

Novell ZENworks Application Virtualization

Novell ZENworks Application Virtualization

Novell ZENworks Application Virtualization

Table of Contents

1.	Legal Notices	1
1.1.	Disclaimer	2
1.2.	Trademarks	3
2.	Overview	4
2.1.	Overview	5
2.2.	What is a virtual application?	6
2.3.	ZENworks virtual application usage scenarios	7
2.4.	Do ZENworks virtual applications require any device drivers?	8
2.5.	How is application virtualization different from hardware virtualization?	9
2.6.	What platforms are supported?	10
2.7.	How is application virtualization different from application streaming?	11
2.8.	What applications can be virtualized using ZENworks Application Virtualization?	12
2.9.	System requirements	13
3.	Getting Started	14
3.1.	Getting Started	15
3.2.	System requirements	16
3.3.	Control panel overview	17
3.4.	Creating virtual applications	18
3.5.	Creating your first virtual application	19
3.6.	Manually configuring a simple virtual application	20
3.7.	Adding runtimes and components	21
3.8.	Loading and saving configurations	22
4.	Configuring Virtual Applications	23
4.1.	Configuring virtual applications	24
4.2.	Snapshotting applications	25-26
4.3.	Specifying a startup file	27
4.4.	Specifying multiple startup files (Jukeboxing)	28

Novell ZENworks Application Virtualization

4.5. Editing the virtual filesystem	29-30
4.6. Editing the virtual registry	31
4.7. Creating and using shared virtual components	32
5. Virtual Application Customization	33
5.1. Virtual application customization	34
5.2. Selecting a project type	35
5.3. Customizing executable metadata	36
5.4. Adding a startup image	37
5.5. Process configuration options	38-39
5.6. Configuring the sandbox location	40
5.7. Virtual services	41
5.8. Internet Explorer 6 emulation mode	42
5.9. Sandbox merge	43
6. Building MSI Setup Packages	44
6.1. Building MSI setup packages	45
6.2. Configuring package information	46-47
6.3. Creating desktop and Start Menu shortcuts	48
6.4. Creating file associations	49
7. Deploying Virtual Applications	50
7.1. Registering virtual applications in the Windows shell	51-52
7.2. Sandbox management	53
7.3. Deploying in Active Directory environments	54-55
7.4. Deploying virtual applications using MSI setup packages	56
7.5. Deploying virtual applications using Microsoft TS RemoteApp	57
7.6. Deploying virtual applications to ZENworks Configuration Management	58-59
7.7. Deploying using the Publish to USB feature	60
8. Advanced Topics	61
8.1. Customizing the ZENworks Application Virtualization interface	62
8.2. Quick snapshot mode	63
8.3. Well-known root folder variables	64-65

Novell ZENworks Application Virtualization

8.4. Building from the command line	66
8.5. Importing configurations from external tools	67
8.6. Platform Merge	68
8.7. Applying the virtual application configuration to the host device	69
8.8. XAppl file format	70-78
8.9. Application Expiration	79
9. Troubleshooting	80
9.1. Troubleshooting	81
9.2. Problems accessing Internet-based resources	82
9.3. Generating diagnostic-mode virtual applications	83

1 Legal Notices

1.1 Disclaimer

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CODE SYSTEMS CORPORATION PROVIDES THIS PRODUCT AS IS AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES AND CONDITIONS, EITHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, OF FITNESS FOR A PARTICULAR PURPOSE, OF ACCURACY OR COMPLETENESS, OF RESULTS, AND OF LACK OF NEGLIGENCE, ALL WITH REGARD TO THIS PRODUCT. THE ENTIRE RISK AS TO THE QUALITY OF OR ARISING OUT OF THE USE OF THIS PRODUCT REMAINS WITH YOU.

THIS PRODUCT MAY CONTAIN TECHNOLOGICAL DEFECTS AND OMISSIONS, TYPOGRAPHIC ERRORS, AND TECHNICAL INACCURACIES. CODE SYSTEMS CORPORATION MAY MODIFY THIS PRODUCT AT ANY TIME.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL CODE SYSTEMS CORPORATION BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, BUT NOT LIMITED TO, DAMAGES FOR LOSS OF PROFITS OR CONFIDENTIAL OR OTHER INFORMATION, FOR BUSINESS INTERRUPTION, FOR PERSONAL INJURY, FOR LOSS OF PRIVACY, FOR FAILURE TO MEET ANY DUTY INCLUDING OF GOOD FAITH OR OF REASONABLE CARE, FOR NEGLIGENCE, AND FOR ANY OTHER PECUNIARY OR OTHER LOSS WHATSOEVER) ARISING OUT OF OR IN ANY WAY RELATED TO THE USE OF OR INABILITY TO USE THIS PRODUCT, THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, OR OTHERWISE IN CONNECTION WITH ANY ASPECT OF THIS PRODUCT, EVEN IN THE EVENT OF THE FAULT, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY, BREACH OF CONTRACT OR BREACH OF WARRANTY OF CODE SYSTEMS CORPORATION, AND EVEN IF CODE SYSTEMS CORPORATION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

CODE SYSTEMS CORPORATION'S CUMULATIVE LIABILITY TO YOU OR ANY OTHER PARTY FOR ANY LOSS OF DAMAGES RESULTING FROM ANY CLAIMS, DEMANDS, OR ACTIONS ARISING OUT OF OR RELATING TO THIS PRODUCT SHALL NOT EXCEED THE LARGER OF THE LICENSE FEE PAID TO CODE SYSTEMS CORPORATION FOR THE USE OF THIS PRODUCT AND U.S. \$5.00.

1.2 Trademarks

ZENworks Application Virtualization, ZENworks Application Virtualization Fox, ZENworks Application Virtualization Postbuild, and ZENworks Application Virtualization are trademarks and/or registered trademarks of Code Systems Corporation.

ZENworks Application Virtualization is a trademark of Novell, Inc.

ThinApp is a trademark of VMware, Inc.

Microsoft, Windows, .NET, and .NET Framework are trademarks of Microsoft Corporation.

All other trademarks are the property of their respective owners.

2 Overview

2.1 Overview

Thank you for using Novell ZENworks Application Virtualization!

This product will allow you to convert your Windows, .NET, Java, Flash, Shockwave, or other Windows-compatible application into a self-contained *virtual application* which runs instantly on end-user devices. Unlike traditional deployment methods, virtual applications do not require separate setup steps for external components and runtimes, reboots, or administrative privileges, and are isolated from other system applications, preventing DLL conflicts and other deployment nightmares.

This guide explains how to use Novell ZENworks Application Virtualization to create your own virtual applications and begin enjoying the benefits of this superior next-generation deployment technology.

2.2 What is a virtual application?

A *virtual application* is a virtual machine image pre-configured with all of the files, registry data, settings, components, runtimes, and other dependencies required for a specific application to execute immediately. Virtual applications allow application publishers and IT administrators to significantly reduce the costs and complexity associated with development, setup, configuration, deployment, and maintenance of software applications.

For example, a publisher of an application based on the Microsoft .NET Framework or Java runtime engine might create a virtual application combining the application with the required runtime engine. Using this virtual application, an end-user can run the application immediately, even if the user has not installed the required runtime engine, or has an incompatible runtime engine installed. Note that this has both improved the user experience and reduced test and support complexity associated with deploying the application.

Furthermore, because virtual applications are *isolated* execution environments, it is possible to concurrently execute multiple applications which would otherwise interfere with one another. For example, applications which overwrite system DLLs or require different runtime engine versions can all run simultaneously on a single host device. As an additional advantage, virtual applications can provide access to internal virtualized copies of privileged system resources, allowing unprivileged users to directly execute many applications without security exceptions or irritating Vista UAC prompts.

Unlike other virtualization systems, ZENworks virtual application technology:

- Does not require any "player" software or separate installation: ZENworks virtual applications are executable files that run immediately on the end-user machine, just like a native executable.
- Do not incur significant processing or filesystem overhead: ZENworks application virtualization technology allows applications to run with essentially the same performance characteristics as when executed natively.
- Does not require any operating system to be installed onto the virtual application: ZENworks virtual applications provide all required virtualized OS functionality within the internal virtual environment.

2.3 ZENworks virtual application usage scenarios

Use your new ZENworks virtual application to:

- **Deploy your application in a single executable that runs immediately:** Improve the user download and startup experience by packaging all application files, registry settings, runtimes, and components into a single executable that runs immediately.
- **Run Java and .NET without separate runtime installations:** Your Java and/or .NET-based application runs immediately, just like a native executable, with no separate installation steps or runtime versioning conflicts. Execute multiple runtime versions concurrently with no conflicts.
- **Improve desktop security:** Execute your applications without granting administrative permissions to end-users. Stabilize desktop images by deploying applications in sandboxed ZENworks virtual applications.
- **Eliminate third-party setup dependencies:** Integrate third-party components, COM/VB controls, and content viewers such as Acrobat, Flash, and Shockwave, directly into your application.
- **Eliminate Vista UAC prompts and compatibility:** Deploying in a ZENworks virtual application eliminates requirements for access to privileged system resources, relieving users of annoying Vista UAC prompts.
- **Leverage Terminal Services and Citrix investments:** By isolating applications from global resource areas, ZENworks virtual application technology allows non-compliant applications to function properly in Terminal Server and Citrix environments.
- **Deploy instantly on USB drives:** Improve mobile worker productivity by placing your ZENworks virtual application onto a USB flash-memory drive. Run your application immediately on remote PCs, with no installation steps, administrative privileges or driver installations.
- **Dramatically reduce test and support costs:** At last, no more "DLL hell", "policy hell", versioning and dependency conflicts, etc. Reduce test complexity and eliminate support requests associated with dependency installation and inter-application resource conflicts.

2.4 Do ZENworks virtual applications require any device drivers?

No. Unlike some other virtualization solutions, application virtualization takes place entirely in user-mode. No device drivers are installed or required.

2.5 How is application virtualization different from hardware virtualization?

Unlike hardware virtualization systems such as Microsoft Virtual PC and VMware, the application virtualization system works at the application level and virtualizes only those operating system features required for application execution. This allows virtualized applications to operate extremely efficiently, with essentially the same performance characteristics as native executables.

Advantages of application virtualization over hardware virtualization include:

Extremely high performance: ZENworks virtual applications execute at essentially the same speed as applications running natively against the host hardware, with only a small additional memory footprint. On the other hand, applications running within hardware-virtualized environments experience significant slowdowns and impose a large memory footprint because the virtual machine includes and virtualizes an entire host operating system.

- ***Dramatically reduced application size:*** ZENworks virtual applications only require a disk footprint proportional to the size of the virtualized application, data, and included components. As a result, ZENworks virtual applications are typically small enough to be conveniently and quickly downloaded by end-users. Because hardware virtualization requires inclusion of an entire host operating system image, including many basic subsystems which will already be present on the end-user device, hardware virtualization typically requires gigabytes of storage per virtual machine.
- ***Ability to run dozens of virtualized applications:*** Because of its low-overhead characteristics, it is easily possible to run dozens of concurrent ZENworks virtual environments per processor. Conversely, due to the high overhead of hardware virtualization, it is generally possible only to run a very small number of hardware-virtualized environments per processor.
- ***Reduced licensing costs:*** Because ZENworks virtual applications do not contain a host operating system, it is not necessary to purchase separate operating system licenses to use a ZENworks virtual application. Hardware virtualization systems require a host operating system in order to function, possibly imposing additional licensing costs and restrictions.

However, hardware virtualization is appropriate in certain specialized scenarios:

- ***Non-Windows operating systems:*** ZENworks virtual applications execute only on the Windows operating system. Hardware virtualization can execute any operating system compatible with the underlying virtualized hardware, such as Linux.
- ***Kernel mode virtualization:*** The ZENworks application virtualization engine only virtualizes user-mode operating system features, whereas hardware virtualization systems emulate the entire OS stack, including kernel mode components. Applications requiring device drivers or other non-user-mode software may require a hardware-virtualized environment to function properly.

You should carefully evaluate the advantages and disadvantages of different virtualization approaches before deciding on a technology to adopt for your deployment scenario.

2.6 What platforms are supported?

ZENworks Application Virtualization supports the following platforms for virtual application build, snapshotting, and execution:

- Windows XP Professional
- Windows Embedded XP
- Windows 2000 Professional
- Windows 2000 Server
- Windows Server 2003 Standard and Enterprise editions
- Windows Vista Business, Ultimate, and Enterprise editions
- Windows Server 2008

ZENworks Application Virtualization supports these operating systems running within VMware and Microsoft hardware virtualization and hypervisor environments.

