SEPTEMBER 11, 2012

Troubleshooting Windows Deployments

MICHAEL NIEHAUS
MICROSOFT
mniehaus@microsoft.com

Table of Contents

Document History

Date	Description
September 6, 2012	Initial version created with log file details.
September 11, 2012	Added sections for MDT return codes and descriptions, along with USMT 5.0 return codes.

Introduction

The Windows deployment process contains many moving parts – many tools and utilities, scripts, and other pieces stitched together into a complete end-to-end deployment process. In a perfect world, all of these parts would work perfectly and there would never been any issues that require troubleshooting.

But most of us realize that "perfection" cannot be achieved. That means we will need to do troubleshooting. So how do you do this effectively? Well, there are really two ways:

- Learn through your own experiences.
- Learn through the experiences of others.

So which is the best? Neither, as you need to be able to do both. You want to leverage the experience of others while also learning how to investigate your own unique issues without depending on others to do it for you.

So the goal of this document – which over time will hopefully grow into a book – is to gather up as many of these experiences as possible. That includes experiences around specific problems and solutions, as well as techniques for getting to the root cause even for uncommon issues that no one has seen before. Combine that with some consolidated reference materials and links to additional sources and hopefully we will end up with something useful.

It sounds good in theory, we'll see how that works out over time.

Log files

Windows logs

There are several logs created during a typical Windows deployment process, but the starting point is usually just with one: the SETUPACT.LOG file. This log keeps track of everything that happens during the Windows installation process.

When someone asks you for a copy of the "Panther logs" generally what they are looking for are the files from C:\Windows\Panther and C:\Windows\Panther\UnattendGC. At the very least, the SETUPACT.LOG and SETUPERR.LOG would be desired, but if in doubt, grab all the files in the Panther folder and all subfolders.

One significant change from Windows XP is that the Sysprep process now creates a log file, so if you see any Sysprep failures while building an image, you can look at the Panther files created specifically by the process. These would be located in the C:\Windows\System32\Sysprep\Panther folder.

You might also need to look on the server for information related to Windows Deployment Services. The WDSSERVER.LOG file, once enabled, is useful whenever investigating PXE and multicast issues, even if that investigation is something simple like "is the request getting to the server" (showing that the networking configuration is correct).

Log file	Description	Where to find it
Setupact.log	Main log file written by the Windows Vista, Windows 7, and Windows 8 installation process. Essential for figuring out what happened during a Windows installation. See http://support.microsoft.com/kb/927521 for more details.	C:\$WINDOWS.~BT\Sources\Panther (for early errors) X:\$WINDOWS.~BT\Sources\Panther (in Windows PE) C:\Windows\Panther (for specialize) C:\Windows\Panther\UnattendGC (for OOBE) C:\Windows\System32\Sysprep\Panther (for sysprep)
Setuperr.log	Log file containing only the error entries from the main setupact.log file. While this can be useful by itself in some cases, often you need the informational messages surrounding the errors to determine the actual cause, so be sure to have both files handy.	Same as for setupact.log.
Dism.log	Shows the results of all DISM commands that were executed during the OS installation process (including those	C:\Windows\Logs\DISM

	executed by the Windows SETUP program itself). Use this file to investigate issues with driver injection, language pack installation, security update installation, etc.	
Cbs.log	A lower-level log file for servicing operations, often corresponding to messages logged in the setupact.log, dism.log.	C:\Windows\Logs\CBS
Setupapi.dev.log	Shows all PnP device driver installation details, useful for determining what drivers were considered for a particular device, issues encountered when installing a driver, and related PnP information.	C:\Windows\Inf X:\Windows\Inf (in Windows PE)
Netsetup.log	Shows domain join attempt details, useful for identifying what domain join details were specified (except for passwords) and the results of each attempt.	C:\Windows\Debug
WindowsUpdate.log	Shows details related to software updates installation from Windows Update, WSUS, or ConfigMgr (SUP).	C:\Windows
Wpeinit.log	Contains details about the Windows PE initialization process, useful for troubleshooting slow startup times, networking initialization issues, and failed commands that prevent Windows PE from rebooting automatically.	X:\Windows\system32 (in Windows PE)
WDSServer.log	Records details about all PXE requests processed by Windows Deployment Services, as well as all multicast transmission details. Useful for troubleshooting PXE and multicast issues. This log is not turned on by default and must be enabled using the instructions found in http://support.microsoft.com/kb/936625.	C:\Windows\Tracing

Task Sequencer logs

Used by System Center Configuration Manager and the Microsoft Deployment Toolkit, the task sequencing engine creates one log file, SMSTS.LOG. But that log bounces around to different locations during the deployment process, so finding it can be a challenge. Additionally, the file can only grow so large before it is renamed with an added timestamp and then a new file is created. To make sure you get files that cover as much as possible, you can always grab every file on the system that matches the pattern "SMSTS*.LOG".

