

ZENworks for Servers 2

Consolidating Data From Many Servers to One

As you are probably aware, Novell's ZENworks for Servers 2 is a directory-enabled server management solution that leverages NDS eDirectory to automate and simplify the tasks involved in managing multiple servers. ZENworks for Servers 2 provides you with a single point from which to control and monitor server and network performance. In addition, ZENworks for Servers 2 enables you to automate the distribution of software applications and content across NetWare, Windows NT, and Windows 2000 servers throughout your company's entire network. As this article explains, you can also extend ZENworks for Servers Server Policies to consolidate data from many servers to one server.

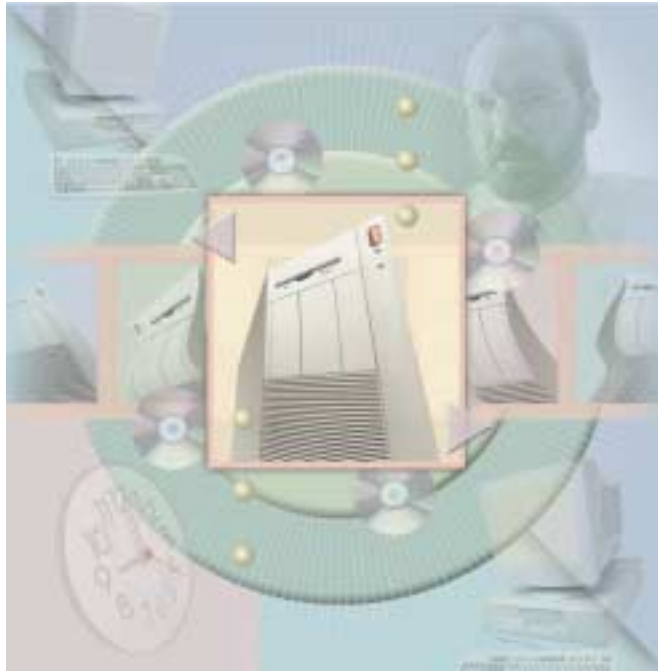
To distribute software applications and content, ZENworks for Servers 2 uses Tiered Electronic Distribution (TED) to automatically distribute software and data files to selected subscriber nodes on the network. Single or multiple distribution source servers can be located anywhere on the network or on the Internet. You can distribute data from these source servers directly to target servers, or you can distribute software data to strategically located parent subscriber nodes. These subscriber nodes can then distribute the software to local target servers.

ZENworks for Servers 2 also features directory-enabled policies that enable you to standardize the configuration of servers in a particular eDirectory tree. Using the Run Policy or the Script Policy, you can run the same policies on many servers at the same time. (For more information about the capabilities of ZENworks for Servers 2, visit www.novell.com/products/zenworks/servers.)

HOW CAN YOU PUT ALL YOUR EGGS IN ONE BASKET?

Although you know that ZENworks for Servers 2 can simplify the task of distributing data to servers, you may not be

aware that by modifying Server Policies, you can use ZENworks for Servers 2 to consolidate data from several servers to one server. For example, you may need to consolidate information such as server logs and installation logs in a central location where that information can be monitored. This data consolidation is particularly useful for companies, such as banks, insurance companies, and retail stores, that have servers located in various geographic locations.



THE ZENWORKS SOLUTION

To consolidate data, you can create a Script Policy that logs a user in to an FTP server and copies the specified files to this FTP server. Script Policies can take one command or script and apply this command or script to many servers. These servers can then run the script or command at a time you specify.

To run this script on many servers, you create a Script Policy. This Script Policy is part of a ZENworks for Servers Server Policy Package, which you then associate with the servers that you want to run the script. (See Figure 1.)

This article outlines the process of creating a Script Policy. Specifically, this article explains how you can create a NetBasic script that logs a user in to an FTP server, creates a subdirectory

on the FTP server for each server you specify, and then copies a file from each server to its corresponding directory on the FTP server.

Note. A Script Policy can be an NCF, NetBasic, or Pearl script. For this example, NetBasic has the required functionality.