ZENworks Application Virtualization also has limited support for the Windows Preinstallation Environment (WinPE), though certain applications depending on operating system features unavailable in WinPE may not function properly.

ZENworks Application Virtualization creates 32-bit executables, which can be run under 32-bit mode on x64-based platforms.

2.7 How is application virtualization different from application streaming?

application virtualization is related to application streaming systems such as Microsoft's SoftGrid in that virtualization takes place at the application level. However, unlike application streaming systems, application virtualization:

- *Does not require any specialized streaming servers:* Streaming systems use specialized streaming servers to deliver application blocks to clients. ZENworks virtual applications are designed to stream to clients using standard SMB-based file shares, allowing them to be hosted on any Windows host, DFS share, or Linux Samba server.
- *Does not require any client installation or device drivers:* Streaming systems require that specialized client software be installed on each end-user device before clients can access hosted applications. Some streaming systems further require installation of client device drivers before use. Because the ZENworks application virtualization engine kernel is embedded and implemented entirely in user mode, ZENworks virtual applications run immediately, with no client installation or device drivers.
- *Works both offline and online:* Many streaming systems require that clients be connected to the streaming server to access applications. Because ZENworks virtual applications are standalone executables, virtual applications can be copied to laptops, USB keys, and other mobile devices for execution both on and off the network.
- *Does not require any additional server systems:* Most streaming solutions require that Active Directory or other infrastructure services be deployed as a prerequisite to application deployment. Because ZENworks virtual applications are standalone executables, no separate server systems need be purchased or deployed. Note that ZENworks virtual applications can optionally be configured to take advantage of Active Directory and SMS infrastructure, if these are present.

application virtualization can be used in concert with application streaming systems: Virtualized applications can themselves be streamed, combining the network transport optimizations of the streaming solution with the isolation, compatibility, and offline execution capabilities of application virtualization. This approach also allows enterprises with a heterogeneous IT infrastructure to enjoy the advantages of application virtualization, even in their non-Active Directory environments.

2.8 What applications can be virtualized using ZENworks Application Virtualization?

ZENworks Application Virtualization and the application virtualization engine supports most major Windows desktop applications. In addition, the ZENworks Application Virtualization Compatibility Lab routinely tests and validates popular applications for deployment using ZENworks Application Virtualization.

However, certain applications, by their nature, are unsuitable for virtualization using ZENworks Application Virtualization's user-mode virtualization technology. These include application features which contain or directly depend on interaction with specialized kernel-mode device drivers or other kernel-mode extensions; operating system components and extensions; anti-virus applications; and kernel event filtering, monitoring, and intrusion detection applications.

ZENworks Application Virtualization applications are compatible with most major anti-virus, runtime, and security packages currently available.

2.9 System requirements

The ZENworks Application Virtualization authoring environment can be run on any of the supported client platforms.

The ZENworks Application Virtualization authoring environment requires a minimum of 512MB of memory. 1GB of memory is recommended for optimal performance. More memory may be required to process unusually large virtual application projects.

A minimum screen resolution of 1024x768 is required to use the ZENworks Application Virtualization graphical user interface.

3 Getting Started

3.1 Getting Started

This section describes the system requirements for installing and running Novell ZENworks Application Virtualization, gives an overview of the ZENworks Application Virtualization user interface, and walks you through the basic steps of creating a virtual application.

3.2 System requirements

Novell ZENworks Application Virtualization requires the Windows XP, 2000 or higher operating system.

The ZENworks Application Virtualization graphical interface assumes a screen resolution of at least 800×600, although a screen resolution of at least 1024×768 is highly recommended.

3.3 Control panel overview

The ZENworks Application Virtualization control panel allows you to configure your virtual application filesystem and registry, embed external runtimes and components, take snapshots of application, and create virtual application executables. The primary interface consists of a ribbon bar and several panes grouped by functional area.

The ribbon bar provides access to common ZENworks Application Virtualization features:

- The **start menu button** located in the circle on the top left of the window, allows virtual application configurations to be opened, saved, and closed.
- The **help bar** provides access to the ZENworks Application Virtualization documentation and version information.
- The **Virtual Application** ribbon provides access to the snapshot and build features, as well as output configuration options such as the startup file, output directory, and diagnostic-mode selection.
- The **Runtimes** ribbon provides a selection of auto-configurable runtime engines which can be embedded into your application with a single click. These include .NET Framework, Java, Flash, and Shockwave runtimes.

The main panel consists of three functional groups, which are accessed by pressing an appropriate buttons along the left of the interface:

- The **Start** panel displays the latest ZENworks Application Virtualization news, including updates, available licenses, and usage suggestions.
- The **Filesystem** panel displays the application virtual filesystem, and allows adding and removing virtual files and directories.
- The **Registry** panel displays the application virtual registry, and allows adding and removing virtual registry keys and data values.
- The **Settings** panel allows configuration of virtual application metadata, startup image, and process configuration options.
- The **Components** panel allows layering of external virtual application components, such as toolbars and optional features.
- The **Setup** panel allows configuration of MSI setup package and shell integration options.
- The **Licensing** panel allows configuration of virtual application licensing modes and options.

Important: You are responsible for assuring compliance with licensing for any third-party redistributable components included using virtualization.

3.4 Creating virtual applications

ZENworks Application Virtualization offers three ways to create and configure virtualized applications. The best method in a given scenario depends on the nature of the application to be virtualized.

- **Use an application template:** ZENworks Application Virtualization includes templates for popular applications which can be built and customized using a guided, step-by-step process. This method is recommended for first-time users of ZENworks Application Virtualization. *(This method is not available in ZENworks Application Virtualization ISV Edition.)*
- **Snapshot an application installation:** Snapshotting captures system state before and after an application is installed and automatically configures virtual application settings based on observed system changes. This method is ideal for virtualizing off-the-shelf applications.
- **Manually configure an application:** This method is most often used by developers virtualizing internally developed applications. Manual configuration requires a high degree of technical knowledge but allows extremely fine-grained control over virtual application settings.

Note that all methods allow additional configuration and customization to be performed once the initial virtual application configuration has been constructed.

3.5 Creating your first virtual application

(This section does not apply to ZENworks Application Virtualization ISV Edition.)

ZENworks Application Virtualization includes automated virtual application configuration wizards for certain popular software applications. It is strongly recommended that first-time users begin by building one of these auto-configurable virtual applications using the ZENworks Application Virtualization Configuration Wizard.

To build an auto-configured application:

1. Open the **ZENworks Application Virtualization Configuration Wizard**. The wizard is displayed on program startup, or can be opened by pressing the **Configuration Wizard** button on the **Virtual Application** ribbon bar.
2. Press the button labeled **Build a virtual application from a template**.
3. Select an application to virtualize from the **Application** dropdown. Some applications may require download of additional configuration information or source application media.
4. Follow the wizard steps to construct the virtual application.

After completing the wizard, the virtual application configuration will remain loaded in the ZENworks Application Virtualization interface. This allows the configuration settings generated by the wizard to be inspected and additional customization to be performed.

3.6 Manually configuring a simple virtual application

This section provides a walkthrough of manual configuration for a simple virtual application based on the Windows Notepad application. In general, manual configuration should only be performed by experienced software developers virtualizing internally developed software applications.

1. Click on the **Filesystem** button.
2. Click on the **Application Directory** folder.
3. Click on **Add Files...**
4. Navigate to to the **System32** folder under your Windows installation directory and double-click on **notepad.exe**. This adds the Windows Notepad executable to the virtual application.
5. (*Windows Vista only*) Click on **Add Folder** and set the new folder name to **en-us**. Navigate into the **en-us** folder and click **Add Files...**. Navigate to the **System32\en-us** folder under your Windows installation directory and double-click on **notepad.exe.mui**. This file is required by the version of Notepad which shipped with Windows Vista.
6. Using Notepad or other text editor (on your machine, not the virtual application), create a file called **hello.txt** containing the text "Hello world" and save it to the **Desktop** folder.
7. Click on **Add Files...** again, navigate to your **Desktop** folder and select the **hello.txt** file that you just created. The display on the right should now show both **notepad.exe** and **hello.txt**.
8. If it is not already visible, click on the **Virtual Application** tab on the ribbon bar to display the virtual application settings.
9. In the **Startup File** dropdown, select **notepad.exe**. The *startup file* indicates which executable or file will be executed when the virtual application is started by the user.
10. Click on the **Browse...** button next to the **Output Directory** textbox. Navigate to your **Desktop** folder and press **OK**.
11. Press the **Build** button. ZENworks Application Virtualization will now display a status dialog while it builds your virtual application.

To use your new Notepad virtual application, navigate to your desktop in a shell window and double-click on **Notepad.exe**. The Notepad application starts.

But how do we know we are inside a virtual application? In the shell, delete the **hello.txt** file from the desktop. Now, inside the Notepad window, click **File / Open...**, and navigate to the **Desktop** folder. Notice that the **hello.txt** file is still present! This is because the Notepad virtual application is using the virtual filesystem, which includes the **hello.txt** file that we added in step 6. You can open and view **hello.txt** exactly as if it were a real file in the physical filesystem.

Congratulations on building your first virtual application!

3.7 Adding runtimes and components

Many components and runtime systems consist of large, complex sets of filesystem entries and registry settings. To simplify configuration of the most common components, ZENworks Application Virtualization contains a collection of pre-configured component settings which can be added to your virtual application with a single click.

To add a runtime or component, click on the **Runtimes** tab on the ribbon bar. Then, click on the appropriate runtime or component to select it for inclusion. Selected components are indicated with a highlighted button. To remove a component, click on the button again. This toggles the component inclusion state.

For example, if your application is a .NET Framework 2.0 application, then selecting the **.NET Framework 2.0** component will allow your executable to run on machines without the .NET Framework installed.

Important: Depending on the size of the component, selecting a component for inclusion can significantly increase the size of the resulting executable. Therefore, you should only select components which are required for proper execution of your application.

Important: You are responsible for assuring compliance with licensing for any third-party redistributable components included in your virtualized application.

Configuring the Java runtime

ZENworks Application Virtualization provides specialized support for the Java runtime. If your application is based on Java runtime, press the **Sun Java Runtime** button on the **Runtimes** ribbon bar. This displays the Java configuration menu.

Select the appropriate version of the Java runtime from the **Java runtime version** dropdown. If you are deploying your application as a set of .class files, then select the **Class** option from the **Startup type** dropdown; if you are deploying within a .jar file, select the **Jar** option. Enter the startup class name or Jar name in the appropriate textbox, along with any additional Java runtime options.

3.8 Loading and saving configurations

Once you have configured your virtual application, you will likely want to save the configuration for future use or modification.

To save a configuration, click on the start button menu and select **Save Configuration As....** Select a filename and location and click **Save**. This saves the virtual application configuration file. By default, configuration files use the extension *.xappl*.

Important: Configuration files do not store the *contents* of virtual filesystem files. The configuration file specifies only the *source path* for each virtual filesystem entry. The source file must exist at build time or the virtual application will not build successfully.

ZENworks Application Virtualization automatically stores source file locations as paths relative to the location of the saved *.xappl* file.

4 Configuring Virtual Applications

4.1 Configuring virtual applications

Virtualization allows application deployment to be dramatically simplified by allowing files, registry settings, components, and other application dependencies to be directly embedded into the application executable. Use of application virtualization reduces setup complexity, prevents DLL collisions, and allows applications to simulate the use of privileged disk and registry resources without requiring administrative privileges on the host machine.

This section describes snapshotting and configuration of virtual applications for use in the ZENworks virtual machine environment.

4.2 Snapshotting applications

Most commercial applications require complex combinations of filesystem and registry entries in order to function properly. In order to facilitate virtualization of these applications, ZENworks Application Virtualization can *snapshot* application installations and automatically configure itself based on modifications made to the host system during application setup.

Snapshotting

Snapshotting uses "before and after" images of the host machine to determine the virtual application configuration:

1. Prior to installing the application, a "before" snapshot is taken. This captures the state of the host device without the target application installed.
2. After installing the application, an "after" snapshot is taken. This captures all changes to the host device during application installation. ZENworks Application Virtualization then computes the changes, or *delta*, between the before and after snapshots, and inserts these changes into the configuration.