Log file	Description	Where to find it

Smsts.log	Main log file written by the task sequencer used by both System Center 2012 Configuration Manager, Configuration Manager 2007, and the Microsoft Deployment Toolkit. This file is useful when investigating failed task sequence steps (especially those that fail without writing any other logs or log entries) and when verifying the evaluation of conditions on task sequence steps and groups.	Several possible locations: • %TEMP%\SMSTSLog (typically in Windows PE) • %WINDIR%\System32\CCM\Logs (ConfigMgr 2007, 32-bit OS) • %WINDIR%\Syswow64\CCM\Logs (ConfigMgr 2007, 64-bit OS) • %WINDIR%\CCM\Logs (ConfigMgr 2012) • C:_SMSTaskSequence\Logs • C:\SMSTSLog • X:\SMSTSLog

Microsoft Deployment Toolkit Logs

All scripts used in the Microsoft Deployment Toolkit write to two different logs, the main BDD.LOG and individual log files associated with each script (e.g. ZTIApplications.wsf will create a ZTIApplications.log file). The information written to both logs is exactly the same, so if you have the BDD.LOG you don't really need the individual script log files (although they might be easier to follow sometimes).

Some scripts will also create additional log files. These additional log files will often be prefixed with the script name, but that's not always the case. For example, the ZTIConfigureDHCP.wsf script will create a ZTIConfigureDHCP_DISM.log file, while the ZTIUserState.wsf script will create USMTCapture.log and USMTRestore.log files. So you might want to keep all of the log files in cases where problems are encountered.

Log file	Description	Where to find it
BDD.log	Main log file written all MDT scripts. Essential for figuring out what happened during any MDT task sequence.	C:\MININT\SMSOSD\OSDLOGS (Lite Touch running) C:_SMSTaskSequence\Logs (ConfigMgr running) C:\WINDOWS\Temp\DeploymentLogs (Lite Touch complete) Others (ConfigMgr, see SMSTS.LOG description)

User State Migration Tool Logs

The User State Migration Tool (USMT) creates one log file for each execution of the Scanstate.exe and Loadstate.exe tools. The default name for the log file will be Scanstate.log and Loadstate.log, but a different file name can be specified on the command line. The Microsoft Deployment Toolkit specifies a different name, while ConfigMgr uses the default names.

Log file	Description	Where to find it
USMTEstimate.log	Log created by the MDT ZTIUserState.wsf script when running Scanstate.exe to determine whether there is enough disk space to save the user state locally. This is only applicable to USMT 3.0, since USMT 4.0 and 5.0 use hardlinks when keeping user state locally meaning there is always sufficient disk space.	In the same location as the BDD.LOG file.
USMTCapture.log	Log created by the MDT ZTIUserState.wsf script when running Scanstate.exe to capture user state from the current computer.	In the same location as the BDD.LOG file.
USMTRestore.log	Log created by the MDT ZTIUserState.wsf script when running Loadstate.exe to restore user state to the current computer.	In the same location as the BDD.LOG file.
Scanstate.log	Log created by the ConfigMgr “Capture User State” step when it runs Scanstate.exe to capture user state from the current computer.	In the same location as the SMSTS.LOG file.
Loadstate.log	Log created by the ConfigMgr “Restore User State” step when it runs Loadstate.exe to restore user state to the current computer.	In the same location as the SMSTS.LOG file.

Configuration Manager Logs

In addition to the task sequencer logs described above, there are additional log files on the ConfigMgr server that are useful for troubleshooting specific issues. See http://technet.microsoft.com/en-us/library/bb932135.aspx for a complete list of log files used by ConfigMgr.

Log file	Description	Where to find it
DriverCatalog.log	Contains information that may be useful when getting errors while importing drivers	C:\Program Files\Microsoft Configuration Manager\Logs

TaskSequenceProvider.log	Useful when getting errors while saving or importing a task sequence	C:\Program Files\Microsoft Configuration Manager\Logs
SMSPXE.LOG	Used when troubleshooting PXE boot issues	C:\Program Files\SMS_CCM\Logs
SMSPROV.LOG	May have more error details while saving or importing a task sequence	C:\Program Files\Microsoft Configuration Manager\Logs

Microsoft Deployment Toolkit Return Codes

When most of the Microsoft Deployment Toolkit scripts were originally written, they each returned one of two results:

- Success, with a return code of 0
- Failure, with a return code of 1

But that wasn't particularly useful in identifying why a particular script failed, so in MDT 2010 each of the scripts was reworked to report different return codes (and corresponding error messages) for each different failure that might occur. So now when you see a particular return code, you immediately have more information available telling you about the error that occurred. And if you like digging into the MDT scripts, this return code will also point you to the specific line in the MDT scripts that generated that return code, because the same return code is never used more than once.

In theory, the MDT documentation would include a full list of these error codes, along with the error messages that would be reported with each one and some potential troubleshooting suggestions: what might have caused the error, what you might want to consider doing about it, etc. Well, there is a table in the documentation called "Identifying Error Codes", but it's not very complete. So the table below can help with that.

Note that each version of MDT could change this list, and if any of the scripts are significantly rewritten (e.g. ZTIUserState, which was rewritten in MDT 2012) the numbers being used within the script could be used for different purposes. So the list below is valid for MDT 2012 Update 1, but might not quite sync up with earlier versions.