THE SCRIPT IN DETAIL

The following is a line-by-line explanation of the NetBasic script. When a server runs this script, the server will automatically send the requested data to the central FTP server.

Note. You can find the NetBasic commands used in this script at http://developer.novell.com/ndk/doc/nb6/index.html?nb6__enu/data/sdk4059.html. All of the NetBasic documentation you need is also located on this web site.

Sub Main

This command indicates the beginning of the program.

```
sHost = "10.0.0.10"
```

sHost contains the IP address of the FTP server that will receive the file(s). You must modify this variable, along with all of the other variables in this script for each application of the script.

```
sUser = "anon"
```

sUser contains the username (account) that will be logged in to the FTP server.

```
sPassword = "password"
```

sPassword is the password for the username specified.

```
sTargetPath = "upload"
```

sTargetPath specifies the directory that holds the target directory structure and files.

```
sTargetDir = "%SERVER_NAME%"
```

sTargetDir specifies the directory that is unique to each server and will hold the individual files copied from each server. In this case, I am using a ZENworks for Servers variable to specify the server name. As a result, ZENworks for Servers 2 will resolve %SERVER_NAME% to the correct server name before sending the script to the NetBasic interpreter.

```
sSourceFile = "sys:\users\report\myfile.zip"
```

sSourceFile contains the full name of the source file, including the path, that is to be uploaded to the FTP server.

```
sTargetFile = "myfile.zip"
```

sTargetFile contains the filename that will be created on the FTP server.

```
FTP:Login (sHost, sUser, sPassword)
```

This command logs the user in to the FTP server.

```
FTP:CD (sSourcePath)
```

This command changes the current directory to the target path.