To use the snapshot feature:

1. Prepare the host device by either removing the target application and all dependencies, or copying ZENworks Application Virtualization onto a "clean" machine.
2. Click on the **Virtual Application** tab on the ribbon bar and click **Capture Before**. This captures the "before" snapshot image. Snapshotting iterates through the filesystem and registry, and therefore may take several minutes to complete.
3. (Optional) We recommend you save the "before" snapshot before continuing. This allows you to skip this step when snapshotting subsequent applications from the same clean machine image. To save the snapshot, click on the down arrow underneath the **Capture Before** button and select **Save Snapshot**. Note that, while ZENworks Application Virtualization automatically saves the last "before" snapshot that was captured, this snapshot is reset once the **Capture and Diff** process is complete.
4. Install your application along with other files or settings you wish to be included in the virtual application. If the application setup requests a reboot, be sure to save the "before" snapshot and then proceed with the reboot.
5. On the Virtual Application tab on the ribbon bar, click **Capture and Diff**. This captures the "after" snapshot, computes the delta between the two snapshots, and populates the virtual application with the delta entries.
6. (Optional) Review the filesystem and registry entries, and remove any files or settings which are not required for proper execution of your virtual application. Removing unused entries reduces virtual application size. However, accidental removal of a required resource may cause your virtual application to no longer function properly.

Saving snapshots

In many cases, the desired "before" snapshot remains fixed while many "after" snapshots are taken. ZENworks Application Virtualization allows you to save the "before" snapshot image so that the snapshot does not need to be re-captured each time. Because snapshotting may take several minutes, this significantly reduces the time required to build virtual applications in this scenario.

To save the "before" snapshot, click on the down arrow underneath the **Capture Before** button on the **Virtual Application** ribbon bar and select **Save Snapshot**. Select an appropriate filename and location and press **Save**. Similarly, to load a saved snapshot, select the **Load Snapshot** menu item and navigate to the saved snapshot file.

To clear the current "before" snapshot image, select the **Clear Snapshot** menu item.

Best practices for snapshotting

The following practices are recommended for optimal use of the snapshotting feature:

- **Perform snapshotting on a clean machine:** Snapshotting on a clean machine assures that all dependencies will be installed by the application setup. Installing on a machine with existing components may cause dependencies to be inadvertently included in the "before" snapshot and therefore excluded from the final virtual application output.
- **Save your "before" snapshot:** Saving the snapshot assures that you need only take the "before" snapshot a single time.
- **Use snapshotting in conjunction with whole-machine virtualization:** Configuring a clean machine using a whole-machine virtualization tool such as Microsoft Virtual PC and saving a "before" snapshot based on this image allows many distinct virtual applications to be snapshotted in rapid succession by reverting the whole-machine virtual state.
- **Cleanup your image:** While ZENworks Application Virtualization automatically excludes many unnecessary files and registry keys, snapshotting often picks up many unnecessary items. If you have adequate technical understanding to do so, you may significantly reduce virtual application size by manually removing unnecessary items from the snapshot delta.
- **Snapshot on the earliest operating system variant you expect to target:** Most applications can be successfully configured by snapshotting on the earliest (least common denominator) base operating system to be targetted. A small number of applications may require multi-platform snapshotting for successful deployment across all operating system variants.

4.3 Specifying a startup file

The virtual filesystem may contain a large number of executable files (such as .exe, .cmd, and .java) and viewable file formats (such as .html and .swf). However, your virtual application is consolidated into a single executable. It is therefore necessary for the virtual application designer to indicate a *startup file*, which is the executable or viewable file which is opened when the user executes the virtual application.

To select the startup file, click on the **Virtual Application** tab on the ribbon bar. Then, click on the **Startup File** dropdown list. This displays a list of all files in the virtual filesystem. Select the file to be used as the startup file. Or, navigate to the desired startup file in the virtual filesystem display, right-click the file, and select **Set as Startup File**.

Files located on the host device (outside of the virtual filesystem) may also be used as startup files. To select a file on the host device as the startup file, enter the full path to the desired startup file in the **Startup File** text box. Remember to use well-known root folder variables such as **@WINDIR@** and **@PROGRAMFILES@** as the root of the full path to ensure that the startup file can be properly located on all base operating systems.

Important: While any file can be selected as the startup file, you should only select a file which is executable or viewable. Selecting a file which cannot be opened will cause an error when the virtual application is started.

4.4 Specifying multiple startup files (Jukeboxing)

In some situations, a virtual application may want to expose multiple startup files. For example, if one is virtualizing an office productivity suite, one may want to launch either the word processor, spreadsheet, or presentation component of the suite, while still deploying a single executable file.

ZENworks Application Virtualization enables this scenario by allowing multiple entry points into the virtual application to be triggered based on a command-line argument to the virtual application executable. For example, in the office suite scenario described above, one might use the command line **office word** to trigger the word processor and **office spreadsheet** to trigger the spreadsheet.

To specify multiple startup files, click the **Multiple** button next to the **Startup File** textbox on the **Virtual Application** ribbon bar. This displays the **Startup Files** selection dialog. To add a new startup file:

1. Click on the **File** column on the first empty row in the startup file list and select the desired file from the dropdown list. Files located on the host device (outside of the virtual filesystem) may also be used as startup files. To select a file on the host device as the startup file, enter the full path to the desired startup file in the **Startup File** text box.
2. Enter the desired command line arguments, if any, in the **Command Line** column.
3. Enter the desired command line trigger in the **Trigger** column. For example, in the command line **office word**, the trigger would be **word**.
4. Check the **Auto Start** checkbox if you want the startup file always to be automatically launched on virtual application startup.

Important: When specifying a startup file located outside of the virtual filesystem, remember to use well-known root folder variables such as **@WINDIR@** and **@PROGRAMFILES@** as the root of the full path to ensure that the startup file can be properly located on all base operating systems.

Tip: The **Auto Start** flag can be specified for multiple startup files to automatically launch multiple applications that are typically used together in a single session (also known as "shotgunning").

4.5 Editing the virtual filesystem

ZENworks Application Virtualization allows you to embed a *virtual filesystem* into your executable. Embedded files are accessible by your ZAV-processed application as if they were present in the real filesystem. Unlike actual files on the host device, virtual files are not visible from and do not require changes to the host device. In particular, the use of virtual files does not require any security privileges on the host device, even if the virtual files reside in a privileged directory such as the Windows directory. Also, because virtual files are embedded in the application executable, shared DLLs embedded in the virtual filesystem will not interfere with or be overwritten by other applications on the host device.


To add virtual files, click on the **Filesystem** button located on the left side of the ZENworks Application Virtualization window. Then, using the view on the right, add the files and folders you wish to embed in the application executable. The **Application Directory** root folder represents the folder containing the virtual application binary on the executing device; the other root folders represent the corresponding folders on the host device.

Virtualization Semantics

In the event of a collision between a file in the virtual filesystem and a file physically present on the host device, the file in the virtual filesystem takes precedence.


Folders may be virtualized in **Full**, **Merge**, or **Write Copy** mode:

- If a folder is virtualized in **Full** mode, *only* files in the virtual filesystem will be visible to the application, even if a corresponding directory exists on the host device, and writes are redirected to the sandbox data area. **Full** mode is generally used when a complete level of virtual application isolation is desired.
- If a folder is virtualized in **Merge** mode, files present in a virtual folder will be merged with files in the corresponding directory on the host machine, if such a directory exists. Writes to host files are passed through to the host device and writes to virtual files are redirected into the sandbox data area. **Merge** mode is generally used when some level of interaction with the host device is desired. For example, **Merge** mode might be used to allow the virtualized application to write to the host device's **My Documents** folder.
- If a folder is virtualized in **Write Copy** mode, files present on the host device are visible to the virtual environment, but any modifications to folder contents are redirected to the sandbox data area. **Write Copy** mode is generally used when a virtual application needs to read from files already be present on the host device but isolation of the host device is still desired.

 To apply the isolation mode to all subfolders, right-click on the folder, choose **Isolation**, click on the checkbox for **Apply to subfolders**, and click **OK**.

File Attributes

Files and folders may optionally be hidden from shell browse dialogs and other applications enumerating virtual directory contents. This is often used to prevent internal components and data files from being displayed to the user. To hide a file or folder, click on the checkbox in the **Hidden** column next to the desired file or folder.


 Enabling the hidden flag only prevents a file or folder from being displayed in browse dialogs or from directory enumeration APIs. It does not prevent the application, and therefore potentially the end-user, from accessing the folder or file contents by direct binding.

Files and folders may optionally be flagged as read-only. This prevents the application from modifying the file or folder contents. To make a file or folder read-only, click on the checkbox in the **Read Only** column next to the desired file or folder.

Filesystem Compression

To reduce executable size, ZENworks Application Virtualization automatically compresses virtual filesystem contents. This typically reduces virtual application payload size by approximately 50%. Filesystem compression

can be disabled by unselecting the **Compress Payload** option in the **Process Configuration** area of the **Settings** panel.

 Disabling payload compression may significantly increase the size of the virtual application binary.

4.6 Editing the virtual registry

ZENworks Application Virtualization allows you to embed a *virtual registry* into your executable. Embedded registry keys are accessible by your ZAV-processed application as if they were present in the real registry. Unlike data present in the registry on the host device, virtual registry keys and values are not visible from and do not require changes to the host device. In particular, the use of a virtual registry does not require any security privileges on the host device, even if the virtual registry entries reside in a privileged section of the registry, such as HKEY_LOCAL_MACHINE. Also, because virtual registry entries are embedded in the application executable, other applications are unable to disrupt application execution by inadvertent modification of registry entries required by the application.

To add virtual registry data, click on the **Registry** button located on the left side of the ZENworks Application Virtualization window. Then, using the view on the right, add the registry keys and values you wish to embed in the application executable. When adding blob data, enter the values in hexadecimal format. The **Classes** root, **Current user** root, **Local machine**, and **Users** root folders correspond to the HKEY_CLASSES_ROOT, HKEY_CURRENT_USER, HKEY_LOCAL_MACHINE, and HKEY_USERS keys on the host machine.

Registry string values may include well-known root folder variables such as **@PROGRAMFILES@** and **@WINDIR@**.

Virtualization Semantics

In the event of a collision between a key or value in the virtual filesystem and data present on the host device registry, information in the virtual registry takes precedence.

Keys may be virtualized in **Full** or **Merge** mode:

- If a key is virtualized in **Merge** mode, then values present in a virtual key will be merged with values in the corresponding key on the host machine, if such a key exists. Writes to host keys are passed through to the host registry and writes to virtual keys are redirected to the user registry area.
- If a key is virtualized in **Full** mode, only values in the virtual registry will be visible to the application, even if a corresponding key exists on the host device, and writes are redirected to the user registry area.



To apply the isolation mode to all subkeys, right-click on the key, choose **Isolation**, click on the checkbox for **Apply to subkeys**, and click **OK**.

Importing Registry Hive Files

Virtual Application Studio can import registry hive (.reg) files into the virtual registry. To import a .reg file, click the **Import** button in the **Registry** panel and select the registry hive file to be imported.

4.7 Creating and using shared virtual components

In some scenarios, multiple virtual applications may share a common set of virtual machine configuration options. For example, multiple applications often share a common set of components or runtime engines; or, system administrators may want to share a common set of configuration options (browser bookmarks, application settings, etc) across a department or enterprise. ZENworks Application Virtualization makes it easy to create, share, and consume virtual machine settings across multiple ZENworks virtual applications using ZENworks Application Virtualization **XLayer**-format virtual components.

Creating and using shared virtual components

4.7.1 To create a shared virtual component:

To create a shared virtual component, configure the virtual application settings exactly as in the case of a standard virtual application (ie using snapshotting, manual configuration, etc). On the **Settings** pane, select **Component** from the **Project type** dropdown. Then, press the **Build** button.

In **Component** mode, the build process results in creation of an **XLayer** file instead of an executable file. An **XLayer** contains the virtual application settings and data payload. **XLayer** files are similar to virtual executable outputs, except that **XLayer** files do *not* contain the ZENworks virtual machine runtime engine. Therefore, an **XLayer** can only be used when combined as part of some other virtual application.

4.7.2 To use a shared virtual component:

To use an existing shared virtual component, click on the **Components** button to navigate to the project components pane. If the component does not already appear in the components table, click **Import Components...**, select the **XLayer** file you wish to load into your project, and click **OK**. The **XLayer** is then loaded into your project and the layer metadata is displayed in the **Components** list. Select the checkbox next to the desired component.

Project virtualization settings take precedence over virtualization settings in any loaded shared components.

To remove a shared virtual component from the project, select the component and click the **Remove Component** button.

