



**Microsoft®**

Microsoft®  
**Windows NT®**  
**Workstation**

*Operating System*

---

Deployment Guide  
Automating Windows NT Setup

---

© 1997 Microsoft Corporation. All rights reserved.

*The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.*

*This document is for informational purposes only. **MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.***

*Microsoft, Windows, Windows NT, the Windows logo, and MS-DOS are registered trademarks of Microsoft Corporation in the United States and/or other countries.*

*Other trademarks or tradenames mentioned herein may be the trademarks of their respective owners.*

*Microsoft Corporation • One Microsoft Way • Redmond, WA 98052-6399 • USA*

*297*

---

Designed for



# Microsoft® **Windows NT®** **Workstation**

---

## **Automating Windows NT Setup**

This deployment guide provides information, tips, and tricks that will help you automate the Microsoft® Windows NT® setup process. It is designed for Information Systems professionals that are tasked with installing either Windows NT Workstation or Windows NT Server on many computers. You should use this guide in conjunction with Part 1 of the Windows NT Workstation Resource Kit.

Microsoft Windows NT includes many tools and features that can be used to automate the setup process. You will find detailed information and examples of these tools in the following five chapters:

- Getting Started
- The Windows NT Workstation 4.0 Setup Script (UNATTEND.TXT) File
- Configuring Machine-Specific Information
- Application Pre-Installation (SYSDIFF.EXE)
- Customizing Windows NT

CONTENTS

INTRODUCTION..... 8

CHAPTER 1 GETTING STARTED ..... 9

- What Microsoft Windows NT Deployment Tools Can Do 9
- What Microsoft Windows NT Deployment Tools Cannot Do 9
- Tools Used with Windows NT Deployment 10
  - Setup Manager - Creating a Windows NT 4.0 Setup Script File (UNATTEND.TXT) 10
  - System Difference Tool: SYSDIFF.EXE 10
  - Windows NT Registry Tools 10
    - REGEDIT.EXE..... 10
    - REGEDIT32.EXE..... 10
    - REGINI.EXE ..... 10
  - Client Connectivity Software 11
  - Windows NT Setup: WINNT.EXE and WINNT32.EXE 11
    - WINNT.EXE and WINNT32.EXE command line parameters..... 11
- Steps to Automating Windows NT Deployment 11
  - Step 1 – Build the Windows NT setup script file (UNATTEND.TXT). 11
  - Step 2 – Build pre-installation packages using the System Difference tool (SYSDIFF.EXE). 12
  - Step 3 – Create process to configure machine-specific information. 12
  - Step 4 - Build process to automate your distribution method. 12
- Building the Distribution Server 12
  - Copy the Windows NT Source Files 12
  - Copy Custom Files Using the \$OEM\$ Directory 12
  - Structure of the Distribution Share Point 13
- Converting Short Filenames to Long Filenames 14
- Using Disk Duplication to Distribute Windows NT 15
- Installation Performance Considerations 16
  - Optimizing Client Connectivity Software 16
  - Optimizing the Character Mode Setup Phase 16
  - Decreasing the Number of Files Copied During Setup 17
    - Removing Peer Web Services..... 17
  - Removing Third-Party Provided Network Adapter Drivers 17

CHAPTER 2 THE WINDOWS NT WORKSTATION 4.0 SETUP SCRIPT FILE (UNATTEND.TXT)..... 18

- Introduction 18
- UNATTEND.TXT File Format and Reference 18
- Description of UNATTEND.TXT File Parameters 19
  - [Unattended] 19
    - OemPreinstall 19
    - NoWaitAfterTextMode 19
    - NoWaitAfterGuiMode 20
    - FileSystem 20
    - ExtendOemPartition 20

Deleted: . Microsoft BackOffice White Paper . 1

ConfirmHardware	21
NtUpgrade	21
Win31Upgrade	21
OverwriteOemFilesOnUpgrade	22
TargetPath	22
ComputerType	22
KeyboardLayout	23
[MassStorageDrivers]	23
<mass storage driver description>	23
[DisplayDrivers]	24
<display driver description>	24
[KeyboardDrivers]	24
<keyboard driver description>	25
[PointingDeviceDrivers]	25
<pointing device driver description>	25
[OEMBootFiles]	25
TXTSETUP.OEM	26
<hal file name>	26
<scsi driver file name>	26
[OEM_Ads]	26
Banner	26
Logo	26
Background	27
[GuiUnattended]	27
OemSkipWelcome	27
OEMBlankAdminPassword	27
TimeZone	27
AdvServerType	28
DetachedProgram	29
Arguments	29
[UserData]	29
FullName	29
OrgName	29
ComputerName	30
ProductID	30
[LicenseFilePrintData]	30
AutoMode	30
AutoUsers	30
[Display]	31
ConfigureAtLogon	31
BitsPerPel	31
Xresolution	31
Yresolution	32
Vrefresh	32
Flags	32

AutoConfirm	32
InstallDriver	33
InfFile	33
InfOption	33
[Modem]	33
InstallModem	34
[<modem parameter section>]	34
<COM port number>	34
[Network]	35
Attended	35
JoinWorkgroup	35
JoinDomain	35
CreateComputerAccount	35
InstallDC	36
DetectAdapters	36
InstallAdapters	36
InstallProtocols	36
InstallServices	37
InstallInternetServer	37
DoNotInstallInternetServer	37
[<Detect Adapters Section>]	37
DetectCount	37
LimitTo	38
<Netcard Inf option>	38
[<Install Adapters Section>]	38
<Netcard Inf option>	38
[<netcard parameter section>]	39
[<Protocols Section>]	39
NBF	39
NWLNKIPX	39
TC	39
DLC	40
RASPPTP	40
STREAMS	40
[<NetBeui Parameters>]	40
[<IPX Parameters>]	40
[<Tcpip Parameters>]	41
DHCP	41
ScopeID	41
[<DLC Parameters>]	43
[<RASPPTP Parameters>]	43
[<STREAMS Parameters>]	43
[<Services Section>]	43
SNMP	43
RAS	43

NWWKSTA	43
NETMON	44
STCPIP	44
SAP	44
TCPPRINT	45
DHCP	45
DNS	45
WINS	45
[<NetWare Client Parameters>]	46
!DefaultLocation	46
!DefaultScriptOptions	46
[<Snmp Parameters>]	46
Accept_CommunityName	46
Send_Authentication	47
Any_Host	47
Limit_Host	47
Community_Name	47
Traps	47
Contact_Name	48
Location	48
Service	48
[<RasParameters>]	48
PortSections	48
DialoutProtocols	48
DialinProtocols	49
NetBEUIClientAccess	49
TcpIpClientAccess	49
UseDHCP	49
StaticAddressBegin	49
StaticAddressEnd	50
ExcludeAddress	50
ClientCanRequestIPAddress	50
IpxClientAccess	50
AutomaticNetworkNumbers	50
NetworkNumberFrom	51
AssignSameNetworkNumber	51
ClientsCanRequestIpxNodeNumber	51
[<port section name>]	51
PortName	51
DeviceType	51
PortUsage	52
[<NETMON Parameters>]	52
[<STCPIP Parameters>]	52
[<SAP Parameters>]	52
[<TCPPRINT Parameters>]	52

[<DHCP Parameters>]	52
[<DNS Parameters>]	52
[<WINS Parameters>]	52
[<internet information server section>]	53
InstallINETSTP	53
InstallADMIN	53
InstallFTP	53
FTPRoot	53
InstallWWW	53
WWWRoot	54
InstallGOPHER	54
GopherRoot	54
InstallDir	54
InstallW3SAMP	54
InstallHTMLA	54
GuestAccountName	55
GuestAccountPassword	55
Sample UNATTEND.TXT Files	55
Sample 2	57
Configuring Network Adapters and Setup Information Files for Automated Installation	60
Building the Windows NT Setup Script Network Adapters Section	60
Example 1 .....	60
Example 2 .....	61
Example 3 .....	61
Determining Network Adapters Parameters in the UNATTEND.TXT	61
Overview of a Network Component .INF that Supports Unattended Installation	62
Setting Parameters for Network Adapter Cards.....	63
Verifying and Testing Component .INFs.....	63
Verifying the OEM File for STF_GUI_UNATTENDED	64
Installation of Network Adapters Drivers Not Supplied on the Windows NT 4.0 Retail CD	65
Bypassing the "Current Netcard Parameters Are Not Verifiably Correct" Message	66
Network Adapter Option Name	67
Driver Supplied in the \I386 Directory	67
Drivers Supplied in the \DRVLIB\NETCARD\X86 and \I386\DRVLIB.NIC Directories	69
OEM Install Options that Can be Used with UNATTEND.TXT	72
TXTSETUP.SIF Entries for Retail-Supplied Files that Work with OEM Options in the UNATTEND.TXT	73
Third-Party Video Display Drivers and Display Settings	77
Options for Microsoft-Supplied Video Drivers (Part of I386/Auto-Detected)	77
Options for OEM-Supplied Video Drivers	78
TXTSETUP.OEM and [OEMBootFiles]	79

TXTSETUP.OEM File: Format and Sample	79
Error Messages When Working with TXTSETUP.OEM	83
CHAPTER 3 CONFIGURING MACHINE-SPECIFIC INFORMATION	85
Create Unique Setup Script Files for Each Computer	85
Modify the Machine-Specific Settings After the Character Mode Portion of Windows NT 4.0 Setup	85
Editing the Windows NT 4.0 Setup Script (UNATTEND.TXT) After Character Mode Setup	86
Editing the Uniqueness Database File (.UDB) After Character Mode Setup	86
Configure Machine-Specific Information Using Uniqueness Database Files (.UDB)	87
Specifying a Unique ID.....	87
Creating the .UDB.....	87
Replacing a Line in Setup Script.....	88
Adding a New Line to Setup Script .....	89
Deleting a Line from Setup Script .....	89
CHAPTER 4 WINDOWS NT 4.0 APPLICATION PRE- INSTALLATION TOOL (SYSDIFF.EXE).....	94
Overview	94
Installing SYSDIFF.EXE	94
SYSDIFF.EXE Parameters and Syntax	94
/snap Mode	95
/diff Mode	95
/apply Mode	95
Dump Mode	96
/inf Mode	97
Building Application Images for Pre-Installation	98
Three Steps to Building an Application Image	98
Adding an Application Image to the Distribution Server	98
SYSDIFF /inf Mode	99
SYSDIFF/apply Mode	99
Troubleshooting SYSDIFF.EXE	100
Error Message: System Error 5.	100
Error Message: An incorrect or duplicate computer name is created after applying the difference file.	100
Error Message: Contact the Manufacturer...	100
Error Message: Diff Failed (error=2)	101
Error Message: Diff Failed (error=32)	101
Problem: Empty directories on the master machine are not processed by SYSDIFF /snap.	102
Problem: Some older applications that use .INI files do not have the .INF file copied.	102
Problem: Networks with limited bandwidth experience problems when doing SYSDIFF /INF to the distribution server.	102

Problem: Package file dates are changed.	102
Problem: The computer stops responding (hangs) when you use the SYSDIFF /apply command.	103
Problem: The SYSDIFF tool takes a long time to finish and the image file is extremely large.	103
Problem: When you run the SYSDIFF tool, it appears on the screen briefly and then nothing else happens.	103
Problem: Some of the changes are not applied when you run the SYSDIFF /apply command.	103
Problem: Network drives appear in My Computer after you apply a difference file.	104
Problem: Temporary files are left in the folder where you are creating SYSDIFF.EXE files.	104
Problem: SYSDIFF.EXE /apply or /inf fails when updating an .INI file or fails to copy the .INI files.	104
CHAPTER 5 CUSTOMIZING WINDOWS NT.....	108
Overview	108
Distributing Files Automatically Using Windows NT Setup	108
Customizing the Start Menu	108
Tools to Customize Windows NT	109
REGEDIT.EXE	109
Configuring a System to Automatically Logon and Execute a Program	111
Configuring System to Skip the Welcome Screen .....	112
Executing a Batch File on First Logon to Customize Windows NT.....	113
Adding Silent Application Setup Commands to a Batch File .....	114
Customizing Windows NT Logon .....	115
Executing Commands During Windows NT Setup - CMDLINES.TXT	117
Using the Windows NT Setup Engine, SETUPAPI.DLL	117
Removing the Gopher and World Wide Web (WWW) Services .....	120
Automating Installation of Windows NT Service Packs	120
Automating Selection of Windows Accessories and Components	121
Automating Installation of Peer Web Services	124
Removing Microsoft Internet Explorer, Microsoft Exchange Client, and Image Viewer	124
Microsoft Internet Explorer 2.0	124
Microsoft Exchange Client	125
Image Viewer	125
Automating Installation of Microsoft Exchange Server Client and Microsoft Internet Explorer 3.0x for Windows NT	126
Automating Installation of the Exchange Client Supplied with Microsoft Exchange Server	126
Installation of the Microsoft Internet Explorer 3.0x	128
For More Information	128

---

## INTRODUCTION

This deployment guide provides information, tips, and tricks that will help you automate the Microsoft® Windows NT® setup process. It is designed for Information Systems professionals that are tasked with installing either Windows NT Workstation or Windows NT Server on many computers. You should use this guide in conjunction with Part 1 of the Windows NT Workstation Resource Kit.

Microsoft Windows NT includes many tools and features that can be used to automate the setup process. You will find detailed information and examples of these tools in the following five chapters:

- Getting Started
- The Windows NT Workstation 4.0 Setup Script (UNATTEND.TXT) File
- Configuring Machine-Specific Information
- Application Pre-Installation (SYSDIFF.EXE)
- Customizing Windows®

This chapter includes an overview of the Microsoft Windows NT deployment tools, the steps to automating the deployment process, and details about creating a distribution share for Windows NT. In addition, it covers what Microsoft Windows NT Deployment Tools can and cannot do.

#### **What Microsoft Windows NT Deployment Tools Can Do**

- Install many standard productivity applications. Standard productivity applications are applications like Microsoft Office or any other non-service type of application.
- Install the core Windows NT operating system. The core Windows NT operating system consists of the normal items required during Text Mode and GUI Mode setup to achieve a functioning installation of the Windows NT operating system.
- Install the core hardware components. Core hardware components include SCSI drivers, display drivers, mouse drivers, keyboard drivers, and the type of processor used (HAL).
- Install Windows NT Retail Services and Protocols. Retail Service and Protocols includes items listed in the Services and Protocol sections of Control Panel Network.
- Install Windows NT Service Packs during the installation process.

#### **What Microsoft Windows NT Deployment Tools Cannot Do**

- Pre-install applications that run as services using the System Difference tool.
- Pre-install multiple hardware profiles. Windows NT 4.0 supports the use of multiple hardware profiles. Hardware profiles can only be configured using the Control Panel System applet on a completely installed Windows NT 4.0 system.
- Pre-install sound cards. Sound cards cannot be installed during Windows NT setup. Instead, sound cards must be installed after Windows NT 4.0 setup is complete.
- Pre-install printers. Printers cannot be installed during Windows NT setup. Instead, printers must be installed after Windows NT 4.0 setup is complete.
- Install multiple language versions of Windows NT.
- Uninstall Windows NT. Windows NT 4.0 does not have an uninstall procedure. To remove Windows NT 4.0 on FAT partition boot from an MS-DOS® disk and run SYS.COM C: to remove the Windows NT boot sector, then delete the directories created by Windows NT setup. If the system is NTFS you must delete and recreate the partition to remove Windows NT.
- Configure Windows NT Auditing. Windows NT Auditing is an advanced feature of Windows NT and there is no interface during setup for this option.
- Configure Windows NT Replication. Windows NT Replication is an advanced feature of Windows NT and there is no interface during setup for this option.

- Install Windows NT Server Macintosh Services or Apple Talk Protocol. Macintosh Service or Apple Talk Protocol is not an automated part of the product. Manual installation and configuration is required.
- Install PNPISA drivers.

## Tools Used with Windows NT Deployment

### Setup Manager - Creating a Windows NT 4.0 Setup Script File (UNATTEND.TXT)

Setup Manager is the starting point for building a basic Windows NT setup script. Setup Manager is located in the \SUPPORT\DEPTOOLS\ directory on the Windows NT 4.0 Retail CD. Setup Manager does not require any special installation procedure. The utility can be run from the CD or copied to a directory.

### System Difference Tool: SYSDIFF.EXE

The System Difference tool enables you to distribute and install applications automatically during or after Windows NT setup. This can significantly reduce deployment time and costs. It can be used to record the changes made to your system when an application is installed, for example capture those changes in a "package" and then "apply" or install the package on another system during or after the setup process.

### Windows NT Registry Tools

#### *REGEDIT.EXE*

REGEDIT.EXE is a Windows NT supplied utility for working with the registry. REGEDIT.EXE is very similar to the one provided with Windows® 95. If you are familiar with the Import and Export features used often in Windows 95, the same functionality is provided with Windows NT. See Chapter 5, "Customizing Windows NT" for examples of how to use REGEDIT.EXE with Windows NT.

#### *REGEDT32.EXE*

REGEDT32.EXE is a Windows NT supplied utility for working with the registry hives. REGEDT32.EXE provides features that REGEDIT.EXE does not. REGEDT32.EXE is used to modify the stock registry hive supplied with the Windows NT 4.0 operating system prior to installation of Windows NT. See Chapter 5, "Customizing Windows NT" for examples of how to use REGEDT32.EXE.

#### *REGINI.EXE*

REGINI.EXE is a Windows NT 4.0 Resource Kit utility, which provides the ability to make simple modifications to the SYSTEM and SOFTWARE registry hives of the Windows NT registry. REGINI.EXE also provides the ability to change/apply security to the registry. REGINI.EXE is included with the

---

Windows NT 4.0 Resource Kit. See Chapter 5, "Customizing Windows NT" for examples of how to use REGINI.EXE.

### Client Connectivity Software

If you plan to install Windows NT from a network distribution point on systems with newly formatted hard drives, it will be necessary to build a client disk that includes a network client. If you have Windows NT Server, a network client is provided on the Windows NT Server retail CD in the \CLIENTS\MSCLIENT directory. The MS Client can also be downloaded from FTP.MICROSOFT.COM.

### Windows NT Setup: WINNT.EXE and WINNT32.EXE

Windows NT includes both a 16-bit and a 32-bit version of setup. Both provide the same basic functionality but the 32-bit version, WINNT32.EXE, will only run on Windows NT and can be used to upgrade an existing installation of Windows NT.

#### *WINNT.EXE and WINNT32.EXE command line parameters*

```
WINNT [/S[:]sourcepath] [/T[:]tempdrive] [/I[:]inffile]
[/U[:scriptfile]] [/R[X]:directory] [/E:command]
/S[:]sourcepath
```

Specifies the source location of Windows NT files. Must be a full path of the form x:\[path] or \\server\share\[path]. The default is the current directory.

/T[:]tempdrive

Specifies a drive to contain temporary setup files. If not specified, setup will attempt to locate a drive for you.

/I[:]inffile

Specifies the filename (no path) of the setup information file. The default is DOSNET.INF.

/B Floppyless operation (requires /s).

/U Unattended operation and optional script file (requires /s).

/R: Specifies optional directory to be installed.

/RX Specifies optional directory to be copied.

/E: Specifies command to be executed at the end of GUI setup.

/W Used with WINNT.EXE only for starting setup from within Windows 3.x or Windows 95. The /B option is disabled in this mode.

## Steps to Automating Windows NT Deployment

### Step 1 – Build the Windows NT setup script file (UNATTEND.TXT).

Start by creating a basic Windows NT setup script. You can do this by using one of the examples that are included or by using the Windows NT Setup Manager. Once you have created and tested your basic script, add the more advanced options of the Windows NT setup script. See Chapter 2, "The

---

Windows NT Setup Script (UNATTEND.TXT) File" for syntax and parameters and more details on creating your setup script. Also review Chapter 5, "Customizing Windows NT" for information on automating your process beyond what is possible with the Windows NT setup script.

### **Step 2 – Build pre-installation packages using the System Difference tool (SYSDIFF.EXE).**

Use the System Difference tool to automate pre-installation of applications during Windows NT setup. See Chapter 4, "Application Pre-Installation (SYSDIFF.EXE)" for details on using the System Difference tool.

### **Step 3 – Create process to configure machine-specific Information.**

Create a Uniqueness Database (UDB) file or create machine specific setup script files for each computer. You have several options for building a setup process that includes configuring systems with machine-specific information without requiring user interaction during setup. See Chapter 3, "Configuring Machine Specific Information" for details.

### **Step 4 - Build process to automate your distribution method.**

The final step is distributing the Windows NT source files and any other files to each computer. You can use a network distribution point, hard drive duplication, software distribution tools such as Microsoft Systems Management Server or a local device such as a CD-ROM. If you are installing Windows NT on a newly formatted hard drive you will need a boot disk to either access the network or possibly access a local device.

## **Building the Distribution Server**

Before beginning the process of automating Windows NT setup you must build a distribution server. In most cases the best distribution point is a network server. Make sure you have read, write, and change privileges on the network server you use.

### **Copy the Windows NT Source Files**

To build the distribution server, copy Windows NT source files from the Windows NT retail CD to your network server. For Intel and Intel compatible-based processors copy the I386 directory and all of its contents to your network server.

### **Copy Custom Files Using the \$OEM\$ Directory**

Windows NT setup includes a feature that can be used to automatically copy directories, standard MS-DOS 8.3 files, and any tools needed for your automated installation process, to the local hard drive during setup. It is based on a pre-defined directory name and structure. If you include this directory and

---

structure in the root of your distribution point, the files and directories will be copied to your local hard drive during Windows NT setup.

The pre-defined directory name is \$OEM\$. If Windows NT setup finds the \$OEM\$ directory in the root of the distribution point, it will copy all of the files found in this directory to the temporary directory created during the text mode portion of Windows NT setup.

---

*Note: Alternatively, use the Application Pre-Installation Tool (SYSDIFF.EXE) to create the \$OEM\$ directory structure. By using SYSDIFF.EXE to create the \$OEM\$ directory you will eliminate the likelihood of errors that are bound to occur when creating the \$OEM\$ directory and subdirectories manually. And because SYSDIFF.EXE also automatically handles long filenames you will not have to create the necessary \$\$RENAME.TXT files. For more information about SYSDIFF.EXE see Chapter 4, "Application Pre-Installation Tool (SYSDIFF.EXE)." For more information about the \$\$RENAME.TXT file see "Converting Short Filenames to Long Filenames" later in this chapter.*

---

### Structure of the Distribution Share Point

```
\<Distribution directory>
  \OEM$
    \Textmode
    \$$
    \Net
    \Display
    \<drive letter>
    \...
    \<drive letter>
```

Where:

**\<Distribution directory>** includes the Windows NT source files and the \$OEM\$ directory.

#### **\OEM\$ directory includes:**

- The CMDLINES.TXT file. This is a text file that contains commands you want to execute during Windows NT setup. This can be used extensively to customize your Windows NT installation. Details can be found in Chapter 5, "Customizing Windows NT".
- All files needed to execute any commands included in CMDLINES.TXT. For example, if you include the following command:

```
SYSDIFF /APPLY /m APPSDIFF.IMG
```

You would copy SYSDIFF.EXE, SYSDIFF.INF and APPSDIFF.IMG to the \$OEM\$ directory.

- **\OEM\$\Textmode** directory contains the hardware-dependent files that Setup Loader and Text Mode Setup install to the target computer. These files can include OEM HALs, SCSI, keyboard, video, and pointing device drivers, and TXTSETUP.OEM, which directs the loading and installing of

these components.

- **\\$OEM\$\\$\$** directory contains the system files (new files or replacement to retail files) that you want to copy to the various subdirectories when Windows NT is installed. The structure of this directory must match the structure of a standard Windows NT installation, where **\\$OEM\$\\$\$** matches **%Windir%**, **\\$OEM\$\\$\$\System32** matches **%Windir%\System32**, and so on. Each subdirectory should contain the files that need to be copied to the corresponding system directory on the target machine. This directory should also contain **\$\$Rename.txt**, which lists all files that need to be renamed, such as files in 8.3 format which must change to long filenames.

For example, if you install Windows NT in a directory named **C:\WINNT** and you want setup to copy a custom bit map file, **MYBITMAP.BMP** to the **C:\WINNT** directory, you can copy **MYBITMAP.BMP** in the **\<Distribution directory>\\$OEM\$\\$\$** directory.

- **\\$OEM\$\NET** directory contains only subdirectories, each of which contains the files for a particular OEM network component (network cards, network services, and network protocol). Files in this directory are used by the network portion of Windows NT Setup.
- **\\$OEM\$\DISPLAY** directory includes files for OEM-supplied video drivers.
- **\\$OEM\$\<drive letter>** directory includes any files or directories you want setup to copy to a drive specified by **<drive letter>**. This directory should also contain **\$\$Rename.txt**, which lists all files that need to be renamed, such as files in 8.3 format which must change to long filenames. For more information see the "Converting Short Filenames to Long Filenames" section later in this chapter.

Example: To create a directory named **\DATA** and copy files to this directory on your D drive, you would create a directory with the following name on your distribution share and copy all of the files you want setup to copy in this directory:

**\<Distribution directory>\\$OEM\$\D\Data**

## Converting Short Filenames to Long Filenames

Windows NT setup uses a special file, **\$\$Rename.txt**, which contains information on converting short filenames to long filenames. You can create the file manually using a text editor or automatically by using the System Difference Tool in **/inf** mode. If you plan to use this file, make sure to place it in the directory of the distribution directory containing the files that need to be converted.

Tip - If your MS-DOS tools cannot copy directories with path names longer than 64 characters, you can use short filenames for the directories and then use **\$\$Rename.txt** to rename them later.

