

# **Software Deployments to Windows Servers using Sysinternals' psexec**

## 1 Document Version

Revision	Editor	Reviewer	Comments
1.0	Tom Clark		Initial draft.



## **2 Table of Contents**

1	Document Version .....	2
2	Table of Contents.....	3
3	PSTools.....	4
4	Remote deployments .....	5
5	Integration with Harvest .....	6

### 3 PSTools

PSTools from [Sysinternals](#) is a powerful suite of free utilities for performing administrative tasks on local and remote Windows systems. One of these utilities, psexec, allows execution of processes on remote machines which is very useful for deploying software builds to remote servers.

A key point of note is that psexec – unlike the Harvest agent – does not require any software to be pre-installed on the target server. This is useful where software installation is unwanted, difficult or unplanned.

Psexec works by making use of the \$IPC administrative share that exists on all Windows systems. The application copies itself to the target server using the admin share, installs itself as a service and then communicates with the client to run the process specified. Once the process has completed, the service uninstalls and deletes itself.

## 4 Remote deployments

Consider the following build and deployment process:

1. Build is created
2. Build is copied to target server
3. Build is deployed on target server

Once the build is created, the deployment might be achieved using psexec with the following script:

```
1 @ECHO OFF
2 SET BUILD_TO_DEPLOY=C:\archive\build.zip
3 SET PS_TOOLS_DIR=C:\pstools
4 SET TARGET_SERVER=192.168.1.15
5 SET TARGET_SERVER_USER=administrator
6 SET TARGET_SERVER_PASS=password
7 REM Copy build to server...
8 NET USE \\%TARGET_SERVER%\c$ %TARGET_SERVER_PASS%
  /USER:%TARGET_SERVER_USER%
9 COPY %BUILD_TO_DEPLOY% \\%TARGET_SERVER%\c$
10 NET USE \\%TARGET_SERVER%\c$ /DELETE
11 REM Unzip build on target server...
12 %PS_TOOLS_DIR%\psexec.exe \\%TARGET_SERVER% -u
  %TARGET_SERVER_USER% -p %TARGET_SERVER_PASS% -w c:\ unzip
  %BUILD_TO_DEPLOY% -d c:\builddir
13 REM Run build script extracted from archive...
14 %PS_TOOLS_DIR%\psexec.exe \\%TARGET_SERVER% -u
  %TARGET_SERVER_USER% -p %TARGET_SERVER_PASS%
  c:\builddir\install.cmd
15 REM Clean up...
16 DEL %BUILD_TO_DEPLOY%
17 RMDIR /Q /S c:\builddir
18 REM Build complete.
19 ECHO Build complete.
20 PAUSE
```

Psexec is used twice in the above script, first to unzip the archive after it has been copied to the target server (line 11), and then a second time to run a script, install.cmd, that was contained in



the archive (line 13). The output from install.cmd could be piped to a log file which is then displayed for inspection to check for errors in the deployment.

Note that some lines have been wrapped for clarity.

## **5 Integration with Harvest**

The above process – and a preceding build step – could easily be integrated with Harvest using custom User Defined Processes (UDPs). For example, the script could be extended to accept parameters for a Harvest state and environment and a build created using those details and a custom build tool such as Ant.

Using Trilogy, Trinem’s middleware integration component, the script could be executed on a remote build server and the build deployed all from a user’s Workbench client.

For more information about Trilogy, see the Trinem website at [www.trinem.com](http://www.trinem.com).