5 Virtual Application Customization

5.1 Virtual application customization

This section describes advanced virtual application customization options, such as executable metadata, startup images, command-line arguments, and process startup options.

5.2 Selecting a project type

ZENworks Application Virtualization supports two project types:

- **Application:** A virtual application project produces an executable file output (.exe file) that can be run directly from the operating system. Application output mode is appropriate for most users and is the default selection.
- **Component:** A component project produces an XLayer output (.xlayer file). XLayer is a ZENworks Application Virtualization file format encoding all virtual application configuration and content in a single binary file. XLayers cannot be executed directly from the operating system. XLayers are used to exchange virtual application and component data between multiple virtual applications. For example, component output mode is used to submit components into the ZENworks Application Virtualization Component Gallery.

To set the project type, press the **Settings** button and select the appropriate option from the **Project type** dropdown.

5.3 Customizing executable metadata

Executable metadata provides external applications such as the Windows shell with information regarding the application's identity, publisher, version, preferred display icon, and description. Metadata may be viewed and edited by clicking on the **Properties** tab of the **Settings** pane.

Standard metadata

Standard metadata includes information such as the product title, publisher, description, icon, web site URL, and version. By default, ZENworks Application Virtualization will apply metadata inherited from the virtual application startup file to the output virtual application executable. However, in some instances, it may be desirable to override the metadata. To manually override executable metadata, uncheck the **Inherit properties** option and enter the desired metadata in the appropriate fields in the **Properties** area. To revert to the default inheritance behavior, recheck the **Inherit properties** option.

Custom metadata

In addition to standard Windows shell metadata, ZENworks Application Virtualization allows introduction of custom metadata into the output executable. Custom metadata may be used by specialized external executable viewer applications, inventory scanners, and other asset and licensing management systems.

To add or modify custom metadata, press the **Custom Metadata...** button. This displays the **Custom Metadata** dialog. Enter the custom metadata property names and values into the dialog. Only string-type custom metadata values are supported.

For information on programmatically reading custom executable metadata, please consult the Microsoft Windows Software Development Kit.

5.4 Adding a startup image

ZENworks Application Virtualization allows you to specify a startup "splash" image to be displayed during virtual application startup. Startup images improve application branding and are especially useful if your application requires several seconds to initialize.

Adding a startup image

To add a startup image, click on the **Startup Image** tab of the **Settings** pane, and then click the **Select...** button next to the **Image** textbox. Navigate to a **BMP**-format image to use as the startup graphic, and click **OK**.

If you wish to remove the current startup image, click the **Reset** button.

Transparency keying

Transparency keying allows the startup image to contain transparent regions. Transparencies can improve the visual effectiveness of your startup image.

To select the transparency key color, click the **Select...** button next to the **Transparency key** label. This displays the transparency key selection dialog. Select the color which represents transparent regions in the startup image and click **OK**.

Previewing the startup image

To preview the startup image, press the **Preview** button. Previewing is particularly useful to assure that the transparency key has been set properly.

5.5 Process configuration options

ZENworks Application Virtualization provides several options that control the startup of the primary and child processes.

Command line arguments

By default, command line arguments specified by the user upon virtual application execution are passed to the virtual application startup executable. However, it is possible to override this behavior and specify a fixed set of command line arguments to be passed to the startup executable. For example, one can use this option to specify Java virtual machine behavior.

To specify an explicit command-line, click on the **Settings** button and enter the command-line arguments in the **Command line** textbox. Note that these arguments *override* any arguments that might be specified by the end-user.

Working directory

The working directory setting determines the active directory at the time the virtual application process is launched.

- The **Use startup file directory** option sets the working directory to the directory of the virtual application startup file. In the case of a jukeboxed application, the working directory is set to the directory of the startup file specified on the jukebox command line.
- The **Use current directory** option sets the working directory to the directory from which the virtual application is launched.
- The **Use specified path** option allows an explicit working directory to be specified. The working directory specification can include environment and well-known root folder variables.

By default, the working directory is set to the directory of the startup file.

Application type

Windows applications may execute in either the GUI- or console-mode subsystems. If you have selected an executable startup file, ZENworks Application Virtualization will automatically configure the virtual application to execute in the same subsystem as the startup file. However, if you have selected a non-executable startup file, it may be necessary for you to manually override the application type. Most applications execute in the GUI subsystem.

To override the application type, select the appropriate mode from the **Application type** dropdown in the **Process Startup Options** section of the **Settings** panel. The **Inherit** mode sets the application type based on the type of the startup file, if possible.

Child processes

Some applications spawn new *child processes* during the course of their execution. Depending on the virtual application context, it may be preferable for such child processes to be created within the virtual application or outside of the virtual application in the host operating system.

Child processes include processes spawned to service COM local server requests.

Important: Child processes created outside of the virtual application will not have access to virtualized filesystem or registry contents. However, these processes will be able to access or modify host operating system contents, even if this would otherwise be forbidden by the virtual application configuration.

By default, child processes are created within the virtual application. To force child processes to be created outside of the virtual application, uncheck the **Spawn child process within virtualized environment** option. COM local servers can optionally be created within the virtual application context. To force COM local servers to be created outside of the virtual application, uncheck the **Spawn COM servers with virtualized environment**

option.

Exceptions to the child process virtualization behavior specified by the **Spawn child process within virtualized environment** and **Spawn COM servers within virtualized environment** flags can be enumerated in the *child process exception list*. Process names listed in the child process exception list behave *opposite* to the master child process virtualization setting. To edit the child process exception list, press the **Child Process Exception List** button. Process names may or may not include the process filename extension.

Environment variables

Some applications may depend on the presence of certain Windows environment variables in order to function properly. ZENworks Application Virtualization allows virtualization of environment variables to support such applications.

To add or modify virtual environment variables, press the **Environment Variables...** button. This displays the **Environment Variables** dialog. Enter environment variable names and values into the environment variable list.

Most virtual environment variables overwrite any environment variables defined in the host environment. However, the special **PATH** and **PATHEXT** environment variables are always merged with the corresponding host environment variables.

Environment variables are automatically captured and merged during the snapshotting delta process. Therefore, it is generally unnecessary to manually configure environment variable settings.

Automatic sandbox reset

The sandbox can optionally be configured to be reset automatically on application shutdown. For example, this allows an application publisher to assure that any changes made to an application's settings are reverted when the application closes.

To enable the automatic sandbox reset feature, select the **Delete sandbox on application shutdown** option.

Read-only virtual environments

In some scenarios, it may be desirable to prevent the user from making any modifications to the virtual environment, including the virtual filesystem and registry. To block all modifications to the virtual environment, select the **Virtual environment is read-only** option.

Startup executable optimization

The startup executable optimization option attempts to launch the startup executable within the initial virtual machine process. This prevents the creation of a separate application process but may be incompatible with some applications.

ZAV command-line arguments

ZENworks Application Virtualization supports command-line arguments of the form **/x[arg]** which modify virtual application behavior at run-time. In rare instances, these arguments may collide with command-line arguments designed for use by the virtualized application. To disable processing of these arguments, unselect the **Enable ZAV command-line arguments** option.

Window class isolation

Window class isolation prevents the virtualized application from viewing window classes that have been registered by external processes. For example, this option may be used to prevent interaction between virtualized and non-virtualized versions of the same application when the application checks for existing class registrations.

5.6 Configuring the sandbox location

Depending on the configured isolation settings, certain edit and write operations may be redirected by the application virtualization engine into an *application sandbox*, a filesystem folder where isolated modifications are persisted. Typically, the sandbox is located in a folder or network share where the user has full read and write permissions, allowing sandbox contents to be accessed and modified by the end user without any authentication or UAC prompts.

Sandbox placement considerations

Please note the following recommended practices when configuring the sandbox location:

- By default, the sandbox is placed in the `@APPDATALOCAL@\Zav\XSandbox\@TITLE@\@VERSION@\@BUILDTIME@` folder, where the `@APPDATALOCAL@` token represents the local **Application Data** folder, and `@TITLE@`, `@VERSION@`, and `@BUILDTIME@` represent the application title, version, and build time, respectively. The application title and version are configured in the **Properties** area. This location is the recommended default location for sandbox contents, as end users have full permissions to this location on standard Windows configurations. Note that distinct builds of the same virtual application use distinct sandbox locations by default; you may want to modify this behavior if persisted user settings should be preserved between virtual application updates.
- When publishing new versions of a virtual application, direct the sandbox to the *same* location as the older version if you want user settings and data to be retained in the new version. Direct the sandbox to a *different* location (typically, by rolling the subdirectory version number forward) if you want user settings and data to be reset.
- If deploying the virtual application on a USB device, place the sandbox in a subfolder of the `@APPDIR@` directory, which represents the location of the virtual application executable. This will have the effect of directing writes to the USB device. The recommended sandbox location for USB deployment is `@APPDIR@\Zav\XSandbox\@TITLE@\@VERSION@\@BUILDTIME@`.
- If deploying the virtual application on an intranet file share, place the sandbox in a user-accessible subfolder on a shared network drive. The recommended sandbox location for intranet deployment is `\\ServerName\ShareName\%USERNAME%\Zav\XSandbox\@TITLE@\@VERSION@\@BUILDTIME@`.
- Generally, you should not place the sandbox under any privileged folders, such as `@WINDIR@` or `@PROGRAMFILES@`. The virtual application may fail to execute properly if the ZENworks Application Virtualization engine is unable to write to the sandbox location at runtime.
- Environment variables may be referenced within the sandbox location by enclosing the variable between percent signs, ie `%VARIABLE%`.

Sandbox location variables

In addition to the standard root folder variables, the sandbox location can contain the following token variables. These variables are based on the values specified in the **Properties** area of the **Settings** pane.

- `@TITLE@`: Product title
- `@PUBLISHER@`: Product publisher
- `@VERSION@`: Full version string, in dotted quad format
- `@WEBSITE@`: Publisher web site
- `@BUILDTIME@`: Virtual application build time, in a format similar to **2008.02.01To8.00**.

5.7 Virtual services

Windows services are specialized applications that execute in the background and are typically responsible for providing system services such as database services, network traffic handling, web request processing, and other server functionality. Many applications install and require specific services in order to function properly.

ZENworks Application Virtualization fully supports virtualization of Windows services. To view or modify virtual service settings, press the **Virtual Services...** button. This displays the **Virtual Services** dialog.

- The **Name** field specifies the internal name of the virtual service. For example, the Windows web server would use the name **w3svc**.
- The **Friendly Name** field specifies the full display name of the service displayed to end users. For example, the Windows web server friendly name is **World Wide Web Publishing Service**.
- The **Command Line** field specifies the full command line (including the service executable name and any parameters) used to launch the service.
- The **Auto Start** flag indicates whether a virtual service automatically starts during virtual application startup, or whether the service must be launched manually or by the virtualized service control manager.
- The **Keep Alive** flag indicates whether the virtual service process is automatically terminated when the primary application executable terminates, or whether the service (and, therefore, the host virtual application executable) continues to run until the service terminates itself.

Service installation and settings are automatically captured during the snapshotting process. Therefore, it is generally unnecessary to manually configure virtual service settings. The primary exception is the case of virtualized applications intended to run as background worker services (for example, virtualized web servers); in this case, it is often required to explicitly enable the **Keep Alive** option.

5.8 Internet Explorer 6 emulation mode

Due to its integration with certain shell features, specialized virtual machine support is required to properly virtualize Microsoft's Internet Explorer 6 web browser. The application virtualization engine provides emulation of such shell functionality via a special Internet Explorer 6 emulation mode. This mode must be enabled when virtualizing the Internet Explorer 6 web browser for use on the Microsoft Windows Vista operating system and later operating systems.

To enable Internet Explorer 6 emulation mode, check the **Enable Internet Explorer 6 emulation mode** checkbox on the **Settings** pane.

Important: Due to the complexity of properly configuring Internet Explorer 6 virtualization settings, it is strongly recommended that the ZENworks Application Virtualization Configuration Wizard be used to configure Internet Explorer 6 virtual applications. The Internet Explorer 6 virtual application wizard can be started by selecting the **Build an auto-configured virtual application** option from the wizard start page and then selecting **Microsoft Internet Explorer 6** from the **Application** dropdown.

5.9 Sandbox merge

Sandbox merge allows sandbox content and settings generated during virtual application execution to be applied to a ZENworks Application Virtualization configuration. Sandbox merge is an alternative to manual registry or filesystem configuration, and is particularly useful for applying additional customizations to existing virtual application configurations or configurations generated from a virtual application template.