---

The syntax for \$\$Rename.txt is as follows:

```
[section_name_1]
short_name_1 = "long_name_1"
short_name_2 = "long_name_2"
.
short_name_x = "long_name_x"
```

```
[section_name_2]
.
```

Where:

- section\_name\_1 and so forth is the path to the directory that contains the files. A section can have no name, or "\" as a name. In this case it indicates that the section contains the name of the files or subdirectories that are on the root of the drive.
- short\_name\_1 and so forth is the name of the file or subdirectory in this directory to be renamed. It must NOT be enclosed in quotes.
- long\_name\_1 and so forth is the new name of the file or subdirectory. Note that this name should be inside double quotes if it contains spaces or commas.

#### Sample \$\$RENAME.TXT

```
[MSO]
MICROS~1.LNK="Microsoft PowerPoint Setup.lnk"
MICROS~2.LNK="Microsoft PowerPoint.lnk"
TEMPLA~1="Templates"

[MSO\Office\MSN]
MICROS~1.MCC="Microsoft Access 95 Forum.mcc"
MICROS~2.MCC="Microsoft Excel 95 Forum.mcc"
MICROS~3.MCC="Microsoft Office 95 Forum.mcc"
```

## Using Disk Duplication to Distribute Windows NT

You can use a disk duplication program or device for clean or new installation environments. This method can save time and effort in your production line. To use this method, you must acquire special equipment or software for duplicating hard disks.

To preinstall to multiple x86-based computers:

1. Follow all of the steps necessary to automate a Windows NT Deployment, including building your setup script, creating packages for applications that you want to pre-install, and any customization of Windows NT. Test and verify your process. These steps are identical to those used in the other methods of distributing Windows NT source files.
2. Run Windows NT setup on one computer but stop Windows NT setup at the

- 
- second reboot, after the text mode portion of setup and before the GUI mode portion of setup.
3. Remove and duplicate the hard drive of that computer.
  4. Install the duplicate hard drive in a new machine.
  5. Optionally, follow the steps in Chapter 3, "Configuring Machine Specific Information" to "Modify the Machine-Specific Settings After the Character Mode Portion of Windows NT 4.0 Setup."
  6. Start the new machine. At this point the GUI mode portion of setup will begin.

To pre-install to multiple RISC-based computers, you must install Windows NT on one of the computer's hard drives and then use that drive to pre-install on a second drive. You then remove the second drive from the computer and duplicate it.

---

*Note: Microsoft does not support duplication of disks if duplicated after the GUI mode portion of Windows NT Workstation 4.0 setup. Using this method compromises the security of your systems. See Q162001 "Do Not Disk Duplicate Installed Versions of Windows NT" in the Microsoft Knowledge Base for more information.*

---

Once a duplicate hard disk has been installed in a target computer, the computer is ready to complete Windows NT setup.

## Installation Performance Considerations

### Optimizing Client Connectivity Software

The 16-bit Windows NT setup program, WINNT.EXE, is subject to the same conventional memory limitations as any other 16-bit MS-DOS application. Memory management and disk caching are important to ensure the best performance during the text mode phase of Windows NT Setup.

If possible, use the Microsoft NetBEUI protocol for network connectivity. The NetBEUI protocol is small and very fast. Consideration has to be given to ensure that access to the distribution point does not require crossing a network router when using NetBEUI. If you wish to use NetBEUI on a segmented network, the deployment image can be copied to a local share point on each segment. The distribution server can be any machine that provides networking support. For example, a laptop with Microsoft Windows 95 used as a portable distribution server if desired.

If using Microsoft Client 3.0, ensure that the Change Redirector Option is set to Use the Basic Redirector. The basic redirector provides all standard workgroup functions. It also uses less memory and disk space than the full redirector.

### Optimizing the Character Mode Setup Phase

Using HIMEM.SYS and EMM386.EXE to maximize the available memory and using SMARTDRV.EXE to manage disk caching can have a significant impact on the time necessary to complete the first phase of setup. In some cases it

---

can cut the time to completion by more than 60%. Refer to the MS-DOS Users Guide for additional information.

### **Decreasing the Number of Files Copied During Setup**

You can eliminate up to 25MB of file transfer during the initial text mode phase of setup by removing unneeded components.

#### *Removing Peer Web Services*

When installing Microsoft Windows NT Workstation 4.0 using an UNATTEND.TXT file, there is no option to install Peer Web Services. Peer Web Services can be automated after the installation of Windows NT, see Chapter 5, "Customizing Windows NT". Since Peer Web Services is not installed during setup, you can remove the \INETSRV directory and can save approximately 5MB of file transfer. Note that this does not apply to Microsoft Windows NT 4.0 Server. The DOSNET.INF controls what optional directories are copied during the text mode phase of setup. To prevent setup from copying the \INETSRV directory, edit DOSNET.INF with a text editor and locate INETSRV under the [OptionalSrcDirs] section and place a semi-colon at the beginning of the line.

Example:

```
[OptionalSrcDirs]
; inetsrv
drvlib.nic
```

#### **Removing Third-Party Provided Network Adapter Drivers**

During the installation of Windows NT 4.0, the optional source directory DRVLIB.NIC is copied by default during the initial text mode phase of setup. The DOSNET.INF controls what optional directories are copied during the Text Mode phase of setup. Commenting out DRVLIB.NIC can save up to 20MB of file transfers. To stop the copy of the DRVLIB.NIC directory, edit DOSNET.INF with a text editor and locate DRVLIB.NIC under the [OptionalSrcDirs] section and place a semi-colon in at the beginning of the line.

Example:

```
[OptionalSrcDirs]
inetsrv
; drvlib.nic
```

Special consideration must be given before commenting out DRVLIB.NIC with a semicolon. If the network adapters being used require that files from the DRVLIB.NIC, do not comment out the line, instead remove all directories that are not needed which will still provide an increase in performance.

## Introduction

You can specify the settings for Windows NT 4.0 installations by creating a custom file in UNATTEND.TXT format and using this setup script for installation. To run Windows NT 4.0 setup using a setup script, you must specify the name of your script file and the location of the Windows NT 4.0 source files:

Examples:

```
WINNT /U:F:\i386\UNATTEND.TXT /S:F:\i386
WINNT32 /U:MYSCRIPT.TXT /S:\\MYSERVER\i386
```

Where F:\i386\UNATTEND.TXT and MYSCRIPT.TXT are the names of setup script files and F:\i386 and \\MYSERVER\i386 are locations of the Windows NT 4.0 installation files. Both can be any location, local or network-based.

---

*Note: The filename for the setup script file must adhere to the MS-DOS 8.3 file format.*

---

In this document, the setup script file will be referred to as the UNATTEND.TXT file. There are two examples of the UNATTEND.TXT at the end of this chapter.

## UNATTEND.TXT File Format and Reference

In general, a setup script file consists of section headers, parameters, and values for those parameters. Most of the section headers are pre-defined whereas some may be user-defined. It is not necessary to specify all of the possible parameters/keys in an UNATTEND.TXT if the installation does not require them. The file format is as follows:

```
[section1]
;
; Section contains keys and the corresponding
; values for those keys/parameters.
; keys and values are separated by "=" signs
; Values usually require double quotes "" around them
;
key = value
.
.

[section2]
key = value
.
.
```

---

## Description of UNATTEND.TXT File Parameters

### [Unattended]

This section header is used to identify whether an unattended installation is being performed or not. The [Unattended] section must exist or the UNATTENDED.TXT file will be ignored. Parameters that can exist in this section are discussed below.

#### **OemPreinstall**

Values: Yes | No

Example Syntax:

OemPreinstall = Yes

Determines whether a special subdirectory, \SOEM\$, will be copied and certain sections will be used during setup. When the value is **Yes**, the \SOEM\$ directory will be copied during setup and the following sections will be used during setup. **No** will cause the \SOEM\$ directory and the following sections to be ignored.

**Note:** If the value of **OemPreinstall** is **Yes** the following sections will be included in the **UNATTENDED.TXT** file:

**[MassStorageDrivers]**  
**[KeyboardDrivers]**  
**[PointingDeviceDrivers]**  
**[OEMBootFiles]**  
**[OEM\_Ads]**

The following options are also enabled under the [Display] section:

**[Display]**  
**InstallDriver**  
**InfOption**  
**InfOptionFile**

See the "OEM Pre-Install Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

#### **NoWaitAfterTextMode**

Value: 0 | 1

Example Syntax:

NoWaitAfterTextMode = 1

This key determines whether the text mode portion of setup should automatically boot into GUI mode or not. It is only valid when **OemPreinstall = Yes**. The default behavior is to halt after text mode during a pre-installation.

---

**0** indicates that setup should halt after text mode and **1** indicates that setup should automatically reboot into GUI setup mode after text mode is complete.

**Note:** If `ExtendOemPartition` is equal to **1** (see below) then setup cannot automatically reboot into GUI setup mode after text mode is complete.

*This key can only be specified in the answer file, not in the UDF.*

### NoWaitAfterGuiMode

Value: **0 | 1**

Example Syntax:

`NoWaitAfterGuiMode = 1`

This key determines whether the GUI mode portion of setup should automatically reboot to the logon screen or not. It is only valid when **OemPreinstall = Yes**. The default behavior is to halt at the end of GUI mode setup.

**0** indicates that setup should halt after GUI mode and **1** indicates that setup should automatically reboot after GUI mode is complete.

*This key can only be specified in the answer file, not in the UDF.*

### FileSystem

Value: **ConvertNTFS | LeaveAlone**

Example Syntax:

`FileSystem = ConvertNTFS`

or

`FileSystem = LeaveAlone`

This key specifies whether the primary partition should be converted to NTFS or left alone. In general, partitions greater than 512MB should be converted to NTFS.

*This key can only be specified in the answer file, not in the UDF.*

### ExtendOemPartition

Value: **0 | 1**

Example Syntax:

`ExtendOemPartition = 1`

The **ExtendOemPartition** key is used to install Windows NT on a disk that is greater than 2GB. This key causes text mode setup to extend the partition on which the temporary Windows NT sources are located into any available unpartitioned space that physically follows it on the disk.

The temporary install source **MUST** be a primary partition and limited to 1024 cylinders only. Writing beyond the 1024 cylinder will cause the installation to fail.

---

**0** implies that the partition will not be extended and **1** indicates that it should be extended. When the value is **1**, the **FileSystem** key must be set to

#### **ConvertNTFS.**

**Note:** With this option, Windows NT setup cannot automatically reboot into GUI setup mode after text mode is complete. The user will be prompted to press a key to reboot the system.

*This key can only be specified in the answer file, not in the UDF.*

#### **ConfirmHardware**

Value: **Yes | No**

Example Syntax:

ConfirmHardware = yes

This key determines whether a user should manually confirm hardware and mass storage devices detected by the setup program.

**Yes** indicates that a user must manually confirm the hardware detected and **No** implies setup should install the detected devices.

For a complete unattended installation, this key should be set to **No**.

*This key can only be specified in the answer file, not in the UDF.*

#### **NtUpgrade**

Values: **Yes | No | Manual | Single**

Example Syntax:

NtUpgrade = No

This key determines whether a previous version of Windows NT Workstation or Server should be upgraded or not. It should be set to **Yes** in order to perform an upgrade.

**Yes** indicates that the detected Windows NT installation should be upgraded. If multiple installations are detected, the first installation found is upgraded.

**No** causes Windows NT setup to halt if a Windows NT installation is found. This is the desired value when OemPreinstall = Yes.

**Manual** implies that the user must specify which previous installation should be upgraded.

**Single** indicates that the upgrade should continue only if a single Windows NT installation is found. If multiple installations are found, the user must manually select which installation to upgrade.

*This key can only be specified in the answer file, not in the UDF.*

#### **Win31Upgrade**

Values: **Yes | No**

---

Example Syntax:  
Win31Upgrade = No

The Win31Upgrade key determines whether previous installations of Windows 3.x or Windows for Workgroups should be upgraded to Windows NT.

**Yes** indicates that the Windows installation should be upgraded and **No** means do not upgrade the installation if found.

*This key can only be specified in the answer file, not in the UDF.*

### OverwriteOemFilesOnUpgrade

Values: **Yes** | **No**

Example Syntax:  
OverwriteOemFilesOnUpgrade = Yes

This key determines whether OEM-supplied files that have the same name as Windows NT system files should be overwritten during an unattended upgrade or not.

**Yes** means overwrite the files and **No** means do not overwrite if found. The default behavior is to overwrite OEM-supplied files.

*This key can only be specified in the answer file, not in the UDF.*

### TargetPath

Values: \* | **<path name>** | **Manual**

Example Syntax:  
TargetPath = \Winnt

This key determines the installation directory in which Windows NT should be installed.

\* implies that setup should generate a unique directory name for the installation. This is usually WINNT.x where x is 0, 1, ..., etc.

**<path name>** is user-defined install directory. Do not use drive letters see the note below.

**Manual** indicates that setup should prompt the user to enter the install path.

**Note:** To place the WINNT directory on any drive other than C:, use the /T: command line switch with WINNT.EXE or WINNT32.EXE. The /T: switch places the \$WIN\_NT\$.~LS directory on the drive specified and Windows NT will by default install to the drive. The system files NTDETECT.COM, NTLDR, and BOOT.INI will still be placed on the active partition which is usually C:.

*This key can only be specified in the answer file, not in the UDF.*

### ComputerType

Values: **<hal description>** [, **Retail** | **OEM**]

---

Example Syntax:

```
ComputerType = "Standard PC - OEM","OEM"
```

This key indicates the type of Hardware Abstraction Layer (HAL) to be loaded by the Setup Loader, and installed by Text Mode Setup. If this key is not present, then setup will attempt to detect the type of computer and install the appropriate retail HAL. It is only valid when **OemPreinstall = Yes**.

The **<hal description>** string identifies the HAL to be installed. It must match one of the strings in the [Computer] section of TXTSETUP.SIF (for a retail HAL), or TXTSETUP.OEM (for an OEM HAL).

**RETAIL** informs setup that the HAL to be installed is part of the Windows NT product.

**OEM** indicates that the HAL to be loaded is OEM-supplied. If the HAL is OEM-supplied, the driver name must be listed in the [OemBootFiles] section. OemBootFiles are to be placed in the \$OEM\$\TEXTMODE directory.

See the "OEM Pre-Install Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **KeyboardLayout**

Value: **<layout description>**

Example Syntax:

```
KeyboardLayout = "US-International"
```

This key indicates the type of keyboard layout to be installed. If this key does not exist, setup will detect and install a keyboard layout.

**<layout description>** must match one of the right hand strings (in "") in the ["Keyboard Layout"] section of TXTSETUP.SIF

See the "OEM Pre-Install Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **[MassStorageDrivers]**

This section contains a list of SCSI drivers to be loaded by the Setup Loader, and installed during Text Mode Setup. If this section is missing or empty, setup will detect the hard disk controllers on the machine, and install the corresponding retail drivers. In most cases you should use this section blank.

**<mass storage driver description>**

Value: **RETAIL | OEM**

Example Syntax:

```
"SCSI Adapter OEM" = "OEM"
```

---

**<mass storage driver description>** This is a string that identifies the driver to be installed. It must match one of the strings defined in the right-hand side of the [SCSI] section of TXTSETUP.SIF (for a retail driver), or TXTSETUP.OEM (for an OEM driver). Multiple <mass storage driver descriptions> can be specified.

**RETAIL** indicates that the driver is part of the retail Windows NT product.

**OEM** indicates that the driver is OEM-supplied. If the value is OEM, the driver must also be listed in the [OemBootFiles] section of the UNATTENDED.TXT file.

**Note:** Use this section if setup is unable to detect your device correctly or if the driver for the device is not included in the retail box. If setup is able to detect your device properly, you do not need this section.

See the "OEM Pre-Install Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

## [DisplayDrivers]

This section contains a list of Display Drivers to be loaded by the Setup Loader, and installed during Text Mode Setup. It is only valid when OemPreinstall = Yes. If this section is missing or empty, setup will attempt to detect the display devices on the machine, and install the corresponding retail drivers. In most cases this section is not necessary.

**Note:** You can get the same functionality by using the settings in the [Display] section described later in this document.

### <display driver description>

Value: **RETAIL | OEM**

This is a string that identifies the driver to be installed. It must match one of the strings defined in the right-hand side of the [Display] section of TXTSETUP.SIF (for a retail driver), or TXTSETUP.OEM (for an OEM driver). Multiple <display driver descriptions> can be specified.

**RETAIL** indicates that the driver is part of the retail Windows NT product.

**OEM** indicates that the driver is OEM-supplied.

**Note:** Use this section if setup is unable to detect your device correctly or if the driver for the device is not included in the retail box. If setup is able to detect your device properly, you do not need this section.

See the "Third-Party Video Display Drivers and Setting Display Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

## [KeyboardDrivers]

This section contains a list of Keyboard Drivers to be loaded by the Setup Loader, and installed during Text Mode Setup. It is only valid when

---

OemPreinstall = Yes. If this section is missing or empty, setup will attempt to detect the keyboard devices on the machine, and install the corresponding retail drivers.

**<keyboard driver description>**

Value: **RETAIL | OEM**

This is a string that identifies the driver to be installed. It must match one of the strings defined in the right-hand side of the [Keyboard] section of TXTSETUP.SIF (for a retail driver), or TXTSETUP.OEM (for an OEM driver). Multiple <keyboard driver descriptions> can be specified.

**RETAIL** indicates that the driver is part of the retail Windows NT product.

**OEM** indicates that the driver is OEM-supplied.

**Note:** Use this section if setup is unable to detect your device correctly or if the driver for the device is not included in the retail box. If setup is able to detect your device properly you do not need this section.

See the "OEM Pre-Install Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

**[PointingDeviceDrivers]**

This section contains a list of pointing device drivers to be loaded by the Setup Loader, and installed during Text Mode Setup. It is only valid when OemPreinstall = Yes. If this section is missing or empty, setup will attempt to detect the pointing devices on the machine, and install the corresponding retail drivers.

**<pointing device driver description>**

Value: **RETAIL | OEM**

This is a string that identifies the driver to be installed. It must match one of the strings defined in the right-hand side of the [Mouse] section of TXTSETUP.SIF (for a retail driver), or TXTSETUP.OEM (for an OEM driver). Multiple <pointing device driver descriptions> can be specified.

**RETAIL** indicates that the driver is part of the retail Windows NT product.

**OEM** indicates that the driver is OEM-supplied.

**Note:** Use this section if setup is unable to detect your device correctly or if the driver for the device is not included in the retail box. If setup is able to detect your device properly you do not need this section.

See the "OEM Pre-Install Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

**[OEMBootFiles]**

This section is used to specify OEM-supplied boot files. It is only valid if

---

OemPreinstall = Yes and the files listed here have been placed in the \$OEM\$\Textmode directory of the Windows NT source files distribution share point.

### **TXTSETUP.OEM**

This file contains descriptions of all of the OEM-supplied drivers listed in this section. It also includes instructions on how to install these drivers. It must exist if this section is included.

#### **<hal file name>**

This <hal file name> maps to a HAL description that has been defined by the **ComputerType** key in the [Unattended] section of the UNATTENDED.TXT file.

#### **<scsi driver file name>**

The <scsi driver file name> maps to a mass storage driver description defined in the [MassStorageDriver] section of the UNATTENDED.TXT file. There can be multiple <scsi driver file names> listed in the [OemBootFiles] section.

See the "TXTSETUP.OEM and [OemBootFiles]" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **[OEM\_Ads]**

In most cases this section is not necessary. This section is for hardware manufacturers (OEMs) that want to customize the user interface.

#### **Banner**

Values: **<text string>**

Example Syntax:

Banner = "XYZ Corporation – Windows NT installation"

This key specifies a **<text string>** to be displayed in the upper left corner of the computer screen. The text must contain the "Windows NT" sub-string or else it will be ignored. To specify more than one line, you can separate the different lines using the \* character.

*This key can only be specified in the answer file, not in the UDF.*

#### **Logo**

Values: **<file name> [,<resource id>]**

Example Syntax:

Logo = Sample.bmp

This key specifies a bitmap to be displayed in the upper right corner of the screen. If this line has only one field, then it is assumed to a .bmp file located

---

in the \\${OEM\$} directory of the distribution share point. However, if two fields are specified, then the first field is the name of a DLL and the second is a base 10 number that represents the resource ID of the bitmap in the DLL. The DLL specified should be located in the \\${OEM\$} directory.

*This key can only be specified in the answer file, not in the UDF.*

### **Background**

Values: <file name> [,<resource id>]

Example Syntax:

Background = back.bmp

This key specifies a background bitmap to be displayed. If this line has only one field, then it is assumed to a .bmp file located in the \${OEM\$} directory of the distribution share point. However, if two fields are specified, then the first field is the name of a DLL and the second is a base 10 number that represents the resource ID of the bitmap in the DLL. The DLL specified should be located in the \${OEM\$} directory.

*This key can only be specified in the answer file, not in the UDF.*

### **[GuiUnattended]**

#### **OemSkipWelcome**

Value: 0 | 1

Example Syntax:

OemSkipWelcome = 1

This key is used to specify whether the introductory "Welcome to Windows NT Setup" page is skipped or not. Default behavior is to show the Wizard page.

*This option can be specified in either the answer file or the UDF.*

#### **OEMBlankAdminPassword**

Value: 0 | 1

Example Syntax:

OEMBlankAdminPassword = 1

This key is used to specify whether the user should see the Administrator Password Wizard page or not. Default behavior is to show the password page.

*This option can be specified in either the answer file or the UDF.*

#### **TimeZone**

Value: <text string>

Example Syntax:

TimeZone = "(GMT-05:00) Eastern Time (US & Canada)"

The TimeZone key determines the time zone of the computer. If the key is empty, the user is prompted to indicate a time zone.

The list of valid TimeZone strings is as follows:

```
(GMT) Greenwich Mean Time; Dublin, Edinburgh, London
(GMT+01:00) Lisbon, Warsaw
(GMT+01:00) Paris, Madrid
(GMT+01:00) Berlin, Stockholm, Rome, Bern, Brussels, Vienna
(GMT+02:00) Eastern Europe
(GMT+01:00) Prague
(GMT+02:00) Athens, Helsinki, Istanbul
(GMT-03:00) Rio de Janeiro
(GMT-04:00) Atlantic Time (Canada)
(GMT-05:00) Eastern Time (US & Canada)
(GMT-06:00) Central Time (US & Canada)
(GMT-07:00) Mountain Time (US & Canada)
(GMT-08:00) Pacific Time (US & Canada); Tijuana
(GMT-09:00) Alaska
(GMT-10:00) Hawaii
(GMT-11:00) Midway Island, Samoa
(GMT+12:00) Wellington
(GMT+10:00) Brisbane, Melbourne, Sydney
(GMT+09:30) Adelaide
(GMT+09:00) Tokyo, Osaka, Sapporo, Seoul, Yakutsk
(GMT+08:00) Hong Kong, Perth, Singapore, Taipei
(GMT+07:00) Bangkok, Jakarta, Hanoi
(GMT+05:30) Bombay, Calcutta, Madras, New Delhi, Colombo
(GMT+04:00) Abu Dhabi, Muscat, Tbilisi, Kazan, Volgograd
(GMT+03:30) Tehran
(GMT+03:00) Baghdad, Kuwait, Nairobi, Riyadh
(GMT+02:00) Israel
(GMT-03:30) Newfoundland
(GMT-01:00) Azores, Cape Verde Is.
(GMT-02:00) Mid-Atlantic
(GMT) Monrovia, Casablanca
(GMT-03:00) Buenos Aires, Georgetown
(GMT-04:00) Caracas, La Paz
(GMT-05:00) Indiana (East)
(GMT-05:00) Bogota, Lima
(GMT-06:00) Saskatchewan
(GMT-06:00) Mexico City, Tegucigalpa
(GMT-07:00) Arizona
(GMT-12:00) Enewetak, Kwajalein
(GMT+12:00) Fiji, Kamchatka, Marshall Is.
(GMT+11:00) Magadan, Solomon Is., New Caledonia
(GMT+10:00) Hobart
(GMT+10:00) Guam, Port Moresby, Vladivostok
(GMT+09:30) Darwin
(GMT+08:00) Beijing, Chongqing, Urumqi
(GMT+06:00) Alma Ata, Dhaka
(GMT+05:00) Islamabad, Karachi, Sverdlovsk, Tashkent
(GMT+04:30) Kabul
(GMT+02:00) Cairo
(GMT+02:00) Harare, Pretoria
(GMT+03:00) Moscow, St. Petersburg
```

### AdvServerType

Value: **SERVERNT** | **LANMANNT** | **LANSECNT**

Example Syntax: Sets value of Windows NT Server to PDC.

AdvServerType = LANMANNT

---

This key is only valid when installing Windows NT Server.  
**SERVERNT** indicates that the computer will be a stand-alone server.  
**LANMANNT** indicates that the computer will serve as a primary domain controller.  
**LANSECNT** indicates that the computer will be a backup domain controller.

#### DetachedProgram

Value: <detached program string>

Example Syntax:  
DetachedProgram = c:\myprogram.exe

The **DetachedProgram** key is used to indicate the path of the custom program that should run concurrently with the setup program. If the program requires any arguments, the **Arguments** key must be specified.

**Note:** In most cases, this option will not be used. Instead, **CMDLINES.TXT** is used which provides greater functionality.

See Chapter 5, "Customizing Windows NT" for further information.

#### Arguments

Value: <arguments string>

The **Arguments** key indicates that arguments or parameters accompany the custom program that should run concurrently with the setup program.