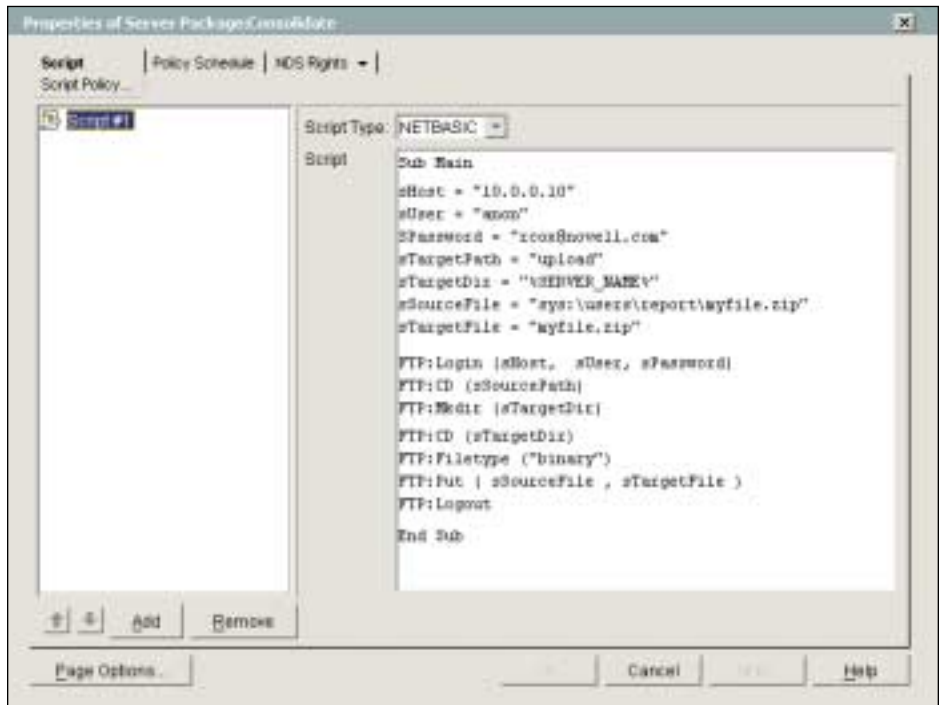
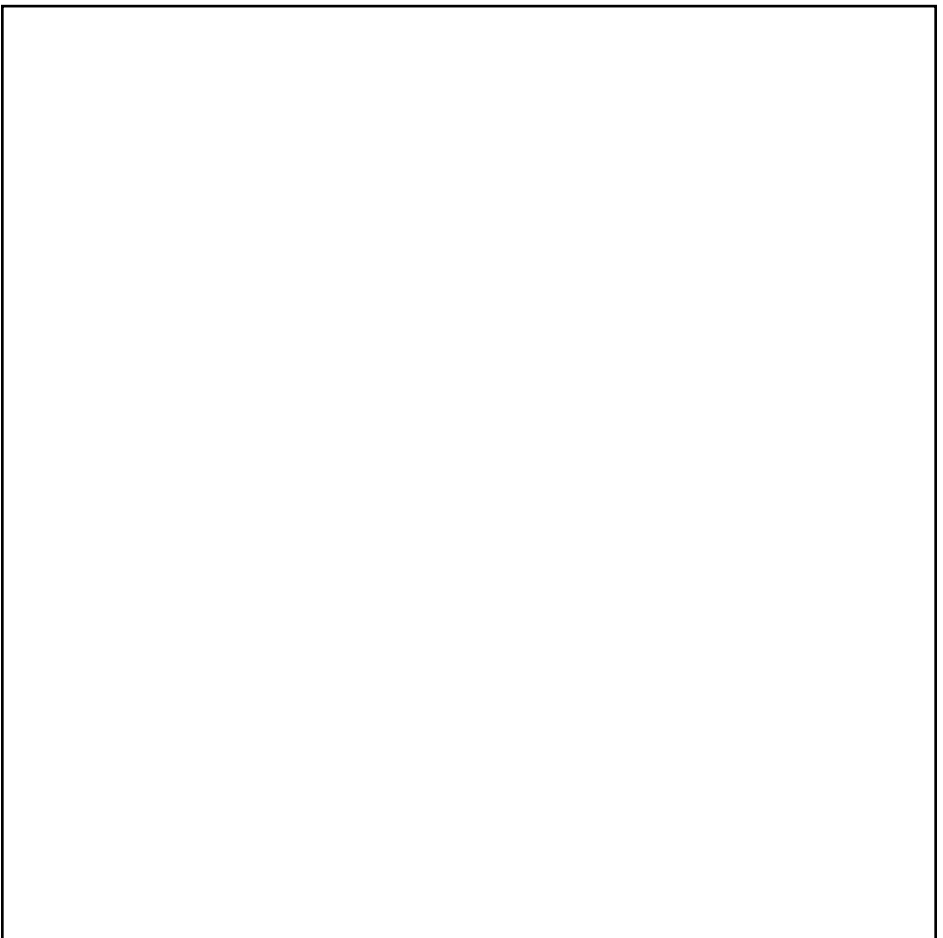


Figure 1. This sample NetBasic script logs in to an FTP server, creates a directory for each server you specify on the FTP server, and then copies files from each server into its corresponding directory. You define this script as part of a Server Policy Package.



Unique File Name

If you need multiple revisions of a file, you simply add a few lines to the Script Policy. These lines will name the file with the date and time, instead of with the standard fixed filename. (In the main article, I have used a fixed filename—myfile.zip.)

To name a file with the date and time, you must set up the Script Policy to make a NetBasic call to retrieve the date and time. You must then modify the returned string to contain only digits. Finally, you must append the extension to the string to create the file name.

Below is a line-by-line example of a script that performs these functions. In this example, the script ran on May 1 at 9:11:02.

```
SysDateObj = SYS:Date:Get
This command gets the date and time in a predefined format.
```

```
sTargetFile = STR:Sub (SysDateObj.Date, 1, 2)
This command copies the month digits into the filename.
sTargetFile is now 05.
```

```
sTargetFile = sTargetFile + STR:Sub (SysDateObj.Date, 4, 2)
This command appends the day digits. sTargetFile is now 0501.
```

```
sTargetFile = sTargetFile + STR:Sub (SysDateObj.Time, 1, 2)
This command adds the hour digits. sTargetFile is now 050109.
```

```
sTargetFile = sTargetFile + STR:Sub (SysDateObj.Time, 4, 2)
This command adds the minutes. sTargetFile is now
05010911.
```

```
sTargetFile = sTargetFile + STR:Sub (SysDateObj.Time, 7, 2)
This command adds the seconds. sTargetFile is now
0501091102.
```

```
sTargetFile = sTargetFile + ".zip"
This command adds .zip to the filename string.
```