To merge an existing sandbox into the active configuration:

1. Click the **Sandbox Merge** button in the **Tools** section of the **Virtual Application** toolbar.
2. Enter the path of the sandbox to be merged into the current configuration.
3. Press **OK**.

For example, to customize the home page of the Firefox virtual application template:

1. Use the Configuration Wizard to create a Firefox virtual application. (The wizard allows customization of the home page, but we will later use the sandbox merge feature to override the setting specified in the wizard.)
2. Press **Build and Run** to launch the virtualized Firefox application.
3. Using the Firefox interface, specify a new browser home page.
4. Exit the Firefox virtual application.
5. Press **Sandbox Merge** to display the sandbox merge dialog. The sandbox path will be pre-populated with the location of the Firefox virtual sandbox.
6. Press **OK**.
7. The virtual application settings are updated with the configuration changes made during Firefox execution, including the updated browser home page.

6 Building MSI Setup Packages

6.1 Building MSI setup packages

ZENworks Application Virtualization includes the ability to generate Microsoft Windows Installer (MSI) setup packages to facilitate deployment of virtualized applications. In addition to deploying the virtual application executable file to the host filesystem, ZENworks Application Virtualization-generated MSI packages also allow creation of desktop and Start Menu shortcuts, creation of shell file extension associations to virtualized applications, and Control Panel uninstallers for application cleanup.

This section describes configuration and build processes for MSI setup packages.

6.2 Configuring package information

This section describes global MSI package configuration options. These options are located on the **MSI** root tree node of the **Setup** settings pane.

Setting the MSI package location

The MSI package generated by ZENworks Application Virtualization will be written to the file specified in the **Output Location** textbox. This textbox should contain the fully qualified name of the desired output file, including the output path and MSI file name.

By default, MSI packages are not automatically generated or updated when the virtual application is rebuilt. To automatically update MSI packages after the virtual application is rebuilt, enable the **Automatically generate MSI after successful application build** option.

Note that regenerating MSIs may significantly increase the time required to complete the build process. Therefore, it is recommended that this option be disabled during the virtual application development process.

It is also possible to manually force the MSI package to be regenerated. To manually build the MSI package, press the **Build MSI** button.

Important: You must build the virtual application executable before the MSI package may be generated. The **Build MSI** button will be disabled if the virtual application executable has not yet been built.

Specifying package metadata

MSI setup packages contain a package manifest describing the product's name, version (can only be numbers and periods), and manufacturer. To configure the MSI package metadata, enter the appropriate values in the **Product Info** area.

Important: The metadata published on the MSI package is distinct from the metadata published on the virtual application executable itself. To modify executable shell metadata, specify the appropriate metadata on the **Settings** pane.

Installation parameters

The **Installation parameters** area allows installation options such as install location and permissions parameters to be configured.

Applications may be installed either for the current user or for all users of the target device. To install the application for all users, enable the **Install applications for All Users** option.

Important: Installing applications for all users requires privileged access to the host device. Do not enable this option if the MSI package is designed for use by end users with standard user permissions.

The **Application Folder** specifies the location where the application executable will be installed. This usually has the form **[Publisher Name]\[Application Name]**.

In the event that a user runs the setup package on a device which already has a version of the application installed, the MSI package may be designed to update the existing application version or side-by-side install with the existing application version. To automatically update existing versions, select the **Automatically upgrade earlier application versions** option; to use side-by-side installation, select the **Allow side-by-side versions of the same application** option.

Important: Building with the **Allow side-by-side versions of the same application** option enabled causes a new setup package GUID to be generated. Once a build is completed with this option enabled, previous installations will no longer be upgraded in place, even if you revert to **Automatically upgrade earlier application versions** mode.

Extended properties

MSI setup packages may also contain extended metadata, such as keywords, product author, product description,

and publisher URL. To configure MSI package extended properties, click on the **Extended Properties** tree node in the **MSI** pane and enter the desired values.

6.3 Creating desktop and Start Menu shortcuts

Desktop and Start Menu shortcuts allow the end user to launch the application directly from the Windows shell.

To add a desktop shortcut, click on the **Desktop** node under the **Shortcuts** node in the MSI tree view. This displays the desktop shortcut list. To add a shortcut, click the **Add Shortcut** button and select the desired shortcut name, target, and options. The **Target** dropdown is populated with the startup file list, allowing shortcuts to be quickly connected to jukebox entry points.

To install additional folders and subfolders on the desktop, click the **Add Folder** button and specify the folder name.

The same procedure is used to add shortcuts and folders to the **Programs** section of the Windows Start bar, except that start bar items are configured under the **Programs Menu** of the **Shortcuts** node in the MSI tree view. Note that Start Menu items are installed either to the current user's Start Menu or **All User's** start menu depending on the **Install application for All Users** setting in the MSI installation parameters section.

6.4 Creating file associations

File associations allow the appropriate viewer or editor application for a given file type to be automatically launched when the user double-clicks on a document in the Windows shell. For example, the **.doc** file extension might automatically launch a virtualized word processing application.

To create a file association:

1. Click on the **ProgIds** node and click **Add ProgId**.
2. Enter a **ProgId** and **Description** in the **Create ProgId** dialog. File associations naming generally follows the convention **[Company Name].[Application Name].[Version]**.
3. In the new ProgId, click **Add Extension** and enter an **Extension** and an optional **MIME Type**.
4. In the new file extension, click **Add Verb** and enter a **Verb, Command**, and choose the **Inherit** behavior and **Default**. Some common verbs are open, edit, print, and view. The **Verb** that is entered will be the text that is displayed when the user right-clicks on the file. When **Inherit** is checked, the behavior of the **Verb** will be controlled by the setup information in the virtual environment. When **Inherit** is unchecked, a **Target Startup File** and **Arguments** will need to be entered manually. The **Arguments** field should contain "%1" which is the full path to the file. When **Default** is checked, the **Verb** will be automatically executed when the file is double-clicked.

File association properties may be modified or deleted by selecting the appropriate **ProgId** in the Setup tree view and modifying the settings as appropriate.

7 Deploying Virtual Applications

7.1 Registering virtual applications in the Windows shell

This section explains how to use the XReg deployment tool via the Windows shell or command-line script to register and manage virtual applications built using ZENworks Application Virtualization.

Introduction to the XReg registration tool

XReg is a tool that provides a simple command-line interface for deploying virtual applications and managing the virtual desktop environment. Users and administrators can use XReg to register virtual applications for a single user or, in the case of administrators, a group of users or devices. XReg can be used to deploy and manage virtual applications and layers built using ZENworks Application Virtualization.

After virtualizing an application with ZENworks Application Virtualization, it is often desirable to make the application Start Menu icons, shortcuts, and file associations available on the users' desktop. XReg allows you to *register* virtual applications created with ZENworks Application Virtualization in the shell, creating all of the shell associations that would generally be created during a standard install process. Unlike performing an installation, however, registration and un-registration can be performed almost instantaneously.

XReg also provides the ability to create, reset, and remove *application sandboxes*, virtual environment "bubbles" where the virtualized applications reside. Sandbox management provides fine-grained control over application linking and intercommunication.

Registering virtual applications using XReg

XReg provides a simple command-line interface for managing the virtual desktop environment. This section describes basic XReg command-line syntax, including steps for registering, updating, and unregistering virtual applications.

Command-line syntax

The following naming conventions are used in this section:

Parameter	Description
AppSpec	An AppSpec is a path (relative or fully qualified) to a virtual executable or layer built with ZENworks Application Virtualization
SandboxSpec	A SandboxSpec is the name or path of a virtual sandbox


Registering a virtual application

To register an application, use the command:

```
XReg.exe AppSpec
```

This command creates all Start Menu items, desktop shortcuts, and file associations associated with the virtual application executable.

By default, registration will create a local cached copy of the virtual application executable and use the user's local profile as the sandbox location.

 The sandbox location specified during virtual application build is ignored when registering applications using the XReg tool.

7.1.1 Advanced registration options

Command-line parameters can be used to control the caching behavior and sandbox where the virtual application should be registered:

```
XReg.exe [Options] AppSpec[@SandboxSpec]
```

Options

Parameter	Behavior
/nocache	The virtual application executable will not be copied to the client machine. All shortcuts and file associations will point to the full path as given by AppSpec .
SandboxSpec	This parameter refers to the name and path to an existing sandbox. If this parameter is specified and a sandbox with that name exists, the application will be registered into that sandbox. For more information on sandboxes, see the <i>Sandbox management</i> section in this document.

Updating a virtual application

To force an application executable to be updated, use the command:

XReg.exe /update AppSpec[@SandboxSpec]

Running the registration command without the **/update** flag will perform an update only if the cached and source versions of the application are different.

Updating registration settings

Application registration settings can be changed by re-executing the registration command with the desired options:

XReg.exe [Option] AppSpec[@SandboxSpec]

Parameter	Behavior
/nocache	Disable caching of the specified application (reverses the /cache setting)
/cache	Enable caching of the specified application (reverses the /nocache setting)

Unregistering a virtual application

Unregistering a virtual application reverses the registration process, removing the virtual application, Start Menu icons, shortcuts, and file associations.

To unregister a virtual application, use the following command:

XReg.exe /unregister AppSpec[@SandboxSpec]

It is also possible to unregister all applications with the single command:

XReg.exe /unregisterall

7.2 Sandbox management

The XReg tool allows creation and management of one or more virtual environment *sandboxes*.

A *sandbox* contains all of a virtual application's isolated data and settings as determined by the virtual application's isolation configuration settings. Applications registered to the same sandbox can view and modify each others' virtualized data and settings.

By default, all applications are registered into a single default sandbox named **Default**. In some cases, it may be desirable to group related applications into a sandbox that can be treated as a single management unit. For example, when a sandbox is reset, all of the application content and data stored in that sandbox is purged and reverted back to the default state.

Creating a sandbox

If no sandbox is specified during registration, the application will be registered to the default sandbox (**Default**). To create an additional sandbox, use one of the following commands:

```
XReg.exe [Profile] /create SandboxName [SandboxPath]
XReg.exe [Profile] /c SandboxName [SandboxPath]
```

If no path is provided, a default path is created under the **AppData** folder under the specified profile.

Resetting a sandbox

Resetting a sandbox reverts all changes made to the sandbox, including any changes to data or settings made by the user. This restores all applications registered to the sandbox to their default state. To reset a sandbox, use one of the following commands:

```
XReg.exe [Profile] /reset [SandboxSpec]
XReg.exe [Profile] /r [SandboxSpec]
```

If a *SandboxSpec* is not supplied, the default sandbox is reset.

Deleting a sandbox

Deleting a sandbox removes all applications, data, and settings from the sandbox. To delete a sandbox, use one of the following commands:

```
XReg.exe [Profile] /delete [SandboxSpec]
XReg.exe [Profile] /d [SandboxSpec]
```

If a *SandboxSpec* is not supplied, the default sandbox will be reset (the default sandbox cannot be deleted). Any applications registered to the deleted sandbox will be moved to the default sandbox.

Moving a sandbox

You can use XReg to move the sandbox location to a given path. To move a sandbox, use the following command:

```
XReg.exe [Profile] /move SandboxSpec SandboxPath
```

7.3 Deploying in Active Directory environments

This section describes how an organization using Microsoft's Active Directory can leverage that infrastructure with the XReg tool to deploy ZENworks virtual applications to their users.

Active Directory

Active Directory allows the network administrator to manage users and groups within an organization. Many organizations use Active Directory to manage their network services. By combining Active Directory with XReg, administrators can deploy virtual applications easily and reliably to one or more users in their organization.

XReg


XReg is the tool for managing the virtual desktop environment for a given user by registering and unregistering virtual applications. A typical XReg command to register an application such as Firefox would be:

```
\\VirtualAppServer\Tools\XReg.exe \\VirtualAppServer\Apps\Firefox.exe
```

This represents the basic functionality of XReg which would copy the virtual application executable, create Start Menu items and desktop shortcuts and setup file associations. For additional functionality dealing with sandboxes, caching and automatic updates, refer to *Deploying using ZENworks Application Virtualization Virtual Desktop*.

Using XReg with Active Directory

In an organization, it is generally more desirable to manage a group of users rather than one at a time. By combining Active Directory with XReg you can manage the virtual environment for a user, group, or Organizational Unit.

 In order to manage virtual applications using Active Directory, the users must have access to a shared network drive where the virtual application executable files exist. This can be specified by a full UNC path, or by using a mapped network drive.