#### [UserData]

##### FullName

Value: <string>

Example Syntax:  
FullName = "your name"

The **FullName** key is used to specify the user's full name. If the key is empty or missing, the user is prompted to enter a name.

##### OrgName

Value: <string>

Example Syntax:  
OrgName = "your company name"

This key is used to specify an organization's name. If the **OrgName** key is empty or missing, the user is prompted to enter an organization name.

---

### ComputerName

Value: <string>

Example Syntax:

ComputerName = "MyComputer"

**Note:** There can be NO spaces in a computer name.

This key is used to specify the computer name. If the **ComputerName** key is empty or missing, the user is prompted to enter a computer name.

### ProductID

Value: <string>

Example Syntax:

ProductID = "29795-oem-0005995-49469" (this is an OEM type of product number)

ProductID = "xxx-xxxxxxx " (10 digit number is the retail format)

The **ProductID** is the CD key on the back of the jewel case that the Windows NT 4.0 CD came in.

### [LicenseFilePrintData]

This section is only valid when installing Windows NT Server.

#### AutoMode

Values: **PERSEAT** | **PERSERVER**

Example Syntax:

AutoMode = PERSEAT

The AutoMode key determines whether Windows NT Server is installed in per seat or per server license mode. If **AutoMode = PERSERVER**, the **AutoUsers** key must also be specified.

**PERSEAT** indicates that a client access license has been purchased for each computer that accesses the server.

**PERSERVER** indicates that client access licenses have been purchased for the server to allow a certain number of concurrent connections to the server.

If AutoMode is empty or missing, setup dialog boxes prompt the user to select the license mode.

*This key can only be specified in the answer file, not in the UDF.*

#### AutoUsers

Value: <decimal number>

Example Syntax:

---

AutoUsers = 10

This key is only valid if **AutoMode = PERSERVER**. The <decimal number> indicates the number of client licenses purchased for the server being installed.

*This key can only be specified in the answer file, not in the UDF.*

## [Display]

This section is used to specify display settings for the particular graphics device being installed. In order for this to work properly, the user must know what settings are valid for the graphics device under consideration. If the pre-specified settings are not valid, the user will be prompted to select them.

### ConfigureAtLogon

Value: **0 | 1**

Example Syntax:

ConfigureAtLogon = 1

This key is used to specify when the graphics devices are configured – during setup or after the first logon by an end user.

**0** implies configure during setup and **1** indicates that the device should be configured during the first logon by the user.

See the “Third-Party Video Display Drivers and Setting Display Options” section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### BitsPerPel

Value: **<valid bits per pixel>**

Example Syntax:

BitsPerPel = 8

This key specifies the **<valid bits per pixel>** for the graphics device being installed.

See the “Third-Party Video Display Drivers and Setting Display Options” section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### Xresolution

Value: **<valid x resolution>**

Example Syntax:

Xresolution = 640

This key specifies a **<valid x resolution>** for the graphics device being

---

installed.

See the “Third-Party Video Display Drivers and Setting Display Options” section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **Yresolution**

Value: **<valid y resolution>**

Example Syntax:

Yresolution = 480

This key specifies a **<valid y resolution>** for the graphics device being installed.

See the “Third-Party Video Display Drivers and Setting Display Options” section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **Vrefresh**

Value: **<valid refresh rate>**

Example Syntax:

Vrefresh = 60

This key specifies a **<valid refresh rate>** for the graphics device being installed.

See the “Third-Party Video Display Drivers and Setting Display Options” section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **Flags**

Value: **<valid flags>**

This key specifies **<valid flags>** for the graphics device being installed.

See the “Third-Party Video Display Drivers and Setting Display Options” section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **AutoConfirm**

Value: **0 | 1**

Example Syntax:

AutoConfirm = 1

The **AutoConfirm** key indicates whether the graphics device should be configured using pre-specified display settings or not.

---

**0** implies do not use the pre-specified settings and **1** indicates that the pre-defined settings should be used.

**AutoConfirm = 1** requires that all of the necessary parameters have been pre-specified in the UNATTEND.TXT file.

You can use the next three parameters instead of the [DisplayDriver], [OemBootFiles] sections and custom TXTSETUP.OEM files to install third-party video drivers. The drivers and files required by the video adapter should exist in the \$OEM\$\Display directory on the distribution share point.

See the "Third-Party Video Display Drivers and Setting Display Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **InstallDriver**

Value: **0 | 1**

Example Syntax:

```
InstallDriver = 1
```

This key specifies whether a third-party driver is being installed or not. If value is **0**, the **InfFile** and **InfOption** keys are skipped.

See the "Third-Party Video Display Drivers and Setting Display Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **InfFile**

Values: <inf file name 1>, <inf file name 2>, ...

This key specifies a list of INF file names for display drivers to be installed. You can specify only one INF per driver. E.g. s3.inf, matrox.inf, ...

See the "Third-Party Video Display Drivers and Setting Display Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **InfOption**

Values: <inf option 1>, <inf option 2>, ...

Example Syntax:

```
InfFile = s3.inf, matrox.inf  
InfOption . S3 765, Millenium 3D
```

See the "Third-Party Video Display Drivers and Setting Display Options" section in Chapter 2 for further information.

*This key can only be specified in the answer file, not in the UDF.*

### **[Modem]**

This section header is used to identify whether a modem should be installed or

---

not. It is used by Remote Access Services (RAS) to install a modem if the **DeviceType = Modem** in the list of RAS parameters. This section cannot be empty if you want to install modems using RAS in unattended mode.

### **InstallModem**

Value: **<modem parameter section>**

Example Syntax:

InstallModem = ModemSection

This key defines a section where modem install parameters are defined. The key must exist in order to install any modems.

*This key can only be specified in the answer file, not in the UDF.*

### **[<modem parameter section>]**

The modem parameter section lists the keys and values required to install a modem on a particular COM port. If the [<modem parameter section>] section is blank, RAS will do modem detection on its pre-configured ports and install any modems it finds.

**<COM port number>**

Values: **<Modem description>** [, **<Manufacturer>**, **<Provider>**]

Example Syntax:

Com2 = "Practical Peripherals PM288HC II V.34"

The **<COM port number>** key specifies the COM ports on which modems are installed. The COM port numbers must match ports configured or to be configured by the RAS installation.

**<Modem description>** must match a modem description in a MDMxxxxx.INF file that corresponds to the modem to be installed. This string must be enclosed in quotes.

The **<Manufacturer>**, **<Provider>** fields are optional fields that identify the manufacturer and provider of a particular modem in cases where the **<modem description>** string is not unique to a particular manufacturer.

To determine the Modem Description use Control Panel\Modem on a machine that has Windows NT and the modem installed.

1. Open Control Panel\Modems.
2. The Modem Description is on the General Tab under the column marked Modem.

You can also locate the Modem Description by searching the %systemroot%\INF\MDMxxxxx.INF files using keywords like the manufacturer with the FIND option on the Start Menu. You can also use FINDSTR.EXE, which is a text string search utility supplied with Windows NT.

---

*This key can only be specified in the answer file, not in the UDF.*

## **[Network]**

This section informs setup that Networking should be installed. If empty, the user will be presented with various error messages. If this section header is missing, network installation will be skipped.

To find out how to make a network component INF support unattended installation, see the "Building the UNATTEND.TXT Network Adapters" section for additional information.

Options in this section can be specified in a UDF or UNATTEND.TXT.

### **Attended**

Value: **Yes | No**

Example Syntax:

Attended = Yes

Omit this key if you want a hands-free setup. If you set this key to Yes, setup will prompt the user for Network Setup information.

### **JoinWorkgroup**

Value: **<workgroup name>**

Example Syntax:

JoinWorkgroup = MyGroup (No spaces in the Workgroup Name)

This key is used to define the workgroup in which the computer will participate.

### **JoinDomain**

Value: **<domain name>**

Example Syntax:

JoinDomain = MyDomain

This key is used to define the domain in which the computer will participate.

### **CreateComputerAccount**

Value: **<username, password>**

Example Syntax:

CreateComputerAccount = jimh, pw01

**Note:** The clear text password is visible to anyone that has access to the UNATTEND.TXT file and it may be visible in the temporary folder that is

---

created during setup. For this reason, it is recommended that you do not use this key. Instead, it is recommended that an administrator setup the Windows NT Workstation computer accounts in advance or create a new user account that is restricted to the right to create computer accounts in the domain so that Domain Administrative privileges are protected.

### **InstallDC**

Value: <domain name>

Example Syntax:

InstallDC = MyDomain

This key is used to specify the name of a domain to be installed. It is only valid when installing a primary or backup domain controller and the AdvServerType key has been set accordingly.

### **DetectAdapters**

Value: <detect adapters section> | ""

Example Syntax:

DetectAdapters = Netcards

This key is used to detect network adapter cards installed on a computer. Either this key or the **InstallAdapters** key must exist in order to install network cards. If the value is "", then the first card detected will be installed.

See "Building the UNATTEND.TXT Network Adapters Section" for additional information.

### **InstallAdapters**

Value: <install adapters section>

Example Syntax:

InstallAdapters = Intelcards

This key defines a section in which the network adapters to be installed are listed.

See "Building the UNATTEND.TXT Network Adapters Section" for additional information.

### **InstallProtocols**

Value: <protocols section>

Example Syntax:

InstallProtocols = Protocols

---

This key defines a section in which the network protocols to be installed are listed.

### **InstallServices**

Value: <services section>

Example Syntax:  
InstallServices = Services

This key defines a section in which the network services to be installed are listed.

### **InstallInternetServer**

Value: <internet information server parameters>

Example Syntax:  
InstallInternetServer = ISSParams

This key defines a section in which parameters for installing the Internet Information Server (IIS) are listed. During installation on Windows NT Server, IIS is installed by default.

### **DoNotInstallInternetServer**

Value: **Yes | No**

Example Syntax:  
DoNotInstallInternetServer = Yes

The presence of this key disables the default installation of IIS on Windows NT Server. The value assigned to it is irrelevant.

## **[<Detect Adapters Section>]**

The name of this section is the value of the **DetectAdapters** key described in the **[Network]** section above.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

Example is [Netcards] defined on the example DetectAdapters = line in the previous section.

### **DetectCount**

Value: <number of detection attempts>

Example Syntax:  
DetectCount = 1

Indicates the number of detection attempts setup should make.

---

See "Building the UNATTEND.TXT Network Adapters" section for additional information.

**LimitTo**

Value: **<netcard inf option>**

Example Syntax:

LimitTo = IEEPRO

IEEPRO for the Intel EtherExpress Pro as defined in the OEMNADEP.INF [options] section. This file is found in the WINNT\SYSTEM32 directory.

This key specifies a list of netcard .INF options to which the detection should be limited. The netcard .INF options for particular cards can be found in the [Options] section of the corresponding OEMNADxx.INF file.

See "Building the UNATTEND.TXT Network Adapters" section for additional information.

**<Netcard Inf option>**

Value: **<netcard parameter section>**

Example Syntax:

IEEPROparm

This key points setup to the section that contains descriptions for a particular network adapter card. The <netcard inf options> for particular cards can be found in the [Options] section of the corresponding OEMNADxx.INF files.

See "Building the UNATTEND.TXT Network Adapters" section for additional information.

**[<Install Adapters Section>]**

*Options in this section can be specified in a UDF or UNATTEND.TXT.*

**<Netcard Inf option>**

Value: <netcard parameter section>

Example Syntax:

netcards = IEEPRO

This key points setup to the section that contains descriptions for a particular network adapter card. The <netcard inf options> for particular cards can be found in the [Options] section of the corresponding OEMNADxx.INF files.

See "Building the UNATTEND.TXT Network Adapters" section for additional information.

---

## [<netcard parameter section>]

Example: [IEEPRO]

This section contains the parameters for a particular network adapter card whose <netcard inf option> has been specified in the [<Detect Adapters Section>] or the [<Install Adapters Section>] of the UNATTEND.TXT file.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

See "Building the UNATTEND.TXT Network Adapters" section for additional information.

## [<Protocols Section>]

This section contains a list of .INF options for network protocols and the corresponding UNATTEND.TXT file section in which the parameters for the particular protocol are list.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### NBF

Value: <Netbeui Parameters>

Example Syntax:

NBF = NetbeuiParams

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [NetbeuiParams]. This section must exist whether the section requires values or not. NetBEUI does not have any values because there are no parameters to configure.

### NWLNKIPX

Value: <IPX Parameters>

Example Syntax:

NWLINK = IPXParams

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [IPXParams]. This section must exist whether the section requires values or not. IPX does not have any values because there are no parameters to configure.

### TC

Value: <Tcpip Parameters>

Example Syntax:

TC = TCPParams

---

This key indicates that TCP/IP should be installed in unattended mode. The corresponding parameter section must exist or setup will fail.

### **DLC**

Value: **<DLC Parameters>**

Example Syntax:  
DCL = DLCParams

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [DLCParams]. This section must exist whether the section requires values or not. DLC does not have any values because there are no parameters to configure.

### **RASPPTP**

Value: **<RASPPTP Parameters>**

Example Syntax:  
RASPPTP = RASPPTPParams

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [RASPPTPParams]. This section must exist whether the section requires values or not. Point-to-Point Protocol does not have any values because there are no parameters to configure.

### **STREAMS**

Value: **<STREAMS Parameters>**

Example Syntax:  
STREAMS = STREAMSParams

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [STREAMSParams]. This section must exist whether the section requires values or not. Streams does not have any values because there are no parameters to configure.

### **[<NetBeui Parameters>]**

This parameter is left empty since NetBEUI does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### **[<IPX Parameters>]**

This parameter is left empty since IPX does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

## [<Tcpip Parameters>]

TCP/IP parameters only support the static IP configuration for a single adapter. If configuring a multi-homed system DHCP has to be used for a keyless install. If static addressing is used the second adapter has to manually configured during setup.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### DHCP

Value: **Yes | No**

Example Syntax:

DHCP = Yes

This key is used to specify whether **DHCP** should be used or not.

### ScopeID

Value: **<scope ID>**

Example Syntax:

ScopeID = ScopeA

See the Windows NT Resource Kit for a definition of ScopeID.

This key is used to specify the computer's scope identifier if required on a network that used NetBIOS over TCP/IP.

**If DHCP = No, the following keys must be specified:**

### IPAddress

Value: **<ip address>**

Example Syntax:

IPAddress = 192.124.254.2

Used to specify the IP address for the computer.

**Note:** You will need the updated TCPCFG.DLL from the Windows NT 4.0 Service Pack 2 to use octets that have a zero. To install the TCPCFG.DLL:

1. Rename the TCPCFG.DL\_ to TCPCFG.ORG on the distribution server.
2. Copy the TCPCFG.DLL from the Windows NT 4.0 Service Pack 2 CD on to the distribution server.

### Subnet

Value: **<subnet address>**

Example Syntax:

---

Subnet = 255.255.255.0

Specifies the subnet mask address.

**Gateway**

Value: <gateway address>

Example Syntax:

Gateway = 192.124.254.3

Identifies the default gateway address for the computer.

**DNSServer**

Value: <IP Addresses>

Example Syntax:

DNSServer = 192.124.254.4

Used to specify up to 3 DNS servers.

**WINSPRIMARY**

Value: <IP Address>

Example Syntax:

WINSPRIMARY = 192.135.144.5

Used to specify the IP address of the primary WINS server.

**WINSSecondary**

Value: <IP address>

Example Syntax:

WINSSecondary = 192.135.144.6

Used to specify the IP address of the secondary WINS server.

**DNSName**

Value: <DNS domain name>

Example Syntax:

DNSName = evv.mjn.bms.com

This key is used to specify the DNS domain name.

---

### [<DLC Parameters>]

This parameter is left empty because DLC does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### [<RASPTP Parameters>]

This parameter is left empty because Point-to-Point Protocol does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### [<STREAMS Parameters>]

This parameter is left empty because Streams does not require additional parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### [<Services Section>]

#### SNMP

Value: <Snp Parameters>

Points to <Snp Parameters>

*Options in this section can specified in a UDF or UNATTEND.TXT.*

#### RAS

Value: <RAS Parameters>

Example Syntax:

RAS = RasParams

Points to <RAS Parameters>

When installing RAS using the UI, a file named SERIAL.INI is created. This file will not be created during the installation of RAS using Windows NT setup script. If you are using SYSDIFF, the SERIAL.INI file can be added to the \$OEM\$\\$\$\SYSTEM32\RAS directory on your distribution server.

See Chapter 1, "Getting Started" for more information on the \$OEM\$ directory. File should contain a semicolon ";" at the very least to make the file greater than 1 byte.

#### NWWKSTA

Value: <NetWare Client Parameters>

Example Syntax:

NWWKSTA = NWPParams

---

Points to <NetWare Client Parameters>

### NETMON

Value: <NetMon Parameters>

Example Syntax:

NETMON = NETMONParams

Points to <NetMon Parameters>

This will install the Network Monitor and agent provided with Windows NT 4.0.

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [NETMONParams]. This section must exist whether the section requires values or not. Network Monitor does not have any values because there are no parameters to configure.

### STCPIP

Value: <STCPIP Parameters>

Example Syntax:

STCPIP = STCPIPParams

Points to <STCPIP Parameters>

This will install simple TCP/IP provided with Windows NT 4.0.

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [STCPIPParams]. This section must exist whether the section requires values or not. Simple TCP/IP does not have any values because there are no parameters to configure.

### SAP

Value: <SAP Parameters>

Example Syntax:

SAP = SAPPParams

Points to <SAP Parameters>

This will install SAP provided with Windows NT 4.0.

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [SAPPParams]. This section must exist whether the section requires values or not. SAPPParams does not have any values because there are no parameters to configure.

---

## TCPPRINT

Value: <TCPPRINT Parameters>

Example Syntax:

TCPPRINT = TCPPRINTParams

Points to <TCPPRINT Parameters>

This will install TCP/IP Printing provided with Windows NT 4.0.

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [TCPPRINTParams]. This section must exist whether the section requires values or not. TCP/IP Printing does not have any values because there are no parameters to configure.

## DHCP

Value: <DHCP Parameters>

Example Syntax:

DHCP = DHCPParams

Points to <DHCP Parameters>

This will install DHCP Server provided with Windows NT 4.0 Server only.

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [DHCPParams]. This section must exist whether the section requires values or not. DHCP does not have any values because there are no parameters to configure.

## DNS

Value: <DNS Parameters>

Example Syntax:

DNS = DNSParams

Points to <DNS Parameters>

This will install DNS Server provided with Windows NT 4.0 Server only.

**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [DNSParams]. This section must exist whether the section requires values or not. DNS does not have any values because there are no parameters to configure.

## WINS

Value: <WINS Parameters>

---

Example Syntax:  
WINS = WINSParams

Points to **<WINS Parameters>**

This will install WINS Server provided with Windows NT 4.0 Server only.  
**Note:** You must define a corresponding section for this value or setup will fail. In this example the section is [WINSParams]. This section must exist whether the section requires values or not. WINS does not have any values because there are no parameters to configure.

### [<NetWare Client Parameters>]

*Options in this section can specified in a UDF or UNATTEND.TXT.*

#### **IDefaultLocation**

Value: **<server\_location>**

Example Syntax:  
!DefaultLocation = NWServer

Example of NDS:

```
!DefaultLocation = "**ABC\MARKETING.US"
```

The **!DefaultLocation** key identifies the default logon server for the NetWare client.

#### **IDefaultScriptOptions**

Values: **0 | 1 | 3**

Example Syntax:  
!DefaultScriptOptions = 1

This key defines the default action to perform with scripts.  
**0** will cause scripts to be ignored, **1** causes NetWare 3.x level scripts to be run only, and **3** implies that either NetWare 3.x or NetWare 4.x level scripts will be run.

### [<Snmp Parameters>]

*Options in this section can specified in a UDF or UNATTEND.TXT.*

#### **Accept\_CommunityName**

Value: **<community names>**

---

Example Syntax:

Accept\_CommunityName = Name1,Name2,Name3

This key is used to specify a maximum of three community names that the computer, on which the SNMP service is running, accept traps from. The **<community names>** are separated by commas.

### Send\_Authentication

Value: **Yes | No**

Example Syntax:

Send\_Authentication = Yes

This key indicates whether an authentication trap should be sent when an unauthorized community or host requests information.

### Any\_Host

Value: **Yes | No**

This key specifies whether the computer, on which the SNMP service is being installed, should accept SNMP packets from any host or not.

### Limit\_Host

Values: **<host names>**

Example Syntax:

Limit\_Host = name1,name2,name3

A maximum of three **<host names>** can be specified separated by commas. This key is valid when **Any\_Host = No**.

### Community\_Name

Value: **<community name>**

Example Syntax:

Community\_Name = name

Indicates the **<community name>** for the computer.

### Traps

Values: **<IP addresses> | <IPX addresses>**

Example Syntax:

Traps = 192.124.134.5

This key is used to specify a maximum of three IP or IPX addresses to

---

which traps should be sent.

#### **Contact\_Name**

Value: <name>

Example Syntax:

Contact\_Name = name

This key is used to specify the computer user's name.

#### **Location**

Value: <computer location>

Example Syntax:

Location = Building2

This key is used to specify the physical location of the computer.

#### **Service**

Values: **Physical, Applications, Datalink, Internet, End-to-End**

Example Syntax:

Service = Physical,Applications,Datalink,Internet,End-to-End

Any combination of the five SNMP services listed here can be specified as values. They must, however, be separated by commas.

**Note:** SNMPTRAP.EXE service is set to manual start. A registry script can be used to set the service to automatic. See Chapter 5, "Customizing Windows NT" to learn about modifying the registry during setup.

### **[<RasParameters>]**

#### **PortSections**

Values: <port section name>

Example Syntax:

PortSections = ComPorts

This key is used to define a port section name. Multiple port section names can be specified, but they must be separated by commas ",". See the [<port section names>] definition below.

#### **DialoutProtocols**

Value: **TCP/IP | IPX | NETBEUI | ALL**

---

Example Syntax:  
DialoutProtocols = ALL

**ALL** implies all installed protocols.  
The remaining parameters in this **<RasParameters>** section only apply to RAS Server installation.

**DialinProtocols**  
Value: **TCP/IP | IPX | NETBEUI | ALL**

Example Syntax:  
DialinProtocols = ALL

**ALL** implies all installed protocols.

**NetBEUIClientAccess**  
Value: **Network | ThisComputer**

Example Syntax:  
NetBEUIClientAccess = ThisComputer

Default is Network.

**TcpIpClientAccess**  
Value: **Network | ThisComputer**

Example: Syntax:  
TcpIpClientAccess = ThisComputer

Default is Network.

**UseDHCP**  
Value: **YES | NO**

Example Syntax:  
UseDHCP = No

Default is Yes.

**StaticAddressBegin**  
Value: **<IP\_address>**

Example Syntax:  
StaticAddressBegin = XXX.XXX.XXX.XXX (where X is the ip address range)

---

This key is required if **UseDHCP = NO**.

#### **StaticAddressEnd**

Value: **<IP\_address>**

Example Syntax:

StaticAddressEnd = XXX.XXX.XXX.XXX (where X is the ip address range)

This key is required if **UseDHCP = NO**.

#### **ExcludeAddress**

Value: **<IP\_address1 - IP\_address2>**

Example Syntax:

ExcludeAddress = XXX.XXX.XXX.XXX -YYY.YYY.YYY.YYY (where X and Y are IP ranges to exclude)

This key is used to *exclude a range* of IP addresses when a range of IP addresses is being assigned manually. It requires that **StaticAddressBegin** and **StaticAddressEnd** be specified already.

#### **ClientCanRequestIPAddress**

Value: **YES | NO**

Example Syntax:

ClientCanRequestIPAddress = Yes

Default is **No**.

#### **IpxClientAccess**

Value: **Network | ThisComputer**

Example Syntax:

IpxClientAccess = ThisComputer

Default is Network.

#### **AutomaticNetworkNumbers**

Value: **YES | NO**

Example Syntax:

AutomaticNetworkNumbers = No

Default is **YES**.

---

### **NetworkNumberFrom**

Value: <IPX\_net\_number>

Example Syntax:

NetworkNumberFrom = number (where number is the hex number defined below)

Valid numbers range from 1 to 0xFFFFFFFF. This key is required if **AutomaticNetworkNumbers = NO**.

### **AssignSameNetworkNumber**

Value: **YES | NO**

Example Syntax:

AssignSameNetworkNumber = No

Default is **YES**.

### **ClientsCanRequestIpxNodeNumber**

Value: **YES | NO**

Example Syntax:

ClientsCanRequestIpxNodeNumber = Yes

Default is **NO**.

## **[<port section name>]**

### **PortName**

Value: **COM1 | COM2 | COM3-COM25**

Example Syntax:

PortName = COM2

This key indicates the names of the ports to be configured in a particular port section.

### **DeviceType**

Value: **Modem**

Example Syntax:

DeviceType = Modem

This key indicates the type of device RAS should install. Today, the only

---

available device type is a modem.

### **PortUsage**

Value: **DialOut** | **DialIn** | **DialInOut**

Example Syntax:

PortUsage = DialInOut

The **PortUsage** key defines the dialing properties for the ports being configured.

### **[<NETMON Parameters>]**

This parameter is left empty because Network Monitor and the agent does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### **[<STCPIP Parameters>]**

This parameter is left empty because simple TC/PIP does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### **[<SAP Parameters>]**

This parameter is left empty because SAP does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### **[<TCPPRINT Parameters>]**

This parameter is left empty because TCP/IP Printing does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### **[<DHCP Parameters>]**

This parameter is left empty because DHCP does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### **[<DNS Parameters>]**

This parameter is left empty because DNS does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

### **[<WINS Parameters>]**

This parameter is left empty because WINS does not require any extra parameters to install.

