the virtual appliance deployment time.

After the activation programs are implemented and verified, they are imported into Image Construction and Composition Tool as the additional software bundles. These activation program bundles are eventually integrated with IBM Virtual Solutions Activation Engine (VSAE) to automatically reconfigure identified resources during the appliance deployment phase without any human intervention.

Another consideration is the inter-component dependency between the software components. In this example, the WebSphere Application Server requires the DB2 host name and port number at deployment time. An activation program for reconfiguring the DB2 specific data source object in WebSphere is implemented in PADK and imported into the tool. Furthermore, the intercomponent dependency determines the deployment sequence of these related software components.

At this time, the activation program development is completed for both the configurable resources and the intercomponent dependency. This scenario then goes through the virtual appliance build process of the tool to add the activation bundles to their corresponding base appliances and to synchronize the extended virtual appliances.

The final step is validation. The virtual appliances are imported and deployed onto the controlled infrastructure of System Director VMControl to ensure that they are properly configured and that they can communicate with each other after the deployment.

The following sections highlight the end-to-end process to implement activation programs into three-tier existing base appliances to manage IBM HTTP Server, WebSphere Application Server, and DB2 activation.

## 10.2 Importing running virtual machines

Before you import the VM image into the IBM Image Construction and Composition Tool, clone the image. Cloning the virtual image protects the original copy from undesired modifications. The *clone* capability provided by the KVM virtual machine manager (*virt-manager*) is used to generate three appliance clones for the IBM HTTP Server, the WebSphere Application Server, and the DB2 Database Server.

If an existing VSAE is on the VM, the tool will upgrade it to the latest version.

The three cloned appliances are imported into the tool individually. The three new images are displayed in the GUI as the base images. For information about this procedure, see “Importing of a running VM function fails” on page 116.

When importing a VM, the tool copies the VM disk files into the virtual appliance repository (*/var/lib/va/workloads/*) and generates a corresponding Open Virtualization Format (OVF) file for the imported image. The OVF file is generated from the *generic.ovf* template file in the */var/lib/vafes/template* directory. In addition, its deployment parameters are shown in the GUI (Figure 10-1 on page 224). These parameters help to reconfigure configuration details, such as the system network interface and system user credentials, at the operating system level. These default values are overridden at deployment time for the virtual appliance.



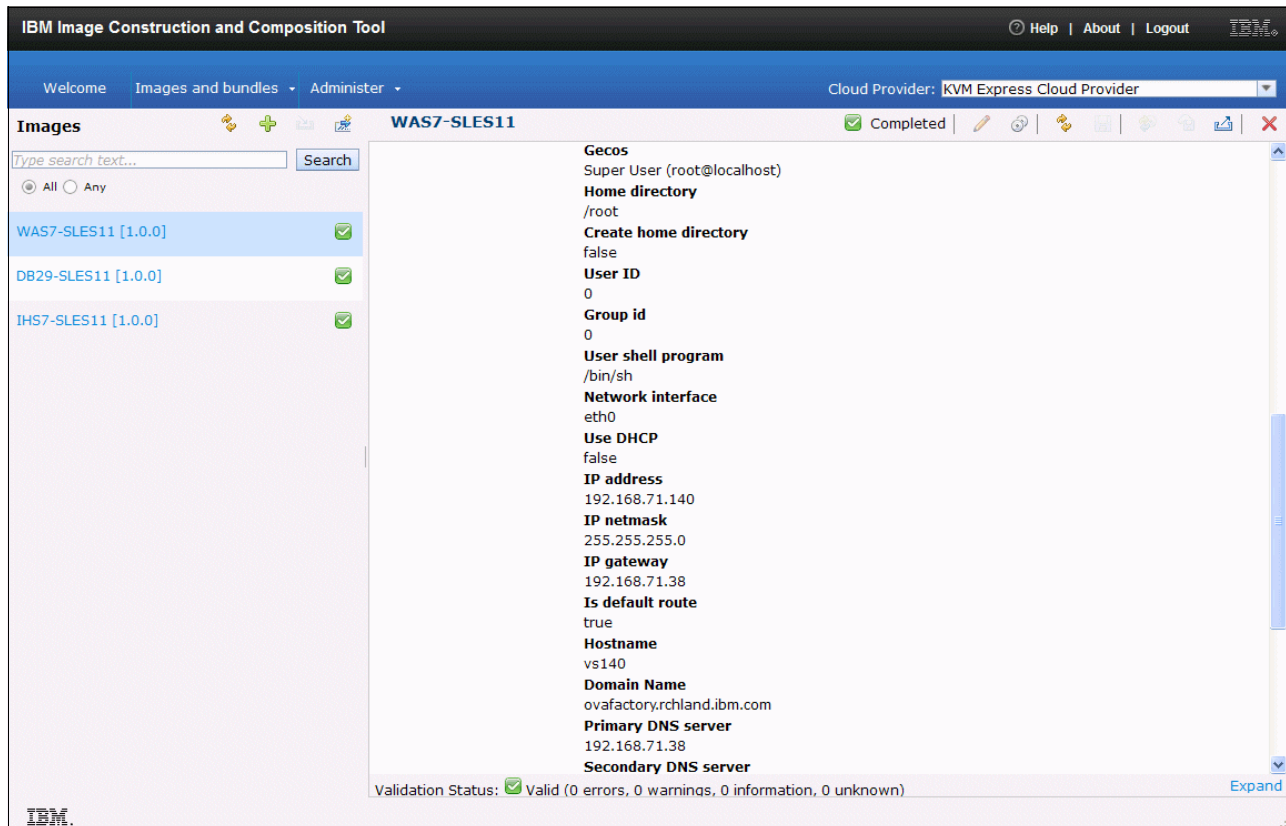


Figure 10-1 Default deployment parameters

The next step is to prepare the activation programs for the IBM HTTP Server, the WebSphere Application Server, and the DB2 server.

## 10.3 Preparing for activation programs

This section explains the process to prepare activation programs.

### 10.3.1 Installing PADK

To install Eclipse and Product Activation Development Kit plug-ins for designing, implementing, and testing the activation programs, see 5.2, “Installing PADK” on page 75.

### 10.3.2 Examining the configurable resources

To properly install and deploy the software components in a new environment, you must thoroughly examine their critical configurable resources.

In the scenario in this chapter, you must set the host name of WebSphere Application Server at the deployment time. For DB2, in addition to its host name, you must configure several runtime-specific parameters, such as instance user names, instance passwords, database names, and license type. For the IBM HTTP Server and its WebSphere plug-in, you must specify the server name, home directory, and configuration file name at deployment time.

In this example, we use the host name customization for the three middleware components to illustrate the additional preparation steps. You can use the **wsadmin** shell script **AdminTask.changeHostName** command to change the WebSphere host name. For more information, see the “Utility command group of the AdminTask object” topic in the WebSphere Application Server Information Center:

[http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Frxml\\_atutility.html](http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Frxml_atutility.html)

The Jython command has the following syntax:

```
wsadmin.sh -lang jython -conntype NONE -c \"AdminTask.changeHostName('[-nodeName  
node_name -hostName host_name]')\"
```

The activation program for reconfiguring the WebSphere host name must provide the **wsadmin** command with two parameters, the node name and new host name.

For DB2 host name customization on Linux, the following major steps are required for most of the DB2 Editions:

1. Update the DB2SYSTEM registry variable:

```
db2iset -g DB2SYSTEM = <new hostname>
```

2. Update the DB2 admin configuration file:

```
db2 update admin cfg using DB2SYSTEM <new hostname>  
db2 update admin cfg using SMTP_SERVER <new hostname>
```

For more information, see the “db2 - Command line processor invocation command” topic in the IBM DB2 database for Linux, UNIX, and Windows Information Center:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/topic/com.ibm.db2.luw.admin.cmd.doc/doc/r0010409.html>

For the IBM HTTP Server web server, you must modify the host name specified in its configuration file, `httpd.conf`, from the current host name to the new host name.

After you identify the critical changeable resources, study the methods (such as the steps, command syntax, and configuration file names) to code the activation programs to reconfigure these resources.

### 10.3.3 Identifying the intercomponent dependency

For a typical multitier topology that involves a set of preinstalled and preconfigured virtual appliances, generally an intercomponent dependency occurs between two adjacent software components.

For the sample three-tier topology, the data source configuration for WebSphere Application Server requires the DB2 server host name, the port number, the database name, and DB2 user credentials. It also requires the IBM HTTP Server host name, user credentials, and server name to modify the plug-in configuration file. This information must be available at deployment time so that the WebSphere applications can interact with its web and database servers.

The intercomponent dependency affects the sequence of the startup order for these software components in the topology. Because a dependency exists between WebSphere Application Server and its database server, you must activate the DB2 virtual appliance before activating the virtual appliance for WebSphere Application Server.

After you examine the configuration parameters, study the methods to modify these parameters, and identify the intercomponent dependencies, you can implement the activation programs.

## 10.4 Implementing the activation programs

This section highlights the process of implementing the activation programs based on the sample scenario. In addition, two code examples are provided for demonstration purposes.

### 10.4.1 Creating PADK projects in Eclipse

For this scenario, three PADK projects are created for reconfiguring the IBM HTTP Server, WebSphere Application Server, and DB2 changeable resources. To create a PADK project, see Chapter 5, “Product Activator Development Kit” on page 73.

Figure 10-2 shows the PADK project that was created for the host name reconfiguration for WebSphere Application Server. The two properties (`was_hostname` and `was_nodename`) added in the project are required for the following command:

```
wsadmin "AdminTask.changeHostName"
```

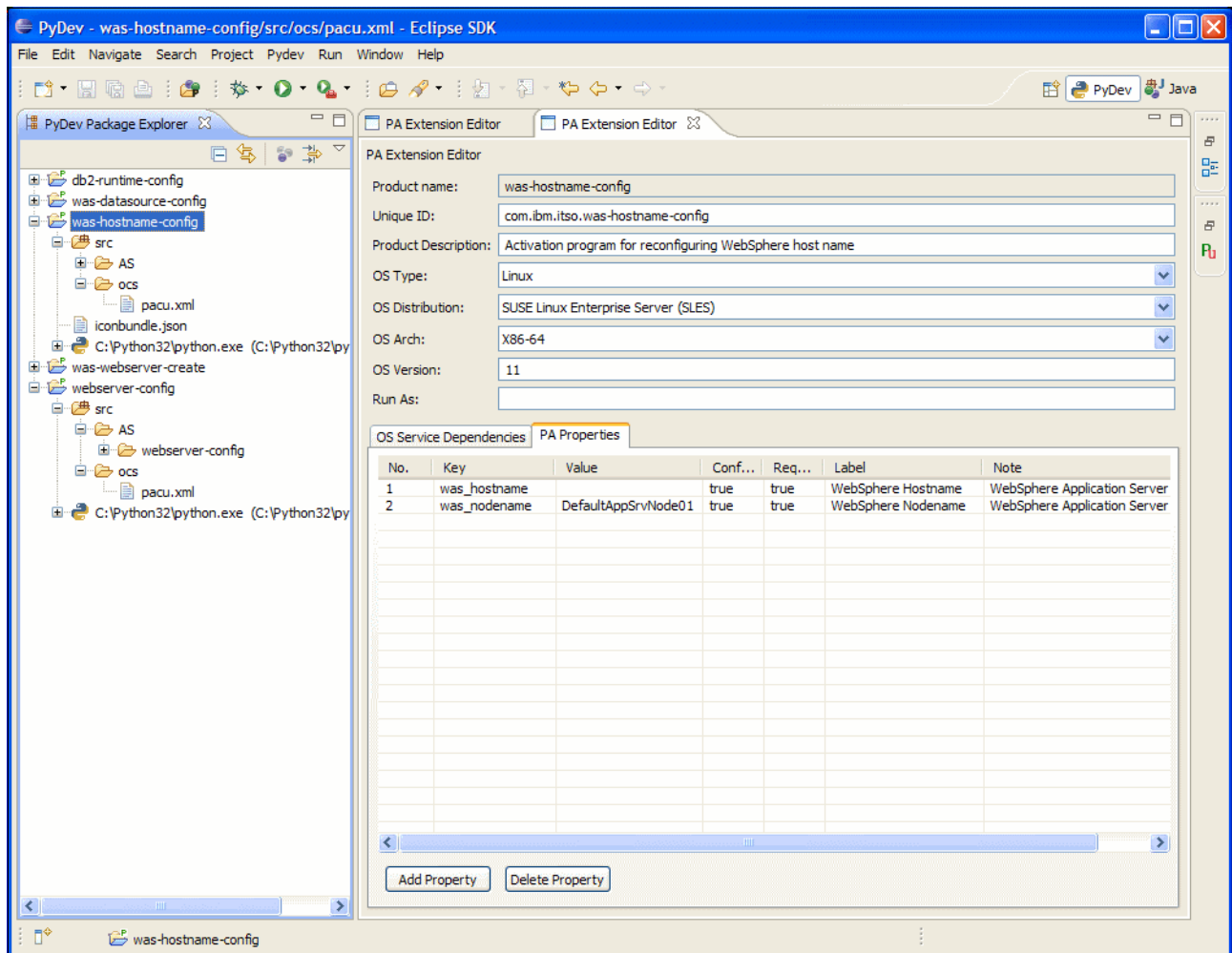


Figure 10-2 PADK project for reconfiguring WebSphere host name

The same principles apply when creating two more PADK projects for reconfiguring the critical parameters for DB2 and the IBM HTTP Server.

Figure 10-3 shows the db2-runtime-config PADK project. The most important parameter for DB2 boot time customization is updating its host name. Furthermore, a set of properties that is relevant to user credentials are identified so that the virtual appliance deployer can modify, for example, the DB2 instance user ID and password, DB2 fenced user password, and DB2 administration server password.

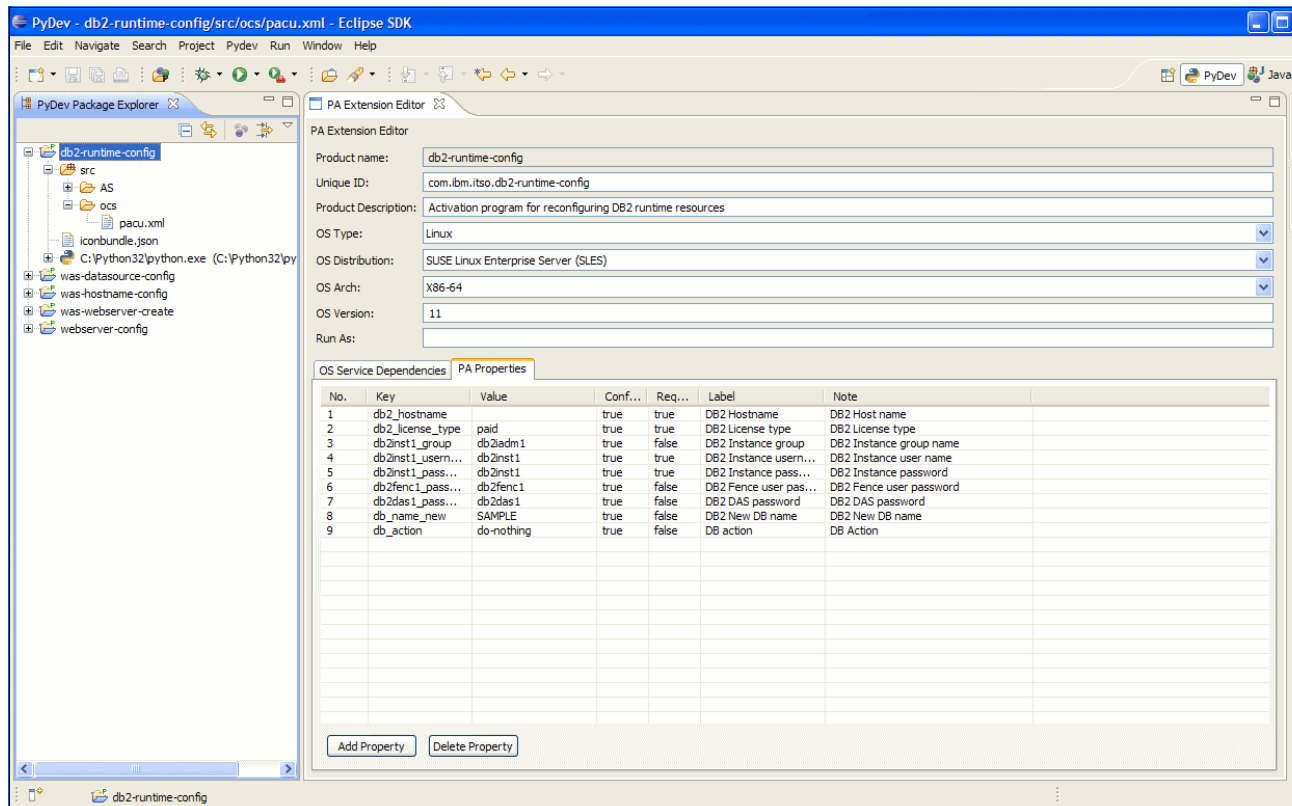


Figure 10-3 PADK project for reconfiguring DB2 run time

For the IBM HTTP Server web server (Figure 10-4), in addition to the server host name, the IBM HTTP Server home directory location and the configuration file name are specified to locate the configuration file for customizing the server host name.

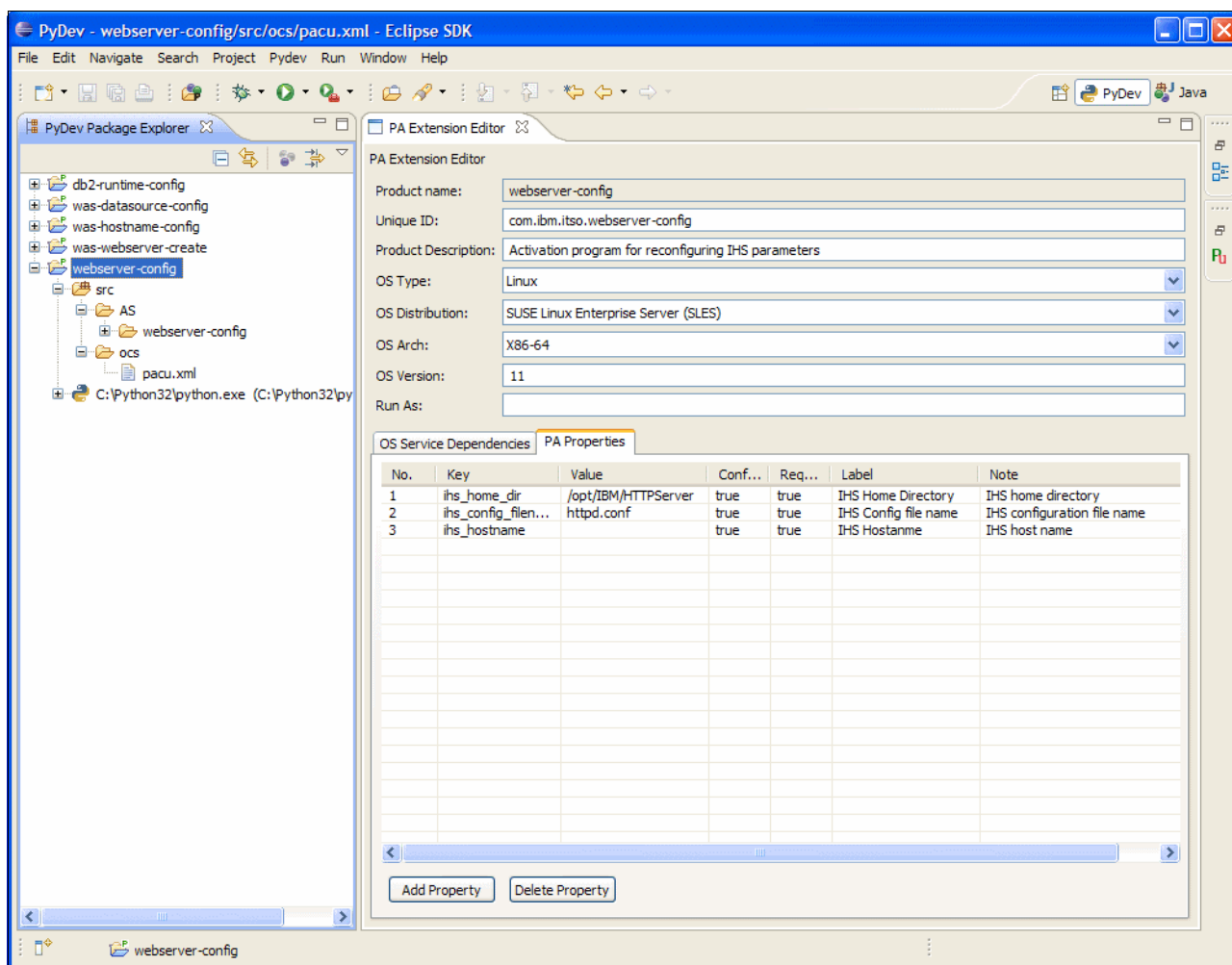


Figure 10-4 PADK project for reconfiguring IBM HTTP Server parameters

## 10.4.2 Coding activation programs for changeable parameters

The PADK plug-in creates one activation program (`activate.py`), one reset program (`reset.py`), one helper program (`esutil.py`), and one metadata file (`pacu.xml`) in each project. The `activate.py` program is the activation program that is started at software deployment time to reconfigure its identified properties.

The base code structure is generated in the `activate.py` source file based on the properties that are added at PADK project creation time. You must examine the source code and customize it to fit with the activation requirements for your software components.

Example 10-1 on page 229 shows the `activate.py` code that was implemented for the `was-hostname-config` PADK project. The program first retrieves the `was_hostname` and `was_nodename` values as provided by the deployment platform at the deployment time. If `was_hostname` is empty, the program retrieves the actual host name from a system call. If there is no input for `nodename`, the default node name (`DefaultAppSrvNode01`) is used. After the two

parameters are set, the program constructs the **wsadmin** command and runs the command to change the WebSphere host name.