Linking an organizational unit (OU) to a group policy object (GPO)

Active Directory offers many ways to manage network services for an organization. Here we are going to look at how we can use an OU in combination with a GPO to manage the virtual application environment for a set of users.

7.3.1 Organizational unit (OU)

In Active Directory, OUs are containers where you can place users, groups and other OUs. Using these containers the administrator can create a structure that models the hierarchical or logical structures within the organization. By setting up the OUs we can isolate the different groups of users that will receive their own set of virtual applications. Some examples would be Accounting, Sales, Marketing, etc.

7.3.2 Group policy object (GPO)

GPOs are a way of applying a set of rules and features to a targeted set of users. Typically GPOs handle security, application installation, logon/logoff scripts, Internet Explorer settings and more. A GPO generally gets applied when a user logs on to a given domain and based on their profile various GPOs will be applied.

7.3.3 XReg and the GPO

For the purposes of virtual application deployment we can configure the GPO to run the necessary XReg commands to register a given set of virtual applications. You can create and edit GPOs using the **Group Policy Object Editor**. This comes as an add on to Microsoft's Active Directory. The simplest way to deal with the virtual application management is by creating a logon script that registers the virtual applications for a particular group in the organization. You would do this under **User Configuration > Windows Settings > Scripts** in

the Group Policy Object Editor. You might add the following logon script for the accounting group:

```
\\VirtualAppServer\Tools\XReg.exe \\VirtualAppServer\AllVirtualApps\Excel.exe
\\VirtualAppServer\Tools\XReg.exe \\VirtualAppServer\AllVirtualApps\Firefox.exe
\\VirtualAppServer\Tools\XReg.exe
\\VirtualAppServer\AllVirtualApps\AcrobatReader.exe
```

Whereas you might add the following for the graphic design group:

```
\\VirtualAppServer\Tools\XReg.exe
\\VirtualAppServer\AllVirtualApps\AdobeIllustrator.exe
\\VirtualAppServer\Tools\XReg.exe \\VirtualAppServer\AllVirtualApps\Firefox.exe
\\VirtualAppServer\Tools\XReg.exe
\\VirtualAppServer\AllVirtualApps\AcrobatReader.exe
```

7.3.4 Linking the GPO to the OU

Once you have set up the OU and the GPO, you simply have to link them through the Group Policy Object Editor. After they are linked, any users that are put into that OU will have the linked GPOs applied when they logon. All of their virtual applications will appear when they logon to any machine on the domain where the GPO applies.



Remember that even though we generally talk about registering virtual applications there are a lot of other capabilities of XReg that can be applied in the GPO logon script, such as unregistering the applications or registering the applications to a specific sandbox.

7.4 Deploying virtual applications using MSI setup packages

ZENworks Application Virtualization allows virtual applications and components to be deployed using legacy MSI setup package technology.

Create MSI setup packages directly within ZENworks Application Virtualization

Novell ZENworks Application Virtualization can be used to create standalone MSI packages directly within the ZENworks Application Virtualization environment. Generated MSI setup packages can include Start Menu items, desktop shortcuts, file associations, and other custom shell integration behaviors.

Deployment using generated MSI packages is appropriate in situations where existing MSI package deployment mechanisms are in place or for deploying applications with shell integration without the XReg Virtual Desktop client tool.

Virtual application and component shell integration settings are shared between MSI- and Virtual Desktop-based deployment, enabling easy migration between deployment models.

Deploy virtual applications into legacy MSI setup packages

ZENworks Application Virtualization also supports deployment of virtual application executables into legacy MSI setup packages.

For additional information on MSI-based deployment, please consult the ZENworks Application Virtualization User Guide.

Import legacy MSI setup packages

Novell ZENworks Application Virtualization also supports one-click import of legacy MSI setup packages into the ZENworks Application Virtualization environment. Following import, the application can be customized and deployed as a ZENworks Application Virtualization-based virtual application or XLayer.

7.5 Deploying virtual applications using Microsoft TS RemoteApp

This section describes how to deploy ZENworks virtual applications using Microsoft Windows 2008 Terminal Services RemoteApp server.

Terminal Services RemoteApp

Terminal Services RemoteApp is a server-side program that provides end-users remote access to applications on a terminal server. Applications configured with TS RemoteApp appear as though they are running locally on the user's machine. End-user's can run RemoteApps side-by-side with local programs or other RemoteApps.

Follow the steps below to utilize virtual applications created with ZENworks Application Virtualization with Microsoft TS RemoteApp:

1. On the TS RemoteApp server, open the **TS RemoteApp Manager** and choose **Add RemoteApp Programs** by right-clicking inside the **RemoteApp Programs** list or through the **Action** drop down menu.
2. Click **Next** in the RemoteApp Wizard.
3. Click **Browse** and select the virtual application executable.
4. After the virtual application is added to the list, select it and click **Properties**.
5. If the virtual application has multiple **Startup Files**, configure the **RemoteApp Program Name** and **Alias**. If this is not done the TS RemoteApp server will not distinguish between the separate applications in the suite.
6. Still in the **Properties** window, select **Always use the following command-line argument**, enter in the **Trigger** for the **Startup File** that is to be executed, and click **OK**.
7. In the **RemoteApp Programs** list, right-click the program that you added and choose **Create .rdp File**.
8. There are no special requirements for the .rdp files.
9. If there are multiple **Startup Files**, repeat these steps for the other applications in the suite and deploy the shortcuts on the host systems.

7.6 Deploying virtual applications to ZENworks Configuration Management

This section describes ZENworks Application Virtualization Startup and Bundle Publishing features. All options described in this section can be found in the **ZENworks** panel.


ZENworks Startup


ZENworks Application Virtualization can be configured to require the ZENworks Configuration Management Agent to be installed on the host device executing the virtual application. By default, virtual applications built with ZENworks Application Virtualization do not require the ZENworks Configuration Management Agent.

To require the ZENworks Configuration Management Agent to be installed on the host system, select **Require ZENworks Configuration Management Agent to be installed on workstation executing the virtual application** from the **ZENworks** panel.

When virtual applications built with ZENworks Application Virtualization require the ZENworks Configuration Management Agent to be installed, they can also be limited to only allow execution in a specific ZENworks Configuration Management Zone.

To only allow execution in a specific ZENworks Configuration Management Zone, select **Only allow devices registered in specific zone to execute the application** from the **ZENworks** panel. When prompted, enter the **ZENworks Server Address**, click **Connect**, enter the appropriate **Username** and **Password**, and click **OK**.

 When specifying the **ZENworks Server Address**, it may be necessary to use the secure HTTP prefix (https://) to connect to a secure ZENworks Configuration Management Server.

 When specifying the **ZENworks Server Address**, it may be necessary to use a port number suffix (:81) if the ZENworks Configuration Management Server has been configured to use a custom port number.

ZENworks Bundle Publishing

ZENworks Application Virtualization can publish virtual applications directly to a ZENworks Configuration Management Zone Bundle.

To publish a virtual application directly to a ZENworks Configuration Management Zone Bundle,

1. Select **MSI** or **Executable** for the **Project Type**. The MSI or Executable will need to be built before publication.
2. Click **Select Zone** under the **ZENworks Bundle Publishing** pane.
3. When prompted, enter the **ZENworks Server Address** and click **Connect**.
4. Log in to the ZENworks Server with the appropriate **Username** and **Password** and click **OK**.
5. Assign a **Bundle Name**.
6. Assign a **Bundle Folder**. The **Bundle Folder** is the location in the bundle hierarchy where the bundle will be published.
7. Assign a **Destination Path**. The **Destination Path** is the location on the host system where the bundle will be installed and must be resolvable by the host system. This option will only be available if the **Project Type** is **Executable**.
8. Click **Publish Now** to publish the bundle to the ZENworks Configuration Management Zone.

Additional options:

- To change the credentials used to publish the bundle, click the **Change Credentials** button and enter the new credentials.

- To store the credentials in the .xappl configuration file, select **Store Zone credentials in .xappl file**. Using this option eliminates the credentials prompt when publishing the bundle. This is a potential security risk as it will store the ZENworks credentials in the .xappl configuration file as plain text.
- To automatically publish the virtual application to a ZENworks bundle after a successful build, select **Automatically publish application as ZENworks bundle after successful build**.

7.7 Deploying using the Publish to USB feature

This section describes how to deploy virtual applications to USB storage devices using the **Publish to USB** feature.

Publishing virtual applications to USB storage devices

The **Publish to USB** feature publishes virtual applications to USB storage devices. When the USB storage device is attached to a host system, the virtual application automatically registers the **Setup** information to the host shell environment. This information is automatically unregistered when the USB device is removed from the host system.


Follow the steps below to deploy virtual applications on USB devices:

1. Open an existing virtual application configuration.
2. Attach a USB storage device to the host system.
3. Click **Publish to USB**, select the USB storage device, and click **Publish**.
4. After the virtual application is published to the USB storage device, click **OK**.

Using virtual applications from USB storage devices

Follow the steps below to virtual applications that are published to USB storage devices:

1. Attach the USB storage device to the host system.
2. (If prompted by **AutoPlay**, choose the **XUsb.exe** option. XUsb will then register the file associations and shortcuts associated with the virtual application.)
3. Remove the USB storage device to unregister the virtual applications from the host system.

 If AutoPlay is disabled on the host system, open the USB storage device's contents and manually run **XUsb.exe**.

8 Advanced Topics

8.1 Customizing the ZENworks Application Virtualization interface

This section describes ZENworks Application Virtualization interface customization options. All options described in this section can be found under the **Options** menu item.

Proxy settings...

ZENworks Application Virtualization uses the Internet to check for product updates and download update packages. If your computer is located behind a firewall, it may be necessary for Internet access to take place through a proxy server. By default, ZENworks Application Virtualization will use the default Internet settings configured on the machine. In some circumstances, however, it may be necessary for you to manually configure the proxy server settings.

To manually configure proxy settings, select the **Proxy settings...** option from the **Options** menu. Please contact your network administrator if you need assistance configuring the proxy settings.

Automatically detect associated runtimes and components

By default, ZENworks Application Virtualization will scan the virtual filesystem at build time and verify that the current configuration includes all available runtimes and components associated with file types contained in the virtual filesystem. This behavior is recommended to assure maximum virtual application reliability.

If you wish to disable this check, unselect the **Automatically detect associated runtimes and components** option in the **Options** menu.

Play sound on build completion

By default, ZENworks Application Virtualization plays a short sound to notify the user of virtual application build completion.

If you wish to disable this sound, unselect the **Play sound on build completion** option in the **Options** menu.

8.2 Quick snapshot mode

By default, ZENworks Application Virtualization uses a "quick" snapshotting algorithm that attempts to minimize the amount of time spent scanning the host system device state during snapshotting. In very rare cases, use of this mode may result in an improperly configured virtual application. Use of quick snapshot mode may also slightly increase the size of the virtual application configuration contents. It is strongly recommended that snapshotting be performed using the quick snapshot mode, as this is compatible with the vast majority of applications. Disabling quick snapshot mode significantly increases the amount of time required to complete the virtual application configuration process.

To disable quick snapshot mode, unselect the **Quick snapshot mode** item from the **Options** menu.

Note that "before" and "after" snapshots must be taken using the same snapshotting algorithm. Loading a saved snapshot image causes ZENworks Application Virtualization to automatically configure the snapshotting mode to be consistent with the algorithm used during the saved snapshot capture.

8.3 Well-known root folder variables

The ZENworks Application Virtualization engine dynamically remaps well-known root folders such as **My Documents** and **Program Files** to the appropriate location based on the host operating system at runtime. This assures, for example, that the virtualized **My Documents** folder will be mapped to `\User\Bob\Documents` when running on Windows Vista or `\Documents and Settings\Bob\My Documents` when running on Windows 2000.

Most of the time, configurations are constructed using snapshotting or in the graphical user interface. However, if manually modifying the .xappl file, the following well-known root folder variables may be used to configure virtual filesystem locations. Root folder variables are case sensitive.