*Options in this section can specified in a UDF or UNATTEND.TXT.*

---

## [<internet information server section>]

This section contains parameters for installing the Internet Information Server (IIS). A value of 1 for each of the parameters below implies the component should be installed, whereas a value of 0 implies the component should not be installed.

### **InstallINETSTP**

Value: **0 | 1**

Example Syntax:

InstallINETSTP = 1

Specifies whether Internet Services will be installed. Default is 1.

### **InstallADMIN**

Value: **0 | 1**

Example Syntax:

InstallADMIN = 1

Specifies whether the Internet Service Manager will be installed.

### **InstallFTP**

Value: **0 | 1**

Example Syntax:

InstallFTP = 1

Specifies whether the FTP Service will be installed.

### **FTPRoot**

Value: **<ftp root directory>**

Example Syntax:

FTPRoot = C:\FTP

Specifies the virtual root for the FTP service.

### **InstallWWW**

Value: **0 | 1**

Example Syntax:

InstallWWW = 1

Specifies whether the WWW Service will be installed.

---

### **WWWRoot**

Value: **<www root directory>**

Example Syntax:

WWWRoot = c:\WWW

Specifies the virtual root for the WWW service.

### **InstallGOPHER**

Value: **0 | 1**

Example Syntax:

InstallGOPHER = 1

Specifies whether the Gopher Service will be installed.

### **GopherRoot**

Value: **<gopher root directory>**

Example Syntax:

GopherRoot = C:\GRoot

Specifies the virtual root for the Gopher service.

### **InstallDir**

Value: **<internet services install directory>**

Example Syntax:

InstallDir = C:\IServ

Specifies the installation directory for all components of Internet Services.

### **InstallW3SAMP**

Value: **0 | 1**

Example Syntax:

InstallW3SAMP = 1

Specifies whether World Wide Web sample files will be installed.

### **InstallHTMLA**

Value: **0 | 1**

Specifies whether the HTML form of the Internet Service Manager will be

---

installed.

**GuestAccountName**

Value: <name>

Example Syntax:

GuestAccountName = name

This key is used to define the anonymous user name used in the WWW, FTP, and GOPHER services.

**GuestAccountPassword**

Value: <password string>

Example Syntax:

GuestAccountPassword = password

This is used to create the guest account password. If it is not defined, IIS will create a random string for the guest account.

## Sample UNATTEND.TXT Files

### Sample 1

This UNATTEND.TXT file is a sample of what might be used in a corporate environment. See the details referenced in this chapter for a description of each of the keys listed here.

```
[Unattended]
ConfirmHardware = no
NtUpgrade = no
Win31Upgrade = no
Win31Upgrade = no
TargetPath = WINNT
OemPreinstall = Yes
FileSystem= ConvertNTFS
OemSkipEula = yes
NoWaitAfterGUIMode = 1

[GuiUnattended]
OemSkipWelcome = 1
OemBlankAdminPassword = 1
TimeZone = "(GMT-06:00) Central Time (US & Canada)"

[UserData]
FullName = "John Doe"
OrgName = "Widgets-Are-Us"
ComputerName = "Computer1"
```

```
ProductId = "123-4567890"

[Display]
ConfigureAtLogon = 0
BitsPerPel = 8
XResolution = 640
YResolution = 480
VRefresh = 60

[Network]
InstallServices = ServicesList
InstallAdapters = AdaptersList
InstallProtocols = Protocols
JoinDomain = SEATTLE
CreateComputerAccount = jimh, pw01

[AdaptersList]
EE16 = EE16Params

[EE16Params]
BusType = 1
Transceiver = 3
BusNumber = 0
IoChannelReady = 2
IoBaseAddress = 784
InterruptNumber = 10

[ServicesList]
NWKSTA = NWCPParams
NETMON = InstallNetMon
STCPIP = InstallSimpleTCP
TCPPRINT = InstallTCPPrint

[NWCPParams]
!DefaultLocation = NWServer
!DefaultScriptOptions = 3

[InstallNetMon]

[InstallSimpleTCP]

[InstallTCPPrint]

[Protocols]
NWLNKIPX = NWLinkParams
```

```
TC = TCPIP
NBF = NetBeuiParams
DLC = DLCParams

[NWLinkParams]

[TCPIP]
DHCP = no

Gateway = 165.89.91.3
Subnet = 255.255.255.0
WINSPrimary = 165.89.91.239
WINSSecondary = 165.89.164.218
DNSName = msd.msn.ehq.com
DNSServer = 165.89.90.228, 165.89.91.241, 165.89.1.118

[NetBeuiParams]

[DLCParams]
```

### Sample 2

This sample UNATTEND.TXT file installs Windows NT Workstation 4.0 and provides examples of all of the options you can specify when OemPreinstall = Yes. This UNATTEND.TXT file would be used if you are installing Windows NT Workstation 4.0 on identically configured systems – for example if you are a hardware manufacturer you would specify all of the devices in the computer. In most cases you would not know this level of detail about the installed base of your organization. If you do not know this level of detail, it is recommended that you omit those sections so Window NT setup will detect the devices installed in your systems. These options should only be used if a device is not detected by Window NT 4.0 setup.

```
[Unattended]
ConfirmHardware = no
NtUpgrade = no
Win31Upgrade = no
TargetPath = winnt
OverwriteOemFilesOnUpgrade = no
OemPreinstall = yes
ComputerType = "Standard PC - OEM", "OEM"
KeyboardLayout = "US-International"
FileSystem = LeaveAlone

ExtendOEMPartition = 0
NoWaitAfterTextmode = 1
```

```
NoWaitAfterGuiMode = 1

[MassStorageDrivers]
"IDE CD-ROM(ATAPI 1.2)/Dual-channel PCI IDE Controller" =
"Retail"
"New OEM SCSI for Adaptec 154x/164x" = "OEM"
[DisplayDrivers]
"OEM Display Driver 1" = OEM
[KeyboardDrivers]
"XT, AT, or Enhanced Keyboard (83-104 keys)" = RETAIL
[PointingDeviceDrivers]
"Microsoft Mouse Port Mouse (includes BallPoint)" = OEM
[OEMBootFiles]
hal.dll
hal486c.dll
elliott.sys
TXTSETUP.OEM

[OEM_Ads]
Banner = "DEC's Windows NT Setup"
Background = test.bmp

[GuiUnattended]
AdvServerType = LANMANNT
TimeZone = "(GMT-08:00) Pacific Time (US & Canada); Tijuana"

[UserData]
FullName = "User Name"
OrgName = "Microsoft"
ComputerName = OEM_Computer
ProductId = "29795-oem-0005995-49469"
[LicenseFilePrintData]
AutoMode = PerSeat

[Display]
ConfigureAtLogon = 0
BitsPerPel = 32
Xresolution = 640
Yresolution = 480
Vrefresh = 60
AutoConfirm = 1

[Network]
DetectAdapters = DetectParms
InstallProtocols = SelectedProtocolsList
```

---

```
InstallServices = SelectedServicesList
InstallDC = OEM_Domain
```

```
[DetectParms]
DetectCount = 1
LimitTo = ELNK3ISA509
ELNK3ISA509 = ENKIIIParms
```

```
[ENKIIIParms]
Transceiver = 0
InterruptNumber = 7
IoBaseAddress = 768
```

```
[SelectedProtocolsList]
TC = TCPIPParms
NWLNKIPX = IPXParms
NBF = NetBeuiParms
```

```
[TCPIPParms]
DHCP = yes
```

```
[NetBeuiParms]
```

```
[IPXParms]
```

```
[SelectedServicesList]
RAS = RemoteAccessParameters
```

```
[RemoteAccessParameters]
PortSections = DialoutSection
DialoutProtocols = TCP/IP
```

```
[Modem]
InstallModem = ModemSection
```

```
[DialoutSection]
PortName = COM2
DeviceType = Modem
PortUsage = DialOut
```

```
[ModemSection]
Com2 = "Sportster 28800-33600 External"
```

---

## Configuring Network Adapters and Setup Information Files for Automated Installation

### Building the Windows NT Setup Script Network Adapters Section

If the network adapter is auto-detected during a manual installation, the DetectAdapters options maybe used. The following is an example of the sections required in a Windows NT setup script to detect two different network adapters. The network adapter option name to the right of the LimitTo is in the "Network Adapter Option Name" section later in this chapter. The network adapter option name can also be found in the options section of each netcard INF.

---

*Note: Not all network adapters are detected. In cases where the network adapter is not detected, you refer to the next example.*

---

#### *Example 1*

This example relies on Windows NT setup to detect the network adapters.

```
[Network]
DetectAdapters = DetectAdaptersSection

[DetectAdaptersSection]
DetectCount = 2
LimitTo = DECETHERWORKSTURBO, EE16
;
; The parameter section is not required if the parameters
; detected are desired. If a particular parameter
; needs to be changed, then use the adapter parameters
; option.
;
DECETHERWORKSTURBO = DECETHERWORKSTURBOParamSection
EE16 = EE16ParamSection

[DECETHERWORKSTURBOParamSection]
InterruptNumber = 5
IOBaseAddress = 1
MemoryMappedBaseAddress = 851968
BusType = 1
BusNumber = 0

[EE16ParamSection]
InterruptNumber = 5
IOChannelReady = 0
Transceiver = 0
IOBaseAddress = 768
```

```
BusType = 1
BusNumber = 0
```

The use of the DectectAdapters provides the ability to have one answer file for multiple network adapters.

### *Example 2*

If you find that the adapter is not auto-detected during a manual installation of Microsoft Windows NT, the Install Adapters options can be used to automatically specify the installation of network adapters. In examples two and three below, you will find the information required to specify a network adapter.

The network adapter option name under the [SelectedAdaptersSection] is in the "Network Adapter Option Name" section later in this chapter.

```
[Network]
InstallAdapters = SelectedAdaptersSection

[SelectedAdaptersSection]
EE16 = EE16ParamSection

[EE16ParamSection]
InterruptNumber = 5
IOChannelReady = 0
Transceiver = 0
IOBaseAddress = 768
BusType = 1
BusNumber = 0
```

### *Example 3*

```
[Network]
InstallAdapters = SelectedAdaptersSection

[SelectedAdaptersSection]
DECETHERWORKSTURBO = DECETHERWORKSTURBOParamSection

[DECETHERWORKSTURBOParamSection]
InterruptNumber = 5
IOBaseAddress = 1
MemoryMappedBaseAddress = 851968
BusType = 1
BusNumber = 0
```

## **Determining Network Adapters Parameters in the UNATTEND.TXT**

Determining an adapter's parameters can be accomplished via the

REGEDT32.EXE utility supplied with Windows NT 4.0. Every installed network adapter has two keys in HKLM\SYSTEM\CurrentControlSet\Services\  
<ProductName> and <ServiceName> that specify the option name for the card and the instance of the card respectively.

To find out the value of <ProductName> and <ServiceName> check the following registry values:

HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\NetworkCards\1

ProductName: REG\_SZ:<adapter\_string>

ServiceName: REG\_SZ:<adapter\_string>

The string assigned to ServiceName is what is needed to locate the parameters, for instance, of the network adapter installed. These parameters can be located in the registry under

HKLM\SYSTEM\CurrentControlSet\Services\<<ServiceName>\Parameters

All numeric values found in the parameters key for the adapter have been converted from hex values to decimal values for the unattend file. The names of the parameters and non-numeric values can be directly translated to the unattend file.

### Overview of a Network Component .INF that Supports Unattended Installation

For a network component (adapters, services, protocols) to be installed using the Unattended Setup mechanism, the component's OEMNxxx.INF files must check certain INF symbols and react accordingly. These symbols, STF\_UNATTENDED, STF\_GUI\_UNATTENDED, and STF\_UNATTENDED\_SECTION, inform Windows NT Setup whether the installation is being run in unattended mode or not. In order to perform this modification, you must be familiar with the Windows NT style INF formats.

To find out which mode setup is running in, your .INF file must contain code similar to the following:

```
ifstr(I) $(!STF_GUI_UNATTENDED) == "YES"
    ifstr(I) $(!AutoNetInterfaceType) != ""
        set BusInterfaceType = $(!AutoNetInterfaceType)
    else
        set BusInterfaceType = 1
    endif
ifstr(I) $(!AutoNetBusNumber) != ""
    set BusNumber = $(!AutoNetBusNumber)
else
    set BusNumber = 0
```

```
endif
goto adapterverify
endif
```

If setup is running in Unattended mode, then you must skip the INF commands that brings up UI, dialogs, etc. These commands are usually of the form:

```
read-syms FileDependentDlg(!STF_LANGUAGE)
ui start "InputDlg"
```

Once the network component INF has verified that setup is running in unattended mode, it then can use the STF\_UATTENDED symbol to get the unattended filename. The STF\_UATTENDED\_SECTION symbol will be set by setup so the .INF code can read it as needed.

### *Setting Parameters for Network Adapter Cards*

If the component is an adapter card, the .INF file will have to pass the parameter section for the card through the AddDefaultNetCardParameters in the UTILITY.INF file that is shipped with Windows NT. To do this, the following key will be specified:

```
ifstr(I) $(!STF_GUI_UNATTENDED) == "YES"
Shell $(Utility.Inf), AddDefaultNetCardParameters, $(KeyParams)
endif
```

This will cause the parameters listed in the UNATTEND.TXT file to be written to the registry.

Note that the code above must be implemented after the default network adapter parameters have been written to the registry. See the OEMNADE2.INF file included in Windows NT 4.0.

### *Verifying and Testing Component .INFs*

Network Adapter Cards:

1. Install the particular network card on Windows NT in attended mode.
2. Check the parameters it installs in the registry  
HKLM\System\CurrentControlSet\Services\<adapter\_name>\Parameters  
where x is an instance of the card. It is usually 1.
3. Note the values of the parameters listed.
4. If the parameter values are not of the type REG\_DWORD, the driver for the adapter must be modified to generate REG\_DWORD values. Note that the NetworkAddress parameter is of the type REG\_SZ.
5. Install the network adapter card in unattended mode using the new INF you created.
6. Check the registry again to verify that the parameters were installed

correctly.

### Verifying the OEM File for STF\_GUI\_UNATTENDED

With the adapter(s) properly configured in the unattend file(s), you must verify that the appropriate OEM file contains correct unattend code.

Using the listing of Network Adapters provided in the "Network Adapter Option Name" section later in this chapter, locate the correct OEMNADxx.INF filename. In the case of the Intel Ether Express 16 LAN Adapter the OEMNADIN.INF file is used. Using a text editor open the OEMNADIN.INF and search for the string "STF\_GUI\_UNATTENDED". If the string is found then the file has been made "unattend aware". See the "Network Adapter Option Name" section for a list of INFs that support unattended setup.

If the adapter is not normally auto-detected or is a third-party network adapter not supported out of the box by Windows NT 4.0, use a text editor to open OEMNADZZ.INF and search for the option name. Once the option name is found, locate the appropriate directory listing which will point to a directory containing the OEMSETUP.INF file. Using a text editor, open the OEMSETUP.INF and search for STF\_GUI\_UNATTENDED. If the string is found, then the file has been made "unattend aware". See the "Network Adapter Option Name" section for a list of .INFs that support unattended setup.

If the STF\_GUI\_UNATTENDED is not found in the file, you will have to add additional code to make the INF file "unattend aware". If the INF is located in \1386 directory, then Microsoft has included the code to make the INF "unattend aware". If the OEMSETUP.INF is used and is part of the \1386\DRVLIB.NIC directory, the hardware vendor provided the INF file and is responsible for including the code to make the INF "unattend aware".

In many cases the automation of the OEMSETUP.INF can normally be handled with the following code additions. For examples of the code see OEMNADTK.INF in the %systemroot%\system32 directory.

Search for adapteroptions in the OEMSETUP.INF and insert the following code:

```
adapteroptions = +
;
;
=====
; This section added for UNATTENDED setups to bypass user
; prompts, called POPS.
;
=====
;
set from = adapteroptions
ifstr(i) $(!STF_GUI_UNATTENDED) == "YES"
    ifstr(i) $(!AutoNetInterfaceType) != ""
        set BusInterfaceType = $(!AutoNetInterfaceType)
```

```

else
    set BusInterfaceType = 1
endif
ifstr(i) $(!AutoNetBusNumber) != ""
    set BusNumber = $(!AutoNetBusNumber)
else
    set BusNumber = 0
endif
    goto skipoptions
endif
endif

```

Notice the goto skipoptions line above. The goto line must be verified for a subroutine called skipoptions. If the skipoptions subroutine does not exist, examine the code of adapteroptions to locate the appropriate goto routine and substitute.

The next part of the automation code is added at to the end of the "registry parameters add" section which is usually part of the writeparameters section. The following code reads the parameters section of the unattend file and writes the values to the registry.

```

Set NewValueList = {{InterruptNumber,$(NoTitle),$(REG_VT_DWORD),$(IRQValue)},+
    {BusType,$(NoTitle),$(REG_VT_DWORD),$(BusTypeNum)},+
    {BusNumber,$(NoTitle),$(REG_VT_DWORD),$(BusNumber)},+
    {AdapterType,$(NoTitle),$(REG_VT_DWORD),0},+
    {MediaType,$(NoTitle),$(REG_VT_DWORD),1},+
    {MemoryMappedBaseAddress,$(NoTitle),$(REG_VT_DWORD),$(MemoryMapValue)},+
    {IoBaseAddress,$(NoTitle),$(REG_VT_DWORD),$(IOBaseAddrValue)}}
Shell $(UtilityInf), AddValueList, $(KeyParameters), $(NewValueList)
;
; =====
; Rewrites the registry from the answer file.
; =====
;
ifstr(i) $(!STF_GUI_UNATTENDED) == "YES"
    Shell $(UtilityInf), AddDefaultNetCardParameters,$(KeyParameters)
endif

```

### Installation of Network Adapters Drivers Not Supplied on the Windows NT 4.0 Retail CD

Installing network adapter drivers that are not supplied on the Windows NT 4.0 Retail CD require all of the same steps as outlined above. The one added item is providing Windows NT Setup with the location of the network adapter file.

The method described provides the best versatility when dealing with vendor provided drivers. Most network installations are started via a batch file. Modify the batch file and insert the following commands before the start of the

---

WINNT.EXE command:

```
NET USE Z:\SERVER\1386
MKDIR C:\MYDRIVERS
COPY Z:\DRIVERS C:\MYDRIVERS
WINNT /B /U:Z:\UNATTEND.TXT
```

Having the drivers on the local driver will also simplify future troubleshooting if the adapter drivers have to be reinstalled.

```
[Network]
InstallAdapters = AdaptersList

[AdaptersList]
OEMAdapter = AdapterParameters, C:\MYDRIVERS

[AdapterParameters]
```

### **Bypassing the "Current Netcard Parameters Are Not Verifiably Correct" Message**

During the installation of a Network Adapter via Attended or Unattended methods, the following message may appear:

"The current netcard parameters are not verifiably correct and may result in usage problems or system failure. Use them anyway?"

The pop-up message is a warning that the parameters entered for the adapter do not match the current physical settings of the card. There are cases when the pop-up message does appear when the settings can not be verified despite the correct values.

During an UNATTENDED installation, the verification of the adapter parameters can be bypassed.

---

*Note: The text that is used in the dialog box resides in UTILITY.INF.*

---

The following example will hold true for the majority of Network Adapter OEMSETUP.INF files provided by Microsoft or Third-Party Vendors. This option should only be used if the parameters for the network adapter have been tested to ensure functionality. To locate the sections to modify, quickly do a search for VERIFY\_WARNING and compare it to the example provided.

The required changes are outlined in the following example:

```
Shell $(UtilityInf),RegistryErrorString,VERIFY_WARNING
ifint $($ShellCode) != $(!SHELL_CODE_OK)
```

```

Debug-Output "ShellCode error: cannot get an error string."
goto ShellCodeError
endif
set Error = $($R0)
; =====
; Start of Changes
; =====
; Comment out the "Goto Warning" entry with a semicolon
;
; Goto Warning
;
; Add the "Goto SkipOptions" entry as follows.
;
Goto SkipOptions
; =====
; End of changes
; =====
skipoptions +=
ifint $(OldVersionExisted) == $(TRUE)

```

### Network Adapter Option Name Driver Supplied in the \I386 Directory

The following adapters are part of the Windows NT 4.0 CD and can be located in the \I386 directory. The options name in the left column is the value needed for the [NETWORK] section of the UNATTEND.TXT. The Literal String in the middle column is the string displayed in Control Panel\Network and used to locate the appropriate Option Name.

\* = STF\_GUI\_UNATTENDED code present, the INF is "unattend aware".

	Source=\I386	
Options Name	Literal String	INF File
ELNKMC	3Com 3C523 Etherlink/MC Adapter	oemnadem.inf
ELNKII *	3Com Etherlink II Adapter (also II/16 and II/16 TP)	oemnade2.inf
ELNK3EISA	3Com Etherlink III EISA Adapter	oemnadee.inf
ELNK3ISA509 *	3Com Etherlink III ISA/PCMCIA Adapter	oemnade3.inf
ELNK3MCA	3Com Etherlink III MCA Adapter	oemnaden.inf
ELNK16 *	3Com Etherlink 16/EtherLink16 TP Adapter	oemnade1.inf
AM1500T *	Advanced Micro Devices AM2100/AM1500T Adapter	oemnadam.inf
AMDPCI	AMD PCNET Family Ethernet Adapter	oemnadap.inf
MAPLE	COMPAQ 32-Bit DualSpeed Token-Ring Controller	oemnadnf.inf
NETFLX	COMPAQ NetFlex/NetFlex-2 ENET-TR Controller	oemnadnf.inf

BONSAI	COMPAQ NetFlex-2 DualPort ENET Controller	oemnadnf.inf
RODAN	COMPAQ NetFlex-2 DualPort TR Controller	oemnadnf.inf
DURANGO	COMPAQ NetFlex-2 TR Controller	oemnadnf.inf
LT200 *	COPS/DayStar Digital LocalTalk Adapter	oemnadlt.inf
LT200MC	COPS/DayStar Digital LocalTalk Adapter (MCA)	oemnadlm.inf
DE425	DEC DE425 EtherWORKS Turbo EISA Adapter	oemnaddt.inf
DE434	DEC DE434 EtherWORKS Turbo PCI TP Adapter	oemnaddt.inf
DE435	DEC DE435 EtherWORKS Turbo PCI Adapter	oemnaddt.inf
DE450	DEC DE450 EtherWORKS Turbo PCI Adapter	oemnaddt.inf
DE500	DEC DE500 Fast Ethernet PCI Adapter	oemnaddt.inf
DEC100 *	DEC EtherWORKS LC Adapter	oemnadd1.inf
DECETHERWORKSTURBO *	DEC EtherWORKS Turbo Adapter	oemnadd2.inf
DEC422	DEC EtherWORKS Turbo EISA Adapter	oemnadd4.inf
DEC101 *	DEC EtherWORKS Turbo/LC Adapter	oemnadd1.inf
DEC300	DEC FDDIcontroller/EISA	oemnadd3.inf
DEFPA *	DEC FDDIcontroller/PCI	oemnaddf.inf
MULTIA	DEC multia's Ethernet Controller	oemnaddt.inf
DC21040	DEC PCI Ethernet DECchip 21040	oemnaddt.inf
DC21041	DEC PCI Ethernet DECchip 21041	oemnaddt.inf
DC21140	DEC PCI Fast Ethernet DECchip 21140	oemnaddt.inf
DC21142	DEC PCI Fast Ethernet DECchip 21142	oemnaddt.inf
DECSTAT *	DEC Turbo Channel Ethernet Adapter	oemnadde.inf
DATAFIREST	Digi DataFire - ISA1S/T Adapter	oemnaddi.inf
DATAFIREU	Digi DataFire - ISA1U Adapter	oemnaddi.inf
DATAFIRE4ST	Digi DataFire - ISA4S/T Adapter	oemnaddi.inf
PCIMACISA	Digi PCIMAC - ISA Adapter	oemnaddi.inf
PCIMACMC	Digi PCIMAC - MC Adapter	oemnaddi.inf
PCIMAC4	Digi PCIMAC/4 Adapter	oemnaddi.inf
NE2000IBMCOMPAT *	IBM Ethernet PCMCIA and Compatible Adapter	oemnadni.inf
IBMTOK*	IBM Token Ring (ISA/PCMCIA) Adapter	oemnadtk.inf
IBMTOKMC *	IBM Token Ring 4/16 Adapter /A	oemnadtm.inf
IBMTOKA *	IBM Token Ring Adapter /A	oemnadtm.inf
IBMTOK2ISA *	IBM Token-Ring Network 16/4 ISA Adapter II	oemnad2.inf
EE16 *	Intel Ether Express 16 LAN Adapter	oemnadin.inf
EE16MC	Intel Ether Express MCA Adapter	oemnadim.inf
IEEPRO *	Intel EtherExpress PRO Ethernet Adapter	oemnadep.inf
MSMDGMPSM16 *	Madge Smart 16 Ringnode	oemnadma.inf
MSMDGMPATP *	Madge Smart 16/4 AT Plus Ringnode	oemnadma.inf
MSMDGMPISA *	Madge Smart 16/4 AT Ringnode	oemnadma.inf
MSMDGMPEISA *	Madge Smart 16/4 EISA Ringnode	oemnadma.inf
MSMDGMPIAC * *	Madge Smart 16/4 ISA Client Plus Ringnode	oemnadma.inf
MSMDGMPPNP *	Madge Smart 16/4 ISA Client PnP Ringnode	oemnadma.inf
MSMDGMPIAC *	Madge Smart 16/4 ISA Client Ringnode	oemnadma.inf
MSMDGMPMCA *	Madge Smart 16/4 MC Ringnode	oemnadma.inf
MSMDGMPMC32 *	Madge Smart 16/4 MC32 Ringnode	oemnadma.inf
MSMDGMPPC *	Madge Smart 16/4 PC Ringnode	oemnadma.inf

MSMDGMPPCI *	Madge Smart 16/4 PCI Ringnode	oemnadma.inf
MSMDGMPPCIBM *	Madge Smart 16/4 PCI Ringnode (BM)	oemnadma.inf
MSMDGMPPCMCIA *	Madge Smart 16/4 PCMCIA Ringnode	oemnadma.inf
MICRODYNEPCMCIA *	Microdyne NE4000 PCMCIA Adapter	oemnadni.inf
LOOP *	MS Loopback Adapter	oemnadlb.inf
NE2000MCA *	NE/2 and Compatible MC Adapter	oemnadnm.inf
NPEISA	Network Peripherals FDDI EISA	oemnadnp.inf
NPMCA	Network Peripherals FDDI MCA	oemnadfd.inf
NE1000 *	Novell NE1000 Adapter	oemnadn1.inf
NE2000 *	Novell NE2000 Compatible Adapter	oemnadn2.inf
NE2000SOCKETEA *	Novell NE2000 Socket EA Adapter	oemnadn2.inf
NE3200	Novell NE3200 EISA Adapter	oemnadne.inf
NE4000PCMCIA *	Novell NE4000 PCMCIA Adapter	oemnadni.inf
AM1500T1 *	Novell/Anthem NE1500T Adapter	oemnadam.inf
AM1500T2 *	Novell/Anthem NE2100 Adapter	oemnadam.inf
P189X *	ProNET-4/16 p189X NIC	oemnadpm.inf
P1390 *	Proteon p139X Adapter	oemnadp3.inf
P1990 *	Proteon p199X Adapter	oemnadp9.inf
WD8003EA	SMC (WD) 8003E /A	oemnadwm.inf
WD8003WA	SMC (WD) 8003W /A	oemnadwm.inf
WD8013EPA	SMC (WD) 8013EP /A	oemnadwm.inf
WD8013WPA	SMC (WD) 8013WP /A	oemnadwm.inf
SMCISA *	SMC (WD) EtherCard	oemnadwd.inf
UBPC *	Ungermann-Bass Ethernet NIUpc Adapter	oemnadub.inf
UBPCEOTP *	Ungermann-Bass Ethernet NIUpc/EOTP Adapter	oemnadub.inf
UBPS	Ungermann-Bass Ethernet NIUps Adapter	oemnadum.inf

#### Drivers Supplied in the \DRVLIB\NETCARD\X86 and \I386\DRVLIB.NIC Directories

The following adapters are part of the Windows NT 4.0 CD and can be located in the \I386\DRVLIB.NIC directory. The Options Name in the left column is the value needed for the [NETWORK] section of the UNATTEND.TXT. The Literal String in the middle column is the string displayed in Control Panel\Network and is used to locate the appropriate Option Name.