*Example 10-1 The activate.py program for reconfiguring the WebSphere host name*

---

```
#!/usr/bin/python
...
class Activator:
    def __init__(self):
        try:
            #don't remove the lines below
            #BEGIN_KEYS_MARKER
            keys=["help", "was_hostname=", "was_nodename="]
            #END_KEYS_MARKER

            argmap = esutil.parse_cmd_args( argv, "hv", keys )
        except getopt.GetoptError, err:
            print str(err)
            self._usage()
            exit(2)

        if argmap.has_key( "-v" ):
            verbose = True
        elif argmap.has_key("-h") or argmap.has_key("--help"):
            self._usage()
            exit()

        # Get returns None if key is not found
        ...

    def _usage(self):
        #don't remove the lines below
        #BEGIN_KEYVALUES_MARKER
        print """\nUSAGE:\n AS/was-hostname-config/"" + os.path.basename(argv[0]) + """
--was_hostname --was_nodename DefaultAppSrvNode01"""
        #END_KEYVALUES_MARKER

    def run_was_hostname_config(self):
        # Construct parameters for the ConfigDB2 shell script. ConfigDB2 sets default values for parameters
        # not provided.
        self.wasadmin_cmd = WAS_PROFILES + "/" + PROF_NAME + "/bin/wsadmin.sh -lang jython -conntype NONE -c
"
        self.wasadmin_cmd = self.wasadmin_cmd + "\"AdminTask.changeHostName ('[-nodeName "

        if self.was_nodename != None and self.was_nodename != "":
            self.wasadmin_cmd = self.wasadmin_cmd + self.was_nodename + " -hostName "
        else :
            self.wasadmin_cmd = self.wasadmin_cmd + NODENAME + " -hostName "

        if self.was_hostname != None and self.was_hostname != "":
            self.wasadmin_cmd = self.wasadmin_cmd + self.was_hostname + " ]')\""
        else :
            import platform; HOSTNAME = platform.node()
            print "Retrieved hostname using platform.node(): " + HOSTNAME
            self.wasadmin_cmd = self.wasadmin_cmd + HOSTNAME + " ]')\"

        self.wasadmin_cmd = self.wasadmin_cmd + " -c \"AdminConfig.save()\""

        ...
        os.system(self.wasadmin_cmd)
```

```

def action(self):
    #Write your activation code here
    self.run_was_hostname_config()

def main():
    #don't remove the line below
    #_____IBMDEBUGMARKER_____
    activator = Activator()
    activator.action()

#####
#### Main ####
#####

if __name__ == "__main__":
    main()

```

---

The activation programs for reconfiguring DB2 host name require several steps to complete the work, as explained in 10.3.2, “Examining the configurable resources” on page 224. In addition, this DB2 virtual appliance reconfigures some of its user credentials as shown in Figure 10-3 on page 227. Because this example is more complicated, a new activation helper program was created to reconfigure DB2 server host name and other parameters that follow the DB2 required commands. At deployment time, the DB2 `activate.py` starts its helper program after all of the parameters are first set.

To replace the host name in the IBM HTTP Server web server, the activation program gets its configuration file (`httpd.conf`) location at deployment time and changes the *ServerName* in the `httpd.conf` file to the new host name.

After the activation programs are coded, follow the steps in Chapter 5, “Product Activator Development Kit” on page 73, to test these newly created activation programs. Then, package them as bundles and transfer the bundles to the Image Construction and Composition Tool.

### 10.4.3 Coding activation programs for intercomponent dependency

In addition to implementing the activation programs for critical changeable resources, you must also consider the intercomponent dependency between the software components. This section uses the dependency between WebSphere Application Server and the DB2 server as an example.

Figure 10-5 shows the WebSphere Application Server console. WebSphere requires database information, database name, server name, and port number to make a connection to the DB2 server. How is this information configured in WebSphere?

The screenshot displays the WebSphere Application Server console interface. On the left is a navigation tree with categories like Welcome, Servers, Applications, Services, Resources, Security, Environment, System administration, Users and Groups, Monitoring and Tuning, Troubleshooting, Service integration, and UDDI. The main content area is titled 'DB2 Universal Driver Datasource' and includes sections for 'Data store helper class name', 'Security settings', and 'Common and required data source properties'.

**Data store helper class name**

☒ Select a data store helper class

Data store helper classes provided by WebSphere Application Server

- DB2 Universal data store helper**  
(com.ibm.websphere.rsadapter.DB2UniversalDataStoreHelper)
- DB2 for iSeries data store helper  
(com.ibm.websphere.rsadapter.DB2AS400DataStoreHelper)

☐ Specify a user-defined data store helper

Enter a package-qualified data store helper class name

**Security settings**

Select the authentication values for this resource.

Component-managed authentication alias: (none)

Mapping-configuration alias: (none)

Container-managed authentication alias: (none)

**Common and required data source properties**

| Name            | Value    |
|-----------------|----------|
| * Driver type   | 4        |
| * Database name | SampleDB |
| * Server name   | vmcvs208 |
| * Port number   | 50001    |

Buttons: Apply, OK, Reset, Cancel

Figure 10-5 WebSphere Application Server Console



A new PADK package, was-datasource-config, is created to reconfigure data source information for WebSphere Application Server a Figure 10-6.

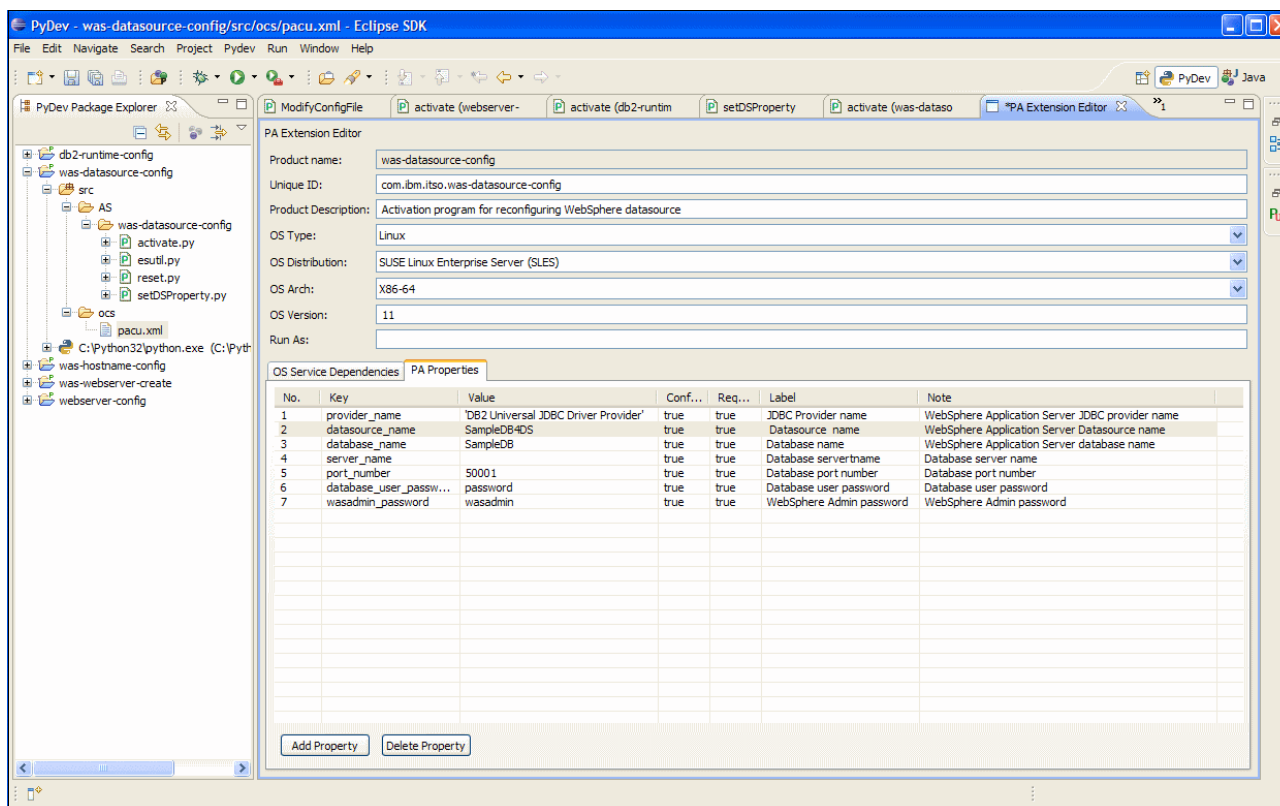


Figure 10-6 PADK package for reconfiguring data source in WebSphere

The code example (setDatasourceProperty.py) in Example 10-2 retrieves and sets the required data source parameter values. It then uses the **wsadmin** commands to reconfigure each data source information, for example, the database server name and port number.

#### Example 10-2 The setDatasourceProperty.py program

```
#!/usr/bin/python
```

```
...
```

```
class Activator:
    def __init__(self):
        try:
            argmap = esutil.parse_cmd_args( argv, "hv", ["help", "provider_name=",
"datasource_name=", "database_name=", "server_name=", "port_number=", "database_user_password=",
"wasadmin_password="] )

            except getopt.GetoptError, err:
                print str(err)
                self._usage()
                exit(2)

            if argmap.has_key( "-v" ):
                verbose = True
            elif argmap.has_key("-h") or argmap.has_key("--help"):
```

```

        self._usage()
        exit()

    # Get returns None if key is not found
    ...

    def _usage(self):
        print '\nUSAGE:\n AS/was-datasource-config/' + os.path.basename(argv[0]) + '
--provider_name provider_name --datasource_name datasource_name --database_name database_name
--server_name server_name --port_number port_number --database_user_password password
--wasadmin_password password'

    def run_datasource_config(self):
        # Construct parameters
        # These two are needed so set to defaults if not passed...
        if self.provider_name == None or self.provider_name == "":
            self.provider_name = "'DB2 Universal JDBC Driver Provider'"
        if self.datasource_name == None or self.datasource_name == "":
            self.datasource_name = "SampleDB4DS"

        #Configure the databaseName resource property
        if self.database_name != None and self.database_name != "":
            os.system("/opt/IBM/WebSphere/AppServer/bin/wsadmin.sh -lang jython -conntype NONE
-f AS/was-datasource-config/setDSProperty.py " + self.provider_name + " " + self.datasource_name
+ " " + "databaseName" + " " + self.database_name)
        #Configure the serverName property
        if self.server_name != None and self.server_name != "":
            os.system("/opt/IBM/WebSphere/AppServer/bin/wsadmin.sh -lang jython -conntype NONE
-f AS/was-datasource-config/setDSProperty.py " + self.provider_name + " " + self.datasource_name
+ " " + "serverName" + " " + self.server_name)
        #Configure the portNumber property
        ...

        os.system(self.cmd)
        self.cmd = "/opt/IBM/WebSphere/Profiles/DefaultAppSrv01/bin/startServer.sh server1"
        os.system(self.cmd)
        print "WAS reconfigured and restarted...."

    def action(self):
        #Configure DB2 instance
        self.run_datasource_config()

def main():
    activator = Activator()
    activator.action()

#####
##### Main #####
#####

if __name__ == "__main__":
    main()

```

---

A similar PADK project is created for handling the dependency between WebSphere Application Server and the IBM HTTP Server web server. This activation program sets up the IBM HTTP Server plug-in connectivity information in WebSphere to communicate with its web server.

After the activation programs that handle the dependencies between software components are completed and tested, they are integrated with the Image Construction and Composition Tool to become software bundles.

## 10.5 Creating software bundles

The software bundles can be used for installing and configuring the operating system, middleware software components, or user applications. In this scenario, as explained in the introduction, they are used to modify the changeable resources of each middleware software component and to resolve the intercomponent dependencies.

Before creating software bundles from the PADK activation program packages to the Image Construction and Composition Tool, you must test all activation programs to ensure successful operation. See Chapter 5, “Product Activator Development Kit” on page 73, to complete the following tasks:

1. Set up a connection to the target VM from Eclipse.
2. Run and debug activation programs.
3. Establish a connection between Eclipse and the Image Construction and Composition Tool.
4. Create a software bundle on the Image Construction and Composition Tool.

Figure 10-7 shows the result of creating the was-hostname-config bundle on the Image Construction and Composition Tool from Eclipse IDE. The bundle in its correct format is stored directly in the bundle repository.

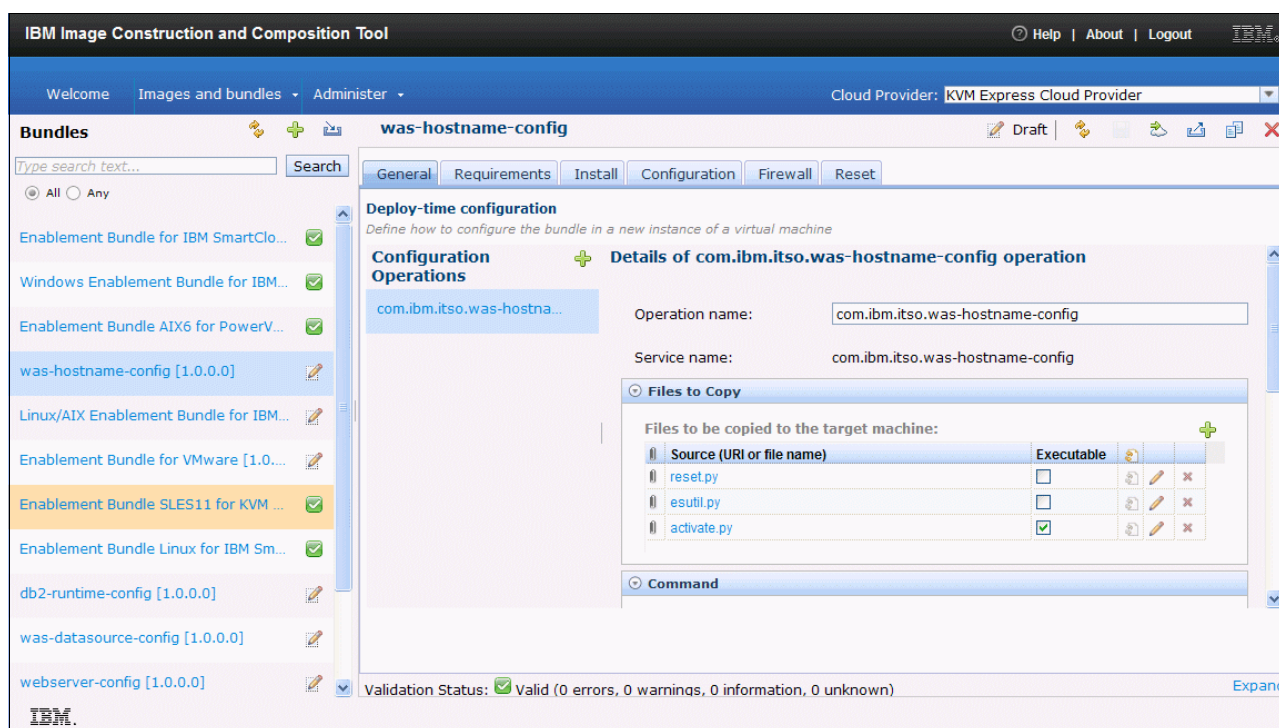


Figure 10-7 The was-hostname-config software bundle

All of the PADK activation program packages for both changeable resources and inter-component dependencies, such as db2-runtime-config, webserver-config, was-datasource-config, and was-webserver-create, are created in the Image Construction and Composition Tool as software bundles.

By using the Image Construction and Composition Tool, the user can specify the required input parameter style for the installation, configuration, and reset software bundle scripts. For example, the input parameter style of the was-hostname-config activation bundle configuration script is a double dash. Therefore, the Long Space Style is selected, as shown in Figure 10-8.

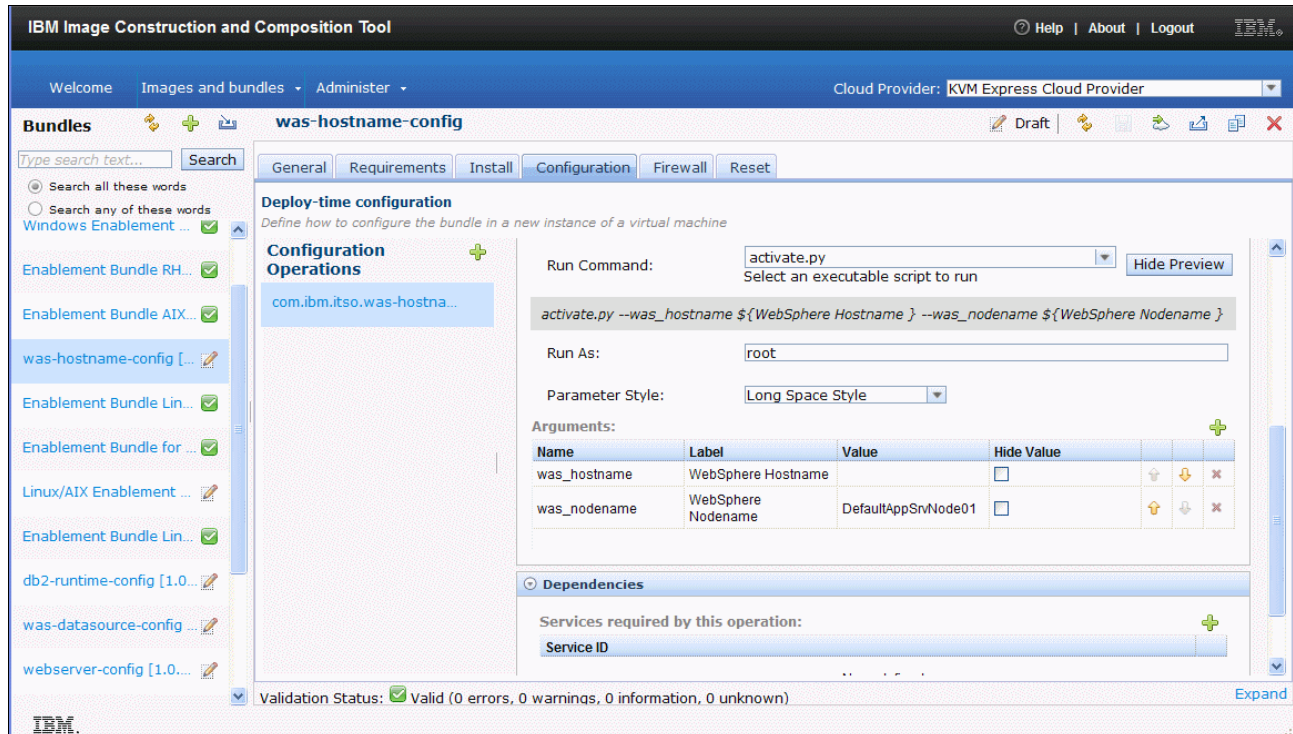


Figure 10-8 The was-hostname-config activation bundle with Long Space Style

Short Space Style is selected for the db2-runtime-config activation bundle as shown in Figure 10-9, because the input parameter style of its configuration script is a single dash.

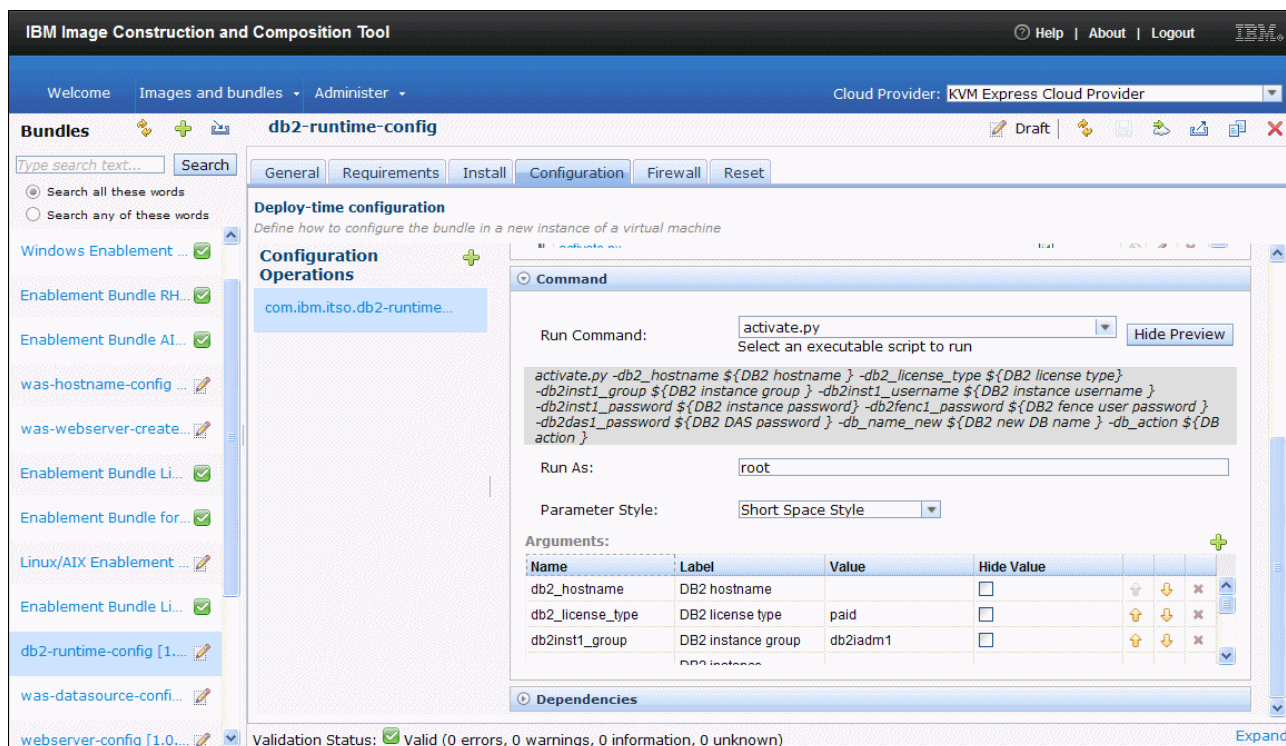


Figure 10-9 The db2-runtime-config activation bundle with Short Space Style

## 10.6 Building virtual appliances

The software bundles for reconfiguring these three software components (IBM HTTP Server, WebSphere Application Server, and DB2) are ready to be added into their associated base images. This section explains how to build three virtual appliances for IBM HTTP Server, WebSphere Application Server, and DB2 based on their base images and the associated activation program bundles. Here, the base image of WebSphere Application Server and its software activation bundles are used to illustrate the process of building a virtual appliance.

For more information, see *Image Construction and Composition Tool, Version 1.0 User Guide*:

[ftp://public.dhe.ibm.com/software/webservers/workloaddeployer/v31/imageconstructiontool\\_guide.pdf](ftp://public.dhe.ibm.com/software/webservers/workloaddeployer/v31/imageconstructiontool_guide.pdf)

### 10.6.1 Adding software bundles to base images

To add the corresponding software activation bundles to the WebSphere Application Server base image, complete the following steps:

1. In the Image Construction and Composition Tool, do these tasks:
  - a. Select **Images and bundles** → **Build and manage images**.
  - b. Select the WebSphere base image, and then click the **Extend** icon (🔗) to create an extended image.

The base image metadata information is copied to the extended image. The extended image name (*VA-WAS7-SLES11*), the universal ID, the version, and the description are provided.

2. Select **VA-WAS7-SLES11**, and click the **Edit** icon (✎) to edit the extended image metadata.
3. In the Software Bundles section, click the **Add bundle** icon (+) to select and add was-hostname-config, was-datasource-config, and was-webserver-create bundles.
4. Customize the installation and deployment parameters, and the license information if necessary.
5. Click the **Save** icon (💾) to persist the changes.

Multiple software bundles can be selected and added at the same time. Remember to use the up (⬆) and down (⬇) icons to specify the order of installation for these software bundles. At synchronization time, the software bundles are run according to the order specified in the image.

After the bundles are added to the base image, their associated changeable parameters are also included as shown in Figure 10-10.