- **@APPDIR@**: Folder where the executing virtual application executable resides
- **@WINDIR@**: The operating system install location root
- **@SYSDRIVE@**: The root folder of the drive containing the operating system installation
- **@PROGRAMFILES@**: The **Program Files** folder
- **@PROGRAMFILESCOMMON@**: The **Program Files\Common Files** folder
- **@SYSTEM@**: The Windows **System32** folder
- **@APPDATALOCAL@**: The folder that serves as a common repository for application-specific data that is used by the current, non-roaming user
- **@APPDATA@**: The folder that serves as a common repository for application-specific data for the current roaming user
- **@STARTUP@**: The folder containing the current user's startup items
- **@PROGRAMS@**: The folder that contains the user's program groups
- **@STARTMENU@**: The folder containing the user's **Start Menu** contents
- **@DESKTOP@**: The current user's **Desktop** folder
- **@TEMPLATES@**: The folder that serves as a common repository for the current user's document templates
- **@FAVORITES@**: The current user's **Favorites** folder
- **@DOCUMENTS@**: The current user's **My Documents** folder
- **@MUSIC@**: The current user's **My Music** folder
- **@PICTURES@**: The current user's **My Pictures** folder
- **@PROFILE@**: The folder that stores the current user's profile data
- **@APPDATACOMMON@**: The folder that serves as a common repository for application-specific data that is used by all users
- **@STARTUPCOMMON@**: The folder containing startup items for **All Users**
- **@PROGRAMSCOMMON@**: The folder for components that are shared across applications
- **@STARTMENUMCOMMON@**: The folder containing the **Start Menu** contents for **All Users**
- **@DESKTOPCOMMON@**: The shared **Desktop** folder
- **@TEMPLATESCOMMON@**: The folder that serves as a common repository for shared document templates
- **@FAVORITESCOMMON@**: The shared **Favorites** folder
- **@DOCUMENTSCOMMON@**: The shared **Documents** folder
- **@MUSICCOMMON@**: The shared **Music** folder

- **@PICTURESCOMMON@**: The shared **Pictures** folder
- **@PROFILECOMMON@**: The folder that stores the shared profile data


8.4 Building from the command line

ZENworks Application Virtualization can optionally be executed from the command line. This is particularly useful for building virtual applications as part of an automated build process.

The command line version of ZENworks Application Virtualization is called **XStudio.exe** and can be found in the ZENworks Application Virtualization installation directory. To build a virtual application from the command line, execute **XStudio Configuration.xappl** from the command prompt, where **Configuration.xappl** is the name of the ZENworks Application Virtualization project created using the graphical interface.

Options specified in the configuration file may be overridden with the command-line flags given in the following table.

Option	Description
<code>/before [/beforepath <i>Snapshot Path</i>]</code>	Performs a before snapshot and saves the snapshot to the optionally specified snapshot folder. If no snapshot folder is specified, the default snapshot folder is used.
<code>/after [/beforepath <i>Snapshot Path</i>] [/o <i>Output Path</i>]</code>	Performs an after snapshot using the optionally specified before snapshot path. If no before snapshot path is specified, the default snapshot folder is used. The output configuration and snapshot files are saved to the optionally specified output path. If no output path is specified, the output configuration and snapshot files are saved to the current user's desktop.
<code>/component or /layer</code>	Forces the output type to component. This mode causes a .xlayer format file to be generated.
<code>/d</code>	Forces a diagnostic-mode output to be generated.
<code>/l <i>License file</i></code>	Applies the specified license file.
<code>/deletesandbox</code>	Forces the virtual application to delete the sandbox on application shutdown.

 Snapshots generated from the command-line using the `/after` flag will not have an output path and specified. When using programmatic snapshotting, it is strongly recommended that additional scripting be performed to apply necessary additional configuration to the generated `xappl` file.

8.5 Importing configurations from external tools

ZENworks Application Virtualization allows configurations from certain external application virtualization tools to be automatically converted into ZENworks Application Virtualization configurations.

To import a configuration from an external tool, select the application control menu (click the program icon in the top left of the main application window or press **Alt-F**) and select **Import Configuration**. This displays the configuration import wizard. Follow the step-by-step wizard instructions to convert your existing virtual application configuration into a Novell ZENworks Application Virtualization configuration.

ZENworks Application Virtualization currently supports import of configurations from Thinstall Virtualization Suite.

8.6 Platform Merge

The **Merge Platforms** feature allows virtual application configurations snapshotted on multiple operating system variants (Windows XP, Vista, etc) to be combined into a single configuration. At runtime, the virtualization engine dynamically applies the configuration options appropriate for the operating system variant used for execution.



The most common platform merge scenario is a merge of snapshots taken on Windows XP and Windows Vista. This is because some newer applications use operating system features specific to Windows Vista.

To merge configurations from multiple platforms:

1. From the **Advanced** tab, click the **Merge Platforms** button.
2. Click **Browse** and open the appropriate configuration for each applicable operating system variant.
3. For operating systems without a configuration, choose which configuration it should use by using the **Inherit** option.
4. When all configurations have been selected or set to **Inherit**, click **Browse** in the **Merge Settings** area, choose where to save the merged configuration, and click **Merge**.

To display or edit a specific operating system from a merged configuration:

1. Open the merged configuration.
2. From the **Advanced** tab, click on the **Display** drop down menu.
3. Select the operating system that you want to display or edit. The filesystem and registry panels will only display settings specific to the selected operating system. Note that you cannot edit configurations which are inherited from other platforms; to edit inherited configurations, you must select and edit the master configuration.

To change the inheritance of a operating system in a merged configuration:


1. Open a merged configuration.
2. From the **Advanced** tab, click the **Display** drop down menu.
3. Select the operating system that you want to modify.
4. Select the platform from which to inherit using the **Inherits** drop down menu.

8.7 Applying the virtual application configuration to the host device

ZENworks Application Virtualization allows the virtual application configuration to be applied to the host system. Applying the virtual application configuration to the host system is helpful when creating XLayer updates for virtual applications.

To apply the virtual application configuration to the host system:

1. Click the **Apply Configuration** button in the start menu of the **Virtual Application Studio** application.
2. Enter the path of the sandbox to be merged into the current configuration.
3. Click **Yes** to acknowledge that the Apply Configuration process cannot be undone.

 The Apply Configuration feature is not intended for use as an installation process for virtual applications.

For example, to create an XLayer update of the Firefox virtual application template:

1. Use the Configuration Wizard to create a Firefox virtual application.
2. With the Firefox configuration loaded, run the Apply Configuration process as above.
3. Open a new virtual application configuration.
4. Capture a before snapshot.
5. Open Firefox, select Help>Check for Updates, and apply any updates.
6. Capture an after snapshot.
7. Build the captured updates as an XLayer.
8. Execute the built XLayer on top of the original virtual Firefox browser and notice that the updates have been applied.

8.8 XAppl file format

Overview

A .xappl file is an XML representation of all the virtual application configuration settings.

All paths in the .xappl file are relative to where the .xappl file resides. For example, the **source** attribute of a File element will begin with ".\Files\". The "." directory is the path where the .xappl file should reside in order for ZENworks Application Virtualization to locate the physical source files during the build process.

The .xappl file must adhere to all XML syntax rules. If there are syntax errors in the .xappl file, ZENworks Application Virtualization will not load the file.

In this document, attribute values are shown in parenthesis after their description and default values are shown in **bold**.

XAppl Configuration Elements and Attributes

OutputLocation

The **value** attribute is the path to the folder where the virtual application executable will be created. This can be a local path, a UNC path, or a mapped drive.

OutputFile

The **value** attribute is the file name of the virtual application executable.

Project Type

The **value** attribute denotes whether this configuration is for a virtual application (Application) or an xlayer component (Component).

Licensing

The **licensing** attribute contains information about the license that was used to build the virtual application.

Output

The **diagnosticMode** attribute denotes when the application output should log diagnostic information (True) or not (False). If true, the virtual application will create diagnostic logs in the directory where it was executed from.

The **sourcePackage** attribute is not used.

MSI

All sub-elements contain settings pertaining to the configuration of the MSI setup file.

The **outputMsiPath** attribute indicates the location where the setup MSI will be built.

The **title** attribute indicates the value of the MSI title property.

The **subject** attribute indicates the value of the MSI subject property.

The **keywords** attribute indicates the value of the MSI keywords property.

The **productName** attribute indicates the value of the MSI product name property.

The **productVersion** attribute indicates the value of the MSI product version property.

The **manufacturer** attribute indicates the value of the MSI manufacturer property.

The **productLanguage** attribute indicates the value of the MSI product language property.

The **author** attribute indicates the value of the MSI author property.

The **description** attribute indicates the value of the MSI description property.

The **manufacturerUrl** attribute indicates the value of the MSI manufacturer URL property.

The **autoBuild** attribute denotes whether the MSI should build when the virtual application build completes successfully (True) or not (False).

The **isolatePerUser** attribute denotes whether the MSI setup should be installed on a per-user basis (True) or installed for all users (False). When installing per-user, the install root path is Application Data. When installing for all users, the install root path is Program Files.

The **applicationFolder** attribute indicates the subfolders into which the virtual application should be installed (Company Name\Product Name).

The **upgradePreviousVersion** attribute denotes whether the setup should maintain the same Upgrade code when it builds (True) or change the Upgrade code for each build (False). This allows the setup to upgrade previous versions when it is installed, or to exist side by side.

The **productCode** attribute indicates the value of MSI product code property.

The **upgradeCode** attribute indicates the value of MSI upgrade code property.

The **componentId** attribute indicates the value of the MSI component id property.

Packages

8.8.1 Clr

The .NET *Clr* runtime element and all sub-elements contain settings pertaining to the configuration of the virtual .NET Framework runtime.

8.8.2 DirectX

The DirectX element and all sub-elements contain settings pertaining to the configuration of the virtual DirectX runtime.

8.8.3 Java

All sub-elements contain settings pertaining to the configuration of the virtual java runtime.

RunTime

The **name** attribute indicates the name of the java runtime (Java).

The **platform** attribute indicates the platform that the java runtime is designed for (x86).

The **version** attribute indicates the version of the java runtime. The available versions are Java 5 (1.5.0.140) and Java 6 (1.6.0.30).

Settings

The **startupType** attribute denotes whether to use the jar file (JAR) or class path (Class) command line parameters for java.exe to launch the application.

The **startup** attribute indicates the jar file path or class name depending on the StartupType.

The **classpath** attribute indicates the path to the class files of the Java runtime.

The **options** attribute denotes any additional command line parameter.

Package

The **name** attribute indicates the name of the component or runtime.

The **platform** attribute indicates the platforms that the component or runtime is supported on. The following are the only available values:

- Any platform (Any)
- x86 platform (x86)

The **version** attribute indicates the version of the component or runtime.

VirtualizationSettings

All sub-elements contain settings pertaining to the configuration of the virtual operating system.

The **suppressBranding** attribute controls the branding pop-up that is displayed (**False**), or not displayed (**True**) in the lower right-hand corner during application startup.

The **isolateWindowClasses** attribute is used to isolate windows classes, as registered via the Windows `::RegisterClass` or `::RegisterClassEx` APIs. For example, this allows a virtualized Firefox instance to run while a non-virtualized instance is running.

The **readOnlyVirtualization** attribute denotes whether the virtual application has the ability to modify virtual files and registry settings (**False**) or not (**True**). Setting this attribute to **True** will prevent modification to the virtual filesystem and virtual registry.

The **disableZENworksApplicationVirtualizationCommandLine** attribute controls the ability to execute (**False**) any file from within the virtual filesystem.

The **subsystem** attribute indicates the application output type. It can be inherited from the startup file (**Inherit**) or set explicitly to be a Windows application (**GUI**) or console application (**Console**). If **Inherit** is set, but the startup file is either not in the virtual filesystem or not an executable, then the output will be a Windows application.

The **ie6Emulation** attribute denotes a special mode required for the Internet Explorer 6 template (**True**). For all other apps, this should be disabled (**False**).

The **sandboxLocation** attribute indicates the base path of the application sandbox (`@APPDATALOCAL@Zav\Sandbox\@TITLE@\@VERSION@\@BUILDTIME@`).

The **workingDirectory** attribute defines what directory the application will run in.

The **compressPayload** attribute controls whether the output executable will be compressed (**True**) or not (**False**).

ChildProcessVirtualization

The **spawnExternalComServers** attribute is controls whether the virtual application launches ComServers in the virtual environment (**False**) or the external environment (**True**).

The **spawnVm** attribute denotes whether the spawned external applications are spawned inside the virtual environment (**True**) or outside the virtual environment (**False**).

ChildProcessException

The **name** attribute indicates the name of the executable file (extension included) to except from the effects of the **spawnVm** attribute.

CustomMetadata

All sub-elements contain settings pertaining to the configuration of the individual custom metadata items.

8.8.4 CustomMetadataItem

The **property** attribute indicates the name of the custom metadata item.

The **value** attribute indicates the value of the custom metadata item.

StandardMetadata

All sub-elements contain settings pertaining to the configuration of the individual standard metadata items.