To find the OEMSETUP.INF file, locate the option name in %systemroot%\system32\OEMNADZZ.INF. OEMNADZZ.INF provides the source path under DRVLIB.NIC.

\* = STF\_GUI\_UNATTENDED code present

Options Name	Source=\I386\DRVLIB.NIC	Literal String
3C508		3Com 3C508 ISA 16-bit Ethernet Adapter
ELINK527		3Com 3C527 Etherlink/MC 32 Adapter
3C592 *		3Com EtherLink III EISA Bus-Master Adapter (3C592)

3C590 *	3Com EtherLink III PCI Bus-Master Adapter (3C590)
3C597 *	3Com Fast EtherLink EISA 10/100BASE-T Adapter (3C597)
3C595 *	3Com Fast EtherLink PCI 10/100BASE-T Adapter (3C595)
3C905 *	3Com Fast EtherLink XL Adapter (3C905)
FLNK	3Com FDDILink EISA LAN Adapter
TLNK3EISA	3Com TokenLink III ISA Adapter in EISA mode (3C619B)
ACCNT *	Accton EN166x MPX2 PnP Ethernet Adapter
ACCTONEN2216 *	Accton EN2216 Ethernet PCMCIA Adapter
ALANE0 *	Adaptec ATM LAN Emulation Adapter
AT1700	Allied Telesyn AT1700 Ethernet Adapter
AT1700	Allied Telesyn AT1720 Ethernet Adapter
A2560PCI	Allied Telesyn AT-2560 Series PCI/100 Ethernet Adapter
ANDTOK	Andrew ISA IIA Token Ring Adapter
E21XX	Cabletron E21XX Ethernet Adapter
E22XX	Cabletron E22XX Ethernet Adapter
F30XX	Cabletron F30XX FDDI Adapter
F70XX	Cabletron F70XX FDDI Adapter
T20XX	Cabletron T20XX Token-Ring Adapter
EMPCI	Cogent eMASTER+ PCI Adapter
CPQNDIS	Compaq Ethernet LAN Card
NetFlex3	Compaq NetFlex-3 Controller
Senet	Compex ENET16 P/PNP Ethernet Adapter
IRMAtrac	DCA IRMAtrac Token-Ring Adapter
DigiSyncFR	Digi SyncPort Frame Relay Adapter
DigiSyncX25	Digi SyncPort X.25 Adapter
DLINKDE220 *	D-Link DE-220 ISA Ethernet Adapter
DLINKDE650 *	D-Link DE-650 Ethernet PCMCIA Adapter
Diehl_DIVA	Eicon DIVA ISDN ISA Adapter
Diehl_DIVAPCM	Eicon DIVA PCMCIA ISDN Adapter
Diehl_DIVAPRO	Eicon DIVA PRO ISDN Adapter with Advanced DSP
Diehl_S2M	Eicon Primary Rate ISDN Adapter
Diehl_QUADRO	Eicon QUADRO ISDN Adapter
Diehl_SCOM	Eicon SCOM ISDN Adapter
Diehl_WAN	Eicon Virtual WAN-Miniport ISDN Interface
ECCARDS	Eicon WAN Adapters
HPTXPCI	HP 10/100TX PCI Ethernet Adapter
HP27245	HP 27245A PC LAN Adapter/8 TP
HPMCA	HP 27246A MC LAN Adapter/16 TP
HP27247A	HP 27247A PC LAN Adapter/16 TP
HP27247B	HP 27247B PC LAN Adapter/16 TP Plus
HP27250	HP 27250 PC LAN Adapter/8 TL
HP27252A	HP 27252A PC LAN Adapter/16 TL Plus
J2573A	HP DeskDirect (J2573A) 10/100 ISA LAN Adapter
J2577A	HP DeskDirect (J2577A) 10/100 EISA LAN Adapter
J2585A	HP DeskDirect (J2585A) 10/100 PCI LAN Adapter
J2585B	HP DeskDirect (J2585B) 10/100 PCI LAN Adapter

J2970A	HP DeskDirect (J2970A) 10BaseT/2 PCI LAN Adapter
J2973A	HP DeskDirect (J2973A) 10BaseT PCI LAN Adapter
IBMFEP	IBM 100/10 PCI Ethernet Adapter
IBMTOK4 *	IBM Auto 16/4 Token-Ring ISA Adapter
STREAMER	IBM Auto LANStreamer PCI Adapter
IBMISAETHER *	IBM ISA Ethernet Adapter
IBMENIIN	IBM LAN Adapter/A for Ethernet
QUADENET	IBM PeerMaster Server Adapter
STREAMER	IBM Streamer Family Adapters
ETH161	ICL EtherTeam16i Adapter
ETH32	ICL EtherTeam32 Adapter
E100BPCI *	Intel 82557-based 10/100 Ethernet PCI Adapter
E10PCI	Intel EtherExpress PRO/10 PCI LAN Adapter
EPRONT	Intel EtherExpress PRO/10+ ISA Adapter
E10PPCI *	Intel EtherExpress PRO/10+ PCI Adapter
E100BEXP *	Intel EtherExpress PRO/100B PCI Adapter
FL32	Intel Flash32 EISA LAN Adapter
TKXP16	Intel TokenExpress 16/4 Adapter
TKXP32	Intel TokenExpress Server Adapter
LINKSYSE16 *	LinkSys Ether16 LAN Card
LINKSYSEC2T *	LinkSys EthernetCard PCMCIA
LEC *	Madge ATM LAN Emulation Client
BLUTOK *	Madge Blue+ Token Ring Adapter
CC10BT	Megahertz CC10BT/2 Ethernet PCMCIA Adapter
XJEM3288 *	Megahertz XJEM3288 Ethernet+Modem PCMCIA Adapter
NE100PCI	Microdyne NE10/100 PCI Adapter
MGSL	MicroGate SyncLink Internet Adapter
NSCNE4100 *	National Semiconductor InfoMover NE4100
NCR TOK	NCR StarLAN 16/4 Token-Ring Adapter
NPAT2	Network Peripherals FDDI - AT2
NPAT3	Network Peripherals FDDI - AT3
NCPF	Network Peripherals NuCard PCI FDDI
NiwRAS	Niwot Networks NiwRAS Adapter
GOCARD	Olicom Ethernet GoCard
OCE2XM	Olicom Ethernet ISA/TV Adapter
O100PCI	Olicom Ethernet PCI 10/100 Adapter
OCE4XMP10 *	Olicom Ethernet PCI/II 10 Adapter
OCE4XMP100 *	Olicom Ethernet PCI/II 10/100 Adapter
GOCARDMF	Olicom GoCard ET/Modem 288
PCMCIA *	Olicom GoCard TR 16/4
COMBO *	Olicom GoCard TR/Modem 144
OCTK16 *	Olicom Token Ring 16/4 Adapter
OCTK32	Olicom Token Ring Server Adapter
OTCJODNT	Ositech Jack of Diamonds Trumpcard
ES3210	Racal Interlan ES3210 EISA Ethernet Adapter
NI6510	Racal InterLan XLerator/EB/NI6510 Adapters

RTL8029	Realtek RTL8029 PCI Adapter
RnsFDDI	RNS 2200 PCI FDDI LAN Controller
SMC8216	SMC 8216 EtherCard Elite16 Ultra
SMC8416	SMC 8416 EtherEZ
SMC8432	SMC 8432 EtherPower PCI Ethernet Adapter
SMC9232	SMC 9232 Fast Ethernet Adapter
SMC9332	SMC 9332 EtherPower10/100 PCI Fast Ethernet Adapter
SMC8232	SMC EISA EtherCard Elite32 Ultra Adapter
ACLSER	Star Gate ACL/Avanstar Family Adapter
SKTOKNT	SysKonnnect SK-NET 4/16+ Token Ring Adapter
SKFENT	SysKonnnect SK-NET EISA FDDI Adapter
SKFPNT	SysKonnnect SK-NET FDDI PCI Adapter
SKETHNT	SysKonnnect SK-NET G16 Ethernet Adapters
SKETHNT	SysKonnnect SK-NET G32+ Ethernet Adapters
SKFINT	SysKonnnect SK-NET ISA FDDI Adapter
SKFMNT	SysKonnnect SK-NET MCA FDDI Adapter
SKTOKNT_PCI	SysKonnnect SK-NET Token Ring PCI Adapter
SKTOKNT	SysKonnnect SK-NET TR4/16+ Token Ring Adapter
TC\$4045e	Thomas-Conrad TC4045 Token Ring Adapter
TC\$4046e	Thomas-Conrad TC4046 Token Ring Adapter
AL56	U.S. Robotics Allegra 56 Frame Relay
ALT1	U.S. Robotics Allegra T1 Frame Relay
USRBRI	U.S. Robotics Sportster ISDN Adapter
WAVELAN_ISA	WaveLAN ISA Bus Adapter
WAVELAN_MCA	WaveLAN MCA Bus Adapter
CENDIS3	Xircom CreditCard Ethernet
CE2XPS	Xircom CreditCard Ethernet Iips
CEM28XPS	Xircom CreditCard Ethernet+Modem 28.8
CM2NDIS3	Xircom CreditCard Ethernet+Modem II
CTNDNT	Xircom CreditCard Token Ring
XCSPE2	Xircom Pocket Ethernet II
XCSPE3	Xircom Pocket Ethernet III

## OEM Install Options that Can be Used with UNATTEND.TXT

```
[Unattended]
OEMPreinstall = Yes
ComputerType = "Standard PC", "RETAIL"

[PointingDeviceDrivers]
"Microsoft Mouse Port Mouse (includes BallPoint)" = "RETAIL"

[MassStorageDrivers]
"Symbios Logic C810 PCI SCSI Host Adapter" = "RETAIL"

[KeyBoardDrivers]
```

---

"XT, AT, or Enhanced Keyboard (83-104 keys)" = "RETAIL"

```
[Display]
InfFile = "chips.inf"
InfOption = "Chips Video Accelerator(64300 64310 65545 65548
65550)"
InstallDriver = 1
```

```
[OEMAds]
```

```
[OEMBootFiles]
TXTSETUP.OEM
OEMHAL.DLL
OEMSCSI.SYS
```

**TXTSETUP.SIF Entries for Retail-Supplied Files that Work with OEM Options in the UNATTEND.TXT**

### **Computer Types**

**Example:**

**[Unattended]**

**ComputerType = "Standard PC","RETAIL"**

Retail Options:

"AST Manhattan SMP"  
"Compaq SystemPro Multiprocessor or 100% Compatible","RETAIL"  
"Corollary C-bus Architecture","RETAIL"  
"Corollary C-bus Micro Channel Architecture","RETAIL"  
"IBM PS/2 or other Micro Channel-based PC","RETAIL"  
"MPS Uniprocessor PC","RETAIL"  
"MPS Multiprocessor PC","RETAIL"  
"MPS Multiprocessor Micro Channel PC","RETAIL"  
"NCR System 3000 Model 3360/3450/3550","RETAIL"  
"Olivetti LSX5030/40","RETAIL"  
"Standard PC","RETAIL"  
"Standard PC with C-Step i486","RETAIL"  
"Wyse Series 7000i Model 740MP/760MP","RETAIL"

### **Keyboard Layouts**

**Example:**

**[Unattended]**

**KeyBoardLayout = "US-International"**

Retail Options:

---

"Albanian"  
"Belarusian"  
"Belgian Dutch"  
"Belgian French"  
"Brazilian (ABNT)"  
"Bulgarian"  
"Bulgarian Latin"  
"Canadian English (Multilingual)"  
"Canadian French"  
"Canadian French (Multilingual)"  
"Croatian"  
"Czech"  
"Czech (QWERTY)"  
"Danish"  
"Dutch"  
"Estonian"  
"Finnish"  
"French"  
"German"  
"German (IBM)"  
"Greek"  
"Greek Latin"  
"Greek (220)"  
"Greek (220) Latin"  
"Greek (319)"  
"Greek (319) Latin"  
"Hungarian"  
"Hungarian 101-key"  
"Icelandic"  
"Irish"  
"Italian"  
"Italian (142)"  
"Latin American"  
"Latvian"  
"Latvian (QWERTY)"  
"Lithuanian"  
"Norwegian"  
"Polish (Programmers)"  
"Polish (214)"  
"Portuguese"  
"Romanian"  
"Russian"  
"Russian (Typewriter)"  
"Serbian Cyrillic"  
"Serbian Latin"

---

"Slovak"  
"Slovak (QWERTY)"  
"Slovenian"  
"Spanish"  
"Spanish variation"  
"Swedish"  
"Swiss French"  
"Swiss German"  
"Turkish F"  
"Turkish Q"  
"Ukrainian"  
"United Kingdom"  
"US"  
"US-Dvorak"  
"US-Dvorak for left hand"  
"US-Dvorak for right hand"  
"US-International"

### **Mouse Drivers**

#### **Example:**

**[PointingDeviceDrivers]**

**"Microsoft Mouse Port Mouse (includes BallPoint)" = "RETAIL"**

#### Retail Options:

"Microsoft Mouse Port Mouse (includes BallPoint)" = "RETAIL"  
"Logitech Mouse Port Mouse" = "RETAIL"  
"Microsoft InPort Bus Mouse" = "RETAIL"  
"Microsoft Serial Mouse" = "RETAIL"  
"Microsoft BallPoint Serial Mouse" = "RETAIL"  
"Logitech Serial Mouse" = "RETAIL"  
"Microsoft (Green Buttons) or Logitech Bus Mouse" = "RETAIL"  
"No Mouse or Other Pointing Device", = "RETAIL"

### **SCSI Drivers**

#### **Example:**

**[MassStorageDrivers]**

**"Symbios Logic C810 PCI SCSI Host Adapter" = "RETAIL"**

#### Retail Options:

"Adaptec AHA-151X/AHA-152X/AIC-6X60 SCSI Adapter" = "RETAIL"  
"Adaptec AHA-154X/AHA-164X SCSI Host Adapter" = "RETAIL"  
"Adaptec AHA-174X EISA SCSI Host Adapter" = "RETAIL"  
"Adaptec AHA-274X/AHA-284X/AIC-777X SCSI Adapter" = "RETAIL"

"Adaptec AHA-294X/AHA-394X/AIC-78XX SCSI Controller" = "RETAIL"  
"AMD PCI SCSI Controller/Ethernet Adapter" = "RETAIL"  
"AMLscsi SCSI Host Adapter" = "RETAIL"  
"BusLogic SCSI Host Adapter" = "RETAIL"  
"BusLogic FlashPoint" = "RETAIL"  
"Compaq 32-Bit Fast-Wide SCSI-2/E" = "RETAIL"  
"Compaq Drive Array" = "RETAIL"  
"Dell Drive Array" = "RETAIL"  
"DPT SCSI Host Adapter" = "RETAIL"  
"Future Domain TMC-7000EX EISA SCSI Host Adapter" = "RETAIL"  
"Future Domain 8XX SCSI Host Adapter" = "RETAIL"  
"Adaptec 2920/2905 / Future Domain 16XX/PCI/SCSI2Go" = "RETAIL"  
"IBM MCA SCSI Host Adapter" = "RETAIL"  
"IDE CD-ROM (ATAPI 1.2)/PCI IDE Controller" = "RETAIL"  
"Mitsumi CD-ROM Controller" = "RETAIL"  
"Mylex DAC960/Digital SWXCR-Ex Raid Controller" = "RETAIL"  
"NCR 53C9X SCSI Host Adapter" = "RETAIL"  
"NCR C700 SCSI Host Adapter" = "RETAIL"  
"NCR 53C710 SCSI Host Adapter" = "RETAIL"  
"Symbios Logic C810 PCI SCSI Host Adapter" = "RETAIL"  
"Olivetti ESC-1/ESC-2 SCSI Host Adapter" = "RETAIL"  
"QLogic PCI SCSI Host Adapter" = "RETAIL"  
"MKEPanasonic CD-ROM Controller" = "RETAIL"  
"Sony Proprietary CD-ROM Controller" = "RETAIL"  
"UltraStor 14F/14FB/34F/34FA/34FB SCSI Host Adapter" = "RETAIL"  
"UltraStor 24F/24FA SCSI Host Adapter" = "RETAIL"

### **Keyboard Driver**

#### **Example:**

[KeyboardDrivers]

"XT, AT, or Enhanced Keyboard (83-104 keys)" = "RETAIL"

Retail Options:

"XT, AT, or Enhanced Keyboard (83-104 keys)" = "RETAIL"

#### **Full Example:**

[Unattended]

OEMPreinstall = Yes

ComputerType = "Standard PC","RETAIL"

KeyboardLayout = "US-International"

[PointingDeviceDrivers]

"Microsoft Mouse Port Mouse (includes BallPoint)" = "RETAIL"

[MassStorageDrivers]

"Symbios Logic C810 PCI SCSI Host Adapter" = "RETAIL"

[KeyboardDrivers]  
"XT, AT, or Enhanced Keyboard (83-104 keys)" = "RETAIL"

## Third-Party Video Display Drivers and Display Settings

### Options for Microsoft-Supplied Video Drivers (Part of I386/Auto-Detected)

The following entries are needed to automate the setup of detected video displays during unattended setup.

```
[Display]
BitsPerPel = 16
XResolution = 1024
YResolution = 768
VRefresh = 60
Flags = 0
AutoConfirm = 1
```

---

*Note: AutoConfirm sets the display adapter values during setup automatically. ConfirmAtLogon allows the user to set the display adapter values on logon.*

---

To determine the exact values needed, the use of REGEDT32.EXE is the best option.

Determining an adapter's parameters can be accomplished via the REGEDT32.EXE utility supplied with Windows NT 4.0. Every installed video adapter has a key in HKLM\SYSTEM\CurrentControlSet\<Display Key>\Device0,

To find out the value of <Display Key> check the following registry values.

HKLM\Hardware\DeviceMap\Video\

Double-click on \Device\Video0 in the left window of REGEDIT32.EXE. Write down the string for the registry path.

Example:

\REGISTRY\Machine\System\CurrentControlSet\Services\ati\Device0

*Note the string right after Services. In the example, it is ATI. ATI is the <Display Key>.*

Now go to HKLM\System\Services\CurrentControlSet\ATI\Device0. The values needed: BitsPerPel, Xresolution, Yresolution, Vrefresh, and Flags can be converted from hex to decimal and entered in the UNATTEND.TXT.

The string assigned to ServiceName is what's needed to locate the parameters for instance of the network adapter installed. These parameters can be located in the registry under HKLM\SYSTEM\CurrentControlSet\Services\<ServiceName>\Parameters.

---

All numeric values found in the parameters key for the adapter have to be converted from hex values to decimal values for the unattend file. The names of the parameters and non-numeric values can be directly translated to the unattend file.

### Options for OEM-Supplied Video Drivers

Setting video drivers supplied by the OEM is a little more difficult and requires parsing the information file supplied for the driver.

The following example shows the use of the InfFile, InfOption, and InstallDriver options. OemPreInstall must be set to "Yes" for options to be used.

```
[Display]
InfFile = "chips.inf"
InfOption = "Chips Video Accelerator(64300 64310 65545 65548 65550)"
InstallDriver = 1
BitsPerPel = 16
XResolution = 800
YResolution = 600
VRefresh = 60
AutoConfirm = 1
```

To properly install the driver and make the files available to setup, the following directory must be created %OEM%\DISPLAY and the .SYS, .DLL, and .INF files have to be copied to the directory.

Determining the InfOption is the most difficult part of the process.

The following information is an example for the above [DISPLAY] section.

In the case of the Chips & Technologies driver, the CHIPS.INF needs to be opened to get the information for the InfOption.

In the CHIPS.INF file search for the following string, [Manufacturer]. Under this heading the following information is found, %chips%=chips.Mfg.

1. Search for [chips.Mfg]. Under the heading the following information is found:

```
Chips Video Accelerator(64300 64310 65545 65548 65550) = chips
```

2. The information to the left of the equal sign is placed in quotes and then placed to the right of the equal sign for InfOption as in the example below:

```
InfOption = "Chips Video Accelerator(64300 64310 65545 65548 65550)"
```

3. Use the same method for determining the options for the display as outlined in the previous section.

### **TXTSETUP.OEM and [OEMBootFiles]**

Any file in the \386\%OEM%\TEXTMODE directory is copied during setup to \TEXTMODE. Files that are specified in the [OEMBootFiles] section are then copied from \TEXTMODE to \WIN\_NT\$.~BT\%OEM%. The \WIN\_NT\$.~BT\%OEM% directory is accessed by setup based on the TXTSETUP.OEM [Disks] section.

For additional information on TXTSETUP.OEM and Windows NT Device Drivers, consult the Microsoft Windows NT Version 4.0 Device Driver Kit (DDK).

#### **TXTSETUP.OEM File: Format and Sample**

A TXTSETUP.OEM file consists of several sections that use the following general format:

```
[SectionName]  
key = value1,value2,...
```

The name of the section is enclosed in square brackets, [ ]. The pound sign (#) or semicolon (;) character at the beginning of a line indicates a comment. Strings with embedded spaces, commas, or hashes must be enclosed in double quotes ("").

The following sections must be included:

#### **Disks Section**

The [Disks] section lists all disks in the disk set.

```
[Disks]  
d1 = description,tagfile,directory  
d2 = description,tagfile,directory  
.  
.
```

d1, d2, ...

Key that can be used in subsequent sections to identify the disk.

#### *description*

Disk name string used to prompt the user to insert the disk.

#### *tagfile*

Specifies the name of the file whose presence on the disk indicates to the Setup program that the correct disk has been inserted. The filename should be specified as a full path from the root – for example, \yourtagfile.ext, but it should not specify a drive.

---

*directory*

Specifies the directory on the disk where the files are located. The directory should be specified as a full path from the root – for example, \diskdir, but it should not specify a drive.

**Defaults Section**

The [Defaults] section lists the default option for each hardware component supported by this file (as indicated by the presence of a [component] section for the component). The default is highlighted in the menu of options presented to the user.

[Defaults]

component = ID

.

*component*

Specifies one of the following components: computer, display, keyboard, mouse, CD-ROM, or SCSI.

*ID*

Specifies a string that identifies the default option. This string matches an ID specified in the corresponding [component] section.

**Component Section**

A [component] section lists the options available for a particular component.

[component]

ID = description

.

*ID*

Specifies a unique string (within this section) that identifies the option. For the computer component, if this string ends in "\_up", setup copies the uniprocessor kernel; if this string ends in "\_mp", Setup copies the multiprocessor kernel; if it ends in neither, the results are unspecified and you might get either kernel.

**Files.component.ID Section**

A [Files.component.ID] section lists the files that should be copied if the user selects a particular component option. One section of this type must be present for each option listed in each component section. The component portion of the section name corresponds to the name of a [component] section, and the ID portion corresponds to an ID key in a [component] section.