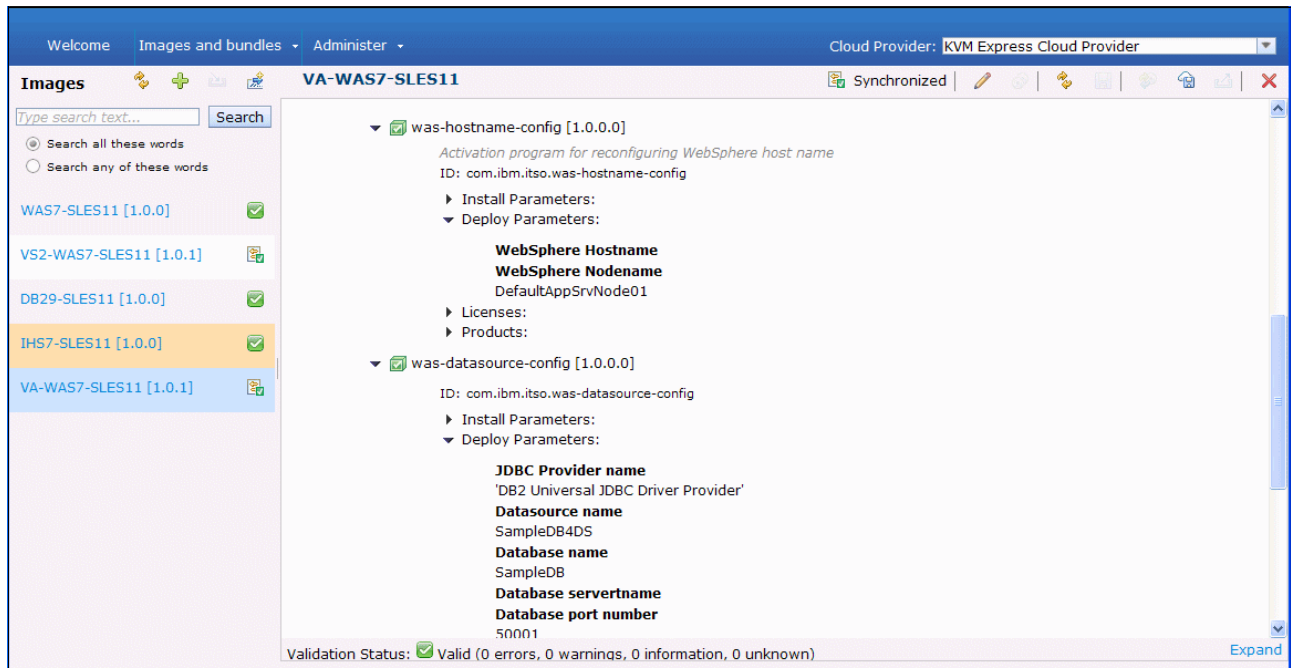


Figure 10-10 Addition of three WebSphere activation program bundles

The same procedure applies to the web server and the database server to build their final virtual appliances. The db2-runtime-config bundle is added to the DB2 extended image, and the webserver-config bundle is added to the web server extended image.

## 10.6.2 Synchronizing virtual appliances

Immediately after you add new bundles and save the changes, the state of an extended image changes to *Out of sync*. Click the **Synchronize** icon (↻) to start the synchronization process. Repeat this action for each of the three virtual appliances for IBM HTTP Server, WebSphere, and DB2.



For each appliance, the KVM cloud provider picks up an inactive IP address from its network configuration list to create a running VM. The actual virtual system information is displayed in the right-side pane of the Image Construction and Composition Tool in the Virtual System section.

During synchronization, the tool starts a VM for the base image clone and then installs the software bundles in the order as specified in the Software Bundles section. Figure 10-11 shows that the synchronization of WebSphere Application Server extended image is completed.

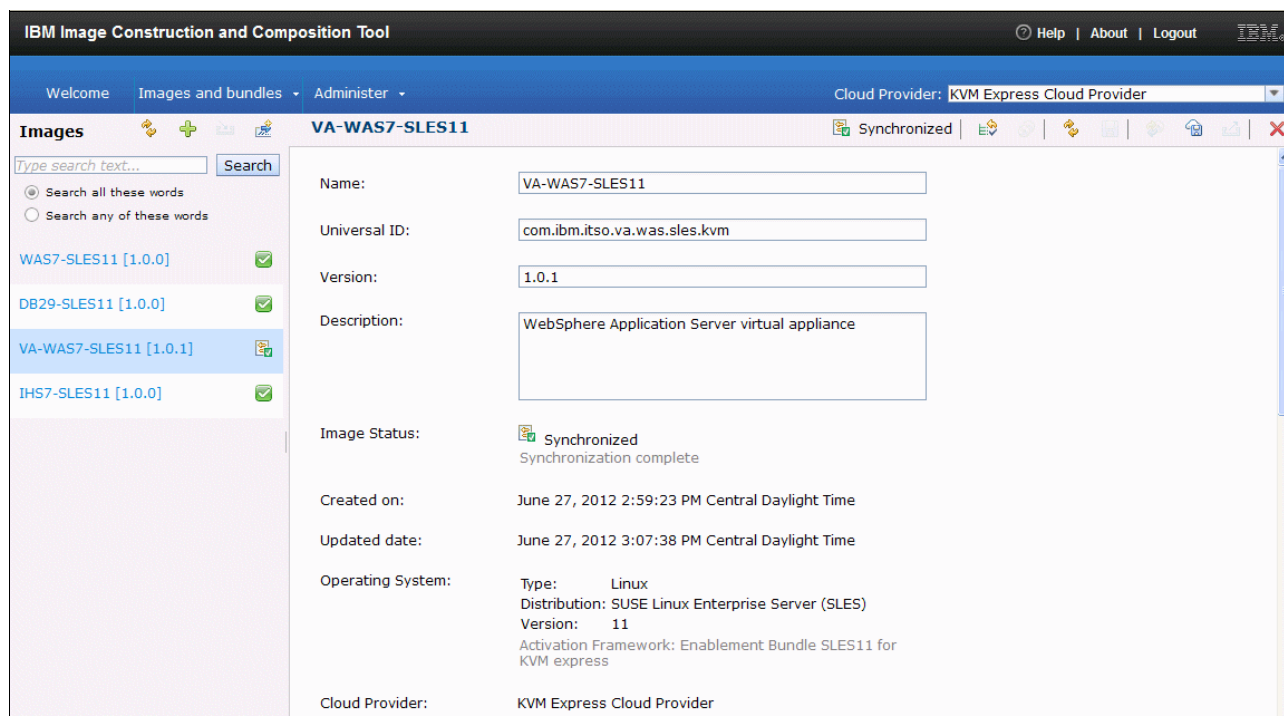


Figure 10-11 Synchronized WebSphere Application Server extended image

After the WebSphere virtual appliance synchronization is completed, use `ssh` to log in to the actual virtual system.

As shown in Figure 10-12, the WebSphere Application Server virtual appliance is successfully configured with its new activation programs. These newly created activation programs are integrated with the VSAE and installed in the `/opt/ibm/ae/AS/` directory. At deployment time, these activation programs are started so that they can reconfigure the WebSphere Application Server host name and its web server and database server information.

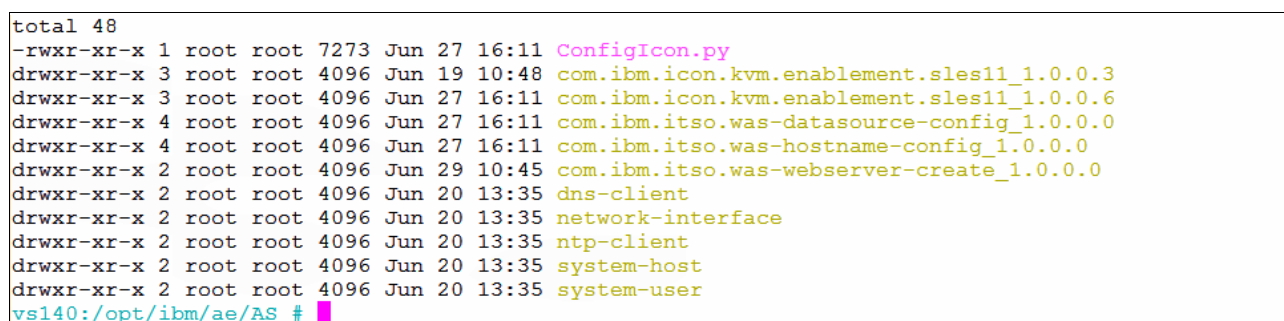


Figure 10-12 New activation programs installed in the application server after synchronization

Complete the same actions, as explained in this section, to synchronize the IBM HTTP Server and DB2 instances.

### 10.6.3 Capturing a virtual appliance

After the three extended images are synchronized and verified successfully, create the Open Virtual Appliance (OVA) tarballs that contain the virtual disk images and the corresponding OVF metadata. The syntax and semantics of the metadata are compliant with the target deployment platform, which is Systems Director VMControl in this case.

Click the **Capture** icon (📷) to create three physical copies of the image disks for IBM HTTP Server, WebSphere Application Server, and DB2. These image disks are stored in the KVM provider virtual appliance repository (VARepo).

Figure 10-13 shows the results of the WebSphere Application Server extended image capture.

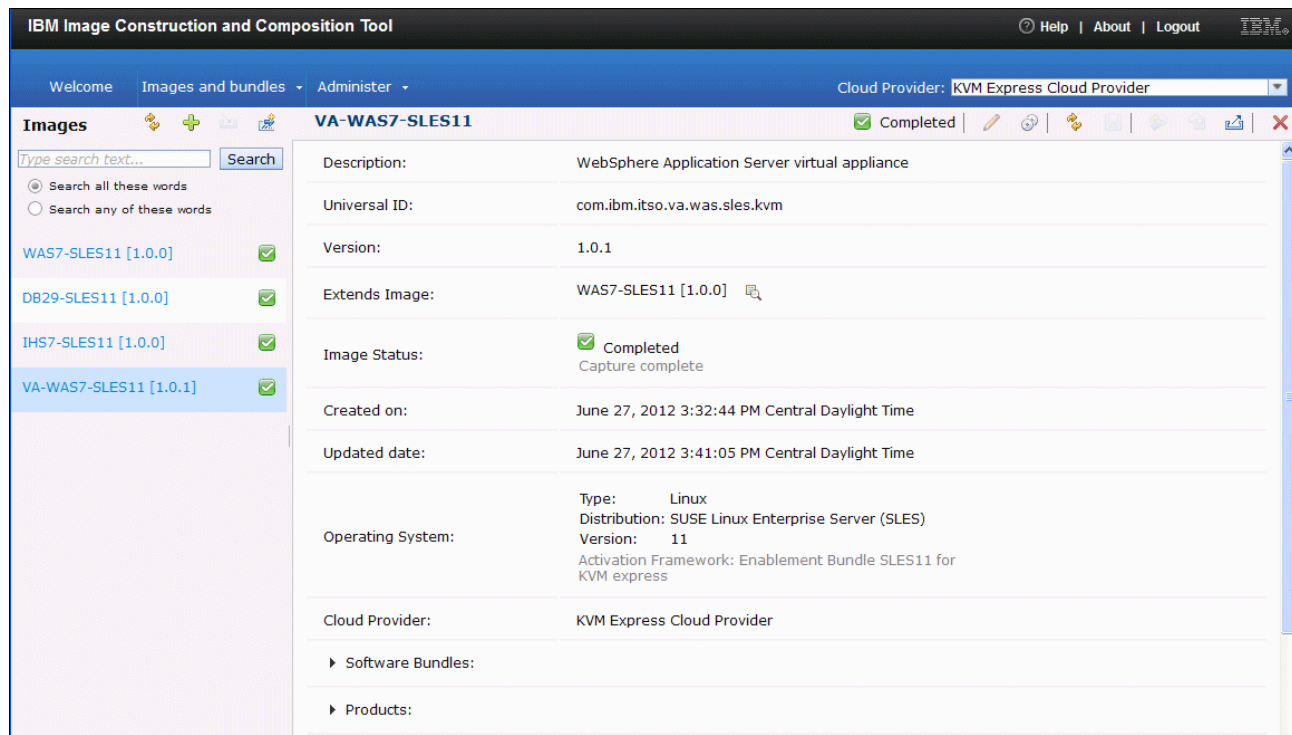


Figure 10-13 Captured extended image for WebSphere Application Server



## 10.6.4 Exporting virtual appliances

After the three virtual appliances are captured, click the **Export** icon (📄) to export them individually as OVA tarballs that are ready for import into the Systems Director VMControl virtual appliance repository (Figure 10-14).

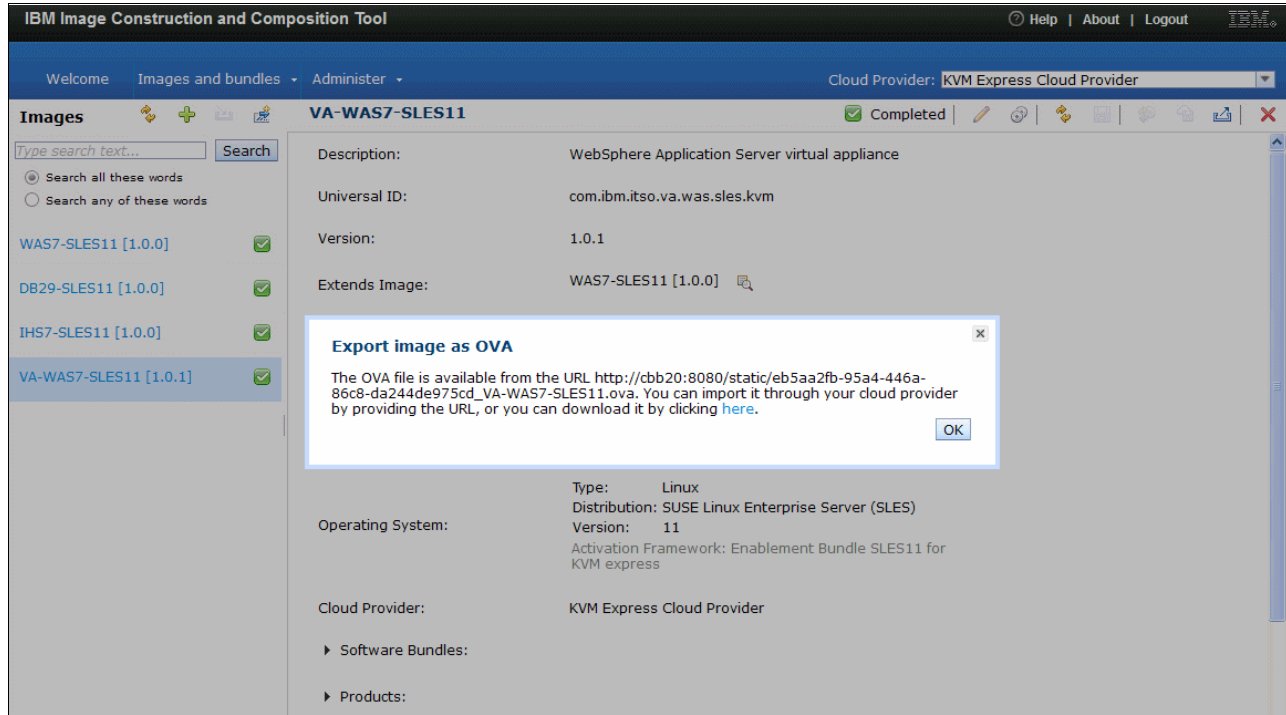


Figure 10-14 Export the image as OVA file

## 10.7 Validating the results

The three software virtual appliances, which are IBM HTTP Server, WebSphere Application Server, and DB2, are now configured with their activation programs to manage the software reconfiguration.

The IBM Systems Director VMControl plug-in, for example on the IBM PureFlex System, provides a graphical interface that a data center administrator can use to easily manage the virtual appliances and virtual server lifecycle. In our method, we use the Systems Director VMControl for the final validation of the software virtual appliances.

For more information, see *IBM Systems Director VMControl Installation and User's Guide Version 2.4.1*:

[http://pic.dhe.ibm.com/infocenter/director/pubs/topic/com.ibm.director.vim.helps.doc/fsd0\\_vim\\_pdf.pdf](http://pic.dhe.ibm.com/infocenter/director/pubs/topic/com.ibm.director.vim.helps.doc/fsd0_vim_pdf.pdf)

To begin, you import each virtual appliance OVA file that is generated by the Image Construction and Composition Tool into Systems Director VMControl. After the OVA file is successfully imported, the virtual appliance is ready to deploy. At deployment time, all of the identified configurable parameters are displayed in the Systems Director VMControl deployment panels. These parameters include the default parameters (see Figure 10-1 on page 224) and the parameters for each software component (see Figure 10-10 on page 237).

For more information about deployment options, see Chapter 11, “Virtual appliance deployment options” on page 245.

If the appliance is deployed, but one or more of the software components in the stack are not properly reconfigured, see “When deploying, reconfiguration of a component fails” on page 121, for troubleshooting guidelines.

## 10.7.1 A deployment documentation template

As explained previously, do not manually deploy multiple interdependent appliances. The IBM SmartCloud Provisioning implements robust and field-hardened patterns.

However, in some simple use cases, you can deploy a set of appliances manually. Be sure to diligently document the parameters that are required by each appliance and their dependencies. This section provides a template for deployment documentation for the three appliances highlighted in this chapter.

Table 10-1, Table 10-2 on page 242, and Table 10-3 on page 242 list all of the configurable parameters and their default values for the WebSphere Application Server, the IBM HTTP Server, and the DB2 as identified in the sample three-tier topology. These tables do not list the installation location parameters.

Because of the inter-component dependencies between WebSphere Application Server and its database server, the order in which the two appliances are deployed is important. In Systems Director VMControl, the DB2 virtual appliance is deployed before the WebSphere Application Server appliance. For example, **server\_name** parameter in Table 10-1 can be set at deployment time only after the **db2\_hostname** parameter in Table 10-3 on page 242 is known or set. If the static IP is used by the deployer, both values are known upfront. However, if the dynamic IP is used, the deployer must wait until the DB2 appliance comes online to retrieve the host name that is allocated by the DNS server.

Table 10-1 lists the configurable parameters for WebSphere Application Server.

*Table 10-1 Configurable parameters for WebSphere Application Server*

| Activation program    | Description                                   | Input parameter                   | Default value                              |
|-----------------------|---|-----------------------------------|--|
| was-hostname-config   | Modify WebSphere Application Server host name | was-hostname (optional)           | Value is retrieved through a system call.  |
|                       |   | was-nodename (optional)           | DefaultAppSrvNode01                        |
| was-datasource-config | Configure data source related information     | provider_name (optional)          | DB2 Universal JDBC Driver Provider         |
|                       |   | datasource_name (optional)        | SampleDB4DS                                |
|                       |   | database_name (required)          |  |
|                       |   | server_name (required)            |  |
|                       |   | port_number (required)            |  |
|                       |   | database_user_password (required) |  |
|                       |   | wasadmin_password (optional)      | wasadmin, assuming JASS-J2C authentication |

| Activation program   | Description                  | Input parameter              | Default value                             |
|----------------------|------------------------------|------------------------------|---|
| was-webserver-create | Build web server information | was_nodename (optional)      | DefaultAppSrvNode01                       |
|                      |                              | wasadmin_password (optional) | wasadmin                                  |
|                      |                              | webserver_name (optional)    | webserver1                                |
|                      |                              | web_port (optional)          | 80  |
|                      |                              | web_install_root (optional)  | /opt/IBM/HTTPServer                       |
|                      |                              | hostname (optional)          | Value is retrieved through a system call. |
|                      |                              | admin_userid (optional)      | httpadmin                                 |
|                      |                              | admin_password (optional)    | httpadmin                                 |

Table 10-2 shows the configurable parameters for IBM HTTP Server.

*Table 10-2 IBM HTTP Server configurable parameters*

| Activation program | Description                     | Input parameter               | Default value       |
|--------------------|---------------------------------|-------------------------------|---------------------|
| webserver-config   | Modify web server configuration | webserver_home_dir (optional) | /opt/IBM/HTTPServer |
|                    |                                 | config_file_name (optional)   | httpd.config        |
|                    |                                 | server_name (required)        |                     |

Table 10-3 shows the configurable parameters for DB2.

*Table 10-3 DB2 configurable parameters*

| Activation program | Description                    | Input parameter              | Default value |
|--------------------|--------------------------------|------------------------------|---------------|
| db2-runtime-config | Modify DB2 runtime information | db2inst1_password (optional) | db2inst1      |
|                    |                                | db2fenc1_password (optional) | db2fenc1      |
|                    |                                | db2das1_password (optional)  | db2das1       |
|                    |                                | db2inst1_username (optional) | db2inst1      |
|                    |                                | db2inst1_group (optional)    | db2iadm1      |
|                    |                                | db_name_new (optional)       | SAMPLEDB      |
|                    |                                | db_action (optional)         | do-nothing    |
|                    |                                | db2_license_type (optional)  | paid          |
|                    |                                | db2_hostname (required)      |               |

## 10.8 Summary

As demonstrated in the example in this chapter, the Image Construction and Composition Tool with the PADK and the activation engine (VSAE) provides a complete development environment for building smart, deployment-ready virtual appliances.

For the sample three-tier topology, the end-to-end development cycle demonstrated in this chapter includes the following basic tasks:

- ▶ Importing three running VMs (IBM HTTP Server, WebSphere Application Server, and DB2) into the Image Construction and Composition Tool as the base images
- ▶ Identifying, implementing, and validating a set of activation programs for the three software components, based on their individual software requirements and intercomponent dependencies between software components
- ▶ Publishing these well-tested activation packages from PADK to the Image Construction and Composition Tool to create a set of activation bundles
- ▶ Extending the three base images to include their associated activation bundles in the Image Construction and Composition Tool to create three virtual appliances
- ▶ Synchronizing three virtual appliances individually to integrate the activation programs with IBM VSAE
- ▶ Capturing and exporting the final three virtual appliances

One of the standards published by the Distributed Management Task Force (DMTF) is the OVF. OVF is the specification used by the Image Construction and Composition Tool to build virtual appliances. The final IBM HTTP Server, WebSphere Application Server, and DB2 virtual appliances created from the Image Construction and Composition Tool development process are DMTF-OVF compliant. Therefore, these virtual appliances work well with the Systems Director VMControl, which implements the same standard.

An advantage demonstrated in this chapter is that you can use the tools provided by the IBM cloud-computing architecture to repackage all of your existing base images to build virtual appliances without changing one line of the existing code. It includes middleware software and customer applications. This end-to-end development process guides you step-by-step to complete the entire development cycle. The virtual appliances that are built by using this process are DMTF-compliant and can be easily deployed into a cloud environment.





## Virtual appliance deployment options

This chapter describes several typical deployment options for the virtual appliances that were built with the IBM Image Construction and Composition Tool (referred to as *the tool* in this book). It illustrates a typical user experience so that you have a better understanding of what it takes to use the appliances that are created with Image Construction and Composition Tool.