StandardMetadataItem

The **property** attribute indicates the name of the standard metadata item. The following are the available standard metadata:

- Product Title(Title)
- Publisher (Publisher)
- Description (Description)
- Website (Website)
- Product Version (Version)

SplashImage

The **path** attribute indicates the source path to the splash image displayed at application startup.

The **transparency** attribute indicates the color in the splash image that should be made transparent when the image is displayed (**Magenta**).

StartupFiles

All sub-elements contain configuration pertaining to the individual startup files.

8.8.5 StartupFile

The **node** attribute indicates the path of the startup file.

The **tag** attribute indicates the command line trigger used to specify this entry as the startup to use.

The **commandLine** attribute indicates the command line arguments to pass to the startup file.

The **default** attribute denotes whether this entry is executed automatically when no tag is specified (True) or not (False).

StartupShim

The startup shim is a virtualized binary that is invoked prior to the startup file. Startup shims are used to perform customized licensing checks or other initialization tasks.

The **useShim** attribute indicates whether to use the startup shim.

The **shimDllPath** attribute indicates the path to the virtual shim DLL implementation.

The **paramOnInitialize** attribute indicates a string to be passed to the shim **OnInitialize** function.

The startup shim signature is **typedef BOOL (__stdcall *FnOnInitialize) LPCWSTR pwcsInitializationToken**). The return value indicates whether virtual machine execution should proceed.

Layers

All sub-elements are individual virtual layers.

8.8.6 Layer

The **Layer** element and all sub-elements contain settings pertaining to the configuration of this layer of the virtual operating system.

The **name** attribute indicates the name of the layer. The default layer (Default) is the only layer for whom the name matters. All other layer names are purely informational.

Condition

The **variable** attribute indicates the host system setting that will be evaluated. The operating system version (OS) is the only available option.

The **operator** attribute indicates the Boolean operation that will be used to evaluate the host system. The available Boolean operations are:

- greater than or equal to (GREATEREQUAL)
- greater than (GREATER)
- equal to (EQUAL)
- not equal to (NOTEQUAL)
- less than (LESS)
- less than or equal to (LESSEQUAL)

The **value** attribute indicates the value against which the host system will be evaluated, using the Boolean operation. The available values in ascending order are:

- Windows 2000 (Win2k)
- Windows XP (WinXP)
- Windows 2003 (Win2k3)
- Windows Vista (Vista)

Filesystem

All sub-elements contain settings pertaining to the configuration of the virtual filesystem.

Directory

All sub-elements contain settings pertaining to the configuration of this directory of the virtual filesystem.

The **rootType** attribute indicates the root system folder that this virtual folder is mapped to on the host filesystem. Directory elements with the **rootType** attribute are always directly beneath the **Filesystem** element. The following are the available **rootType** values:

- Application Directory (Application)
- Windows\System32 (System)
- Windows (OS)
- System Drive Root Directory (SysDrive)
- Program Files\Common (AllProgramsCommon)
- Program Files (AllPrograms)
- Current User - Start Menu (StartMenu)
- Current User - Start Menu\Programs (Programs)
- Current User - Start Menu\Programs\Startup (Startup)
- Current User - Application Data (AppData)
- Current User - LocalSetting\Application Data (AppDataLocal)
- Current User - Desktop (Desktop)

- Current User - Templates (Templates)
- Current User - Favorites (Favorites)
- Current User - Music (Music)
- Current User - Pictures (Pictures)
- Current User - My Documents (Documents)
- %PROFILE% (Profile)
- All Users - Start Menu (StartMenuCommon)
- All Users - Start Menu\Programs (ProgramsCommon)
- All Users - Start Menu\Programs\Startup (StartupCommon)
- All Users - Application Data (AppDataCommon)
- All Users - Desktop (DesktopCommon)
- All Users - Templates (TemplatesCommon)
- All Users - Favorites (FavoritesCommon)
- All Users - Music (MusicCommon)
- All Users - Pictures (PicturesCommon)
- All Users - My Documents (DocumentsCommon)
- %ALLUSERSPROFILE% (ProfileCommon)

The **isolation** attribute indicates the isolation setting of the virtual folder. The available values are:

- Full isolation (Full)
- WriteCopy isolation (WriteCopy)
- Merge isolation (Merge)

The **name** attribute indicates the name of the virtual directory.

The **hide** attribute denotes whether the directory is marked as hidden (True) or visible (**False**).

File

The **name** attribute indicates the name of the file.

The **hide** attribute denotes whether the file is marked as hidden (True) or visible (False).

The **source** attribute indicates the source path to the file.

Registry

All sub-elements contain settings pertaining to the configuration of the virtual registry.

Key

All sub-elements contain settings pertaining to the configuration of this key of the virtual filesystem.

The **rootType** attribute indicates the root system folder that this virtual folder is mapped to on the host filesystem. Key elements with the rootType attribute are always directly beneath the Registry element. The following are the available **rootType** values:

- HKEY_CLASSES (ClassesRoot)
- HKEY_CURRENT_USER (CurrentUser)
- HKEY_LOCAL_MACHINE (CurrentUser)

- HKEY_USERS (Users)

The **name** attribute indicates the name of the key.

The **namePathInformationTuples** indicates that there is a path in the name or value of the registry item. There are 3 comma delimited integers for each path found in the name/value.

1. Flags that indicate the state of the path (valid combinations: 0x0, 0x1, 0x2, 0x4 0x5 0x6)

- 0x1 – All Uppercase
- 0x2 – All Lowercase
- 0x4 – Uses Short Path Names

2. Start index of the path

3. Length of the path

The **isolation** attribute indicates the isolation setting of the virtual folder. The available values are:

- Full isolation (Full)
- Merge isolation (Merge)

Value

The **name** attribute indicates the name of the value.

The **type** attribute indicates the type of the value. The available values are:

- REG_SZ and REG_EXPAND_SZ (String)
- REG_DWORD (DWORD)
- REG_QWORD (QWORD)
- REG_BINARY (Binary)
- REG_MULTI_STRING (StringArray)

The **namePathInformationTuples** indicates that there is a path in the name or value of the registry item. There are 3 comma delimited integers for each path found in the name/value.

1. Flags that indicate the state of the path (valid combinations: 0x0, 0x1, 0x2, 0x4 0x5 0x6)

- 0x1 – All Uppercase
- 0x2 – All Lowercase
- 0x4 – Uses Short Path Names

2. Start index of the path

3. Length of the path

The **value** attribute indicates the value of the value. This is true for all types, except StringArray, which contains the String sub-element.

Environment Variables

The **name** attribute indicates the name of the environment variable.

The **value** attribute indicates the value of the environment variable.

Services

The **name** attribute indicates the name of the windows service.

The **autoStart** attribute denotes whether the windows service starts when the virtual application starts (**True**) or

not (False).

The **commandLine** attribute indicates the startup command line of the windows service.

The **friendlyName** attribute indicates the friendly name of the windows service.

The **description** attribute indicates the description of the windows service.

The **objectName** attribute indicates the account under which the windows service ran when not virtualized.

The **keepAlive** attribute denotes whether the windows service should continue running after the startup application has closed (**True**) or not (False).

The **start** attribute indicates the value of the Start DWORD value in the Windows Services registry key.

The **type** attribute indicates the value of the Type DWORD value in the Windows Services registry key.

The **errorControl** attribute indicates the value of the ErrorControl DWORD value in the Services registry key.

8.8.7 Shortcuts

All sub-elements contain settings pertaining to the configuration of the MSI shortcuts.

Folder

All sub-elements contain settings pertaining to the configuration of the MSI shortcuts in this folder.

The **name** attribute indicates the name of the folder. The two top level folders represent the Desktop (Desktop) and the Programs menu on the Start menu (Programs Menu).

Shortcut

The **name** attribute indicates the name of the shortcut.

The **targetPath** attribute indicates the path of the StartupFile that is the target of the shortcut.

The **targetParameter** attribute indicates the Trigger or Tag of the StartupFile that is the target of the shortcut.

The **arguments** attribute indicates the arguments passed to the target of the shortcut at runtime.

The **showCmd** attribute denotes whether the application should start in a maximized(3), minimized(7) or regular (1) window state.

The **description** attribute indicates the description of the shortcut.

IconResource

The **IconResource** sub-element contains an identifier of the icon that is used for the Shortcut.

8.8.8 ProgIds

All sub-elements contain settings pertaining to the configuration of the ProgId.

The **name** attribute indicates the name of the ProgId.

The **description** attribute indicates the description of the ProgId.

IconResource

The **IconResource** sub-element contains an identifier of the icon that is used for the file association.

Extension

All sub-elements contain settings pertaining to the configuration of the file extensions for the ProgId.

The **extension** attribute indicates the file extension that is associated with the ProgId.

The **contentType** attribute indicates the mime type of all files with the extension.

Verb

All sub-elements contain settings pertaining to the configuration of the Verb for the file extension.

The **title** attribute indicates the title of the verb.

The **verb** attribute indicates the verb value.

The **arguments** attribute indicates the arguments passed to the target of the verb at runtime.

The **default** attribute denotes whether this verb is the default verb (**True**) or not (**False**).

8.9 Application Expiration

This section describes the **Expiration** feature. With the **Expiration** feature, virtual applications can be set to expire after a certain number of days or after a certain date.


To set a virtual application to expire after a specific number of days:

1. Open on the **Expiration** panel.
2. Check the **Disallow execution after number of days** checkbox.
3. Select the number of days after the application is first executed on a system before it will expire.
4. Choose the **Time Source** the virtual application will use to validate the date.

To set a virtual application to expire after a certain date:

1. Open on the **Expiration** panel.
2. Check the **Disallow execution after date** checkbox.
3. Select the date the virtual application will expire.
4. Choose the **Time Source** the virtual application will use to validate the date.

For all expiration modes, the **System clock** setting will use the host system's clock to validate the date. The **Web server clock** setting will validate the date against an HTTPS-based web server. Check the **Disallow execution if web server is unreachable** checkbox to prevent the application from being executed offline.

 The **Web server clock** setting is more secure than the **System clock** setting since it prevents the expiration mechanism from being circumvented by modifying the system clock. However, this setting will prevent applications from executing on devices which cannot connect to the time server source.

Optionally, an **Expiration Warning** can be set to warn the user when the virtual application is about to expire. The message will be displayed each time the virtual application is executed when it is within the specified threshold.

9 Troubleshooting

9.1 Troubleshooting

This section describes the most common configuration errors which occur when using ZENworks Application Virtualization.

If you encounter a problem with a virtual application, please carefully read this section before using other support options. It is very likely that the issue you have encountered is addressed in this section.

9.2 Problems accessing Internet-based resources

Several ZENworks Application Virtualization features require access to Internet-based resources in order to function properly. These features may be unavailable if ZENworks Application Virtualization is unable to connect to the Internet.

In many corporate environments, access to the Internet is filtered through a firewall or proxy server. In these cases, ZENworks Application Virtualization will attempt to automatically configure itself for Internet access based on the system Internet settings. In some cases, however, it may be necessary to manually configure the proxy server settings.

To configure proxy server settings, click the **Proxy Settings...** option on the **Options** menu. This displays the **Proxy Settings** dialog. Enter appropriate proxy server settings in the dialog. It may be necessary for you to consult your system administrator to obtain your organization's proxy server settings.

9.3 Generating diagnostic-mode virtual applications

Occasionally, errors during virtual application configuration result in an executable which fails to run properly; that is, to emulate exactly the behavior of the original source application. Usually this is a result of an error in the virtualization configuration, such as a missing file or registry entry.

To assist in diagnosis of these problems, ZENworks Application Virtualization offers the option of creating *diagnostic-mode executables*. Diagnostic-mode executables generate logging data during execution that can assist in diagnosis of problems related to virtualization.

To generate a diagnostic-mode executable, select the **Generate diagnostic-mode executable** option on the **Settings** section of the **Virtual Application** ribbon bar. Then click to generate the instrumented executable.

Execution of the instrumented executable will generate a *xclog_<id>.txt* file in the application startup directory that contains detailed diagnostic data gathered during execution. Inspection of this file, particularly of entries labeled *WARNING* or *ERROR*, often allows diagnosis of virtualization errors. If you require assistance from Code Systems technical support to resolve your problem, we strongly encourage you to submit this information with along with your support request to facilitate resolution of your issue.

Important: Because diagnostic-mode executables run significantly slower than standard executables and generate very large log files, diagnostic-mode executables should not be distributed to your end-users.