[Files.component.ID]

file\_type = source\_disk,filename[,keyname]

### *filetype*

Identifies the type of file. One of the following is specified:

- **driver** – Valid for all components. File is copied to systemroot\system32\drivers.
- **Port** – Valid for keyboard, mouse, and SCSI components. Allows distinction between port and class driver – but equivalent to driver type.
- **Class** – Valid for keyboard and mouse components. If specified, replaces the standard class driver. File is copied to systemroot\system32\drivers.
- **dll** – Valid for all components. Useful for GDI portion of a display driver. File is copied to systemroot\system32.
- **hal** – Valid only for computer component. File is copied to systemroot\system32\hal.dll (80386/80486), or to \os\winnt\hal.dll on the system partition (ARC).
- **inf** – Valid for all components. Used to copy a GUI INF file for use with system maintenance setup. File is copied to systemroot\system32.
- **detect** – Valid for the computer component (80386/80486 only). If specified, replaces the standard 80386/80486 hardware recognizer. File is copied to c:\ntdetect.com.

### *source\_disk*

Identifies the disk from which to copy the file and must match an entry in the [Disks] section.

### *filename*

Name of the file not including the directory path. The filename is appended to the directory specified for the disk in the [Disks] section to form the full path of the file on the disk.

### *DriverKey*

Name of the key to be created in the registry services tree for this file – if the file is of type driver, port, or class. This value is used to form [Config.DriverKey] section names.

### **Config.DriverKey Section**

A [Config.DriverKey] section specifies values to be set in the registry for particular component options. Required values in the Services\DriverKey key are created automatically. Use this section to specify additional keys to be created under Services\DriverKey and values under Services\DriverKey and Services\DriverKey\subkey\_name.

[Config.DriverKey]

value = subkey\_name,value\_name,value\_type,value...

*subkey\_name*

Specifies the name of a key under the **Services**\DriverKey tree in which to place the specified value. The key is created if it does not exist. If the empty string ("") is specified, the value is placed under the **Services**\DriverKey tree.

*value\_name*

Specifies the name of the value to be set within the key.

*value\_type*

A string like REG\_DWORD that identifies the type of data for this value.

*value*

Specifies the actual value; its format depends on value\_type.

The following types can be specified in the value\_type field in [Config.DriverKey] sections:

REG\_DWORD

One value is allowed; it must be a string of 1-8 hex digits.

- For example:

**value** = parameters,NumberOfButtons,REG\_DWORD,0X2

REG\_SZ, REG\_EXPAND\_SZ

One value is allowed; it is interpreted as the zero-terminated string to be stored.

- For example:

**value** = parameters,Description,REG\_SZ,"This is a text string"

REG\_BINARY

One value is allowed; it is a string of hex digits, each pair of which is interpreted as a byte value.

- For example (stores the byte stream 00,34,ec,4d,04,5a):

**value** = parameters,Data,REG\_BINARY,0034ec4d045a

REG\_MULTI\_SZ

Multiple value arguments are allowed; each is interpreted as a component of the multisz.

- For example:

**value** = parameters,Strings,REG\_MULTI\_SZ,String1,"String 2",string3

**Example of the [OemBootFiles] Section for the Example TXTSETUP.OEM**

```
[OemBootFiles]
myhal.dll
aha154x.sys
txtsetup.oem
```

#### **Example TXTSETUP.OEM File**

```
[Disks]
d3 = "HAL Support for Windows NT",\oemhal.tag,\
d1 = "Storage Support for Windows NT",\oemstor.tag,\

[Defaults]
HAL = e_isa_up
scsi = aha154x

[Computer]
e_isa_up = "Custome HAL",files.none

[Files.Computer.e_isa_up]
HAL = d3,myhal.dll

[SCSI]
aha154x = "Adaptec AHA-154x/1640 - OEM"

[Files.scsi.aha154x]
driver = d1,aha154x.sys.sys, aha154x
[Config.aha154x]
```

#### **Error Messages When Working with TXTSETUP.OEM**

##### **Error:**

Windows NT Setup

```
File caused a unexpected error (0) at
the line 1213 in d:\nt\private\ntos\boot\setup\oemdisk.c
```

Press any key to continue.

##### **Solution:**

The variable HAL is not defined correctly for the section [Files.Computer.xxxxxx] in the TXTSETUP.OEM.

##### **Incorrect Structure**

```
[Files.Computer.e_isa_up]
e_isa_up = d3,myhal.dll
```

---

**Correct Structure**

```
[Files.Computer.e_isa_up]  
HAL = d3,myhal.dll
```

Note the differences of the line under the [Files.Copmuter.e\_isa\_up] for both examples.

**Error:**

Windows NT Setup

```
File \ $WIN_NT$.~BT\ $OEM$\HAL\HAL.DLL could not be loaded.  
The error code is 18
```

Setup cannot continue. Press any key to exit.

**Solution:**

The path defined under the [Disks] section is incorrect. Windows NT Setup does not allow for subdirectories under \ \$OEM\$\TEXTMODE. All component files must be placed in the TEXTMODE directory.

Syntax for the [Disks]

```
[Disks]
```

```
d1 = description,tagfile,directory
```

```
d2 = description,tagfile,directory
```

**Incorrect Structure**

```
[Disks]
```

```
d3 = "HAL Support for Windows NT",\oemhal.tag,\HAL
```

```
d1 = "Storage Support for Windows NT",\oemstor.tag,\SCSI
```

**Correct Structure**

```
[Disks]
```

```
d3 = "HAL Support for Windows NT",\oemhal.tag,\
```

```
d1 = "Storage Support for Windows NT",\oemstor.tag,\
```

In many cases when using a TXTSETUP.OEM provided by a third-party vendor, the directory would have to be modified and set to \.

Windows NT 4.0 setup includes a feature that allows you to configure machine-specific information without having to create a unique setup script file for each machine. This allows you to create a generic setup script for multiple systems and specify settings unique to each machine as needed in a separate file called a Uniqueness Database File (UDB). This and other methods of configuring settings that are machine-specific allow you to minimize the time spent by a setup technician at each desktop. In this section you will find the following methods discussed:

- Create unique setup script files for each machine.
- Modify the machine-specific settings after character mode setup. Useful for those using disk duplication to distribute Windows NT 4.0.
- Configure machine-specific information using Uniqueness Database Files (.UDB).

### **Create Unique Setup Script Files for Each Computer**

Creating unique setup script files for each computer is the most simple method of configuring machine-specific information. In addition, this method can have a positive impact on cost of ownership since you can save the original machine setup script file and use it in the future if you have a hardware problem that requires you to re-install the operating system and software applications. You can simply replace the failed hardware component or the entire computer and run Windows NT setup with your script again. There can be an up front cost associated with this method since you will likely find the need for a tool to automate the process of customizing each machine-specific setup script, a tedious and error prone task when done with a simple text editor.

The overall process is simple, start by creating a Windows NT 4.0 setup script file that includes all of the settings necessary for your preferred client configuration. This setup script file will be your template. Then copy the template to a unique filename, modify it to include machine-specific information with a text file editor such as Notepad.exe, and save the file.

Automating the process of customizing the template setup script with machine-specific information will likely mean creating your own tool. For example, you could create a program that would replace tokens you have placed in your setup script template file. The program could provide the setup technician with a user interface prompt asking for the computer name and network card, including network card settings, and replacing the tokens you placed in the setup script file with strings you have pre-tested then writing the unique setup script to disk, preventing user error and speeding the process.

### **Modify the Machine-Specific Settings After the Character Mode Portion of Windows NT 4.0 Setup**

Windows NT 4.0 setup has two distinct modes – during the first mode, called character mode, one of the tasks accomplished is to copy all of the files necessary to complete the setup process to a temporary directory on the local

---

hard drive. If you intend to use disk duplication to distribute Windows NT 4.0 this is the point where you turn off the computer and duplicate the hard drive for other systems. This method of distribution presents a problem if your goal is to create an entirely automated setup process that requires no input from a setup technician or an end user. This is because there is not an obvious way to set machine-specific settings before shipping the machine to the end user. However, you can do this in one of two ways. If you have set the machine-specific information in the setup script you created, you can modify those settings on each of the duplicated hard drives. The second method can be used if you included a .UDB file as one of the parameters for your Windows NT setup command line.

#### **Editing the Windows NT 4.0 Setup Script (UNATTEND.TXT) After Character Mode Setup**

During text mode setup a modified version of your original setup script file is copied to a temporary directory on your local hard drive. By editing this file on each of the duplicated hard drives before shipping the computers to your end users, you can create a process that requires no input from the end user during the graphical mode portion of Windows NT setup. To do this use an MS-DOS boot disk to start the computer after installing the duplicated hard drive. Then edit the C:\\$WIN\_NT\$.~BT\WINNT.SIF file and replace the settings unique to the computer. Note that the WINNT.SIF file is not an identical copy of your original setup script (UNATTEND.TXT), so you cannot replace the entire file. For example, if you set the computer name in your original setup script file to "ReplaceMe" you would open the file and search for that string, then replace it with the unique computer name for that computer.

#### **Editing the Uniqueness Database File (.UDB) After Character Mode Setup**

During text mode setup a copy of the .UDB file that you included on the Windows NT 4.0 setup command line is copied to the temporary directory on the local hard drive. You can replace this file with one that contains the unique settings for this computer. To do this use an MS-DOS boot disk to start the computer after installing the duplicated hard drive. Then either edit or replace the .UDB file in the temporary directory created by Windows NT setup. You can locate this file by using the following MS-DOS command (for MS-DOS 5.0 or later):

```
DIR C:\*.UDB /S
```

---

*Note: The ID that you specified on the Windows NT 4.0 setup command line will be the same for each of the drives you have duplicated. You must make changes to the sections specified in the .UDB file for the ID you specified. You will find more information about creating .UDB files later in this section.*

---

---

## Configure Machine-Specific Information Using Uniqueness Database Files (.UDB)

Windows NT setup includes a feature designed to enable you to create one common Windows NT 4.0 setup script for use with a number of target computers to overcome the problem of "How to specify machine-specific information?" without creating a unique setup script for each computer by specifying the unique settings in a separate database file. This file is a Uniqueness Database File (.UDB). .UDBs are used to provide replacements for sections of setup script, or supply additional section. This file is indexed using strings called UniqueIDs.

The .UDB is used to specify a set of sections that should be merged into the setup script file at the start of GUI setup. This process takes place before any affected components actually read the internal representation of the setup script file, and is transparent to the user.

---

*Note: The Windows NT setup command line parameter is /UDF but the file name extension for the Uniqueness Database File is .UDB.*

---

### *Specifying a Unique ID*

To specify a UniqueID during setup, you must run WINNT.EXE or WINNT32.EXE command using the following parameter:

```
/UDF:ID[,database_filename]
```

Where:

ID is the UniqueID that is used while installing Windows NT on this computer, and database\_filename is the filename, including the full path, of the .UDB.

If both the UniqueID and the filename of the .UDB are specified, the .UDB is copied to the local drive during Text Mode Setup, and is used during GUI mode setup without user intervention. The .UDB file can be any legal MS-DOS filename.

If only the UniqueID is specified on the setup command line, setup will prompt the user for a disk with a .UDB file named \$Unique\$.udb. This disk must be prepared by the administrator in advance. The user is prompted for this disk during the graphical (GUI) mode of setup.

In either case, if the supplied .UDB is corrupt or if setup cannot locate the specified UniqueID, the user is prompted to insert a disk that contains the fixed .UDB, or Cancel. If the user clicks Cancel, the values in the setup script file will be used. These might not be appropriate for the computer.

### *Creating the .UDB*

---

The .UDB file is a text file, use any standard text editor to create the .UDB file. The first section of the .UDB is [Uniquelds]. This section lists all UniqueIDs that are included in your Uniqueness Database File. The information on the left is the UniqueID, which can contain any character except an asterisk (\*), space, comma, or equal (=) character. The information on the right is a list of sections, the names which should match the names in the corresponding section in the setup script file.

The format is as follows:

```
[Uniquelds]
id1 = section1,section2
id2 = section1,section2
id3 = section1,section3,section4
```

For example, if you planned to use computer names based on user email names as the UniqueIDs, this section might resemble the following:

```
[Uniquelds]
computer1 = UserData,Unattended
computer2 = UserData,Unattended
computer3 = UserData,GuiUnattended,Network
```

The sections following [Uniquelds] are referenced in [Uniquelds]. These names can take either of two forms: they can match the section name in the setup script file (for example, [Unattended]), or they can be preceded by the UniqueID and a colon (for example, [computer1:UserData]). This allows you to create specialized replacement sections for each computer name.

If both a general section (such as [Unattended]) and an ID-specific section (such as [computer2:Unattended]) are included, Windows NT setup gives precedence to the ID-specific section.

The sections in the .UDB can contain any possible keys and values for the same-named sections in the setup script file. You can replace, add, or delete values in your setup script file. During setup, each key specified in a referenced section overrides the value for the same key in the setup script file. If a key is specified in the setup script file, but not in the .UDB section referenced by the UniqueID, the value specified in the setup script file is used.

If a section is referenced in the [Uniquelds] section but does not exist, the user will be prompted to insert a disk containing a valid .UDB file.

### *Replacing a Line in Setup Script*

If both the setup script and the .UDB section contain a line that is referenced by the UniqueID, the value specified in the .UDB is used instead of the value in

---

setup script. This is equivalent to replacing the line in setup script.

#### *Adding a New Line to Setup Script*

If the .UDB section contains a line that is referenced by the UniqueID but the setup script file does not, the value specified in the .UDB is used. This is equivalent to adding the line to setup script.

#### *Deleting a Line from Setup Script*

If a key is specified in the setup script file, and it appears in the .UDB section that is referenced by the UniqueID with no value to the right of the equal sign, the default value is used. This is equivalent to commenting out the key in the setup script file.

If a key is specified in the .UDB, but the value is left blank, no value is used for that key even if it is specified in setup script. This might result in the user being prompted for the information.

The following sections are not permitted in the .UDB file.

```
[UNATTENDED]
[MassStorageDrivers]
[KeyboardDrivers]
[PointingDeviceDrivers]
[OEMBootFiles]
[OEM_Ads]
[Display]
[Modem]
```

You can create a single .UDB file with UniqueIDs for all of your computers or many .UDB files, one for each machine. You might find that creating individual .UDB files for each computer is easier to manage than a single encompassing .UDB file.

#### **Example 1**

In this example, the [Userdata], [GuiUnattended], and [Network] sections will be merged into the setup script file. See the sample setup script file at the end of this chapter as a reference. The following example .UDB file includes settings for a single computer.

```
[UniqueIDs]
  ComputerID1 = Userdata,GuiUnattended,Network

[UserData]
  FullName = "User ID-1"
  ComputerName = "MACHINE-1"

[GuiUnattended]
```

---

```
TimeZone = " (GMT-05:00) Eastern Time (US & Canada)"
```

```
[Network]  
JoinDomain = "DomainEast"
```

The settings specified in this .UDB file will be merged during the GUI mode of Windows NT setup.

### Example 2

This example shows the use of the UniqueID to specify multiple machine IDs within the same UDB. Note that each section has the UniqueID included. The UniqueID specified on using the /UDF parameter would be change to match the UniqueID for each computer but the .UDB filename would be the same. Also note that this .UDB file would merge the same values for ComputerID1 in example 1 above:

```
[UniqueIDs]  
ComputerID1 = Userdata,GuiUnattended,Network  
ComputerID2 =  
GuiUnattended,Userdata,Modem,AdaptersList,EE16Params,ProtocolsList,TCP  
IPParams, SelectedServicesList,InstallCSNW
```

```
[ComputerID1-UserData]  
FullName = "User ID-1"  
ComputerName = "MACHINE-1"
```

```
[ComputerID1-GuiUnattended]  
TimeZone = " (GMT-05:00) Eastern Time (US & Canada)"
```

```
[ComputerID1-Network]  
JoinDomain = "DomainEast"
```

```
[ComputerID2:GuiUnattended]  
OEMSkipWelcome = 1  
OemBlankAdminPassword = 1  
TimeZone = "(GMT-06:00) Central Time (US & Canada)"  
DetachedProgram = "C:\REGINI.EXE"  
Arguments = "C:\REGINI.INI"
```

```
[ComputerID2:UserData]  
FullName = "User Name"  
OrgName = "Engineering Department"  
ComputerName = "OPKNT40-3"  
ProductID = "123-1234567"
```

---

[ComputerID2:NetWork]  
JoinWorkGroup = engineering

[ComputerID2:AdaptersList]  
EE16 = EE16Params ; Works Fine

[ComputerID2:EE16Params]  
BusType = 1  
Transceiver = 3  
BusNumber = 0  
IoChannelReady = 2  
IoBaseAddress = 768  
InterruptNumber = 5

[ComputerID2:ProtocolsList]  
NBF = NetBeuiParams  
NWLNKIPX = NWLINKIPXParams  
TC = TCPIPPParams  
DLC = DLCParams

[ComputerID2:TCPIPPParams]  
DHCP = Yes

[ComputerID2:SelectedServicesList]  
NWWWKSTA = InstallCSNW  
NETMON = InstallNetMon

[ComputerID2:InstallCSNW]  
!DefaultLocation = TheNovellNet  
!DefaultScriptOption = 1

**Sample setup script file used for reference for the above .UDB.**

[Unattended]  
OEMPreinstall = Yes  
NoWaitAfterTextMode = 1  
NoWaitAfterGUIMode = 1  
FileSystem = LeaveAlone  
ExtendOemPartition = 0  
ConfirmHardware = No  
NtUpgrade = no  
Win31Upgade = no  
OverwriteOemFilesOnUpgrade = yes  
TargetPath = winnt  
ComputerType = "My Super Dupper HAL - OEM", "OEM"  
KeyBoardLayout = "US-International"  
OemSkipEula = Yes

---

```
[MassStorageDrivers]
"Adaptec AHA-154X/AHA-164X - OEM" = "OEM"

[DisplayDrivers]

[KeyBoardDrivers]
"XT, AT, or Enhanced Keyboard (83-104 keys)" = "RETAIL"

[PointingDeviceDrivers]
"Microsoft Mouse Port Mouse (includes BallPoint)" = "RETAIL"

[OEMAds]
Banner = "Windows NT Sample OEM Banner"
Logo = oemlogo.bmp
Background = oembgnd.bmp

[OEMBootFiles]
TXTSETUP.OEM
AHA154X.SYS
MYHAL.DLL
OEMHAL.TAG
OEMSTOR.TAG

[GuiUnattended]

[UserData]

[Display]
BitsPerPel = 16
XResolution = 1024
YResolution = 768
VRefresh = 60
Flags = 0
AutoConfirm = 1

[Modem]

[NetWork]
InstallAdapters = AdaptersList
InstallProtocols = ProtocolsList
InstallServices = SelectedServicesList

[AdaptersList]
```

---

[EE16Params]

[ProtocolsList]

[NetBeuiParams]

[NWLINKIPXParams]

[TCPIPParams]

[DLCParams]

[SelectedServicesList]

STCPIP = InstallSimpleTCP

TCPPrint = InstallTCPPrint

[InstallCSNW]

[InstallNetMon]

[InstallSimpleTCP]

[InstallTCPPrint]

## Overview

Windows NT 4.0 includes a tool that enables you to distribute and install applications automatically during or after Windows NT setup which can significantly reduce deployment time and costs. This tool, the System Difference Tool (SYSDIFF.EXE), can be used to record the changes made to your system when an application is installed, for example, capture those changes in a "package", and then "apply" or install the package on another system during or after the setup process.

The System Difference Tool can be used to distribute files or make application configuration changes during the setup process in addition to distributing and installing applications. It can be used to pre-install applications that do not provide a "silent" or "unattended" setup. It can also be used to create "packages" to be distributed after Windows NT has been setup. Later in this section you will find step by step instructions on how to use the System Difference Tool to pre-install an application.

## Installing SYSDIFF.EXE

The System Difference Tool is included on the Windows NT Retail CD. To install the tool, simply copy SYSDIFF.EXE and SYSDIFF.INF to a directory on your hard drive. These files are located in the \support\deptools directory on your Windows NT Retail CD.

## SYSDIFF.EXE Parameters and Syntax

The form of a SYSDIFF command line is as follows:

```
SYSDIFF.EXE [/snap | /diff | /apply | /dump | /inf]
           [/log:log_file]
           [switches]
           ...
```

One of the command line parameters, /snap, /diff, /apply, /dump, or /inf must be specified. This parameter is the System Difference Tool mode. See below for descriptions of these modes and the options they use.

*log\_file* is the optional name of a log file, to which SYSDIFF

*switches* provide additional control over various operations. Some are general to all modes and others are per-mode and documented in the sections that follow. Switches that are common to all modes are as follows:

`/u` — Generate Unicode text files

When the System Difference Tool runs in /snap or /diff mode, it looks for a file called **SYSDIFF.INF** in the same directory that SYSDIFF.EXE is located.

---

SYSDIFF.INF contains information that is used to exclude certain files, registry keys, etc. from being included in a package.

### **/snap Mode**

Run the System Difference tool in **/snap mode** to create a "snap shot" of the system which will be the point of comparison for **/diff mode**.

```
SYSDIFF /snap [/log:log_file] snapshot_file
```

*snapshot\_file* may be any valid Win32 filename. A snapshot of the system will be recorded in this file.

#### **Example:**

```
SYSDIFF /SNAP /log:C:\IMAGES\SNAP.LOG C:\IMAGES\APPSNAP.IMG
```

### **/diff Mode**

Run the System Difference Tool in **/diff mode** to create a "package" or *sysdiff\_file* that includes changes made to the system after the *snapshot\_file* was created. The *sysdiff\_file* can include all of the files copied to the system, changes to the system registry, and changes made to configuration files such as the System.ini.

```
SYSDIFF /diff [/log:log_file] [/c:comment] snapshot_file sysdiff_file
```

*comment* is optional and specifies a human-readable comment to be inserted in the SYSDIFF package file. Only specify the */c* switch if you will be applying the SYSDIFF package using the **/apply mode** to be described later in this document. This comment is then used during apply mode to tell the user what is being applied. Be sure to use quotes when appropriate: i.e., */c:"This is a package title"* requires quotes (as shown).

*snapshot\_file* specifies a file generated by an earlier invocation of **SYSDIFF /snap** on the same Windows NT installation. (SYSDIFF will fail if *snapshot\_file* is from a different Windows NT installation.)

*sysdiff\_file* may be any valid Win32 filename. The specified file will be the output of SYSDIFF and will be suitable for application to a Windows NT installation and end-user setup time (via **SYSDIFF/apply**).

#### **Example:**

```
SYSDIFF /DIFF /log:C:\IMAGES\DIFF.LOG C:\IMAGES\APPSNAP.IMG  
C:\IMAGES\APPDIFF.IMG
```

### **/apply Mode**

The syntax for using SYSDIFF in **/apply mode** is:

```
SYSDIFF /apply /m sysdiff_file
```

---

*sysdiff\_file* is the file created when running the System Difference Tool in **/diff** mode.

Note the %SystemRoot% must be the same as it was on the system that was used to generate *SYSDIFF\_file*. In other words, if you generate a *SYSDIFF\_file* on a Windows NT installation in C:\WINNT, then that *SYSDIFF\_file* can be applied on other computers only if they are running Windows NT installed in C:\WINNT.

**/m** – Use this switch if you are using the System Difference Tool to pre-install a package during Windows NT setup by including the *SYSDIFF /apply* command to the CMDLINES.TXT file on your distribution share. When the System Difference Tool generates a package during **/diff** mode, it captures changes made to the directory where the current user's profile is stored. When you specify this flag, the System Difference Tool will apply these changes to the Default User profile instead. A new user that logs on to Windows NT 4.0 for the first time will get a copy of the Default User profile which will include the shortcuts to the applications created during that application setup.

For example, if you use the Administrator profile to create the System Difference Tool package, as is suggested below, shortcuts (.LNK files) for the Start Menu are typically created in the %WinDir%\Profiles\Administrator\Start Menu\Programs directory. By using the **/m** switch with the System Difference Tool in **/apply** mode, the same shortcuts will be copied to %WinDir%\Profiles\Default User\Start Menu\Programs instead of the Administrator profile. When new users log on to the system they will get a copy of the "Default User" profile.

**Example:**

```
SYSDIFF /APPLY /m C:\IMAGES\APPSDIFF.IMG
```

**Dump Mode**

This mode is useful as a diagnostic. It outputs a human-readable form of the contents of a *SYSDIFF* package to a text file. The syntax is:

```
SYSDIFF /dump sysdiff_file dump_file
```

*sysdiff\_file* is a Win32 path to a file that was created by *SYSDIFF*'s **/diff** mode.

*dump\_file* is a Win32 path to a text file that will be created and will contain the dump.

For instance, if you create a package and want to know what is in it, you can use the **/dump** switch to create a text file for later analysis.

**Example:**

---

SYSDIFF /DUMP C:\IMAGES\APPSDIFF.IMG C:\IMAGES\APPSDIFF.DMP

### **/inf Mode**

This mode is used to prepare a Windows NT distribution point for application pre-installation. The /inf mode accomplishes three primary tasks:

1. Creates an .INF that includes all of the changes to configuration files tracked when /diff mode was run. Primarily this includes changes made to the system registry.
2. Creates an \$OEM\$ directory tree that includes the files that were copied to the system between the /snap and the /diff mode. These files are extracted from the sysdiff\_file created during /diff mode and copied to the "mirror" directory structure in the \$OEM\$ directory on the Windows NT distribution share point. The directory tree is created using only 8.3 names to ensure maximum compatibility with file distribution tools. A \$\$RENAME.TXT file is created in the directory of the distribution directory containing the files that need to be converted to long filenames.
3. Adds the appropriate command line to CMDLINES.TXT to ensure that the changes are applied during setup.