This chapter includes the following sections:

- ▶ Deploying KVM virtual appliances to Systems Director VMControl or IBM Flex System Manager
- ▶ Deploying PowerVM virtual appliances to Systems Director VMControl or IBM Flex System Manager
- ▶ Deploying a virtual appliance with IBM SmartCloud Entry

## 11.1 Deploying KVM virtual appliances to Systems Director VMControl or IBM Flex System Manager

The target deployment platform for the appliances built on the KVM provider is IBM Systems Director VMControl. The steps in this section are based on a working instance of IBM Systems Director 6.3 and Systems Director VMControl 2.4 or higher with a properly set up image repository. The deployment process consists of two fundamental tasks:

- ▶ Importing an appliance into the Systems Director VMControl image repository
- ▶ Deploying an appliance on a KVM host or into a KVM system pool

### 11.1.1 Importing an appliance into the image repository

To import an appliance into the Systems Director VMControl image repository, use the following steps:

1. After the virtual appliance construction process is finalized, capture the appliance and then export it. To export, do the following tasks:
  - a. In the Image Construction and Composition Tool, select **Images and bundles** → **Build and manage images**.
  - b. From the list of available appliances (images), select the image that you want to deploy through Systems Director VMControl. The image details are displayed in the right panel.
  - c. Click the **Export** icon in the upper right corner of this panel.
  - d. In the Export image as OVA window (Figure 11-1), copy the address, which is the following address in this example:

`http://192.168.71.86:8080/static/677cdda4-4f97-422e-994c-9079b36b40a3_myApp.tomcat6-SLES11.1-1G-KVM.ova`

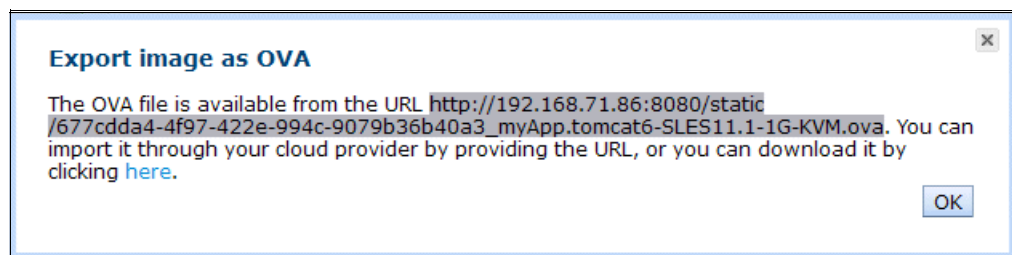


Figure 11-1 Exporting virtual appliance from Image Construction and Composition Tool

2. In the Systems Director console, open the Systems Director VMControl window.
3. On the Virtual Appliances tab, click **Import**.

4. In the Import window (Figure 11-2), under Source, paste the address that you copied in step 1 on page 246d on page 246 into the text box. Click **Next**.

**Source**

Specify the virtual appliance package to import.

Location and name of virtual appliance package, specified as an .ovf or .ova file:

[View Details](#)

Example: <http://www.vappliances4sale.com/aix61/aix61TL4wDB2.ova>

Acceptable specification forms:

On any management server:

[relativepath]filename.ovf or [relativepath]filename.ova  
http://path/filename.ovf or http://path/filename.ova

On IBM® Systems Director Server on AIX and Linux only:

/localpath/filename.ovf or /localpath/filename.ova  
file:///localpath/filename.ovf or file:///localpath/filename.ova

On IBM® Systems Director Server on Windows only:

file://c:/localpath/filename.ovf or file://c:/localpath/filename.ova  
c:\localpath\filename.ovf or c:\localpath\filename.ova  
\\computername\path\filename.ovf or \\computername\path\filename.ova

< Back   Next >   Finish   Cancel

Figure 11-2 Importing virtual appliance into Systems Director VMControl

5. Follow the remaining steps in the Import window, and then click **Next**. After the appliance is imported, it is ready for deployment.



## 11.1.2 Deploying an appliance on a KVM host or to a KVM system pool

To deploy an appliance on a KVM host or to a KVM system pool, use the following steps:

1. On the Virtual Appliances tab (Figure 11-3), select the newly imported appliance, and click **Deploy Virtual Appliance**.

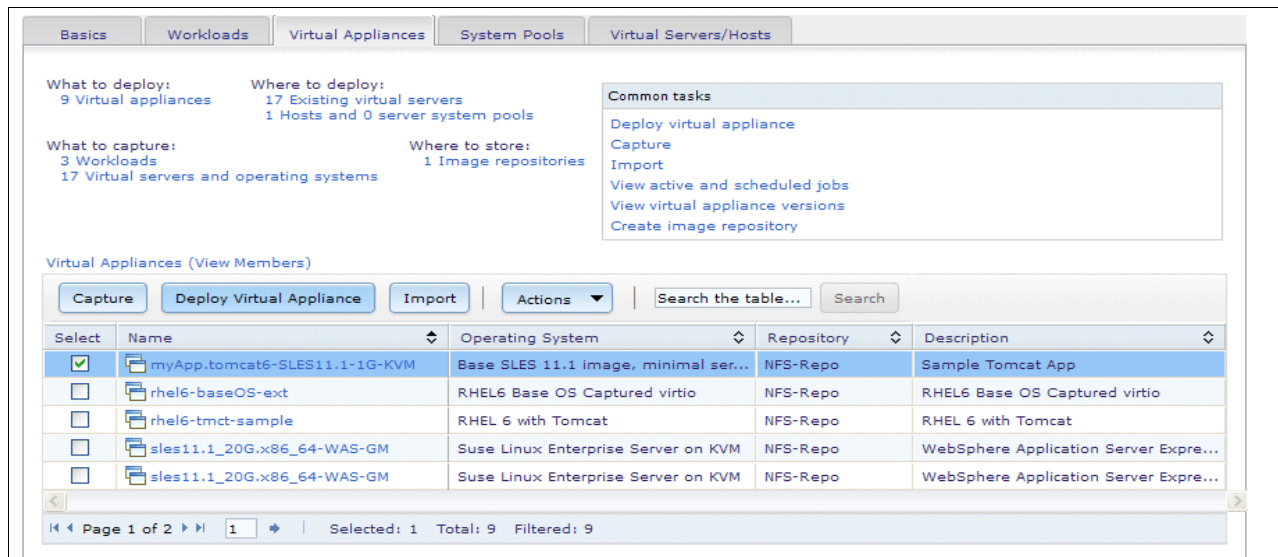


Figure 11-3 Deploying a virtual appliance

2. Follow the prompts in the Deploy Virtual Appliance wizard that opens. Be sure to select the target for the deployment (a KVM host or a KVM system pool), specify the workload name, the virtual server name, the storage mapping, and the network mapping.

3. In the last Deploy Virtual Appliance wizard panel (Figure 11-4), complete the runtime parameters for the various product sections in the OVF associated with the appliance. In this window, the parameters for each product section in the corresponding OVF are grouped together (see the arrows for the activation program name). Typically, most of the values have defaults set by the appliance architect.

**Password value:** Systems Director VMControl does not encrypt the password value in this window. Therefore, a user must provide a hashed password value. The alternative is not to require a hashed value, which results in a root password stored in clear text in the ISO transport file and is unacceptable in most environments. The user must manually create the hashed value and paste it into the User password field.

**Product**  
Specify the product settings you want to use when you deploy the virtual appliance.

**Summary:**  
Activate system user  
Product: activate-system-user  
Version: 2.1  
User name: root  
User password: i4N0bZpIUigP5WRl3vU5InE8UgB2hHZQDdbKF1  
Gecos: Super User (root@localhost)  
Home directory: /root  
Create home directory: false  
User ID: 0  
Group id: 0  
User shell program: /bin/sh

**Summary:**  
System network interface configuration  
Product: activate-network-interface  
Version: 2.1  
Network interface: eth0  
Use DHCP: false  
IP address: 192.168.71.129  
IP netmask: 255.255.255.0  
IP gateway: 192.168.71.38  
Is default route: true

**Summary:**  
System name and domainname  
Product: activate-system-host  
Version: 2.1  
Hostname: vs129  
Domain Name: ovafactory.rochester.ibm.com

**Summary:**  
activates the dns client  
Product: activate-dns-client  
Version: 2.1

Figure 11-4 Specifying the product activation parameters

### 11.1.3 Generating a password hash value

You can generate the necessary hash value for the password parameter by using the following methods:

- ▶ If you have access to the KVM provider host, at deployment time, go to the `/var/lib/va/vafes` directory, and then run the following command:

```
python -c "import cfesUtils; print cfesUtils.hashPass('your_password')"
```

The result is a hash value similar to the following example:

```
$6$G2PjnHn2Vi$sUzmu3ViPB0nh7fJm3piVvaI1c3sdhsjQoW8LR1yw9.Pt1qUqu9AFFVr4y49HyIKC  
PwBANpvy0nFkYeMatW9B/
```

Paste this value into the VMC password prompt.

- ▶ If you do not have access to the KVM provider host, on any Linux system, create a dummy user, and then set the password to the value you want to hash as in the following example:

```
Useradd dummy  
passwd dummy
```

The hash value that is generated by the operating system can be determined by looking up the contents of the `/etc/shadow` file:

```
cat /etc/shadow
```

Cut and paste the hash value between the first and the second colon for the dummy user profile:

```
dummy:$6$TFgbqfaZ$UzzkZHqTSn1YvRmHI0BSWS47fAZ7BEr0.9W4rugbui2UiJA00qy4WB.wrJErg  
objb56.mLdb8kKE0xkGxspLun0:15412:0:99999:7:::
```

## 11.2 Deploying PowerVM virtual appliances to Systems Director VMControl or IBM Flex System Manager

Similarly to the KVM provider, the target deployment platforms for the virtual appliances that are built with the PowerVM Express cloud provider are IBM Systems Director VMControl and IBM PureFlex System Flex System Manager. The steps in this section are based on a working instance of IBM Systems Director 6.3 and Systems Director VMControl 2.4 or later with an image repository with the correct setup. The deployment process consists of two fundamental tasks:

- ▶ Importing a virtual appliance into the Systems Director VMControl image repository
- ▶ Deploying the virtual appliance on a PowerVM host or system pool

### 11.2.1 Importing a virtual appliance into the Systems Director VMControl image repository

After the virtual appliance construction process is finished and the appliance is captured and exported as an OVA, import the virtual appliance into a production PowerVM environment:

1. Place the virtual appliance `.zip` file on a system to prepare for extraction.

**Filesystem space:** Ensure that you have enough space on the file system because the `.zip` file contains a raw disk image file that can be large.

2. Extract the virtual appliance, which results in the following files:
  - <ICCT Image Name>.raw
  - <ICCT Image UID>\_<ICCT Image Version>.ovf
  - osNode.xml
  - osNodeC.xml
3. By using the secure copy (**scp**) utility, transfer the four files to a IBM Systems Director VMControl management server, to a Flex System Manager system (for example, /home/userid), or to an HTTP server that is accessible by the cloud provider.
4. Open the IBM Systems Director console. In the left pane, expand **System Configuration**, and select **VMControl**.
5. On the **Virtual Appliances** tab, under Common tasks (Figure 11-5), select **Import**.

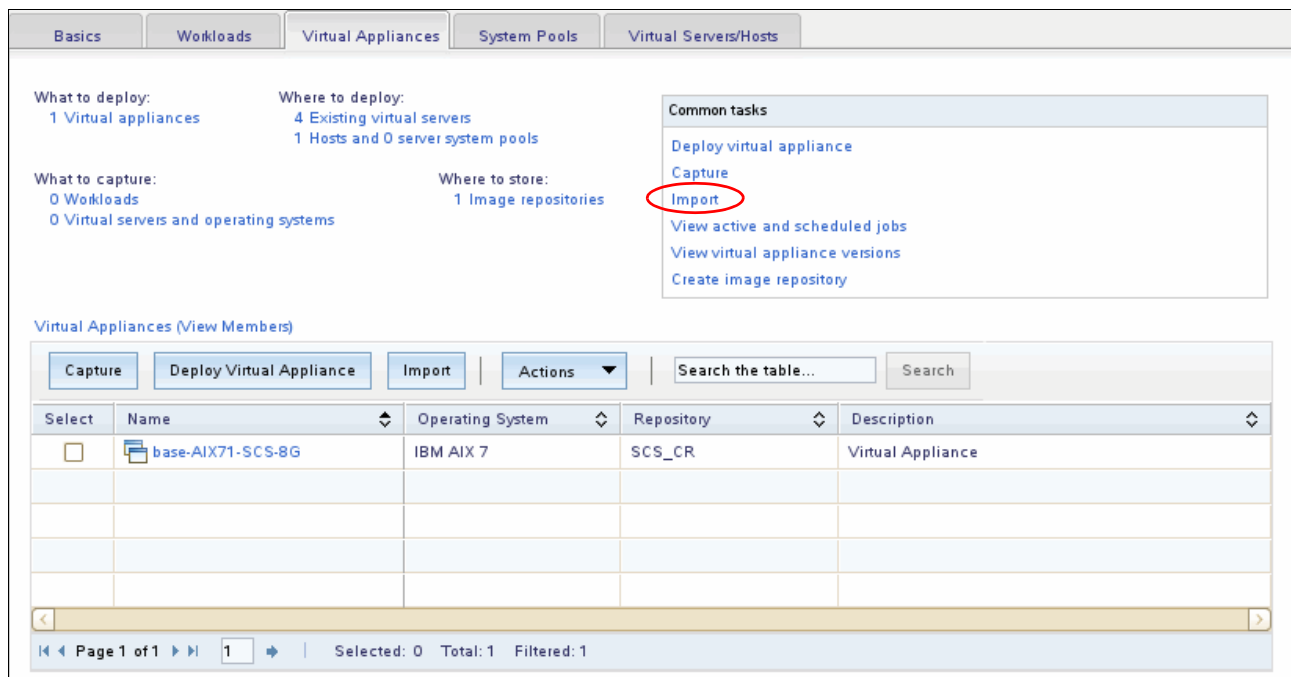


Figure 11-5 Importing the virtual appliance

6. In the Welcome window of the Import wizard, click **Next**.
7. In the Source window, enter the path (local or http) and the file name of the OVF file (for example, <ICCT Image UID>\_<ICCT Image Version>.ovf). Click **Next**.
8. In the Digital Signature window, click **Next**.
9. In the Name window, where you see the name, which is filled from the OVF file, click **Next**.
10. If more than one image repository exists, when prompted, select one. Then in the Repository window, select the repository that you want to use. Click **Next**.
11. In the Version Control window, leave the default if it is a new appliance. If it is an update to an existing appliance, use version control to select an existing appliance to be the parent version. Click **Next**.

12. In the Summary window (Figure 11-6), review the information, and then click **Finish**.

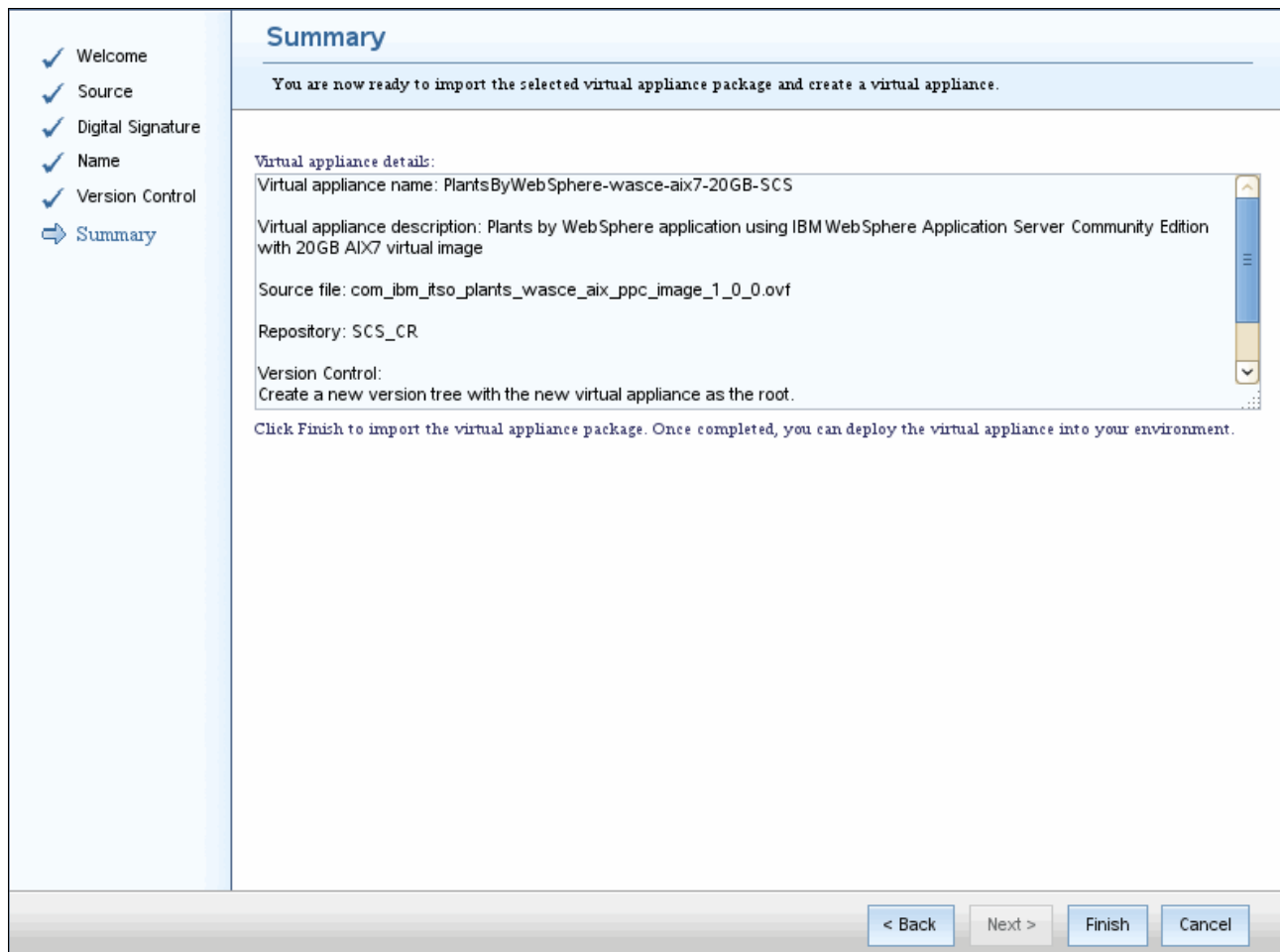


Figure 11-6 Summary of virtual appliance import

## 11.2.2 Deploying the virtual appliance on a PowerVM host or system pool

After the virtual appliance is imported, it is ready for deployment. To deploy the virtual appliance, complete the following steps:

1. Open the IBM Systems Director console.
2. In the left pane, expand **System Configuration**, and select **VMControl**.

- On the **Virtual Appliances** tab (Figure 11-7), where you see all the virtual appliance available to deploy from the image repository, select the newly imported virtual appliance, and click **Deploy Virtual Appliance**.

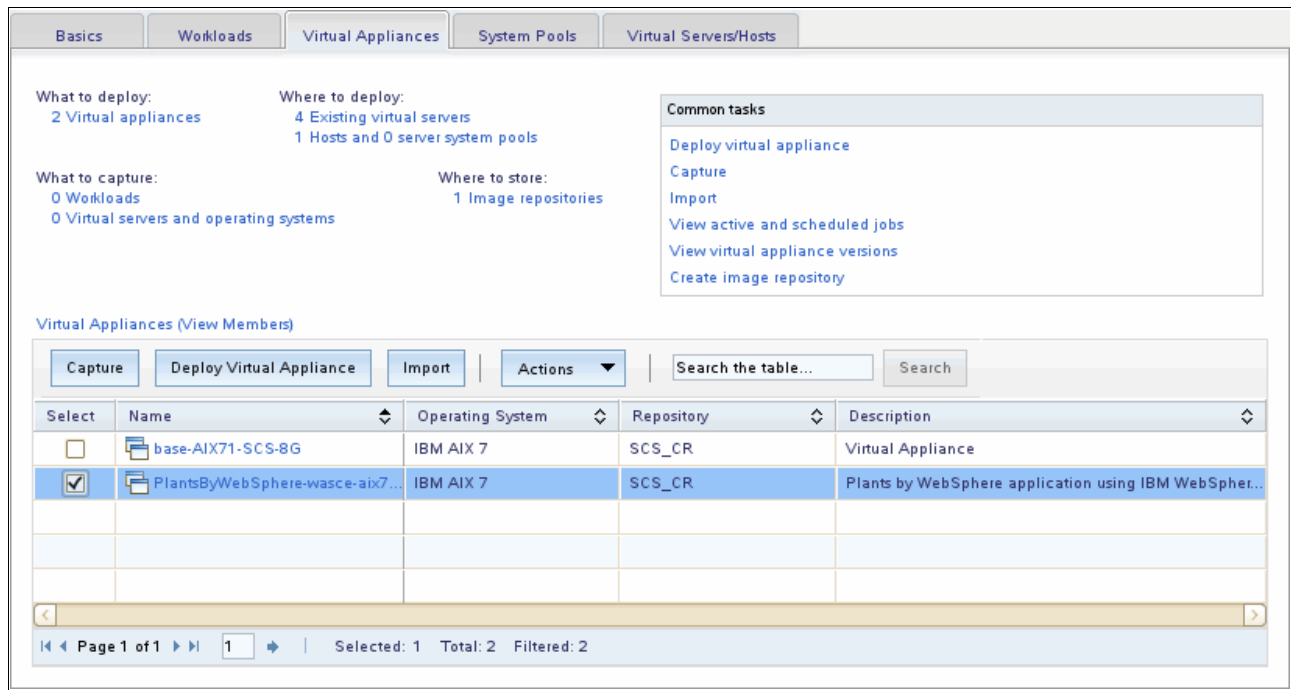


Figure 11-7 Deploying the virtual appliance

- In the Welcome window of the Deploy Virtual Appliance wizard, click **Next**.
- In the Target window, select to deploy to a new virtual server, and select the PowerVM system to host the new virtual server. Click **Next**.
- In the Workload Name window, specify a unique name for the newly deployed workload. Then click **Next**.
- In the Storage Connection window, specify how to attach the storage. For example, use virtual SCSI to attach SAN storage to the virtual server. Click **Next**.
- In the Storage Mapping window, select the virtual appliance disk, and then assign to either a storage volume or a storage pool. Click **Next**.
- In the Network Mapping window, select **Virtual Networks on Host** from the list. Click **Next**.

10. In the Product window (Figure 11-8), enter the System Level Networking information and the arguments that were specified in the configuration section of software bundles created in the IBM Image Construction and Composition Tool. These arguments are transferred from and defined in the Product section of the OVF. Click **Next**.

**Product**

Specify the product settings you want to use when you deploy the virtual appliance.

**General System Product Section**

Time zone setting for the virtual system.

**System Level Networking**

Short host name for the system.

DNS domain name for the system.

IP addresses of DNS servers for system.