In this example, sTargetFile ends up as 0501091102.zip. This unique filename will not overwrite earlier versions of the file that have been placed in the directory by the same script. ●

```
FTP:Mkdir (sTargetDir)
```

This command creates the subdirectory under the target path. If this subdirectory already exists, the command returns an error that is ignored.

```
FTP:CD (sTargetDir)
```

This command changes the current directory to the server unique directory. All files in this directory come from the target server.

```
FTP:Filetype ("binary")
```

This command sets the file type to

binary. If text files will be transmitted, you should omit this command.

```
FTP:Put (sSourceFile, sTargetFile)
```

This command puts the source file to the target file. If a file with this name already exists, the file will be overwritten. (For more information, see "Unique File Name.")

```
FTP:Logout
```

This command logs the user out of the FTP connection. Obviously, this command is a more elegant solution

than just terminating the script.

```
End Sub
```

This command specifies the end of the script.

SCHEDULING THE SCRIPT

You can schedule the script to run as often as needed. To schedule the script to run, you enter the schedule in the Policy Schedule Tab, which is located in the Properties of Server Package window. I suggest that you set a window of time for the file transfer and then check the random box in the schedule. (See Figure 2.) This type of schedule ensures that all of the servers running this script do not hit the server at the same time, thus preventing congestion at the server and on slow WAN links.

CONCLUSION

ZENworks for Servers 2 is a powerful tool that leverages eDirectory to provide centralized management of network servers. Using policies and scripts, you can automate almost any server function across your company's entire network.

By implementing the simple Script Policy described in this article, you can automate a task and manage your servers more efficiently. By implementing this Script Policy, you can free up resources for more productive, less tedious work.

Rick Cox is the product manager for ZENworks for Servers. You can contact Rick Cox at rcox@novell.com. ●

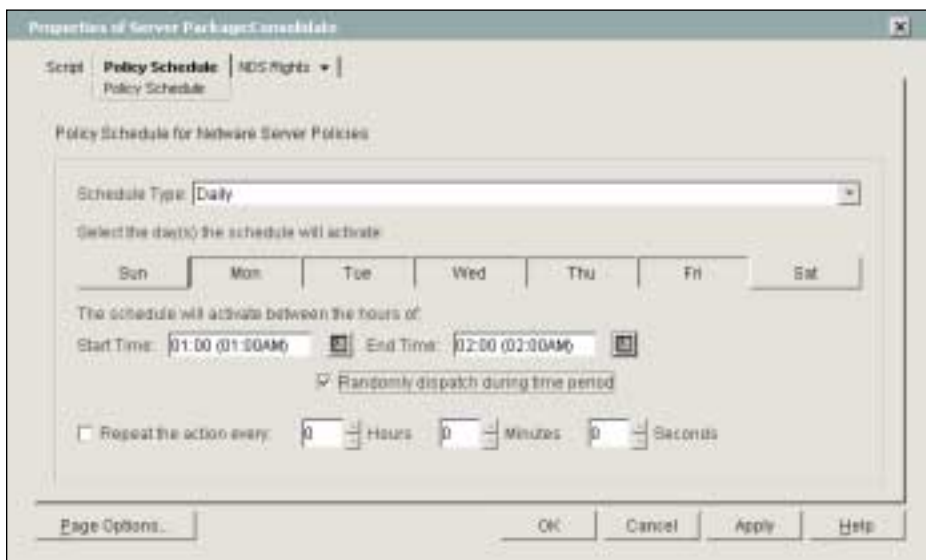


Figure 2. ZENworks for Servers 2 can run a Script Policy at a random time within the time window that you specify. Scheduling the Script Policy in this way prevents the FTP server from being deluged with files and conserves valuable bandwidth.