The syntax for /inf mode is:

```
SYSDIFF /inf [/m] sysdiff_file oem_root
```

*/m* is the same as for /apply mode. Use of this flag is mandatory – failure to use it will result in the final result on the end user's machine not working properly.

*sysdiff\_file* is a Win32 path to a file that was created by the System Difference Tool in /diff mode.

*oem\_root* is the Win32 path of a directory. The \$OEM\$ structure will be created in this directory, and the .INF will be placed in the \$OEM\$ directory with a name based on the name of the SYSDIFF\_file. A CMDLINES.TXT will also be generated or appended (this file will always be ANSI regardless of whether /u was specified).

### **Example:**

```
SYSDIFF /INF /m C:\IMAGES\APPSDIFF.IMG Z:\
```

Tip - After you have created the snapshot file using the System Difference Tool in /snap mode, be very careful not to make changes to the system that you do not want to include in the SYSDIFF\_file created during /diff mode. It is not uncommon for someone to mistakenly change something about the system configuration, nor is it unusual because the base system is likely in a lab where a lot of testing is taking place. For example, if an application error occurs during application setup or configuration you should start over and do not make changes to the color depth, video resolution, computer name, user name, or user password. These type of mistakes can result in a lot of wasted time and energy trying to determine the source of the error. In general it is a good idea to "quarantine" your base system after the snapshot file is created.

## Building Application Images for Pre-Installation

The System Difference Tool (SYSDIFF.EXE) can be used to pre-install most standard productivity applications. When possible you should use an application's "silent mode" or "unattended mode" setup feature if it is available. In some cases you will find that SYSDIFF.EXE will not install a standard productivity application correctly.

Microsoft does not support the use of SYSDIFF.EXE for the installation of system services, applications that include system services, device drivers, or network client software. System services and device drivers write to trees or values in the registry which are dynamic or were found to be sensitive to access by SYSDIFF.EXE during testing. These trees and values are excluded from the /snap and /diff modes in the SYSDIFF.INF file. In some cases you may find that SYSDIFF.EXE will successfully install these types of software, but you cannot be sure until you test unless you know which trees and values the service or device driver you are installing writes to in the registry.

When you are in the process of creating System Difference Tool packages, it is a good practice to create and test each package separately rather than creating a snapshot file then installing several applications before creating a SYSDIFF\_file package. When you are testing it is also a good practice to test both /apply mode and /inf mode. If one mode fails try the other.

---

*Note - While you can use SYSDIFF /apply to install an application or perform testing of a SYSDIFF package, you should be aware that running the same command by adding the command to CMDLINES.TXT operates very differently (see Chapter 5, "Customizing Windows NT" for more information on CMDLINES.TXT). By running SYSDIFF /apply using CMDLINES.TXT or by using /inf mode to add a package to your distribution share, you are adding changes to the registry file NTUSER.DAT for Default User in the Profiles directory. During Windows NT setup a special User Hive is loaded into HKEY\_Current\_User. This is the only time that this Hive is accessible. After Windows NT setup is complete this Hive is written as NTUSER.DAT in the Default User profile. Each new user that logs in gets a copy of the Hive.*

---

### Three Steps to Building an Application Image

1. Create an initial snapshot of your system. Use an account that has administrative privileges on the local machine to take a "snapshot" of the state of your system. Use SYSDIFF /snap to create a snapshot file.
2. Install the application and make any configuration changes. Be very careful not to make any unintended changes to the system and use the same administrative account when installing the application.
3. Create the application package. Use SYSDIFF /diff to create the package.

### Adding an Application Image to the Distribution Server

The System Difference Tool has two modes, /inf and /apply mode, that can be used to add an application image to your distribution server. The two modes are interchangeable but have some subtle differences that may determine

---

which method would work best in your environment.

In both cases, you must add OEMPreinstall=Yes to the [Unattended] section of your Windows NT setup script file.

### **SYSDIFF /inf Mode**

SYSDIFF /inf mode is the preferred method because:

- It requires less manual effort on your part.
- It is possible to edit the package if problems arise or if you find editing necessary for some other reason.
- It can be more efficient on the network because the files being copied are usually much smaller.

Adding an application image to your distribution server in /inf mode is automated. All necessary files in the package are copied to the distribution server and the command line that installs the package is added to the CMDLINES.TXT. See Chapter 5, "Customizing Window NT" for more information on CMDLINES.TXT.

Using /inf mode makes it easier to make manual changes to the contents of the package. The Windows 95 style .INF file that is used by Windows NT setup to make the necessary changes to the registry and other system configuration files is also extracted from the System Difference Tool package. This file is a text file that can be examined and edited to alter the configuration changes that have been made in the registry or another system configuration file. Also, each of the files that were included in the application and subsequently added to the SYSDIFF package are extracted from the package, and then copied to the \$OEM\$ directory on your distribution share. If needed, you can add files or remove files from that directory. See Chapter 1, "Getting Started" for more information on the \$OEM\$ directory.

Because each file is extracted from the System Difference Tool package in /inf mode, the installation of the package will likely result in significantly smaller files being copied during the text mode phase of Windows NT setup than if /apply mode were used. Copying smaller files over a network can be more efficient than copying a very large file.

### **SYSDIFF/apply Mode**

SYSDIFF /apply mode is an alternative to /inf mode. In general it is better to use /inf mode, but you may find that /apply mode works at times when /inf mode fails. Using /apply mode can result in somewhat slower copying of the files necessary to install the application image since a single large file may copy more slowly over a network than smaller files. If a package is a large application or application suite, such as Microsoft Office, it can be many megabytes in size.

To view the changes being made using SYSDIFF /apply mode, you must use SYSDIFF /dump mode. If you want to make changes to the application image package such as changing the files copied or editing configuration

---

changes applied during the application setup, you would have to first install the application, manually edit the registry and manually remove or add any files, then create your System Difference package again.

To use /apply mode you must edit the CMDLINES.TXT to add the SYSDIFF /apply command and you must copy all necessary files including SYSDIFF.EXE, SYSDIFF.INF, and the package to the \$OEM\$ directory on your distribution server. See Chapter 5, "Customizing Windows NT" for more information on CMDLINES.TXT and Chapter 1, "Getting Started" for more information on the \$OEM\$ directory.

## Troubleshooting SYSDIFF.EXE

TIP- Check the status windows displayed by SYSDIFF.EXE for a "sysdiff aborted" message. The file listed above this entry is usually the file that SYSDIFF.EXE failed on.

SYSDIFF.EXE uses the standard Windows error message reporting facilities. These are listed in the Win32 SDK the Windows NT Resource Kit. Some of the common error messages you may come across when running SYSDIFF.EXE are listed below.

### **Error Message: System Error 5.**

Solution:

This error translates to "access denied" and may be generated when the SYSDIFF tool attempts to access restricted keys in the registry. The SYSDIFF tool does not support system service, hardware driver, or other driver installations. To resolve this issue, follow these steps:

1. Identify which program installed the system service, hardware driver, or other driver.
2. Create a new difference file without the program identified in step 1.
3. Apply the new difference file.
4. Install the program using its own installation program.

### **Error Message: An incorrect or duplicate computer name is created after applying the difference file.**

Solution:

The master computer (the computer on which the SYSDIFF tool snapped the original, or snap\_file and created the difference file) has had its computer name changed between the snap and the creation of the difference file. To correct this problem, re-create both the original file and the difference file making sure the computer name does not change or re-diff the machine if possible using the SYSDIFF.INF to exclude this registry key.

### **Error Message: Contact the Manufacturer...**

Solution:

---

If you attempt to apply a difference file to a computer on which the %SystemRoot% folder is named differently from the %SystemRoot% folder on the computer on which the difference file was created, you may receive the following error messages.

A problem exists with a file supplied by your computer's manufacturer. Contact the manufacturer and report the following:

A snapshot or /diff file specified on the command line was created with a different sysroot and cannot be used now.

Click OK. Setup will continue but certain applications or other features may not work correctly.

#### **Error Message: Diff Failed (error=2)**

Solution:

This error message indicates you spelled one of the SYSDIFF command line arguments incorrectly. For example, sysdiff /diff base.img offic.img, but base.img is really called base2.img.

#### **Error Message: Diff Failed (error=32)**

Solution:

This error message indicates that some of the files the SYSDIFF tool is trying to read are in use. To resolve this issue, use one of the following methods:

- Close all programs, restart the computer, and then run the SYSDIFF tool without running any other programs.
- Use the [ExcludeFiles] section in the SYSDIFF.INF file.

To verify that you are using the correct SYSDIFF utilities, check the time/date stamps:

```
Sysdiff.exe 8/2/96 1:30AM 68,368 bytes NTW
Sysdiff.exe 8/9/96 1:30AM 68,368 bytes NTS
```

Also, if problems are suspected with the SYSDIFF.INF, try using the one that is on the Windows NT CD:

```
Sysdiff.inf 8/2/96 1:30AM 2,500 bytes NTW
Sysdiff.inf 8/9/96 1:30AM 2,500 bytes NTS
```

---

**Problem: Empty directories on the master machine are not processed by SYSDIFF /snap.**

Solution:

Place a dummy file in the directory.

**Problem: .INI files may not be updated or copied during SYSDIFF/apply or SYSDIFF/inf.**

Solution:

Instead of relying on SYSDIFF to update the .INI file, manually update the .INI file, then copy it to the appropriate directory under the \$OEM\$\<Drive Letter>.

**Problem: Networks with limited bandwidth experience problems when doing SYSDIFF /inf to the distribution server.**

Solution:

Perform the operation to the local drive, then manually move the \$OEM\$ tree over to the distribution server via XCOPY.EXE.

**Problem: Package file dates are changed.**

When the SYSDIFF package is used to apply software to computers running Windows NT Server or Workstation, the dates on files that are applied via the package are changed from their original date to the current date. This happens whether you are doing a SYSDIFF /apply or a SYSDIFF /inf.

Solution:

Get the updated version of SYSDIFF.EXE from Microsoft's FTP site. Consult the Microsoft Knowledge Base for the exact location.

---

Tip - If your SYSDIFF.EXE package is causing Windows NT setup to hang or SYSDIFF.EXE stops responding (hangs) when using /apply mode or /inf mode, in some cases the cause of the hang is a single entry or update that SYSDIFF.EXE is trying to apply to the system. To determine where it is failing, you can comment out portions of the .INF file that are created in /inf mode. For example, you could comment out portions of the .INF file until the package installs completely. Then narrow the the portion that is being commented out until you identify the line or section that is causing the problem. Once this has been identified, you can determine whether there are other methods that can be used to apply that change to your system.

**Problem: The computer stops responding (hangs) when you use the SYSDIFF /apply command.**

**Solutions:**

The target computer is running low on disk space or has run out of disk space. Free some disk space and reapply the difference file.

There may be a problem with one of the applications contained in the image. Try smaller images or image the applications one at a time until you determine which one is causing the problem. Create a new package without the problem application.

There may be a problem updating an .INI file. See "SYSDIFF.EXE /apply or /inf fails when updating an .INI file or fails to copy the .INI files." below.

The difference file may be damaged. This can occur for various reasons, but the most common one is the difference file becomes damaged while being created across the network. To verify the integrity of the difference file, follow these steps:

1. Try to manually apply the difference file instead of using the /inf switch.
2. Create the difference file on the local computer and then copy it to the network server instead of creating the file across the network.
3. Manually apply the difference file from the local hard disk.

**Problem: The SYSDIFF tool takes a long time to finish and the image file is extremely large.**

**Solution:**

If you are creating the image on the local hard disk, make sure to exclude the folder being used to store the image file in the SYSDIFF.INF file in the [ExcludeDirectoryTrees] section. For example, if you were running the SYSDIFF tool from the C:\Image folder, you would create an entry similar to:

```
[ExcludeDirectoryTrees]
C:\Image
```

**Problem: When you run the SYSDIFF tool, it appears on the screen briefly and then nothing else happens.**

**Solution:**

This can occur when an incorrect syntax for the SYSDIFF tool is used on the command line.

**Problem: Some of the changes are not applied when you run the SYSDIFF /apply command.**

---

Solution:

When you apply a difference file, it may seem that some user-specific settings are not applied. These settings are contained in HKEY\_USERS or HKEY\_CURRENT\_USER in the registry. Because these settings are user-specific, they are not displayed if you log on with a different account than the one that created the difference file. The use of Profiles and Policy files provide the functionality needed to provide a uniform and controlled registry setting for users.

**Problem: Network drives appear in My Computer after you apply a difference file.**

Solution:

Network drives have been created after snapping the base file but before the difference file are applied to the new installation. These settings are in the registry keys and are shown on a per-user basis. If you do not want additional network drives to be mapped for a user, you must disconnect them before creating the difference file.

**Problem: Temporary files are left in the folder where you are creating SYSDIFF.EXE files.**

Solution:

This behavior can occur if you end the SYSDIFF tool prematurely. Delete the temporary files and re-create the SYSDIFF files.

**Problem: SYSDIFF.EXE /apply or /inf fails when updating an .INI file or fails to copy the .INI files.**

Solution:

If SYSDIFF.EXE is hanging when updating an .INI file or is failing to update or copy .INI files, then the .INI files can be updated manually and copied to the appropriate directories under \$OEM\$ to ensure they are distributed properly.

---

### Sample SYSDIFF.INF File

```
[Version]
Signature = "$Windows NT$"

[ExcludeDrives]
;
; The first character on each line is the drive letter
; of a drive to exclude.
;
;c
;d

; General notes for file/dir exclusion sections:
;
; *: refers to all drives.
; ?: refers to the drive with the system on it.
; :: is substituted with %systemroot%
;
; Lines that are not in valid format (such as those that
; don't start with x:\) are ignored.
;

[ExcludeDirectoryTrees]
;
; Each line is a fully-qualified path of a tree to
; be excluded. The directory and all of its subtrees
; are excluded.
;
*:\recycled
*:\recycler

[ExcludeSingleDirectories]
;
; Each line is a fully-qualified path of a directory to be
; excluded. The directory's subdirs are NOT excluded.
;
::\system32\config
::\system32\logfiles

[ExcludeFiles]
; Each line is a fully-qualified path of a file to be excluded.
; If it does not start with x:\ then we assume it's a filename part
```

```

; for a file to be excluded where ever it is found.
;
*:\pagefile.sys
ntuser.dat
ntuser.dat.log

[IncludeFilesInDir]
;
; Each line in here is a fully qualified path of a directory
; whose files are all to be included in a diff (marked as
; added/changed). Use this if you want to include files in the diff
; that might not have actually been changed.
;

[ExcludeRegistryKeys]
;
; Each line indicates a single registry key to be excluded.
; Subkeys of this key are not excluded.
;
; The first field is one of HKLM or HKCU
; The second field is the subkey, which must NOT start with a \.
;
HKLM,System\Disk
HKLM,System\Select
HKLM,System\Setup
HKCU,Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU

[ExcludeRegistryTrees]
;
; Each line indicates a registry key and subkeys to be excluded.
;
; The first field is one of HKLM or HKCU
; The second field is the subkey, which must NOT start with a \.
;
HKLM,SYSTEM\ControlSet001
HKLM,SYSTEM\ControlSet002
HKLM,SYSTEM\ControlSet003
HKLM,SYSTEM\ControlSet004
HKLM,SYSTEM\ControlSet005
HKLM,SYSTEM\ControlSet006
HKLM,SYSTEM\ControlSet007
HKLM,SYSTEM\ControlSet008
HKLM,SYSTEM\ControlSet009
HKLM,SYSTEM\CurrentControlSet\Enum
HKLM,"SOFTWARE\Microsoft\Windows NT\CurrentVersion\perflib"

```

---

[ExcludeRegistryValues]

;

; Each line indicates a registry value entry to be excluded.

;

; The first field is one of HKLM or HKCU.

; The second field is the subkey, which must NOT start with \.

; The third field is the value entry name.

## Overview

After you have automated Windows NT setup and the installation of applications, you will likely find that you still have a list of customizations or tasks that must be completed before turning the computer over to the end user. You may want to change the password policy for local machine accounts, disable and delete a service, or execute some other task that is outside of the capabilities of the Windows NT deployment tools. In this chapter you will find examples of tools and methods that can be used to automate these tasks. The examples provided are solutions to some common questions. For the answer to your specific question about how to automate a post-setup task, consult the Microsoft On-line Knowledge Base and Microsoft Tech-Net, your question may have already been asked and answered.

## Distributing Files Automatically Using Windows NT Setup

In some cases the only customization you may want to perform is as simple as copying files to the computer you are installing Windows NT on. These files can be just about any file, for example they might be templates for word processing, macros for a spreadsheet application or email forms. See the "Copy Custom Files Using the \$OEM\$ Directory" section in Chapter 1, "Getting Started" for more information about distributing files using Windows NT setup.

### Customizing the Start Menu

Manually customizing the Start Menu involves organizing the files and directories in the "<SYSTEM ROOT>\Profiles\Start Menu" directory. The easiest way to automate any customizations you want to make is to use the Application Pre-Installation Tool (SYSDIFF.EXE) to track and apply additions to this directory. Alternatively, you can use System Policies to store all or part of the Start Menu on a network share point. This will allow you to centrally manage the files and shortcuts for each user's Start Menu and Desktop.

See the Windows NT Resource Kit for more information about System Policies. Refer to Chapter 4, "Application Pre-Installation (SYSDIFF.EXE)" in this document for detailed information on using SYSDIFF.EXE.

Follow the instructions in Chapter 4, "Application Pre-Installation (SYSDIFF.EXE)" for creating a package, but rather than installing an application, copy any files or shortcuts to the Start Menu for the current user that you would like to appear on your custom Start Menu. You can also edit the "\Profiles\Default User\Start Menu" directory or the "All Users" directory.

If you copy files to the current user's Start Menu profile, you must use the optional /m switch when running "SYSDIFF /inf." The /m switch will "re-map" the changes from the current user's Start Menu to the "Default User." By mapping the changes to the "Default User", you ensure that each user that logs on will get a copy of your customized Start Menu. If you do not use the /m switch, a copy of the current user's Start Menu will be copied to each system that you install from your network distribution point.

---

When you run "SYSDIFF /inf /m" SYSDIFF.EXE will create an \$OEM\$ directory and copy all of the files that you added to the \$OEM\$\\$\\$Profiles directory on your distribution point. SYSDIFF will also create a special file that is used to translate the short filenames on the distribution share to long filenames.

---

*Note: If you only intend to make and track changes to the system in your directory structure, you should delete the command that SYSDIFF adds to the CMDLINES.TXT file in the \$OEM\$ directory on your distribution share. This command line executes the .INF file which includes all of the registry and other configuration file changes that SYSDIFF tracked. To determine which command line to delete, there will be more than one if you have run SYSDIFF /inf with more than one package you have created – find the command line that has the same suffix as your SYSDIFF\_file. For example if you used the following command when running SYSDIFF /inf:*

---

```
SYSDIFF /inf /m mydiff
```

Then the command line the CMDLINES.TXT would be:

```
"rundll32 setupapi,InstallHinfSection DefaultInstall 128 .MYDIFF.INF"
```

The "MYDIFF.INF" file in this example contains any configuration changes made when customizing the Start Menu. If all you did was copy files, this line is unnecessary and should be removed from the CMDLINES.TXT file.

## Tools to Customize Windows NT

The first step to automating the process of customizing Windows NT is to find a tool that will make the necessary changes in your system configuration. To automate the process, the tool should include a feature that will allow you to execute the process without user interaction. In many cases this means that the tool can be run from a command line with command line parameters that specify the action to be taken. In this section you will find some tools and examples that demonstrate how they might be used.

### REGEDIT.EXE

Windows NT 4.0 includes two registry editing tools: REGEDT32.EXE and REGEDIT.EXE. Both tools allow you to edit the Windows NT registry but REGEDIT.EXE includes a command line switch feature, "/s", that allows you to silently add and change registry keys and values. This is a powerful feature that can be used to customize the Windows NT 4.0 configuration.

REGEDIT.EXE is one of several tools that can be used to make "silent" or "scripted" changes to the registry. If you find that limitations of REGEDIT.EXE, like the inability to delete a registry key or value make it difficult or impossible to accomplish the needed customizations, you should evaluate the functionality of other registry editing tools such as REGINI.EXE. For more information about REGINI.EXE see the Windows NT Resource Kit.

The first step when using REGEDIT.EXE to modify your Windows NT 4.0 configuration is to locate the key(s) or value(s) that you must edit. If you do not

---

know which to edit or add, the easiest way to locate them is to export the registry before and after making the change you wish to automate and then compare the two text files using a file comparison utility. For example, to determine what keys and values are changed when changing the Wallpaper on your desktop:

1. Run REGEDIT.EXE, highlight "My Computer", then choose "Registry" and "Export Registry file..." from the menu. Select a filename for the exported registry file.
2. Change the Wallpaper on your desktop.
3. Repeat step 1, but choose a different filename for the exported registry file.
4. Run the file comparison utility, FC.EXE, to compare the two files. Here is an example command line if the filenames you selected were "BEFORE.REG" and "AFTER.REG":  
FC before.reg after.reg
5. From the file comparison utility you would learn that the value in the registry that has changed is "wallpaper". With this information you can use the search feature of REGEDIT.EXE to find the branch that includes "wallpaper". In this case it is "HKEY\_CURRENT\_USER\Control Panel\Desktop".
6. Using your mouse, select the branch in REGEDIT.EXE and export the selected branch to a file – WALLPPR.REG for example. It is recommended that you edit the text file that is exported so only the value which makes the change is included. This will ensure that you do not make any unintended changes to the registry.

The next step is to test your exported registry file. To do this simply double-click the .REG file you created to import the changes into the registry. Once you have verified that the imported .REG file works as expected, add a command to your CMDLINES.TXT file that imports the .REG file silently. Using the example above the command would be:

```
"REGEDIT.EXE /S .\WALLPPR.REG"
```

You must also copy REGEDIT.EXE and WALLPPR.REG to the \$OEM\$ directory on your distribution server. The "/S" switch causes REGEDIT.EXE to import the registry file in "silent" mode. If you do not use this switch, the user will be prompted with a message stating that "The information has been successfully entered into the registry" and an "OK" button the must be clicked to continue.

For information about the CMDLINES.TXT, see "Using CMDLINES.TXT to Automate Program Execution " later in this chapter.

### *Configuring a System to Automatically Logon and Execute a Program*

By configuring a system to automatically logon the administrator and execute a program or a batch file, you can automate many of the steps in your procedure for customizing Windows NT 4.0.

The following example uses the "Auto Admin Logon" feature and the "Run Once" feature to automatically logon the Administrator after Windows NT 4.0 has completed setup on the first logon, then execute a program named "CUSTOM.COM" at this first logon.

1. Using a text editor, create a file named AUTOLOG.REG with the following information:

```
REGEDIT4
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\
Winlogon]
"DefaultUserName"="Administrator"
"AutoAdminLogon"="1"
"DefaultPassword"=""
```

DefaultUserName can be any valid account. During setup, the only account installed that has access to configuring the system is the local administrator account.

The AutoAdminLogon feature disables itself after the first logon if the DefaultPassword is blank. The user will be prompted for a logon name and password on any subsequent logons.

---

*Note: To create text files, you should use the EDIT.EXE utility included with Windows NT instead of NOTEPAD.EXE since EDIT.EXE does not use UNICODE characters.*

---

2. Using a text editor, create a file named RUNONCE.REG with the following information:

```
REGEDIT4

[Hkey_Local_Machine\Software\Microsoft\Windows\CurrentVersion\Ru
nOnce]
"RunThis"="c:\batch\custom.cmd"
```

3. If a \$OEM\$ directory does not exist on your Windows NT distribution share create one. See "Structure of the Distribution Share Point" in Chapter 1, "Getting Started" for information about the \$OEM\$ directory. Copy AUTOLOG.REG, RUNONCE.REG,

---

and REGEDIT.EXE to the \$OEM\$ directory. REGEDIT.EXE is located in the WINNT (%SYSTEMROOT%) directory by default.

4. Create a \$OEM\$\C\BATCH directory on your distribution share and copy your CUSTOM.COMD file in this directory. The CUSTOM.COMD file should be included and command lines needed to automate your customizations for Windows NT 4.0. See the next section in this chapter for an example.
5. If a CMDLINES.TXT file does not exist in the \$OEM\$ directory, create one with a text editor. Enter the following information:

[Commands]

```
".\REGEDIT.EXE /S .\AUTLOG.REG"
```

```
".\REGEDIT.EXE /S .\RUNONCE.REG"
```

If a CMDLINES.TXT exists, edit it so it includes the two new lines. Do not add another [Commands] section.

6. Edit the Windows NT 4.0 setup script file that is being used and add the following line to the [UNATTENDED] section:

```
OEMPreInstall = YES
```

### *Configuring System to Skip the Welcome Screen*

1. Using a text editor, create a file named AUTOLOG.REG with the following information:

```
REGEDIT4
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\  
Winlogon]
```

```
"DefaultUserName"="Administrator"
```

```
"AutoAdminLogon"="1"
```

```
"DefaultPassword"=""
```

DefaultUserName can be any valid account. During setup, the only account installed that has access to configuring the system is the local Administrator account.

The AutoAdminLogon feature disables itself after the first logon if the DefaultPassword is blank. The user will be prompted for a logon name and password on any subsequent logons.