Default IPv4 gateway.

**Network adapter configuration for Network adapter 1 on Discovered-1-0**

**Internet Protocol Version 4**

Static IP address for the network adapter "Network adapter 1 on Discovered-1-0".

Static network mask for network adapter "Network adapter 1 on Discovered-1-0".

**Internet Protocol Version 6**

Static IP address for the network adapter "Network adapter 1 on Discovered-1-0".

Static default gateway for network adapter "Network adapter 1 on Discovered-1-0".

Use IPv6 stateless address autoconfiguration for network adapter "Network adapter 1 on Discovered-1-0".

**Deployment use**

The adapter order for network adapter "Network adapter 1 on Discovered-1-0".

The slot number for network adapter "Network adapter 1 on Discovered-1-0".

Captured from virtual server helium81 connected to Discovered-1-0 on host helium53-IVM

**Config WAS CE**

Number Of Servers

WAS CE Home

WASCE\_ADMIN\_USER

WASCE\_ADMIN\_PASSWORD

< Back Next > Finish Cancel

Figure 11-8 Product information window

11. In the Summary window, review the deployment details, and then click **Finish**.
12. In the Launch job window, choose when to run the job. Select **Run Now** or **Schedule**. Then click **OK** to finish the deployment wizard.

After you finish the deployment wizard, you can monitor the log of your deployment job to see when the deployment completes. When the Systems Director VMControl job completes, you might need to wait additional time for the activation of the appliance because Systems Director VMControl does not monitor this part of the deployment.

## 11.3 Deploying a virtual appliance with IBM SmartCloud Entry

IBM SmartCloud Entry uses Flex System Manager, Systems Director VMControl, or VMware vSphere with vCenter to provide the underlying virtual infrastructure environment. With IBM SmartCloud Entry, you can do the following tasks:

- ▶ Provisioning and deprovisioning servers
- ▶ Drafting and cloning workloads
- ▶ Capturing workloads
- ▶ Starting and stopping servers as part of a workload
- ▶ Resizing existing server

This section guides you through the process of deploying a virtual appliance with IBM SmartCloud Entry.

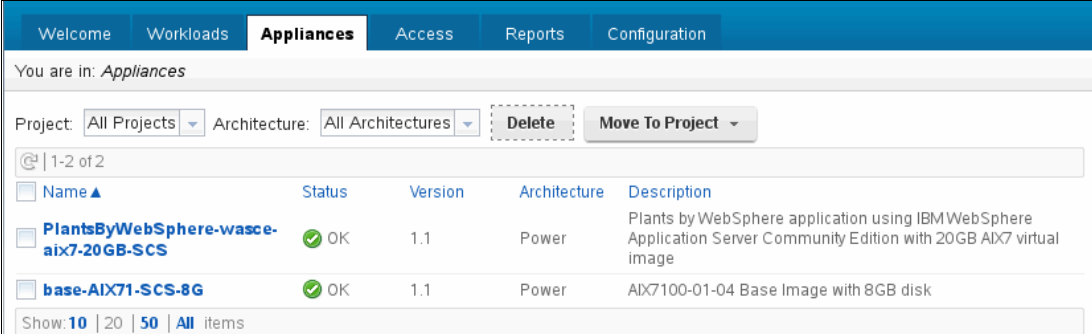
### 11.3.1 Importing PowerVM and KVM virtual appliances into IBM SmartCloud Entry

You cannot import a PowerVM or KVM virtual appliance directly into IBM SmartCloud Entry. Therefore, you must import the virtual appliance into VMControl as described in the previous sections. After you complete that task, you can see the appliance in IBM SmartCloud Entry. For information about importing to the cloud provider, see 11.2.1, “Importing a virtual appliance into the Systems Director VMControl image repository” on page 250 and 11.1.1, “Importing an appliance into the image repository” on page 246.

### 11.3.2 Managing PowerVM and KVM virtual appliances in IBM SmartCloud Entry

To deploy a virtual appliance with IBM SmartCloud Entry, complete the following steps:

1. Enter the address of your IBM SmartCloud Entry in a browser, such as the following example:  
`http://hostname:8080/cloud/web/login.html`
2. Log in to IBM SmartCloud Entry.
3. On the **Appliance** tab (Figure 11-9), from the list of virtual appliances available for deployment, click on the name of the virtual appliance that you want to deploy.



The screenshot shows the 'Appliances' tab in the IBM SmartCloud Entry interface. At the top, there are navigation tabs: 'Welcome', 'Workloads', 'Appliances' (selected), 'Access', 'Reports', and 'Configuration'. Below the tabs, it says 'You are in: Appliances'. There are filters for 'Project' (set to 'All Projects') and 'Architecture' (set to 'All Architectures'). To the right of these filters are buttons for 'Delete' and 'Move To Project'. Below the filters is a table with two rows of appliances. The table has columns for 'Name', 'Status', 'Version', 'Architecture', and 'Description'. The first row is 'PlantsByWebSphere-wasce-aix7-20GB-SCS' with status 'OK', version '1.1', architecture 'Power', and description 'Plants by WebSphere application using IBM WebSphere Application Server Community Edition with 20GB AIX7 virtual image'. The second row is 'base-AIX71-SCS-8G' with status 'OK', version '1.1', architecture 'Power', and description 'AIX7100-01-04 Base Image with 8GB disk'. At the bottom of the table, there is a 'Show' dropdown set to '10' and a link to 'All items'.

| Name                                  | Status | Version | Architecture | Description   |
|---------------------------------------|--------|---------|--------------|---|
| PlantsByWebSphere-wasce-aix7-20GB-SCS | OK     | 1.1     | Power        | Plants by WebSphere application using IBM WebSphere Application Server Community Edition with 20GB AIX7 virtual image |
| base-AIX71-SCS-8G                     | OK     | 1.1     | Power        | AIX7100-01-04 Base Image with 8GB disk  |

Figure 11-9 List of appliances



4. As shown in Figure 11-10, click **Deploy**, and then select **Advanced**.

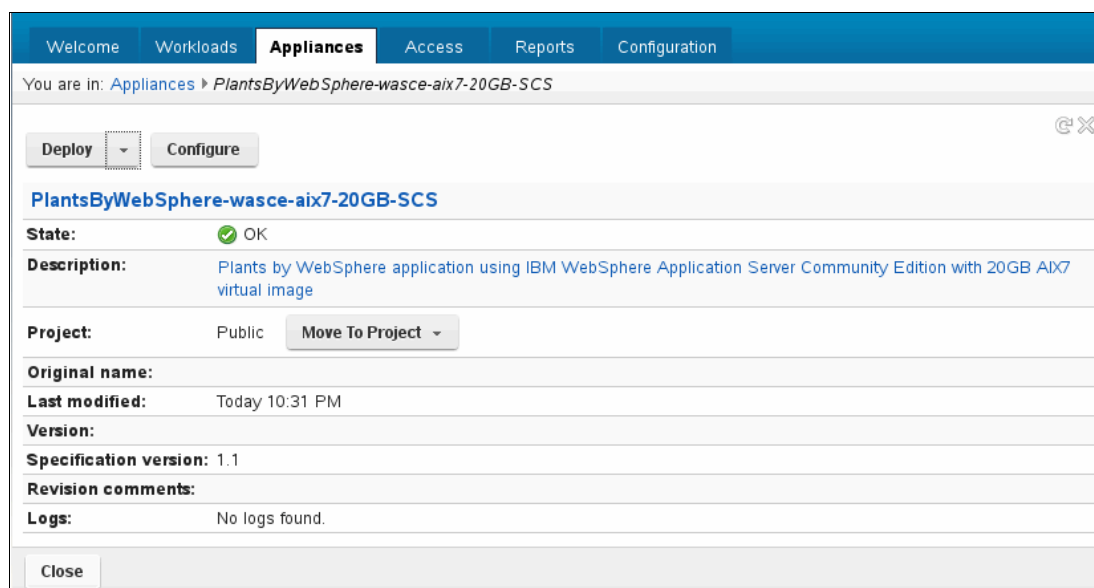


Figure 11-10 Plants By WebSphere virtual appliance

5. Enter the custom parameters for the deployment (Figure 11-11 on page 257):
- Enter a name. We enter Plants by WebSphere Workload 1.
  - Enter a description. Again, we enter Plants by WebSphere Workload 1.
  - Select the type of project. We select **Public**.
  - Under Processor Settings, do the following tasks:
    - Select whether the physical server will use a dedicated or shared process. In this example, we select **Shared**.
    - Enter the values for the shared virtual processors.
    - Enter the values for the shared processing units.
    - Select the processing units sharing mode of the virtual server. We select **Uncapped**.
    - Select the priority of the virtual server. We enter **128**.
  - Under Memory Settings (MB), select the three values.
  - Under Network, select the network configuration for System and for Adapter 1. We select **Network1** for both options.
  - Under Other Settings, for System, enter the following information:
    - Enter the number of servers. We enter 1.
    - Enter the path for WAS CE Home.
    - Enter the WASCE\_ADMIN\_USER name.
    - Enter the WASCE\_ADMIN\_PASSWORD.
  - Under Other Settings, configure the storage mapping table.
  - Click **Deploy**.

Welcome
Workloads
**Appliances**
Access
Reports
Configuration

You are in: [Appliances](#) > [PlantsByWebSphere-wasce-aix7-20GB-SCS](#) > [Deploy](#)

Deploying *PlantsByWebSphere-wasce-aix7-20GB-SCS*

**\*Name:**  
Plants By WebSphere Workload 1

**Description:**  
Plants By WebSphere Workload 1

**Project:**  
Public [New Project](#)

**Expiration Period**  
☐ Set an expiration date

**Processor Settings**  
\*Indicates whether the virtual server will use physical or virtual processors (dedicated or shared mode).:  
Shared  
Shared virtual processors:  
1 ≤ 2 ≤ 4  
Shared processing units:  
0.1 ≤ 0.2 ≤ 4  
The processing units sharing mode of the virtual server.:  
Uncapped  
The priority of the virtual server to available processing units in the shared processor pool.:  
128

**Image Target**  
Workloads based on this appliance will go to the selected target:  
Server-7895-42X-SN10FA72A (Host)

**Memory Settings (MB)**  
Memory Settings (MB):  
256 ≤ 2,048 ≤ 8,192

**Network**  
**System**  
Network configuration: Network1 [Edit Network1](#)  
[Show settings](#)  
**Adapter 1**  
Network configuration: Network1 [Edit Network1](#)  
[Show settings](#)

**Other Settings**  
**System**  
Internet Protocol Version 6  
Static IP address for the network adapter "Network adapter 1 on Discovered-1-0":  
Static default gateway for network adapter "Network adapter 1 on Discovered-1-0":  
Use IPv6 stateless address autoconfiguration for network adapter "Network adapter 1 on Discovered-1-0":  
Time zone setting for the virtual system:  
Number of Servers  
1  
WAS CE Home  
/opt/IBM/WebSphere/AppServerCommunityEdition  
WASCE\_ADMIN\_USER  
system  
WASCE\_ADMIN\_PASSWORD  
manager

**Other Settings**  
Configure storage mapping table. Disk Name: disk1, Size (MB): 20480:  
1-3 of 3 Page: 1

| Name ▲  | Location                                   | VIOS Count | Maximum Allocation (MB) | Description                                 |
|---|--|------------|-------------------------|---|
| <input type="radio"/> KVM_SCS                         | SAN: Storwize V7000-2076-V7000a<br>NGP-IBM | 1          | 1713920                 | SAN pool accessed through one or more VIOS. |
| <input checked="" type="radio"/> POWER_SCS_DEPLO<br>Y | SAN: Storwize V7000-2076-V7000a<br>NGP-IBM | 1          | 2697984                 | SAN pool accessed through one or more VIOS. |
| <input type="radio"/> POWER_SCS_REPO                  | SAN: Storwize V7000-2076-V7000a<br>NGP-IBM | 1          | 2285824                 | SAN pool accessed through one or more VIOS. |

Figure 11-11 Deployment information for PlantsByWebSphere

6. On the **Workloads** tab (Figure 11-12), view the appliance deploying as a new workload.

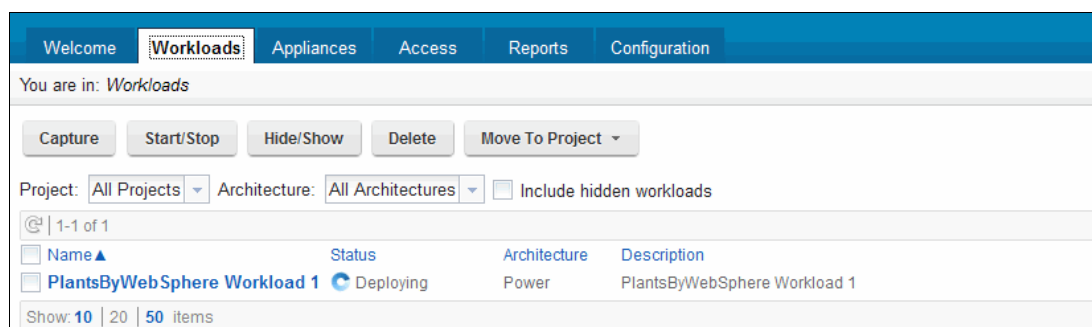


Figure 11-12 New workload deploying

When you finish the deployment through the UI, you can monitor the status of your workload to determine when the deployment completes. A status of OK indicates a successful deployment of the workload. You might need to wait additional time for the activation of the appliance because IBM SmartCloud Entry does not monitor this part of the deployment.

### 11.3.3 Importing OVA into VMware vSphere with vCenter

You cannot import a virtual appliance directly into IBM SmartCloud Entry. Therefore, you must import it in to the underlying virtual infrastructure environment. In this case, VMware vSphere with vCenter is the underlying virtual infrastructure.

To import a virtual appliance into vCenter, complete the following steps:

1. Export the OVA from Image Construction and Composition Tool to a remote server. From Image Construction and Composition Tool, click the **Export as OVA** icon (📄).
2. In the Export image as OVA window (Figure 11-13 on page 259):
  - a. In the Remote host field, enter the IP address or host name of the remote host to copy the OVA to.
  - b. Enter the destination folder on the remote host.
  - c. Leave the file name fields as is because it is populated with the UUID name for the image.
  - d. Select the authentication method.
  - e. Enter the user name and password to access the remote host.
  - f. For OVA file format, select the correct OVA file format. For ESX, select **IBM Workload Deployer 3.1 (or higher), VMware VirtualCenter**.

**Export image as OVA**

To what location should the image be exported?  
*The destination host must support Secure Copy (SCP)*

Remote host: 192.168.10.100

Destination folder: /images

File name: com.ibm.itso.sample.tomcat6.rhel62.e

**Authentication Method:**

☐ Private Key File ☒ Password

User name: root

Password: ●●●●●●●●●●

Verify password: ●●●●●●●●●●

OVA file format: 1(or higher), VMware VirtualCenter ▼

OK Cancel

Figure 11-13 Export OVA from Image Construction and Composition Tool

3. After the OVA is exported to a remote host, extract the OVA. First, log in to the host to which you just exported the OVA. Then extract the OVA.

In Linux, you can use the following command:

```
tar -xvf filetoextract
```

4. Import into VMware vCenter. Start a vSphere client and connect to vCenter.
5. When you are connected, click **File** → **Deploy OVF Template**.

6. In the Deploy OVF Template field of the Source window (Figure 11-14), enter the location of the OVF file. It can be a URL or a local directory. Click **Next**.

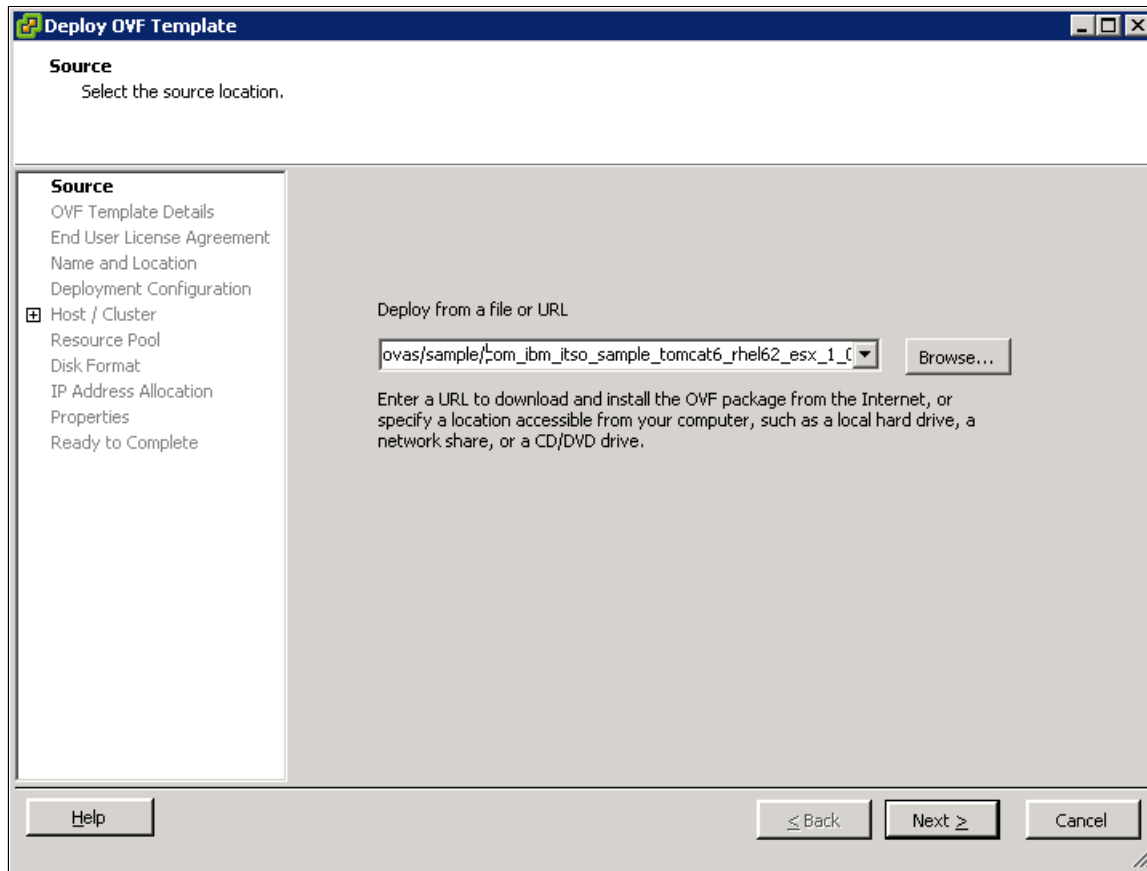


Figure 11-14 Importing an OVF in vCenter

7. Review the OVF details, and then click **Next**.
8. Review and accept any license agreements. Click **Accept**, and then click **Next**.
9. Enter a unique name and select an Inventory location. Then click **Next**.
10. Select the ESX host to which to deploy, and then click **Next**.
11. If you receive a warning message (Figure 11-15) that states that the operating system of the image is not supported, click **Yes** to continue. You correct this situation in a later step.

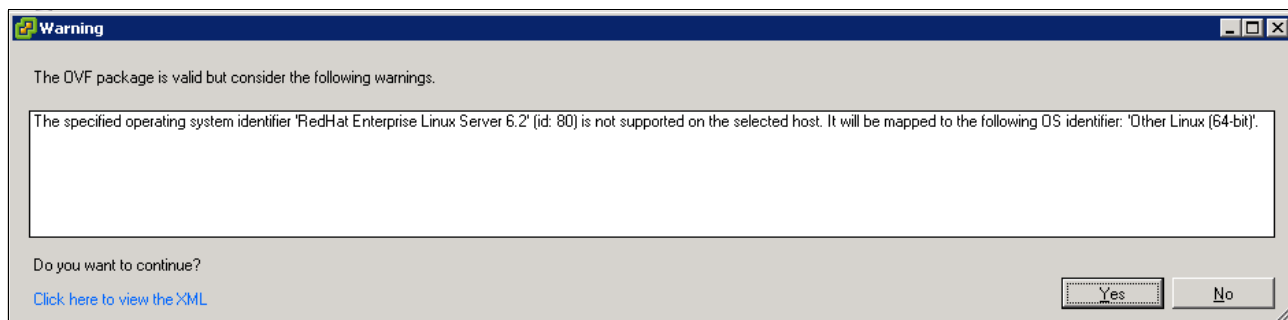


Figure 11-15 Warning message for OS not supported on host

12. Select the disk format of the virtual machine. Then, click **Next**.
13. Edit the settings so that you can verify that the virtual machine imported correctly. Then click **Next**.
14. Review the Summary page, and verify that the deployment settings are correct. Then click **Finish**. Now the OVA is a VM on the ESX that can be deployed.

**Important:** If a warning message indicates that the specified operating system identifier is not supported, complete steps 15 and 16. Otherwise skip to step 17 on page 262.

15. Edit the setting of the virtual machine. Right-click the virtual machine and select **Edit Settings**.
16. On the **Options** tab, on the right side (Figure 11-16), select the Guest Operating System and the version. If the version is not listed, select the closest match. In this example, Red Hat Enterprise Linux 6.2 (64-bit) is installed in the image, but we select **Red Hat Enterprise Linux 5 (64-bit)** as the one being the closest match. Click **OK**.

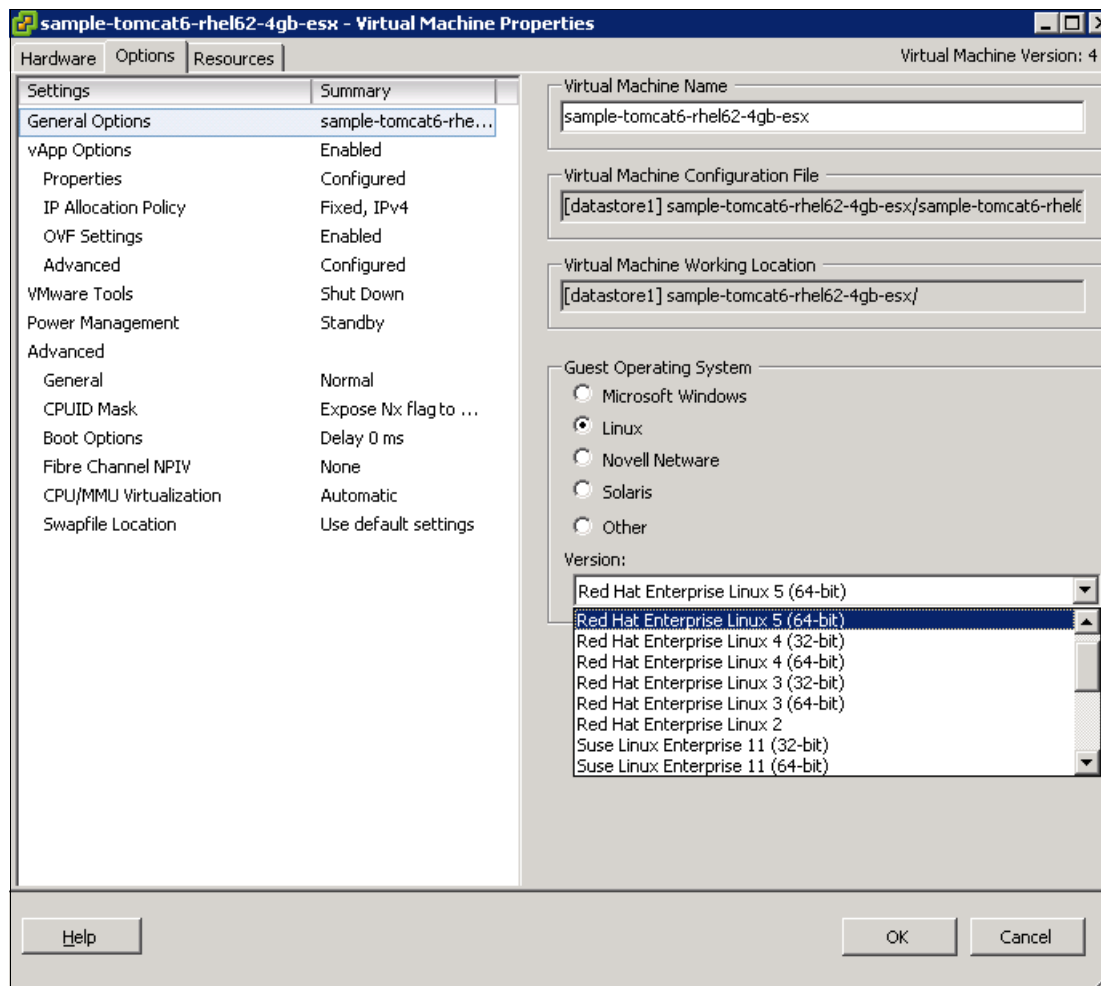


Figure 11-16 Selecting the guest operating system of the virtual machine

17. Clone or convert the VM to a template before IBM SmartCloud Entry is able to discover it. In this example, we clone to a template.
  - a. Click the virtual machine, and then select **Template** → **Clone to Template**.
  - b. Enter a unique name, and select an inventory location. Click **Next**.
  - c. Select the ESX server to host the template. Click **Next**.
  - d. Select the data store where you want to store the template files. Click **Next**.
  - e. Select the disk format of the virtual machine. Click **Next**.
  - f. Review the Summary page, and verify that the deployment settings are correct. Click **Finish**.

### 11.3.4 Managing virtual appliances in IBM SmartCloud Entry

To manage a virtual appliance with IBM SmartCloud Entry, complete the following steps:

1. Enter the URL of your IBM SmartCloud Entry in a browser as shown in this example:  
<http://hostname:8080/cloud/web/login.html>
2. Log in to IBM SmartCloud Entry.
3. On the **Appliances** tab (Figure 11-17), from the list of the virtual appliances available for deployment, click on the name of the virtual appliance that you want to deploy.

Welcome

Workloads

Appliances

Access

Reports

Configuration

You are in: Appliances

Project: All Projects

Architecture: All Architectures

Delete

Move To Project

1-8 of 8

| <input type="checkbox"/> Name ▲   | Status | Version | Architecture | Description   |
|---|--------|---------|--------------|---|
| <input type="checkbox"/> 2012_06_06 itso-client3 rhel62-baseos snapshot | OK     | vmx-04  | x86          | Appliance created as a snapshot taken on 6/14/12 3:35 PM. |
| <input type="checkbox"/> rh62-tmct6-sample-va-2                         | OK     | vmx-04  | x86          | RHEL 6.2 with Sample Tomcat                               |
| <input type="checkbox"/> rh62-tmct6-sample-va-tmpl                      | OK     | vmx-04  | x86          | RHEL 6.2 with Sample Tomcat                               |
| <input type="checkbox"/> rhel60-2G-baseOS-tmpl                          | OK     | vmx-07  | x86          | Red Hat Enterprise Linux 6                                |
| <input type="checkbox"/> rhel60-tmct-tmpl                               | OK     | vmx-07  | x86          | Red Hat Enterprise Linux 6                                |
| <input type="checkbox"/> rhel62-baseOS-tmpl                             | OK     | vmx-07  | x86          | Red Hat Enterprise Linux 6                                |
| <input type="checkbox"/> rhel62-tmct6-tmpl                              | OK     | vmx-07  | x86          | Red Hat Enterprise Linux 6                                |
| <input type="checkbox"/> sample-tomcat6-rhel62                          | OK     | vmx-04  | x86          | RHEL 6.2 with Sample Tomcat                               |

Figure 11-17 IBM SmartCloud Entry list of virtual appliances

4. Click the arrow next to **Deploy**, and select **Advanced** (Figure 11-18).

IBM SmartCloud Entry

Welcome Workloads Appliances Access Reports Configuration

You are in: Appliances > sample-tomcat6-rhel62 > Deploy

Deploying *sample-tomcat6-rhel62*

\*Name: sample Tomcat app

Description: Sample Tomcat App

Project: Public [New Project](#)

Expiration Period

☒ Set an expiration date

\*Expiration date: 6/5/2013

Days until expiration: 30

Hardware

System

Memory and CPU Settings

Memory (MB): 2,048

Virtual CPUs: 1

Software

System

Default locale: en

Default country: US

Default encoding: UTF-8

Hostname: scevs101

Domain: rchland.ibm.com

IP address: 10.5.169.159

Netmask: 255.255.255.0

Gateway: 10.5.169.1

Primary DNS: 10.10.244.200

Secondary DNS: 10.10.244.100

Password (root): ibm1234

Password (virtuser): ibm1234

VNC Username: virtuser

VNC Password:

NTP Server 0: time-a.nist.gov

avoidSkipped: no

Sample App Home Directory: /usr/share/tomcat6/webapps/sample

The file in which to modify the hostname: index.html

Current hostname: scevs101

Other Settings

Boot protocol: static

☐ Enable VNC

Network

System

Network configuration: None [Create new](#)

[Hide settings](#)

TCP/IP Network Settings

Host name: tomcat Manually entered

\*Domain name: rchland.ibm.com Manually entered

Primary DNS: 10.10.10.100 Manually entered

Backup DNS: Manually entered

Domain name server suffix list (comma separated):

Adapter 1

Network configuration: Default Network Configuration [Edit Default Network Configuration](#)

[Show settings](#)

Storage

System

Storage Settings

Disk Size of Hard disk 1 (MB): 2,560

Target Storage: Default

Deploy Save Cancel

Figure 11-18 Details on the virtual appliance to deploy



5. Enter the custom parameters for the deployment.

**Tip:** During deployment, you can modify hardware parameters, such as the number of processors, memory, and disk size.

**Note:** If you created your KVM virtual appliance with an older version of the Image Construction and Composition Tool, the labels for several Other Setting fields in IBM SmartCloud Entry might not be visible. To correct this problem, update your KVM cloud provider to the latest level that will replace the current `generic.ovf` file that is used when creating virtual appliances. This is the file that is used as a template to create the OVF file for the virtual appliance. You can also manually edit the `generic.ovf` file for all new virtual appliances or the `.ovf` file of a specific virtual appliance to correct this problem. For details, see Appendix C, “Editing the `generic.ovf` file for IBM SmartCloud Entry deployment” on page 289.

6. Click **Deploy**.



## Proven practices

This chapter highlights tested and proven practices in various real-life engagements. By adopting these practices, you can reduce the time required to write software bundles and master the virtual image creation process. This chapter includes the following sections:

- ▶ Virtual appliance naming convention
- ▶ 12.2, “Software bundle naming convention” on page 267
- ▶ 12.3, “Preinstalling software or creating a bundle with the installation image” on page 269
- ▶ 12.4, “Best practices for software bundles” on page 270
- ▶ 12.5, “Writing shell scripts for software bundle operations” on page 271
- ▶ 12.6, “Testing the product activation” on page 278
- ▶ 12.7, “Split large virtual images” on page 279

## 12.1 Virtual appliance naming convention

In more complex environments, you manage multiple virtual appliances. Therefore, you must adopt a consistent naming convention. For each virtual appliance in the Image Construction and Composition Tool, provide the following information:

- ▶ 12.1.1, “Image name” on page 266
- ▶ 12.1.2, “Universal ID” on page 266
- ▶ 12.1.3, “Version” on page 267

The combination of the universal ID and the version uniquely identifies the virtual image.

### 12.1.1 Image name

Provide image names that are descriptive and meaningful. We use the following format:

[Product] - [Software] - [OS] - [Size] - [Hypervisor] - [Method]

The line has the following definitions:

|                   |  |
|-------------------|--|
| <b>Product</b>    | The solution or application name.  |
| <b>Software</b>   | Optional: The name of the software component or middleware installed in the image. |
| <b>OS</b>         | The guest operating system.  |
| <b>Size</b>       | The size of the disk capacity assigned to the appliance.                           |
| <b>Hypervisor</b> | The target hypervisor. This value is optional.                                     |
| <b>Method</b>     | Approach based on PowerVM, SCS, or NIM. This value is optional.                    |

For an appliance based on PowerVM, you might use the following image name:

MySampleApp-WAS8-AIX71-50GB-PowerVM-SCS

For an appliance based on VMware, you might use the following image name:

MySampleApp-Tomcat6-RHEL62-10GB-VMware

### 12.1.2 Universal ID

Use the universal ID to provide enough information about the appliance content so that your co-workers or appliance users can quickly decide whether it meets their requirements. Use the following format:

[domain].[product][version].[software][version].[os][version].[hypervisor]

The line has the following definitions:

|                 |  |
|-----------------|--|
| <b>domain</b>   | The domain name of the organization that produced the appliance. For example com.ibm.itso.   |
| <b>product</b>  | The user solution or application name.   |
| <b>version</b>  | The version of the preceding component.  |
| <b>software</b> | The name of the middleware or software component on which this solution depends. For example, DB2, WebSphere Application Server, Apache Tomcat, and mySQL. This value is optional. |

|                   |   |
|-------------------|---|
| <b>os</b>         | The guest OS. For example, Red Hat Enterprise Linux (RHEL), SUSE Linux Enterprise Server (SLES), AIX and IBM i.                                   |
| <b>hypervisor</b> | The target hypervisor for the appliance, for example, <code>powervm</code> , <code>kvm</code> , and <code>vmware</code> . This value is optional. |

For an appliance based on PowerVM, you might use the following universal ID:

```
com.ibm.itso.mysampleapp213.tomcat6.aix71.powervm
```

For an appliance based on KVM, you might use the following universal ID:

```
com.ibm.itso.mysampleapp132.was8.rhel62.kvm
```

### 12.1.3 Version

At least three levels are required for the version number. The version number that you assign when extending a base image is not kept as the version number in the cloud provider image repository when the capture of your new virtual appliance is complete. System Director VMControl automatically generates a version number when a capture occurs as you are creating a new virtual appliance. The version number in System Director VMControl is in the form of `branch.version`, starting with 1.1. The version number that you assign when extending a base image is used to identify your appliance release and revision levels within the Image Construction and Composition Tool.

## 12.2 Software bundle naming convention

In a collaborative virtual appliance development environment, you must manage several software bundles for different applications supported on different operating systems and cloud providers. We prefer to adopt a consistent naming convention and create one software bundle to install, configure and reset each application or software component. For each software bundle in the Image Construction and Composition Tool, you must provide the following information:

- ▶ 12.2.1, “Bundle name” on page 267
- ▶ 12.2.2, “Universal ID” on page 268
- ▶ 12.2.3, “Version” on page 268

### 12.2.1 Bundle name

Provide a descriptive name to identify software bundles. We use the following format:

```
[Product] - [OS] - [Hypervisor] - Bundle
```

The line has the following definitions:

|                   |  |
|-------------------|--|
| <b>Product</b>    | The solution or application name.              |
| <b>OS</b>         | The supported operating system.                |
| <b>Hypervisor</b> | The target hypervisor. This value is optional. |

For a Lotus IBM Domino® 8.5.3 software bundle, supported on IBM i 7.1 based on PowerVM, you might use the following name:

```
LotusDomino8.5.3-IBMi7.1-PowerVM-Bundle
```

## 12.2.2 Universal ID

The universal ID and version combination must be unique among all software bundles. We use the following format for the universal ID:

```
[domain].[product][version].[os][version].[hypervisor].bundle
```

The line has the following definitions:

|                   |   |
|-------------------|---|
| <b>domain</b>     | The domain name of the organization that produced the bundle. For example <code>com.ibm.itso</code> .   |
| <b>product</b>    | The software product or application name.   |
| <b>version</b>    | The version of the preceding component.   |
| <b>os</b>         | The OS where the software product, component or application is supported. For example, Red Hat Enterprise Linux (RHEL), SUSE Linux Enterprise Server (SLES), AIX and IBM i. |
| <b>hypervisor</b> | The target hypervisor for the bundle. For example, <code>powervm</code> , <code>kvm</code> , and <code>vmware</code> . This value is optional.                              |

For a Lotus Domino 8.5.3 software bundle supported on IBM i 7.1 based on PowerVM, you might use the following universal ID:

```
com.ibm.itso.lotusdomino853.ibm71.powervm.bundle
```

## 12.2.3 Version

At least three levels are required for the version number. This number identifies the release and revision levels of your software bundle within the Image Construction and Composition Tool. You can implement your own Concurrent Versions System (CVS) or version control system for software bundles by checking in the bundle directory:

```
/drouter/ramdisk2/mnt/raid-volume/raid0/local/bundles/<bundle ID>
```

Read the `properties.txt` file first, to make sure about the software bundle you want to check in. You can log in to the server where the Image Construction and Composition Tool is installed and enter the following command to find your software bundle directory.

```
find /drouter/ramdisk2/mnt/raid-volume/raid0/local/bundles -type "f" \
-name "properties.txt" | xargs grep "<bundle name>"
```

All bundles artifacts are under the artifacts directory. The bundle files are in the following subdirectories:

|                     |  |
|---------------------|--|
| <b>imageBuild</b>   | Install operation directory. All files uploaded in the install operation are saved in the <code>imageBuild/newstep</code> directory.   |
| <b>imageCapture</b> | Reset operation directory. All files uploaded in the reset operation are saved in the <code>imageCapture/newstep</code> directory.   |
| <b>imageDeploy</b>  | Configuration operation directory. You can have one or more configuration operations, all the files uploaded in each operation are saved in the <code>imageDeploy/&lt;lowercase operation name&gt;</code> directory. |

## 12.3 Preinstalling software or creating a bundle with the installation image

Decide whether to capture an image with the software that is installed or to create bundles to separately install and configure each piece of software that is required for the virtual appliance.

Generally, the following two methods are possible for creating virtual appliances that contain the entire software stack that is necessary to automatically deploy a software solution into a cloud:

- First method

Create a base image with the operating system, and then extend it with any required software bundles and the user application bundles. For more information about this approach, see Chapter 9, “Constructing simple virtual appliances” on page 181.

- Second method

Create a virtual machine, and install all the necessary software. Then, import the running VM to the Image Construction and Composition Tool, and extend it with bundles that contain only configuration and reset scripts. For more information about this approach, see Chapter 10, “Constructing complex virtual appliances” on page 221.

We prefer the first method of creating a base appliance and then extending it because of the following advantages:

- It eliminates redundancy and duplication of effort. If it is well designed and implemented, a single bundle can be used to install a software component into several guest operating systems that target different hypervisors. This approach truly fulfills the vision of the Image Construction and Composition Tool architects: *build once, run everywhere*.
- It clearly delineates the roles and responsibilities. An operating systems specialist is responsible for creating and maintaining the base image. The software specialist creates and maintains bundles. And the cloud architect assembles and validates the final product: a virtual appliance.
- It simplifies the lifecycle management. There is a fairly small number of basic building blocks such as base images, and software bundles.
- It simplifies the patch management of the base image. The base image can be used to build different software solutions that are supported in the same operating system level.
- The software components are configured at deployment time to meet specific environment requirements or business needs.

This method works well for organizations that do not have many existing virtual images that were built over time. Sometimes it is advantageous to start adopting new technology, such as cloud, when the technology reaches a certain level of maturity.

The method to create a virtual image (second approach) might be practical in cases where an organization wants to quickly move to the cloud deployment model. Such organizations prefer this method because they do not want the up-front investment that is required to create fully functional bundles for all the software components in the solution stack.

A typical use case for the second approach is an organization that uses a stable, hardened, and matured software solution. The organization wants to cut the time required to deploy this well-defined and slowly changing stack. In this case, the organization can use the tool to point to a virtual machine that is not part of a workload and is running the software stack, and import it as a gold master image. Such an appliance is not self-configurable or ready for

automatic deployment. The tool injects the IBM Virtual Solutions Activation Engine (VSAE) during the import process. However, there are no activation programs for the software components that run above the operating system (for example, middleware and user applications). A solution architect can either insert the activation programs manually or use the Product Activator Development Kit (PADK) to quickly and easily implement the necessary activation programs.

Advantages of the second method are as follows:

- ▶ It can be used to quickly convert existing stacks into self-configurable virtual appliances.
- ▶ It uses existing assets such as previously built virtual images (built for VMware).
- ▶ Lowers the test and validation effort (assuming that the stack was previously tested in production).

Inconveniences of the second method are as follows:

- ▶ Software with hard dependencies in the network configuration might be difficult to reconfigure during the virtual appliance deployment.
- ▶ Some software components or applications may be easier to install from scratch rather than upgrading directly in the base image.

Although the second approach might be appealing at first, the organizations that adopt it must make long-term plans on how to gradually make the transition to the first approach. The advantages of the first approach outweigh the initial savings of the second approach.

## 12.4 Best practices for software bundles

We suggest the following practices when you create software bundles in the Image Construction and Composition Tool:

- ▶ Use either of the following options when creating software bundles to install and configure applications:
  - Install during the image synchronization and configure during the virtual appliance deployment.
  - Install and configure during the virtual appliance deployment. Some applications might be difficult to reconfigure after installing with default values; the best approach is to perform both steps, installation and configuration, when the appliance deploys. Also, some applications might require an activation license key during the installation, which the user enters and accepts. Further, some licenses might have dependencies on the machine serial number or partition ID. The process takes longer with this approach.
- ▶ Use the reset operation when a software bundle includes a configuration operation and you are creating a new base image to support other software solutions. The reset operation returns the virtual system that is deployed during the image synchronization to an initial state.
- ▶ Consider that the reset operation runs only if the activation was previously done. If you run any testing in your virtual system after the image synchronization, you cannot expect a reset program to run and cleanup the system without a previous activation.

- Use sequence numbers to name each configuration operation. For example, if you are deploying an enterprise application running on a WebSphere Application Server instance, which has a dependency on a database configuration on IBM DB2, you can define the operation names as follows:
  - 01activate.DB2: This configuration operation configures the DB2 product and has network configuration dependencies, for example, activate.vmc.system-networking and activate.vmc.network-interface.
  - 02activate.EnterpriseApplicationDatabase: This configuration operation creates and configures the application database. It is dependent on 01activate.DB2 operation.
  - 03activate.WAS: This configuration operation creates the WebSphere Application Server profile and starts the application server. It is dependent on 02activate.EnterpriseApplicationDatabase operation.
  - 04activate.EnterpriseApplicationOnWAS: This configuration operation installs and configures the enterprise application in the WebSphere Application Server. It is dependent on 03activate.WAS operation.

Notice that the configuration operations could be in one software bundle or in different separate software bundles.

- Only one script is run by each command operation. If more scripts must run, mark them as executable and run them from the activation program script.
- If you use a text editor in Microsoft Windows to write a shell script, convert its disk operating system (DOS) format to the UNIX format. DOS new line characters prevent the script from running. You can install CygUtils for Windows and convert the format using the **dos2unix** command. CygUtils is available at the following location:

<http://gnuwin32.sourceforge.net/packages/cygutils.htm>

## 12.5 Writing shell scripts for software bundle operations

As mentioned previously, you can write your scripts in any language supported by the base image operating system. The choice of a programming language depends of your programming skills. However, we prefer to use Korn shell scripts for AIX and IBM i, and Bourne shell scripts for Linux software bundles. The Korn shell `/QOpenSys/usr/bin/sh` is the default shell in the IBM Portable Application Solutions Environment for IBM i (PASE).

This section describes several best practices that a software specialist can use to write shell scripts for install, configure, and reset applications on AIX, Linux (x86 and Linux on Power), and IBM i.

**About Windows scripts:** You can use any programming language supported in your base image to write your scripts (for example, batch scripts, CSharp, or Visual Basic). You can also run Python scripts if a wrapper is provided to start the `.py` script (because Python scripts are not directly executable). For return-code processing, you can handle the exit codes at your convenience. Non-zero exit codes are treated as an error.

Windows bundle scripts are run as the NT Authority\System user.



## Parsing arguments

Use the short-space style to pass arguments into shell scripts. Example 12-1 shows how to parse short-space style parameters using the **getopts** command.

*Example 12-1 Parsing arguments with the getopts command*

---

```
while getopts "a:b:" option
do
    case $option in
        a) InsParm1="$OPTARG";;
        b) InsParm2="$OPTARG";;
        esac
    done
```

---

Each required argument must be followed by the colon character. Use delimited values of single quotation marks for arguments that accept spaces. You must add each argument as well in the software bundle command. Even if a single character is used to name each argument, you can use a more descriptive name in the Label field. Label names are displayed during the image synchronization in the Image Construction and Composition Tool and during the virtual appliance deployment in the VMControl deploy wizard or in the IBM SmartCloud Entry deploy page.

## Bundle operation directories

use absolute paths to access any file under the bundle operation directories. Use the following commands to get the current operation directory:

- ▶ IBM i PASE

```
INSTALL_DIRECTORY=$(pwd)
CONFIGURATION_DIRECTORY=$(dirname $0)
RESET_DIRECTORY=$(dirname $0)
```
- ▶ AIX or Linux

```
INSTALL_DIRECTORY=`pwd`
CONFIGURATION_DIRECTORY=`dirname $0`
RESET_DIRECTORY=`dirname $0`
```

## Sharing parameters

Parameters are important for communicating with the software and applications; they are common in the install and configuration operations because you can pass values during the image synchronization and the virtual appliance deployment. However, the reset operation is performed by the activation engine and you cannot pass parameters during the image capture. One option is to have only one configuration operation per bundle, as explained in 6.7.1, “Passing parameters to the reset operation” on page 107. Another option is to create a properties file which contains a collection of key-value pairs to share any parameter or variable across software bundle operations. We recommend saving the properties file in /etc for AIX or Linux, and /QOpenSys/opt/ibm/ae for IBM i.

Example 12-2 and Example 12-3 show how to create a properties file from a configuration script.

- IBM i PASE is shown in Example 12-2.