---

*Note: To create text files you should use the EDIT.EXE utility included with Windows NT instead of NOTEPAD.EXE because EDIT.EXE does not use*

---

*UNICODE characters.*

---

- Using a text editor, create a file named NOWELC.REG that contains the following six lines.

```
REGEDIT4
```

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explore  
\Tips]  
"DisplayInitialTipWindow"=dword:00000000  
"Show"=hex:00,00,00,00  
"Next"=hex:03,00
```

---

*Note: Line two is blank, but must be present.*

---

- If a \$OEM\$ directory does not exist on your Windows NT distribution share, create one. See "Structure of the Distribution Share Point" in Chapter 1, "Getting Started" for information about the \$OEM\$ directory. Copy NOWELC.REG and REGEDIT.EXE to the \$OEM\$ directory. REGEDIT.EXE is located in the WINNT (%SYSTEMROOT%) directory by default.
- Create a \$OEM\$\C\BATCH directory on your distribution share and copy your CUSTOM.COMD file in this directory. The CUSTOM.COMD file should include the command lines needed to automate your customizations for Windows NT 4.0. See the next section in this chapter for an example.
- If a CMDLINES.TXT file does not exist in the \$OEM\$ directory, create one with a text editor. Enter the following information:

```
[Commands]  
".\REGEDIT.EXE /S .\NOWELC.REG"
```

If a CMDLINES.TXT exists, edit it so it includes the two new lines. Do not add another [Commands] section.

- Edit the Windows NT 4.0 setup script file that is being used and add the following line to the [UNATTENDED] section:

```
OEMPreInstall = YES
```

*Executing a Batch File on First Logon to Customize Windows NT*

The previous section includes an example of the Windows NT "RunOnce" feature. In this example you will find a sample batch or .CMD file that includes

examples of programs which can be used to customize Windows NT 4.0. Be sure that the batch file has been thoroughly tested before implementing. Example commands for .CDM or .BAT file:

```
net use z:\myserver\setupshare
net use z: /d
net accounts /forcelogoff:20
net accounts /minpwlen:5
net accounts /maxpwage:120
net accounts /minpwage:30
net accounts /uniquepw:2
net config server /srvcomment:"Windows NT 4.0"
net user SuperUs1 /ADD
net localgroup MYGROUP1 /ADD /COMMENT:"Super Group 1"
net localgroup MYGROUP1 SuperUs1 /ADD
net share ROOT=C:\ /unlimited /remark:"The door is wide open"
net user administrator password
```

For more information on the NET.EXE program and other programs included that may be used to help customize Windows NT, run the NTCMDS.HLP help file.

Some applications or utilities may not include a feature which allows you to silently execute the program without user interaction. In some cases you can work around simple prompts by these programs by using the "pipe" to send a character to the application. For example, if you wanted to use the "Change ACLS" utility, CACLS.EXE, to setup NTFS security features you would find that the user is prompted with a "ARE YOU SURE?" message. The CACLS.EXE command line utility does not provide a /Y switch that automatically answers with Y for Yes to the "ARE YOU SURE? Y/N" prompt. However, you can use the echo command to pipe the character Y to CALCS.EXE in a batch file:

```
echo y| cacls <filename> /g <username>:<permission>
```

---

**Note:** *There is no space between the "y" and the pipe symbol "|".*

---

---

**SPECIAL NOTE ABOUT CACLS.EXE.:** *If you do not want to replace the Access Control List (ACL), use the /E (edit) option. You should be careful when editing the ACL because it is very possible to change the system so it becomes inaccessible. Do not deny access to any Windows NT system files or directories.*

---

#### *Adding Silent Application Setup Commands to a Batch File*

Applications that can be setup silently can also be run using the "Run Once" feature. For example, to install Microsoft Office 95 or Microsoft Office 97 you could add the following command line to your batch file to run Microsoft Office setup in "quiet mode":

---

```
\\products\msoffice\setup /q /k"<CD Key>"
```

<CD Key> is the CD Key from your Microsoft Office CD

### *Customizing Windows NT Logon*

The following sample .REG files include examples of some commonly requested Windows NT Logon customizations:

#### *Changing the Logon Caption and Text*

LOGON.REG:

REGEDIT4

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon]
"LegalNoticeCaption"="This is the Legal Notice Caption"
"LegalNoticeText"="This is the Legal Notice Text"
"LogonPrompt"="Enter a user name and password that is valid for this system."
"Welcome"="Good morning and welcome to a new day at Widgets Are Us!"
```

---

*Note: If you are using the previous examples to automatically logon as administrator and run program, you should not import the LegalNoticeCaption and LegalNoticeText registry changes using the CMDLINES.TXT file. If you do the system will not automatically logon the administrator because of the dialogue boxes these registry settings add to the logon process. Instead, add the command line to import these registry settings to the "RunOnce" key.*

*"Welcome" - Default is title only, no message. The text entered appears in the caption bar beside the title of the Begin Logon, Logon Information, Workstation Locked, and Unlock Workstation dialog boxes. This value entry does not appear in the Registry unless you add it.*

---

#### *Customizing Windows NT Logon Options*

REGEDIT4

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon]
"DontDisplayLastUserName"="1"
"PowerdownAfterShutdown"="1"
"RunLogonScriptSync"="0"
"ShutdownWithoutLogon"="1"
```

*"DontDisplayLastUserName" - By default, Windows NT displays the name of the last person to logon in the Username space of the Logon Information dialog box. If you add this value entry and set it to 1, the Username space is always blank when the Logon Information dialog box appears.*

*"PowerdownAfterShutdown" - Default: 0 on Windows NT Server, 1 on*

Windows NT Workstation. Determines whether the Shut Down and Power Off option appears in the Shut Down Computer dialog box. (This dialog box appears when you press CTRL+ALT+DELETE and then click the Shut Down button.) The option appears only if the value of this entry is 1. See also ShutdownWithoutLogon and NoClose.

"RunLogonScriptSync" - Default: 0. Determines whether the logon script and Program Manager are synchronized. If the value of this entry is 1, Program Manager does not begin loading the desktop until the logon script has finished running. If the value is 0, the logon script and Program Manager can run simultaneously.

---

*Note: This value entry also appears in HKEY\_CURRENT\_USER\Software\Microsoft\Windows NT\CurrentVersion\Winlogon. The HKEY\_LOCAL\_MACHINE value applies to all users. The HKEY\_CURRENT\_USER value applies only to the current user. You can use the System Policy Editor to change this value.*

---

"ShutdownWithoutLogon" - Default: 0 on Windows NT Server, 1 on Windows NT Workstation. Specifies whether the Shut Down button in the Logon Information dialog box is enabled. If the value is set to 1, users can click the Shut Down button to stop the operating system without logging on or turning off power to the computer. If it is set to 0, the Shut Down button is disabled. See also NoClose and PowerdownAfterShutdown.

#### Specifying Default Shut Down Settings

#### REGEDIT4

```
[HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Shutdown]
"LogoffSetting"="0"
"ShutdownSetting"="0"
```

The Shut Down subkey contains values that enable you to specify and retain default Shut Down settings. This subkey is not in the Registry unless you add it.

"LogoffSetting" - Default: 0. Specifies the default option for the Logoff dialog box. Valid values are:

- 0 = Logoff
- 1 = Shut Down
- 2 = Shut Down and Restart
- 3 = Shut Down and Power Off (when supported).

"ShutdownSetting" - Default: 0. Specifies the default value for the Shut Down Computer dialog box. This dialog box appears when you press CTRL+ALT+DELETE and then click the Shut Down button. Valid values are:

- 0 = Logoff
- 1 = Shut Down

- 2 = Shut Down and Restart
- 3 = Shut Down and Power Off (when supported)

### Executing Commands During Windows NT Setup - CMDLINES.TXT

Windows NT setup includes a feature that will allow you to execute a program during the graphical mode portion of the setup. To enable this feature, you must use a Windows NT setup script (UNATTEND.TXT) file. The setup script must include "OEMPreInstall=Yes" in the [Unattended] section of the setup script. Each of the programs or commands that you want to execute must be added to a text file named CMDLINES.TXT. Then you must copy the file to \$OEM\$ directory on your Windows NT distribution share. For more information on the \$OEM\$ directory see Chapter 1, "Getting Started."

---

*Note: Some applications will not run during the graphical mode portion of setup because users have not been created by Windows NT setup. To run applications that fail to run during Windows NT setup, use the "Run Once" feature described in the "Configuring System to Automatically Logon and Execute a Program" section earlier in this chapter. Also, the application or command should have a silent scripted feature so no user interaction is required.*

---

When the commands listed in CMDLINES.TXT are executed, the current directory will be the \$OEM\$ directory. To change the current directory you must specify the change directory command, CD, in the CMDLINES.TXT file.

Syntax for the CMDLINES.TXT file:

```
[Commands]
"<command_1>"
"<command_2>"
.
.
"<command_x>"
```

Where:

<command\_1>, <command\_2> and so forth refer them to the commands in the order you want them to run when CMDLINES.TXT is called by GUI Mode Setup. Note that all commands must appear in quotation marks ("").

### Using the Windows NT Setup Engine, SETUPAPI.DLL

Windows NT 4.0 includes a setup engine, SETUPAPI.DLL, that can be used to copy files, update the registry and other configuration files, and reboot the system if necessary. This is a very powerful tool that can be used to customize Windows NT 4.0 and distribute software. SETUPAPI.DLL is the same engine that the Application Pre-Installation Tool (SYSDIFF.EXE) uses to install software. You must use a Windows 95 style .INF file to provide the

setup engine with the instructions necessary to complete your task. Information on writing and editing Windows 95 style .INF files can be found in the Win32 Software Development Kit (SDK) and Appendix C of the "Windows 95 Resource Kit." Because SETUPAPI.DLL is not an executable file, you must use an executable "wrapper" that will load and call the routines contained in SETUPAPI.DLL. You can use the RUNDLL32 system command to execute an Install section in an .INF file. The syntax of the command line is:

```
RUNDLL32 SETUPAPI.DLL,InstallHinfSection <section> <reboot-mode> <inf-name>
```

<section> parameter is any Install section in the .INF file.

<reboot-mode> parameter determines which of five reboot modes should be used.

---

*Note: The recommended values for <reboot-mode> are four, if the .INF file is a Windows NT-supplied .INF file, or 132 if the INF file is provided by you. Using any of the other values shown below may cause the machine to be rebooted unnecessarily or cause the machine not to be rebooted when it should be.*

---

The following list describes each reboot mode:

#### Never Reboot

Set <reboot-mode> to 0 or 128. Whatever happens, the PC will not be rebooted. It's up to the client to determine if the PC should be rebooted. For setup, this means there is a file C:\WINDOWS\WININIT.INI that is not zero bytes in size.

#### Always Silent Reboot

Set <reboot-mode> to 1 or 129. The user will not be prompted with a "Reboot the machine, Yes/No" dialog and the PC will always reboot.

#### Always Prompt Reboot

Set <reboot-mode> to 2 or 130. The user will always be asked to respond to a "Reboot the machine, Yes/No" dialog. Setup does not attempt to determine if a reboot is necessary.

#### Silent Reboot

Set <reboot-mode> to 3 or 131. If setup determines that the PC needs to reboot, there is no user interaction.

#### Prompt Reboot

Set <reboot-mode> to 4 or 132. If setup determines that the PC needs to reboot, it prompts the user with a "Reboot the machine, Yes/No" dialog.

For example, the following command line installs the Games optional

component and, if Windows NT setup determines a reboot is necessary, asks the user if they want the computer to reboot immediately after the installation is complete.

RUNDLL32 SETUPAPI.DLL, InstallHinfSection games 4 applets.inf

---

*Notes: SETUPAPI does not check for disk space required to install any files being copied. This is the responsibility of the administrator.*

---

Your installation process should not include commands that are to execute after the RUNDLL32 command has executed because the system may reboot immediately at that point, before returning control of the system to another process. If your installation process requires other commands that are to execute after the RUNDLL32 command, you should use the RunOnce feature. See the "Configuring System to Automatically Logon and Execute a Program" section earlier in this chapter for more information on the RunOnce feature.

The following example makes the same changes to the registry as the example in the "Configuring System to Automatically Logon and Execute a Program" section above. The system is configured to automatically logon the administrator and execute a program or a batch file.

AUTOLOG.INF

[Version]

Signature = "\$Windows NT\$"

[DefaultInstall]

AddReg = AddRegistryKeys

[AddRegistryKeys]

HKLM,"SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon","DefaultPassword",,""

HKLM,"SOFTWARE\Microsoft\Windows

NT\CurrentVersion\Winlogon","DefaultUser",,"Administrator"

HKLM,"SOFTWARE\Microsoft\Windows

NT\CurrentVersion\Winlogon","AutoAdminLogon",,"1"

HKLM,"SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce","RunThis",,"c:\batch\custom.cmd"

To execute AUTOLOG.INF the following line would be added to  
CMDLINES.TXT.

[Commands]

"rundll32 setupapi,InstallHinfSection DefaultInstall 128 .\autolog.INF"

### *Removing the Gopher and World Wide Web (WWW) Services*

This example .INF file removes the Gopher Service and the World Wide Web (WWW) services. These services are installed with Peer Web Services.

REGDEL.INF

[version]  
signature="\$Windows NT\$"

[DefaultInstall]  
DelReg = DeleteRegistryKeys

[DeleteRegistryKeys]  
HKLM,"System\CurrentControlSet\Services\GOPHERSVC"  
HKLM,"Software\Microsoft\NetMgmt\Parameters\AddOnServices","Gopher"  
HKLM,"System\CurrentControlSet\Services\W3SVC"  
HKLM,"Software\Microsoft\NetMgmt\Parameters\AddOnServices","WWW"

To execute REGDEL.INF the following line would be added to a batch file after you have installed Peer Web Services:

```
RUNDLL32 SETUPAPI,InstallHinfSection DefaultInstall 128 .\regdel.inf
```

## **Automating Installation of Windows NT Service Packs**

There are three ways to automate the installation of a service pack.

### **Method 1**

Automatically logon as the administrator, then use the "Run Once" feature to run the Windows NT Service Pack Installation Program, UPDATE.EXE, in silent mode.

See the earlier section, "Configuring System to Automatically Logon and Execute a Program", for an example of automatically logging on as the administrator and running a program. Add the command line for running UPDATE.EXE in silent mode to the "RunOnce" key in the registry. Example command line:

```
\\NTFILES\SP\UPDATE.EXE /U /Z
```

### **Method 2**

Use the Windows NT setup switch, /E, to specify a command to be executed at the end of graphical mode setup. Example:

```
WINNT.EXE /U:UNATTEND.TXT /S:X:\ /E:"X:\SP\UPDATE.EXE /U /Z"
```

### **Method 3**

---

If you are using the Disk Duplication method of distributing Windows NT this is the recommended method of automating installation of Windows NT Service Packs because all needed files are copied to the hard drive before duplication rather than copying the files after the graphical mode portion of setup is complete. See Chapter 1 in this guide for more information about disk duplication.

1. Copy the Windows NT Service Pack source files to your network distribution point.

Example:

```
copy d:\i386\*. * c:\dist\%oem$
copy d:\spcdrom.40 c:\dist\%oem$
copy d:\disk1 c:\dist\%oem$
```

Where d: is the CD ROM drive letter and i386 is the name of the target platform.

2. Edit or create a CMDLINES.TXT file (using Notepad, for example) with the following content. Save the file in the %oem\$ directory of your network distribution sharepoint.

```
[Commands]
".\update /u /z"
```

3. Edit or create your Window NT setup script file (UNATTEND.TXT) to ensure that it contains the following line under in the [Unattended] section:

```
[Unattended]
OemPreinstall = yes
```

## Automating Selection of Windows Accessories and Components

Each Windows NT component or accessory has an associated setup information file (.INF). Each .INF uses the value InstallType to specify whether the application is to be installed.

InstallType values:

```
0 = Manual Only
10 = Typical or Custom
14 = Typical, Custom, or Portable
```

The following .INF files use the InstallType variable. The list of .INF files can be

---

found under the [BaseWinOptionsInfs] in the SYSSETUP.INF.

```
accessor.inf
communic.inf
games.inf
mmopt.inf
multimed.inf
optional.inf
pinball.inf
wordpad.inf
```

If none of the options in a particular .INF are desired, a semicolon can be placed at the beginning of the line.

The following example is the section for Free Cell from the GAMES.INF for Windows NT 4.0 Server. By default, games are not installed on Windows NT Server 4.0. Since Unattended Setup uses Typical for installation purposes, a value of 10 can be used for InstallType. By default InstallType for Free Cell is 0.

This is the value that is change. To install Free Cell, change InstallType value to 10.

```
[Freecell]
OptionDesc = %Freecell_DESC%
Tip        = %Freecell_TIP%
IconIndex  = 62 ;Windows mini-icon for dialogs
Parent     = Games
InstallType = 0 ;Manual only
CopyFiles  = FreecellCopyFilesSys, FreecellCopyFilesHelp
AddReg     = FreecellAddReg
UpdateInis = FreecellInis
Uninstall  = FreecellUninstall
Upgrade    = FreecellUpgrade
Detect     = %11%\freecell.exe
```

To take advantage of the InstallType variable for manual or unattended installations of Windows NT, a certain amount of preparation is needed.

1. The contents of the I386 directory from the Windows NT 4.0 CD needs to be copied to a distribution share.
2. For each .INF file that needs to be modified, the file first needs to be expanded and the original file renamed.
  - A. EXPAND GAMES.IN\_ GAMES.INF
  - B. RENAME GAMES.IN\_ GAMES.SAV

*Note: The Windows NT version of EXPAND.EXE needs to be used.*

3. Edit the .INF file and change the InstallType value to one of the following values:

- 0 = Manual Only
- 10 = Typical or Custom
- 14 = Typical, Custom, or Portable

#### **List of Accessory .INFs and their Settings**

##### **ACCESSOR.INF**

Calculator	InstallType = 10 ;Typical, Custom
Character Map	InstallType = 10 ;Typical, Custom
Clipboard Viewer	InstallType = 14 ;Typical, Portable, Custom
Clock	InstallType = 14 ;Typical, Portable, Custom
Desktop Wallpaper	InstallType = 0 ;Manual
Document Templates	InstallType = 10 ;Typical, Custom
Mouse Pointers	InstallType = 0 ;Manual
Object Package	InstallType = 14 ;Typical, Portable, Custom
Paint	InstallType = 10 ;Typical, Custom
Quick View	InstallType = 10 ;Typical, Custom
Screen Savers Open GL	InstallType = 10 ;Typical, Custom
Screen Savers Standard	InstallType = 10 ;Typical, Custom

##### **COMMUNIC.INF**

Chat	InstallType = 14 ;Typical, Portable, Custom
Phone Dialer	InstallType = 14 ;Typical, Portable, Custom
Hyper Terminal	InstallType = 14 ;Typical, Portable, Custom

##### **GAMES.INF**

Freecell	InstallType = 0 ;Manual
Mine Sweeper	InstallType = 0 ;Manual
Solitaire	InstallType = 0 ;Manual

##### **PINBALL.INF**

Pin Ball	InstallType = 0 ;Manual
----------	-------------------------

##### **MMOPT.INF**

Media Options	InstallType = 10 ;Typical, Custom
Musica Sound Scheme	InstallType = 0 ;Manual
Jungle Sound Scheme	InstallType = 0 ;Manual
RobotZ Sound Scheme	InstallType = 0 ;Manual
Utopia Sound Scheme	InstallType = 0 ;Manual

##### **MULTIMED.INF**

CD Player	InstallType = 14 ; Custom, Typical, Laptop
-----------	--

---

Media Player	InstallType = 14 ; Custom, Typical, Laptop
Sound Recorder	InstallType = 14 ; Custom, Typical, Laptop
Volume Control	InstallType = 14 ; custom, typical, laptop

#### **OPTIONAL.INF**

Accessibility Options	InstallType = 14 ;Typical, Portable, Custom
-----------------------	---

#### **WORDPAD.INF**

Word Pad	InstallType = 10 ;Typical, Custom
----------	-----------------------------------

### **Automating Installation of Peer Web Services**

Windows NT Workstation Peer Web Services includes a silent installation mode but this cannot be installed until after Windows NT has been completely installed.

To automatically run Peer Web Services setup in silent mode, add the following at the command line to the RunOnce registry key.

```
INETSTP -b setup.txt
```

Where SETUP.TXT is a text file with nothing but a semi-colon in it, Peer Web Services setup requires a setup script file to run in silent mode like Internet Information Server for Windows NT Server, but does not make use of any settings included in the file.

### **Removing Microsoft Internet Explorer, Microsoft Exchange Client, and Image Viewer**

#### **Microsoft Internet Explorer 2.0**

To disable the installation of Microsoft Internet Explorer 2.0 during setup, comment out or delete the IEXPLORE.INF entry in SYSSETUP.INF.

SYSSETUP.INF is compressed on the Windows NT 4.0 CD-ROM.

1. On your network distribution share use the EXPAND utility to uncompress SYSSETUP.IN\_ to SYSSETUP.INF.  
Note: EXPAND.EXE must be run from within Windows NT.  
Example: `expand SYSSETUP.IN_ SYSSETUP.INF`
2. Rename SYSSETUP.IN\_ to SYSSETUP.OLD. By default, setup will use the compressed version if both exist.
3. Open SYSSETUP.INF and search for IEXPLORE.INF and "comment out" or delete the line with IEXPLORE.INF. To "comment out" IEXPLORE.INF insert a semicolon in front of IEXPLORE.INF.

---

Example:  
[Infs.Always]  
; iexplore.inf,DefaultInstall

### Microsoft Exchange Client

To disable the installation of the Microsoft Exchange Client ICON on the desktop during setup, "comment out" or delete the line with the MSMAIL.INF entry in SYSSETUP.INF.

SYSSETUP.INF is compressed on the Windows NT 4.0 CD-ROM.

1. On your network distribution share use the EXPAND utility to uncompress SYSSETUP.IN\_ to SYSSETUP.INF.  
Note: EXPAND.EXE must be run from within Windows NT.  
Example: expand SYSSETUP.IN\_ SYSSETUP.INF
2. Rename SYSSETUP.IN\_ to SYSSETUP.OLD. By default, setup will use the compressed version if both exist.
3. Edit SYSSETUP.INF and search for MSMAIL.INF and comment out that line by inserting a semicolon in front of MSMAIL.INF.

Example:

```
[BaseWinOptionsInfs]
accessor.inf
communic.inf
games.inf
imagevue.inf
mmopt.inf
; mmail.inf
multimed.inf
optional.inf
pinball.inf
wordpad.inf
```

### Image Viewer

To disable the installation of Image Viewer during setup, the IMAGEVUE.INF entry in SYSSETUP.INF needs to be commented out or deleted.

SYSSETUP.INF is compressed on the Windows NT 4.0 CD-ROM.

1. On your network distribution share use the EXPAND utility to uncompress SYSSETUP.IN\_ to SYSSETUP.INF.  
Note: EXPAND.EXE must be run from within Windows NT.  
Example: expand SYSSETUP.IN\_ SYSSETUP.INF
2. Rename SYSSETUP.IN\_ to SYSSETUP.OLD. By default, setup

---

will use the compressed version if both exist.

3. Open SYSSETUP.INF and search for IMAGEVUE.INF, then insert a semicolon in front of the line with IMAGEVUE.INF.

The change should look like this:

```
[BaseWinOptionsInfs]
accessor.inf
communic.inf
games.inf
; imagevue.inf
mmopt.inf
msmail.inf
multimed.inf
optional.inf
pinball.inf
wordpad.inf
```

## Automating Installation of Microsoft Exchange Server Client and Microsoft Internet Explorer 3.0x for Windows NT

### Automating Installation of the Exchange Client Supplied with Microsoft Exchange Server

You can automate the installation of the Microsoft Exchange Client provided with the Microsoft Exchange server using the Application Pre-Installation Tool (SYSDIFF.EXE), however you must first disable the installation of the Microsoft Exchange Client that is included with Windows NT.

1. Follow the instructions outlined in Removing Internet Explorer, Microsoft Exchange Client, and Image Viewer in the previous section of this chapter to disable the installation of the Microsoft Exchange Client. You must disable this before installing Windows NT on the computer that will be used to create the application image with SYSDIFF.EXE.
2. Follow the instructions for creating an application package in Chapter 4, "Application Pre-Installation (SYSDIFF.EXE)". Install the Exchange Client provided on the Microsoft Exchange Server CD.

---

*Note: Do not begin the configuration Wizard for the Exchange Client before creating a package with SYSDIFF.EXE. Doing so will cause all Exchange Clients to be configured incorrectly. Exchange provides a profile utility that can be used to configure each client.*

---

Consult the Microsoft Knowledge Base or your Microsoft Technet CD

---

subscription for additional information on Microsoft Exchange Server and Clients.

---

### **Installation of the Microsoft Internet Explorer 3.0x**

To automate the installation of latest version of Microsoft Internet Explorer 3.0x, you should obtain the Microsoft Internet Explorer Administrators Kit. You will find instructions for creating a silent scripted installation of Microsoft Internet Explorer. You can then use the RunOnce feature of Windows NT to add a command line that will run a batch file including the commands necessary to install Microsoft Internet Explorer.

The Microsoft Internet Explorer Administration Kit which provides full documentation on customizing Microsoft Internet Explorer for deployment. More information and the sign-up form are available at Microsoft's Web site: <http://www.microsoft.com/ie/ieak/>

### **For More Information**

For the latest information on Windows NT Workstation, check out our World Wide Web site at <http://www.microsoft.com/ntworkstation> or the Windows NT Server Forum on the Microsoft Network (GO WORD: MSNTS).