---

*Example 12-2 Creating a properties file in IBM i PASE shell script*

---

```
IMAGE_PROPS=/QOpenSys/opt/ibm/ae/virtualimage.properties
touch $IMAGE_PROPS
chmod +r $IMAGE_PROPS
echo "KEY_A=VALUE_A" >> $IMAGE_PROPS
echo "KEY_B=VALUE_B" >> $IMAGE_PROPS
echo "KEY_C=VALUE_C" >> $IMAGE_PROPS
```

---

By saving the properties file in the /QOpenSys/opt/ibm/ae directory, you ensure that the QPGMR user can read and write the file. If you decide to create the properties file using your installation script during the synchronization, then change its user owner to QSYS. This user owns the /QOpenSys/opt/ibm/ae directory. You can change the user owner by including the following command in your script:

```
chown qsys /QOpenSys/opt/ibm/ae/virtualimage.properties
```

- AIX or Linux is shown in Example 12-3.

---

*Example 12-3 Creating a properties file in AIX or Linux shell script*

---

```
IMAGE_PROPS=/etc/virtualimage.properties
touch $IMAGE_PROPS
chmod +r $IMAGE_PROPS
echo "KEY_A=VALUE_A" >> $IMAGE_PROPS
echo "KEY_B=VALUE_B" >> $IMAGE_PROPS
echo "KEY_C=VALUE_C" >> $IMAGE_PROPS
```

---

Add the following command to make the parameter or variable values visible in a shell script.

```
. $IMAGE_PROPS
```

Where \$IMAGE\_PROPS is the absolute path of the image properties file. Save the values with spaces in the properties file as single quotation-mark-delimited, as in the following example:

```
echo "KEY_A='VALUE_A WITH SPACES'" >> $IMAGE_PROPS
```

## Script output

Use the **set -vx** command at the beginning of a script only when hidden parameters are not used. Otherwise, hidden values such as passwords are shown in the Image Construction and Composition Tool log files or the activation engine report files. Use the code in Example 12-4 to turn on debugging options during script execution.

---

*Example 12-4 Sending the output of a script execution portion to the log files*

---

```
set -vx
<command 1>
<command 2>
...
<command N>
set -
```

---

## Script exit code

The error handling within the executable scripts in a software bundle is optional. However, handling an exit code in an installation script helps the Image Construction and Composition Tool to display the correct state of the image synchronization task. Likewise, handling the exit code in the configuration and reset scripts helps the activation engine to log the correct state of each operation. Non-zero exit codes are treated as an error.

## Software binary files

Large binary installation files (in the range of gigabytes), can be available in an internal Network File System (NFS) server or File Transfer Protocol (FTP) server. An installation script can access the software repository server by mounting a remote file system or transferring the file into the active virtual system that is deployed during the image synchronization. A large binary file can also be split into smaller pieces. Use a size in the range of 400 - 500 MB. Each chunk file can be easily uploaded into a software bundle. First, upload the binary file to the AIX or Linux server where the Image Construction and Composition tool is installed and then run the following command:

```
split -b <size of smaller files in MB>m <large binary file name> <prefix>_
```

The following line is an example:

```
split -b 400m Notesi853.zip Notesi853.zip_
```

Upload each chunk file in to the bundle install files and add the following command in the installation script to merge them:

- ▶ IBM i PASE

```
/QOpenSys/usr/bin/cat Notesi853.zip_* > Notesi853.zip
```

- ▶ AIX or Linux

```
cat Notesi853.zip_* > Notesi853.zip
```

## Run commands as a different user

On AIX and Linux, all bundle operations run as root by default. If a command must run under a different user ID, first ensure the user exists in the base image OS or ensure this is created during the bundle operation. Use the **su** command to run a command as a different user ID. We recommend to put all commands that must run under a different user ID into a new shell script file, and then run this script from the activation program script using the **su** command. This approach is useful when running DB2 commands or the WebSphere Application Server **wsadmin** tool.

## 12.5.1 Best practices when writing shell scripts for IBM i

Take into account the following recommendations when writing shell scripts for IBM i:

- ▶ Run system calls as follows:

```
/QOpenSys/usr/bin/system "<CL command or program call>"
```

- ▶ Do not use the **i** flag for system calls.

- ▶ The files that are uploaded in the install operation of a bundle, are transferred to the QOpenSys file system during the image synchronization. Use the **CPYFRMSTMF** command to copy a file into a member or a save file located in the integrated file system (IFS).

- ▶ Use the **call qp2term** or the **QP2SHELL2** command to test PASE Korn shell scripts.

- ▶ Use the **SBMJOB** command to run a CL program during the configuration operation when the CL program in a multi-threaded job is not thread-safe.

- Use the **STRQSH** command to run Qshell scripts from a PASE Korn shell script. For example, use the following command to run WebSphere Application Server Qshell scripts that are in the application server root or the profile root directory.

```
/QOpenSys/usr/bin/system \
"STRQSH CMD('<Qshell script> <parameter one> <parameter two>')"
```

- Installation and reset scripts can run under the QSECOFR user profile. Consider that configuration operations are startup services that run once during the virtual appliance deployment and are run under the QPGMR user profile.

**Important:** The activation engine is installed in the /QOpenSys/opt/ibm/ae directory. The scripts listed in /QOpenSys/opt/ibm/aescripts call the activation engine by passing the configuration operation name defined in the software bundle, and are started in alphabetical order at IBM i boot-up time (IPL). On IBM i, only product activation dependencies are supported. Dependencies are defined on each bundle configuration operation.

## Using profile handles

Activation programs run under the QPGMR user. If you need to adopt \*SECOFR or other authority at deployment time to perform your configuration, create a CL program to change the QPGMR configuration job to the appropriate user profile that has all the required privileges. Example 12-5 shows a CL program to change a job to the QSECOFR user. This CL program can be created during the image synchronization, and must be owned by QSECOFR and adopt \*OWNER authority.

*Example 12-5 BM i CL program to change a profile handle to adopt \*SECOFR authority*

```
/*=====*/
/* MODULE NAME: ICCTCONFIG */
/* FILE NAME: ICCTConfig.mbr */
/* */
/*=====*/
/* IMPORTANT: */
/* */
/*This program must be owned by QSECOFR and adopt *OWNER authority, */
/*otherwise the automated configuration fails due to lack of authority.*/
/* */
/*=====*/

PGM PARM(&SCRIPT &CFGPARM1 &CFGPARM2)
/* */
/*-----Input Parameter Declarations-----*/
DCL VAR(&CFGPARM1) TYPE(*CHAR)
DCL VAR(&CFGPARM2) TYPE(*CHAR)
DCL VAR(&SCRIPT) TYPE(*CHAR)
DCL VAR(&NULL) TYPE(*CHAR) LEN(1) VALUE(X'00')
DCL VAR(&CMD) TYPE(*CHAR) LEN(100)
/* */
/*-----Program Declarations -----*/
DCL VAR(&CURUSER) TYPE(*CHAR) LEN(10) +
VALUE(*CURRENT)
DCL VAR(&CURHANDLE) TYPE(*CHAR) LEN(12)
DCL VAR(&ADMUSER) TYPE(*CHAR) LEN(10) +
VALUE('QSECOFR')
DCL VAR(&ADMHANDLE) TYPE(*CHAR) LEN(12)
/* */
/*-----Global Monitor Message-----*/
/* */
```

```

MONMSG      MSGID(CPF0000) EXEC(GOTO CMDLBL(FAILED))

/*
/*-----Change Profile to SEQCOFR-----*/
/*
/*Generate a profile handles for the user ID this program is
/*currently running under. When *CURRENT is specified for the
/*the user ID, the password field is ignored
/*
CALL      PGM(QSYGETPH) PARM(&CURUSER ' ' +
                        &CURHANDLE)

/*-----Generate profile handle for QSECOFR-----*/
CALL      PGM(QSYGETPH) PARM(&ADMUSER '*NOPWDCHK' +
                        &ADMHANDLE)

/*-----Change the job to use QSECOFR-----*/
CALL      PGM(QWTSETP) PARM(&ADMHANDLE)

/*
/*Perform any required configuration here using the parameters
/*
/*PASE command example using a shell script
/*

CHGVAR VAR(&CMD) VALUE(&SCRIPT *CAT +
'-a' *BCAT &CFGPARAM1 *BCAT +
'-b' *BCAT &CFGPARAM2)

CHGVAR VAR(&CMD) VALUE(&CMD *TCAT &NULL)

CALL PGM(QP2SHELL2) PARM('/QOpenSys/usr/bin/sh' &CMD)

/*CL command example
/*

ADDLIB    LIB(&CFGPARAM1)
ADDLIB    LIB(&CFGPARAM2)
MONMSG    MSGID(CPF2103)
QGPL/<IBM PROGRAM>
/*-----Reset job back to original profile-----*/
/*
CALL      PGM(QWTSETP) PARM(&CURHANDLE)

/*release handles*/
CALL      PGM(QSYRLSPH) PARM(&CURHANDLE)
CALL      PGM(QSYRLSPH) PARM(&ADMHANDLE)

EXIT:      /*normal end of program*/

RETURN

/*-----Exception Routine-----*/
FAILED:    SNDPGMMSG MSGID(CPF0001) MSGF(QCPFMSG) +
MSGDTA('Configuration failed.') +
MSGTYPE(*ESCAPE)
ENDPGM

```

---

Some configuration parameters may accept values without a fixed length. The CL program can use *variables* to assign these configuration parameters. Likewise, the activation program script can write a properties file and share all the necessary parameters that are passed at deployment time.

Add the Example 12-6 code in the installation script to create the CL program from Example 12-5.

*Example 12-6 Creating an IBM i CL program*

---

```
INSTALL_DIRECTORY=$(pwd)
/QOpenSys/usr/bin/system "CPYFRMSTMF \
FROMSTMF('$INSTALL_DIRECTORY/ICCTConfig.mbr') \
TOMBR('/QSYS.LIB/QGPL.LIB/QCLSRC.FILE/ICCTCONFIG.MBR') MBROPT(*REPLACE)"
/QOpenSys/usr/bin/system "CRTCLPGM PGM(QGPL/ICCTCONFIG) \
SRCFILE(QGPL/QCLSRC) SRCMBR(*PGM) TEXT('ICCT Configuration') USRPRF(*OWNER)"
```

---

## Calling a helper script

You can use Example 12-5 on page 275 to call a *helper* shell script that requires \*SECOFR authority to perform the required configuration. Upload this script as part of the installation files and mark it as executable. You can move this *helper* script to a well-known directory during the image synchronization by using the Example 12-7 code in your installation script.

*Example 12-7 Moving a helper script to a well-known directory*

---

```
INSTALL_DIRECTORY=$(pwd)
HELPER_ACTIVATION_DIR=/tmp/my_application
mkdir -p $HELPER_ACTIVATION_DIR
mv $INSTALL_DIRECTORY/helper.sh $HELPER_ACTIVATION_DIR
```

---

Example 12-8 shows how to call the CL program created in Example 12-6 to run a *helper* script.

*Example 12-8 Calling a helper script using parameters*

---

```
HELPER_ACTIVATION_SCRIPT=/tmp/my_application/helper.sh
/QOpenSys/usr/bin/system "CALL PGM(QGPL/ICCTCONFIG) \
PARAM('$HELPER_ACTIVATION_SCRIPT ' 'parm_A' 'parm_B')"
```

---

The space after \$HELPER\_ACTIVATION\_SCRIPT is important to pass the parameters properly. The code in Example 12-8 can be used in the activation program script that is run by the activation engine under the QPGMR user. Here, the helper.sh script is called from **ICCTCONFIG** to adopt \*SECOFR authority and perform the configuration. You can use the approach of passing the parameters to helper.sh, or create a properties file from your activation program script to share parameters or variables.

**Using a well-known directory:** Although the *helper* script can be added to the configuration files within your bundle, the configuration path might be too long to be passed to a CL program. In addition, you might need to change the &CMD variable length in Example 12-5 on page 275 if the bundle universal ID or version changes. Using a well-known directory keeps the *helper* script absolute path with a fixed length. Be sure the &CMD variable length is equal to or greater than the *helper* script absolute path plus its parameters.

## 12.6 Testing the product activation

You must deploy the virtual appliance that was created in the Image Construction and Composition Tool to be able to validate its activation. The activation may include the configuration of several applications or different software components. In addition, it may take place in a multitiered application where service dependencies are involved. In this case, you might want to test the product activation for each configuration operation before the image capture. The virtual system that is deployed during the image synchronization can be used as a test system to validate the activation. When the activation is working properly, you can extend the base image, add your bundles, synchronize again and capture.

An activation program or *product activator* is the executable script in a configuration operation that runs during the virtual appliance deployment. You can test your activation programs in the activation engine interactive mode, as explained in “When deploying, reconfiguration of a component fails” on page 121.

The PADK plug-in can help you to test the activation programs in a running virtual system before capturing the image. On AIX or Linux, the PADK can also be used with a wrapper to test the activation programs that use shell scripts. The PADK is not supported to run activation programs on IBM i.

### 12.6.1 Testing the product activation on IBM i

Ensure that your activation program runs a CL program that changes a profile to adopt \*SEC0FR authority or any other user profile with the authority that is required to perform the configuration during the virtual appliance deployment. Each activation program runs under the QPGMR user profile and this user cannot run the AE.sh shell script in the /QOpenSys/opt/ibm/ae directory. Therefore, you must run the activation engine as QSEC0FR.

To test the product activation, complete the following steps:

1. Download the /tmp/ovf-env.xml file from your IBM i system with FTP.
2. Rename the file as test-env.xml. Use your preferred text editor to enter an argument value on each property element. In Example 12-9 notice that values with spaces for *activate.Domino.admin\_name* and *activate.Domino.admin\_last\_name* are delimited by single quotation marks.

*Example 12-9 Test file test-env.xml*

---

```
<?xml version="1.0" encoding="ASCII"?> <ovfenv:Environment
xmlns:ovfenv="http://schemas.dmtf.org/ovf/environment/1" ovfenv:id="vs0">
  <ovfenv:PropertySection>
    <ovfenv:Property ovfenv:key="Activation.traceEnabled"
ovfenv:value="true"/>
    <ovfenv:Property ovfenv:key="Activation.file.log.level"
ovfenv:value="DEBUG"/>
    <ovfenv:Property ovfenv:key="Activation.stream.log.level"
ovfenv:value="DEBUG"/>
    <ovfenv:Property ovfenv:key="activate.Domino.server_ipv4_address"
ovfenv:value="172.26.45.201"/>
    <ovfenv:Property ovfenv:key="activate.Domino.server_name"
ovfenv:value="vmDomSrv"/>
    <ovfenv:Property ovfenv:key="activate.Domino.organization_name"
ovfenv:value="vmOrg"/>
```

```

        <ovfenv:Property ovfenv:key="activate.Domino.domain_name"
ovfenv:value="vmdomain"/>
        <ovfenv:Property ovfenv:key="activate.Domino.admin_name"
ovfenv:value="'J Moises'"/>
        <ovfenv:Property ovfenv:key="activate.Domino.admin_last_name"
ovfenv:value="'Romo C'"/>
        <ovfenv:Property ovfenv:key="activate.Domino.admin_password"
ovfenv:value="passw0rd"/>
        <ovfenv:Property ovfenv:key="activate.Domino.time_zone"
ovfenv:value="cst"/>
    </ovfenv:PropertySection>
</ovfenv:Environment>

```

---

3. Upload the test-env.xml file into the /tmp directory of your IBM i system.
4. Open a 5250 session to your IBM i system. Log in to the system as QSEC0FR.
5. Start an interactive terminal session for PASE. Enter the following command:

```
call qp2term
```
6. Change into the activation engine directory.

```
cd /Q0penSys/opt/ibm/ae
```
7. Enter the following command to test your activation program in the interactive mode:

```
./AE.sh -i /tmp/test-env.xml
```
8. Answer **y** for each activation you want to test.
9. You can also test the reset operations by running the following command:

```
./AE.sh -I /tmp/test-env.xml
```
10. Answer **y** for each reset operation you want to test. Answer **n** when prompting to shut down the system.

## 12.7 Split large virtual images

An Open Virtual Appliance (OVA) archive can be too large when exporting most of the virtual appliances in the Image Construction and Composition Tool. Eventually, you might want to transfer OVA archives within your private network or across the Internet. Typically, the Image Construction and Composition Tool server itself is used to receive the OVA archive when exporting an appliance. The **split** command can be used in AIX or Linux to split the OVA archive into smaller pieces that can be easily transferred to a different location.

To split the OVA archive in AIX, complete the following steps:

1. Log in to the AIX server.
2. Change into the OVA archive directory.
3. Run this command:

```
split -b <size of smaller files in MB>m <ova archive name> <prefix>
```

An example of this command is as follows:

```
split -b 4500m MySampleApp_Domino853_IBMi71_100GB_SCS.ova.zip \
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_
```



4. Verify whether the files are created, as in the following example:

```
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_aa  
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_ab  
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_aN
```

In the example, *aN* is ac, ad, ae, and so on.

5. Optional: Rename the files as follows:

```
mv MySampleApp_Domino853_IBMi71_100GB_SCS.ova_aa \  
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_1ofN.zip  
  
mv MySampleApp_Domino853_IBMi71_100GB_SCS.ova_ab \  
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_2ofN.zip  
...  
mv MySampleApp_Domino853_IBMi71_100GB_SCS.ova_aN \  
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_NofN.zip
```

To merge the files again, use the following steps:

1. Change into the directory of the files you want to merge.
2. Run the command:

```
cat MySampleApp_Domino853_IBMi71_100GB_SCS.ova_* > \  
MySampleApp_Domino853_IBMi71_100GB_SCS.ova.zip
```

### 12.7.1 Validating files when splitting an OVA archive

After splitting an OVA archive, validate the file integrity using MD5 hashes when transferring the chunk files to a different location. Use the following steps to install the **md5sum** command in AIX and check MD5 hashes:

1. Download the **coreutils** RPM from the AIX Toolbox.
2. Install **coreutils** using the following command:  

```
rpm -iv coreutils-5.2.1-2.aix5.1.ppc.rpm
```
3. Get the MD5 hashes for each chunk file and redirect the output to a file.

```
md5sum MySampleApp_Domino853_IBMi71_100GB_SCS.ova_1ofN.zip >> MD5SUMS  
md5sum MySampleApp_Domino853_IBMi71_100GB_SCS.ova_2ofN.zip >> MD5SUMS  
md5sum MySampleApp_Domino853_IBMi71_100GB_SCS.ova_NofN.zip >> MD5SUMS
```

4. If the OVA chunk files are transferred to a separate location along with the MD5SUMS file, you can validate all MD5 hashes as follows:

```
md5sum -c MD5SUMS
```

The following output shows that the MD5 hashes are valid:

```
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_1ofN.zip: OK  
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_2ofN.zip: OK  
MySampleApp_Domino853_IBMi71_100GB_SCS.ova_NofN.zip: OK
```



## Sample script for KVM host setup

Use the shell script listing shown in Example A-1 to guide you through the process of a KVM host setup. Remember to modify the network settings to match your environment.

*Example A-1 KVM host setup script*

---

```
#!/bin/bash

yum update --nogpgcheck --assumeyes

# kvm install
yum --nogpgcheck --assumeyes install kvm
yum --nogpgcheck --assumeyes install virt-manager libvirt libvirt-Python
Python-virtinst virt-viewer perl-XML*
modprobe kvm kvm-intel
#lsmod | grep kvm
chkconfig libvirtd on
service libvirtd restart
virsh list -all

# web.py install; you may want to check the website for the latest stable version
wget http://webpy.org/static/web.py-0.36.tar.gz
tar xvzf web.py-0.36.tar.gz
cd web.py-0.36
python setup.py install

# bridged networking
chkconfig NetworkManager off
chkconfig network on
cd /etc/sysconfig/network-scripts
interface=eth0
bridge=br0
```

```

#sudo sed -i "s/^iface $interface inet \(.*\)$/iface $interface inet
manual\n\nauto br0\niface $bridge inet \1/" /etc/network/interfaces
tee -a /etc/sysconfig/network-scripts/ifcfg-$bridge <<EOF
DEVICE=$bridge
TYPE=Bridge
BOOTPROTO=none
ONBOOT=yes
#HOSTNAME=hmb40.rchland.ibm.com
IPADDR=192.168.71.87
NETMASK=255.255.255.0
EOF

mv ifcfg-$interface __ifcfg-$interface
mac=`ifconfig $interface | egrep -o '([[:xdigit:]]{2}[:])}{5}[[:xdigit:]]{2}'`
tee -a /etc/sysconfig/network-scripts/ifcfg-$interface <<EOF
DEVICE=$interface
#Remember to use the actual MAC address
HWADDR=$mac
ONBOOT=yes
BRIDGE=$bridge
EOF

#Default Gateway
tee -a /etc/sysconfig/network <<EOF
GATEWAY=192.168.71.1
EOF

service network restart

#forward across the bridge
iptables -I FORWARD -m physdev --physdev-is-bridged -j ACCEPT
service iptables save
service iptables restart

service libvirtd reload
brctl show

#DNS setup
mv /etc/resolv.conf /etc/__resolv.conf
> /etc/resolv.conf
tee -a /etc/resolv.conf <<EOF
search ovafactory.rchland.ibm.com
nameserver 192.168.71.38
EOF

```

---



## Sample scripts for simple virtual appliance

The scripts in this appendix are provided as reference for Chapter 9, “Constructing simple virtual appliances” on page 181. This appendix includes the following scripts:

- ▶ Installation script for WebSphere Application Server Community Edition
- ▶ Configuration script for WebSphere Application Server Community Edition
- ▶ Configuration script for PlantsByWebSphere

# Installation script for WebSphere Application Server Community Edition

Example B-1 shows the installation script for the WebSphere Application Server Community Edition.

*Example B-1 WebSphere Application Server Community Edition installation script*

---

```
function usage
{
    echo ""
    echo " Usage : "
    echo "      install.sh -wasce_install_path <WASCE_INSTALL_PATH>
-installed_java_home <INSTALLED_JAVA_HOME> -wasce_install_exec <WASCE_INSTALL_EXEC>
-wasce_zip_name <WASCE_ZIP_NAME>"
    echo ""
    exit 1
}

echo "$0: === Install IBM WAS CE === "
echo "$0:   Arguments are: $* "

#Init
WASCE_INSTALL_PATH=
INSTALLED_JAVA_HOME=
WASCE_INSTALL_EXEC=
WASCE_ZIP_NAME=
INSTALL_FLAG='0'

#Read arguments

while [ $# -ne 0 ]
do
    case $1 in
        -wasce_install_path*)
            WASCE_INSTALL_PATH=$2
            ;;
        -installed_java_home*)
            INSTALLED_JAVA_HOME=$2
            ;;
        -wasce_install_exec*)
            WASCE_INSTALL_EXEC=$2
            ;;
        -wasce_zip_name*)
            WASCE_ZIP_NAME=$2
            ;;
        *)
            ;;
    esac
    shift 1
done

#create the temporary directory
mkdir -p "/tmp/wasce"
if [ ! -d "/tmp/wasce" ]; then
    echo "$0: failed to create /tmp/wasce" >&2
```

```

    exit 1
fi
echo "DONE CREATING TEMP DIR"

#install the utilities
echo "Installing sed and unzip"
rpm -Uvh sed-4.1.1-1.aix5.1.ppc.rpm
rpm -Uvh unzip-5.51-1.aix5.1.ppc.rpm
echo "DONE INSTALLING sed and unzip"

echo "Install method correctly mentioned. Received the install file."

#unzip the installation package to temporary directory
unzip $WASCE_ZIP_NAME -d /tmp/wasce
if [ $? -gt 0 ]; then
    echo "$0: unzip <file.zip> decompress command failed"
    exit 1
fi
echo "UNZIP DECOMPRESSED THE FILE SUCCESSFULLY"

# Make sure Java is in the path for installation
export PATH=$INSTALLED_JAVA_HOME/jre/bin:$PATH
if [ $? -gt 0 ]; then
    echo "$0: failed to add java bin to environment variable path"
    exit 1
fi
echo "ADDED JAVA HOME PATH to PATH VARIABLE SUCCESSFULLY"
echo "PATH=$PATH"

#Change into the temporary directory
cd /tmp/wasce
if [ $? -gt 0 ]; then
    echo "$0: failed to change directory to /tmp/wasce"
    exit 1
fi
echo "DONE CHANGING DIRECTORY"

# Initiate the silent installation fo WebSphere Application Server Community Edition
chmod +x ./$WASCE_INSTALL_EXEC
./$WASCE_INSTALL_EXEC -i silent -DUSER_INSTALL_DIR=$WASCE_INSTALL_PATH
if [ $? -gt 0 ]; then
    echo "$0: WASCE install failed"
    exit 1
fi
echo "WASCE INSTALL COMPLETED SUCCESSFULLY"

# Remove the temporary directory
cd /tmp
rm -rf /tmp/wasce
exit 0

```

---

# Configuration script for WebSphere Application Server Community Edition

Example B-2 shows the configuration script for WebSphere Application Server Community Edition.

*Example B-2 WebSphere Application Server Community Edition configuration script*

---

```
logfile=/tmp/wasceConfigout.log
HOSTNAME=`hostname`
echo "Configuring WAS CE with host $HOSTNAME" >> $logfile

WASCE_HOME=/opt/IBM/WebSphere/AppServerCommunityEdition
num_servers=1
WASCE_ADMIN_USER="system"
WASCE_ADMIN_PASSWORD="manager"

while [ $# -ne 0 ]
do
    case $1 in
        -num_servers*)
            num_servers=$2
            ;;
        -WASCE_HOME*)
            WASCE_HOME=$2
            ;;
        -WASCE_ADMIN_USER*)
            WASCE_ADMIN_USER=$2
            ;;
        -WASCE_ADMIN_PASSWORD*)
            WASCE_ADMIN_PASSWORD=$2
            ;;
        *)
            ;;
    esac
    shift 1; shift 1
done

# provide variables to be consumed by other scripts
echo "WASCE_HOME=$WASCE_HOME" > /etc/virtualimage.properties
echo "WASCE_ADMIN_USER=$WASCE_ADMIN_USER" >> /etc/virtualimage.properties
echo "WASCE_ADMIN_PASSWORD=$WASCE_ADMIN_PASSWORD" >> /etc/virtualimage.properties

svr_ct=1

/opt/freeware/bin/sed -i s/"ServerHostname = 0.0.0.0"/"ServerHostname =
$HOSTNAME"/g $WASCE_HOME/var/config/config-substitutions.properties
/opt/freeware/bin/sed -i s/"RemoteDeployHostname =
localhost"/"RemoteDeployHostname = $HOSTNAME"/g
$WASCE_HOME/var/config/config-substitutions.properties
/opt/freeware/bin/sed -i
s/"system=master"/"$WASCE_ADMIN_USER=$WASCE_ADMIN_PASSWORD"/g
$WASCE_HOME/var/security/users.properties
/opt/freeware/bin/sed -i s/"admin=system"/"admin=$WASCE_ADMIN_USER"/g
$WASCE_HOME/var/security/groups.properties

$WASCE_HOME/bin/startup.sh
```

```

echo "Configuring $num_servers server instance(s)" >> $logfile

while [ $svr_ct -lt $num_servers ]
do
    let svr_ct++
    instName="instance$svr_ct"
    echo "Creating $instName instance" >> $logfile
    mkdir $WASCE_HOME/$instName
    cp -r $WASCE_HOME/var $WASCE_HOME/$instName
    let y=$svr_ct-1
    /opt/freeware/bin/sed -i s/"PortOffset=0"/"PortOffset=$y"/g
    $WASCE_HOME/$instName/var/config/config-substitutions.properties
    GERONIMO_OPTS=-Dorg.apache.geronimo.server.name=$instName
    export GERONIMO_OPTS=$GERONIMO_OPTS
    $WASCE_HOME/bin/geronimo.sh start
    echo "Started $instName instance" >> $logfile
done

exit 0

```

---

## Configuration script for PlantsByWebSphere

Example B-3 shows the configuration script for PlantsByWebSphere.

### *Example B-3 PlantsByWebSphere configuration script*

---

```

echo "Configuring WAS CE Datasource" >> $logfile
HOSTNAME=`hostname`
WASCE_HOME=/opt/IBM/WebSphere/AppServerCommunityEdition
WASCE_ADMIN_USER="system"
WASCE_ADMIN_PASSWORD="manager"

# source environment variables
. /etc/virtualimage.properties

# -----
# -- deploy the datasource
# -----
${WASCE_HOME}/bin/deploy.sh --host $HOSTNAME --user $WASCE_ADMIN_USER --password
$WASCE_ADMIN_PASSWORD deploy
${WASCE_HOME}/repository/org/tranql/tranql-connector-derby-embed-xa/1.7/tranql-connector-de
rby-embed-xa-1.7.rar plan.xml
echo "Datasource configured!"

# -----
# -- deploy the application
# -----
${WASCE_HOME}/bin/deploy.sh --host $HOSTNAME --user $WASCE_ADMIN_USER --password
$WASCE_ADMIN_PASSWORD deploy PlantsByWebSphereEAR-3.0.0.3.ear pbw-plan.xml
echo "PlantsByWebSphere Deployed!"

exit 0

```

---







# Editing the generic.ovf file for IBM SmartCloud Entry deployment

To correct the issue of no labels when deploying a KVM virtual appliance in IBM SmartCloud Entry, add the bold-font lines, shown in Example C-1, to your generic.ovf file so that all new appliances have the correct parameters. You can also edit the OVF file for a specific virtual appliance.

*Example C-1 Add the bold font lines*

---

```
<ovf:ProductSection ovf:class="com.ibm.ovf.vmcontrol.adapter.networking" ovf:instance="4"
  <ovf:info>Network adapter configuration for Network adapter 0 on Discovered-breth0-0
</ovf:info>
  <ovf:Category>Networking adapter configuration for Network adapter 0 on
  Discovered-breth0-0</ovf:Category>
</ovf:ProductSection>
<ovf:ProductSection ovf:class="com.ibm.vsaes.2_1.system-user">
  <ovf:info>Activate system users</ovf:info>
  <ovf:Product>activate-system-user</ovf:Product>
  <ovf:Version>2.1</ovf:Version>
  <ovf:Category>Activate system user</ovf:Category>
  <ovf:Property ovf:key="username" ovf:type="string"
    ovf:userConfigurable="true" ovf:value="root"
    <ovf:Description>User name</ovf:Description>
    <ovf:Label>User name</ovf:Label>
  </ovf:Property>
  <ovf:Property ovf:key="password" ovf:type="string" ovf:userConfiguration="true"
    ovf:value="s3vur3!">
    <ovf:Description>User password</ovf:Description>
    <ovf:Label>User password</ovf:Label>
  </ovf:Property>
  <ovf:Property ovf:key="gecos" ovf:type="string" ovf:userConfiguration="true"
    ovf:value="Super User (root@localhost)">
    <ovf:Description>Gecos</ovf:Description>
    <ovf:Label>Gecos</ovf:Label>
```

```

    </ovf:Property>
<ovf:Property ovf:key="homedir"
  ovf:type="string" ovf:userConfiguration="false" ovf:value="/root">
  <ovf:Description>Home directory</ovf:Description>
  <ovf:Label>Home directory<ovf:Label>
</ovf:Property>
<ovf:Property ovf:key="create_home_dir" ovf:type="string" ovf:userConfiguration="false"
  ovf:value="false">
  <ovf:Description>Create home directory</ovf:Description>
  <ovf:Label>Create home directory<ovf:Label>
</ovf:Property>
<ovf:Property ovf:key="uid" ovf:type="string" ovf:userConfiguration="false"
  ovf:value="0">
  <ovf:Description>User ID</ovf:Description>
  <ovf:Label>User ID<ovf:Label>
</ovf:Property>
<ovf:Property ovf:key="gid" ovf:type="string" ovf:userConfiguration="false"
  ovf:value="0">
  <ovf:Description>Group ID</ovf:Description>
  <ovf:Label>Group ID<ovf:Label>
</ovf:Property>
<ovf:Property ovf:key="shell" ovf:type="string" ovf:userConfiguration="false"
  ovf:value="/bin/sh">
  <ovf:Description>User shell program</ovf:Description>
  <ovf:Label>User shell program<ovf:Label>
</ovf:Property>
</ovf:ProductSection>
<ovf:ProductSection ovf:class="com.ibm.vase.2_1.network-interface">
  <ovf:info>System network interface configuration</ovf:info>
  <ovf:Product>activate-network-interface</ovf:Product>
  <ovf:Version>2.1<ovf:Version>
  <ovf:Category>System network interface configuration</ovf:Category>
  <ovf:Property ovf:key="interface" ovf:type="string"
    ovf:userConfigurable="false" ovf:value="eth0">
    <ovf:Description>Network interface</ovf:Description>
    <ovf:Label>Network interface<ovf:Label>
  </ovf:Property>
  <ovf:Property ovf:key="usedhcp" ovf:type="string"
    ovf:userConfigurable="true" ovf:value="false">
    <ovf:Description>Use DHCP</ovf:Description>
    <ovf:Label>Use DHCP<ovf:Label>
  </ovf:Property>
  <ovf:Property ovf:key="ipaddr" ovf:type="string"
    ovf:userConfigurable="true" ovf:value="192.168.71.129">
    <ovf:Description>IP address</ovf:Description>
    <ovf:Label>IP address<ovf:Label>
  </ovf:Property>
  <ovf:Property ovf:key="netmask" ovf:type="string"
    ovf:userConfigurable="true" ovf:value="255.255.255.0">
    <ovf:Description>IP netmask</ovf:Description>
    <ovf:Label>IP netmask<ovf:Label>
  </ovf:Property>
  <ovf:Property ovf:key="gateway" ovf:type="string"
    ovf:userConfigurable="true" ovf:value="192.168.71.38">
    <ovf:Description>IP gateway</ovf:Description>
    <ovf:Label>IP gateway<ovf:Label>
  </ovf:Property>
  <ovf:Property ovf:key="defaulttroute" ovf:type="string"
    ovf:userConfigurable="true" ovf:value="true">
    <ovf:Description>ls default route</ovf:Description>

```

```

        <ovf:Label>ls default route</ovf:Label>
    </ovf:Property>
</ovf:ProductSection>
<ovf:ProductSection ovf:class="com.ibm.vase.2_1system-host">
    <ovf:info>System name and domainname</ovf:info>
    <ovf:Product>activate-system-host</ovf:Product>
    <ovf:Version>2.1</ovf:Version>
    <ovf:Category>System name and domainname</ovf:Category>
    <ovf:Property ovf=key"hostname" ovf:type="string"
        ovf:userConfigurable="true" ovf:value="myhost">
        <ovf:Description>Hostname</ovf:Description>
        <ovf:Label>Hostname</ovf:Label>
    </ovf:Property>
    <ovf:Property ovf=key"domiannname" ovf:type="string"
        ovf:userConfigurable="true" ovf:value="mydomain.com">
        <ovf:Description>Domain Name</ovf:Description>
        <ovf:Label>Domain Name</ovf:Label>
    </ovf:Property>
</ovf:ProductSection>
<ovf:ProductSection ovf:class="com.ibm.vsa.2_1.dns-client">
    <ovf:info>activates the dns client</ovf:info>
    <ovf:Product>activate-dns-client</ovf:Product>
    <ovf:Version>2.1</ovf:Version>
    <ovf:Category>activates the dns client</ovf:Category>
    <ovf:Property ovf=key"sec_dns" ovf:type="string"
        ovf:userConfigurable="true" ovf:value="192.168.71.38">
        <ovf:Description>Secondary DNS</ovf:Description>
        <ovf:Label>Secondary DNS</ovf:Label>
    </ovf:Property>
    <ovf:Property ovf=key"search" ovf:type="string"
        ovf:userConfigurable="true" ovf:value="mydomain.com">
        <ovf:Description>Domain search list</ovf:Description>
        <ovf:Label>Domain search list</ovf:Label>
    </ovf:Property>
</ovf:ProductSection>
<ovf:ProductSection ovf:class="com.ibm.vsa.2_1.ntp-client">
    <ovf:info>activates the openntp client</ovf:info>
    <ovf:Product>activate-ntp-client</ovf:Product>
    <ovf:Version>2.1</ovf:Version>
    <ovf:Category>activates the openntp client</ovf:Category>
    <ovf:Property ovf=key"ntp-server" ovf:type="string"
        ovf:userConfigurable="true" ovf:value="0.pool.ntp.org">
        <ovf:Description>Ntp server</ovf:Description>
        <ovf:Label>Ntp server</ovf:Label>
    </ovf:Property>
</ovf:ProductSection>

```

---





D

## Additional material

This book refers to additional material that can be downloaded from the Internet as described in the following sections.

### Locating the web material

The web material associated with this book is available in softcopy on the Internet from the IBM Redbooks web server:

<ftp://www.redbooks.ibm.com/redbooks/SG248042>

Alternatively, you can go to the IBM Redbooks website:

[ibm.com/redbooks](http://ibm.com/redbooks)

Select the **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG248042.

### Using the web material

The additional web material that accompanies this book includes the following files:

| <i>File name</i>    | <i>Description</i> |
|---------------------|--------------------|
| <b>SG248042.zip</b> | Code samples       |

### System requirements for downloading the web material

The web material requires the following system configuration:

|                          |                        |
|--------------------------|------------------------|
| <b>Hard disk space:</b>  | 50 MB minimum          |
| <b>Operating System:</b> | Windows, Linux, or AIX |
| <b>Processor:</b>        | Power or Intel x86     |
| <b>Memory:</b>           | 256 MB                 |

## **Downloading and extracting the web material**

Create a subdirectory (folder) on your workstation, and extract the contents of the web material .zip file into this folder.

# Abbreviations and acronyms

|               |  |             |                                     |
|---------------|--|-------------|-------------------------------------|
| <b>CAS</b>    | Common Agent Services                      | <b>VM</b>   | virtual machine                     |
| <b>CLI</b>    | command-line interface                     | <b>VMFS</b> | Virtual Machine File System         |
| <b>CRM</b>    | customer relationship management           | <b>VSAE</b> | Virtual Solutions Activation Engine |
| <b>DMTF</b>   | Distributed Management Task Force          |             |                                     |
| <b>DNS</b>    | Domain Name System                         |             |                                     |
| <b>EAR</b>    | enterprise archive                         |             |                                     |
| <b>GUI</b>    | graphical user interface                   |             |                                     |
| <b>GUID</b>   | Global Unique Identifier                   |             |                                     |
| <b>HAL</b>    | hardware abstraction layer                 |             |                                     |
| <b>HMC</b>    | Hardware Management Console                |             |                                     |
| <b>IDE</b>    | integrated development environment         |             |                                     |
| <b>ISV</b>    | independent software vendor                |             |                                     |
| <b>IVM</b>    | Integrated Virtualization Manager          |             |                                     |
| <b>IaaS</b>   | infrastructure as a service                |             |                                     |
| <b>JSON</b>   | JavaScript Object Notation                 |             |                                     |
| <b>NIC</b>    | network interface card                     |             |                                     |
| <b>NIM</b>    | Network Installation Management            |             |                                     |
| <b>OPEX</b>   | operational expenses                       |             |                                     |
| <b>OS</b>     | operating system                           |             |                                     |
| <b>OVA</b>    | Open Virtual Appliance                     |             |                                     |
| <b>OVF</b>    | Open Virtualization Format                 |             |                                     |
| <b>PADK</b>   | Product Activator Development Kit          |             |                                     |
| <b>PaaS</b>   | platform as a service                      |             |                                     |
| <b>REST</b>   | REpresentational State Transfer            |             |                                     |
| <b>RHEL</b>   | Red Hat Enterprise Linux                   |             |                                     |
| <b>RXA</b>    | Remote Execution and Access                |             |                                     |
| <b>SCS</b>    | Storage Copy Services                      |             |                                     |
| <b>SEA</b>    | shared Ethernet adapter                    |             |                                     |
| <b>SLES</b>   | SUSE Linux Enterprise Server               |             |                                     |
| <b>SMB</b>    | small and medium businesses                |             |                                     |
| <b>STG</b>    | Systems and Technology Group               |             |                                     |
| <b>SaaS</b>   | software as a service                      |             |                                     |
| <b>tar</b>    | tape archive                               |             |                                     |
| <b>UML</b>    | Unified Modeling Language                  |             |                                     |
| <b>VAF</b>    | Virtual Appliance Factory                  |             |                                     |
| <b>VAFES</b>  | Virtual Appliance Factory Express Services |             |                                     |
| <b>VARepo</b> | virtual appliance repository               |             |                                     |
| <b>VIOS</b>   | Virtual I/O Server                         |             |                                     |





# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

## IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *IBM Flex System Manager*, TIPS0862
- ▶ *IBM PureFlex System and IBM Flex System Products and Technology*, SG24-7984
- ▶ *IBM Systems Director VMControl Implementation Guide on IBM Power Systems*, SG24-7829
- ▶ *IBM Workload Deployer: Pattern-based Application and Middleware Deployments in a Private Cloud*, SG24-8011

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Online resources

These websites are also relevant as further information sources:

- ▶ IBM Virtual Solutions Activation Engine User's Guide  
<https://www.ibm.com/developerworks/mydeveloperworks/groups/service/html/communityview?communityUuid=a8d4a3e3-e134-44ed-bdac-f6b3795f6e8e>
- ▶ “Working with IBM Image Construction and Composition Tool” topic in the IBM Workload Deployer Information Center  
[http://pic.dhe.ibm.com/infocenter/worlodep/v3r1m0/topic/com.ibm.worlodep.doc/IC0N/topics/iwd\\_cicn\\_overview.html](http://pic.dhe.ibm.com/infocenter/worlodep/v3r1m0/topic/com.ibm.worlodep.doc/IC0N/topics/iwd_cicn_overview.html)
- ▶ “Working with IBM Image Construction and Composition Tool” topic in IBM SmartCloud Orchestrator Information Center  
[http://pic.dhe.ibm.com/infocenter/tivihelp/v48r1/topic/com.ibm.sco.doc\\_2.2/ICON/topics/iwd\\_cicn\\_overview.html](http://pic.dhe.ibm.com/infocenter/tivihelp/v48r1/topic/com.ibm.sco.doc_2.2/ICON/topics/iwd_cicn_overview.html)
- ▶ IBM Systems Director 6.3.1 Information Center  
<http://publib.boulder.ibm.com/infocenter/director/pubs/index.jsp>
- ▶ IBM Construction and Composition Tool Software Bundles  
<https://www.ibm.com/software/brandcatalog/ismlibrary/search#query=ICCT:ICCT>
- ▶ IBM Image Construction and Composition Tool - Tutorial in YouTube  
<http://www.youtube.com/watch?v=RY66qTdY0gQ&feature=youtu.be&noredirect=1>

- ▶ IBM Image Construction and Composition Tool - Tutorials in IBM Virtual Appliance Factory  
[https://www-304.ibm.com/partnerworld/wps/servlet/ContentHandler/stg\\_com\\_eis\\_vaf\\_education\\_roadmap](https://www-304.ibm.com/partnerworld/wps/servlet/ContentHandler/stg_com_eis_vaf_education_roadmap)
- ▶ Tips and Best Practices for Software Bundle Creation for the IBM Image Construction and Composition Tool version 1.1.1  
[https://www.ibm.com/developerworks/mydeveloperworks/blogs/9e696bfa-94af-4f5a-ab50-c955cca76fd0/entry/icct\\_software\\_bundle\\_creation\\_whitepaper6?lang=en](https://www.ibm.com/developerworks/mydeveloperworks/blogs/9e696bfa-94af-4f5a-ab50-c955cca76fd0/entry/icct_software_bundle_creation_whitepaper6?lang=en)
- ▶ IBM Integrated Service Management Library  
<http://www.ibm.com/software/brandcatalog/ismlibrary>
- ▶ Using IBM Image Construction and Composition Tool: Best practices to start building virtual appliances on IBM PowerVM cloud providers  
<http://www.ibm.com/developerworks/aix/library/au-aix-image-construction/>

## Help from IBM

IBM Support and downloads

[ibm.com/support](http://ibm.com/support)

IBM Global Services

[ibm.com/services](http://ibm.com/services)



## Creating Smart Virtual Appliances with IBM Image Construction and Composition Tool

(0.5" spine)

0.475" <-> 0.873"

250 <-> 459 pages







# Creating Smart Virtual Appliances with IBM Image Construction and Composition Tool



**Redbooks®**

**Create smart appliances that are cloud-ready**

**Develop PowerVM, KVM, and ESX customized virtual appliances**

**Learn from experts how to cloudify your applications**

In a traditional deployment model, software is installed on a physical server, and it is configured for the particular data center environment. The cloud deployment model requires that the dependency on a specific hardware configuration is severed. This IBM Redbooks publication guides you through the transition from the traditional application deployment model to the cloud-friendly deployment model. It explains how to achieve these goals by packaging the software stacks into industry standard virtual appliances.

A key part of this transition involves using the IBM Image Construction and Composition Tool. This tool is the IBM tool for creating virtualized workloads that target several private cloud deployment platforms, including platforms from IBM and not from IBM. In fact, this tool is unique in its ability to support such a wide range of cloud offerings. It is also the only tool in the marketplace that can create virtual appliances for both x86 and IBM Power hardware architectures.

This book provides an in-depth look at the capabilities and internal workings of Image Construction and Composition Tool. It focuses on the capabilities of this tool, which target the virtualization and cloud offerings of IBM Systems and Technology Group. These offerings include IBM Systems Director VMControl, IBM SmartCloud Entry, and IBM PureFlex System with IBM Flex System Manager appliance. The Image Construction and Composition Tool also has a much richer set of capabilities. Specifically, it supports IBM Workload Deployer, IBM PureApplication Systems, and IBM SmartCloud Provisioning.

This publication targets software architects, cloud solutions architects, and cloud administrators. Its goal is to provide you with the expert-level skills required to package the existing and newly created applications into self-configurable, smart virtual appliances.

## **INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION**

### **BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
**[ibm.com/redbooks](http://ibm.com/redbooks)**

SG24-8042-01

ISBN 0738438480