Each of the MDT scripts also has a range of return codes allocated to it, so the error numbers also can tell you from which script the error came. To simplify things, the script name is also included below. Each of the messages is also rated on a scale of 0 to 10 in two different areas:

- Likelihood. If the error is highly unlikely, it's not worth worrying about so no troubleshooting suggestion will be provided. If by chance you ever see the error, contact Microsoft Support for assistance.
- Quality. Some error messages are self-explanatory; others make little sense without further explanation. So each has been rated, with 0 being "absolutely useless" and 10 being "perfectly self-explanatory."

Also, you'll notice that some items have been highlighted in red. These indicate issues in the MDT scripts: numbers that were reused, numbers that don't fit into the right number range for each script, spelling and grammar errors, etc. At some point, I might files bugs for these…

Return code	Script	Error Message	Likely ?	Quality	Troubleshooting Suggestions
5212	LiteTouch.wsf	Welcome wizard failed or was cancelled	10	5	This is normal if you cancel the Lite Touch welcome wizard (the initial wizard that shows up

					in Windows PE unless you set SkipWizard=YES). But it also might mean that the wizard crashed. You can’t really tell the difference though. (The wizard sets a variable WizardComplete to “Y” when it succeeds. The error means WizardComplete wasn’t set to “Y”.) You might also see this message show up in the log file for the next task sequence executed on the computer because of a left-over BDD.LOG found on the computer in the C:\MININT folder structure. This is harmless, but will result in a yellow summary screen at the end of the deployment.
5204	LiteTouch.wsf	The logged-on user does not have Administrator rights.	5	10	This can happen in two scenarios: • You try to run LiteTouch.vbs from Windows XP or Windows Server 2003 when you aren’t logged on as an account with administrator rights. • You try to run LiteTouch.wsf from any other OS when you aren’t running elevated (with UAC enabled). The solution in the first case is obvious, but in the second case it’s a little more subtle: Make sure you always run LiteTouch.vbs because it will trigger a UAC prompt to elevate the resulting LiteTouch.wsf process. (That’s one of the main reasons that LiteTouch.vbs exists.)
5206	LiteTouch.wsf	The Deployment Wizard was cancelled or did not complete successfully. The deployment will not proceed.	10	5	This is normal if you cancel the Lite Touch deployment wizard. But it also might mean that the wizard crashed. You can’t really tell the difference though. (The wizard sets a variable WizardComplete to “Y” when it succeeds. The error means WizardComplete wasn’t set to “Y”.) You might also see this message show up in the log file for the next task sequence executed on the computer because of a left-over BDD.LOG found on the computer in the C:\MININT folder

					structure. This is harmless, but will result in a yellow summary screen at the end of the deployment.
5208	LiteTouch.wsf	Invalid DeploymentType value %DeploymentType% specified.	1	7	Normally the DeploymentType task sequence variable is set automatically. But if for some reason the variable was set through some other means (e.g. a command line parameter or CustomSettings.ini) to an invalid value, MDT will generate an error before other bad things happen.
5208	LiteTouch.wsf	Unable to find the SMS Task Sequencer. The deployment will not proceed.	4	7	There are a set of executables and DLLs needed to run the task sequencing engine. These are copied to the local machine before the task sequence initially starts, and it's expected that these files remain through the process. If something happens where MDT can determine that there is an in-progress task sequence (by the presence of an _SMSTaskSequence folder) but can't find the task sequencer files that are supposed to be present, you'll see this error. This can happen with rouge scripts that delete the MININT folder, or images that accidentally contain a MININT folder (which can cause the "real" MININT folder to be moved to WINDOWS.OLD).
5210	LiteTouch.wsf	ValidateDeployRootWithRecovery: Cancel	7	5	MDT tried to connect to the deployment share specified by the DeployRoot task sequence variable, but it wasn't able to connect. A popup was displayed to the user asking to retry or cancel, and the user chose cancel. So this message means MDT is giving because the user told it to do so.
5211	LiteTouch.wsf	ValidateDeployRootWithRecovery: Cancel	0	5	It's presently impossible for this error to occur. When MDT is unable to connect to a deployment share, there used to be an option to suspend the task sequence, letting you manually fix the problem and then resume the task sequence, but this thoroughly confused people, so that option has been disabled. The logic is still in the script

					though, so it could be re-enabled if you really wanted to do so. That said, it's really not clear why this error even exists.
5614	LTIApply.wsf	ERROR - Unable to run cross platform installation of WinNT32.exe form WinPE.	3	9	When cross-platform OS deployment was added in MDT 2012, it was discovered that it was possible to attempt to do an unattended install of Windows XP or Server 2003 from the wrong architecture of Windows PE, which would fail miserably. So this error was added to tell you that. But you should never be able to get this far, as the task sequence wizard will filter out these task sequences to keep you from selecting them in this situation. So you should only see this error if you skipped the task sequence wizard pane and specified the TaskSequenceID value through CustomSettings.ini or other means.
7002	LTISysprep.wsf	Computer is a member of a domain, should be in a workgroup when sysprepping.	2	9	Microsoft recommends to not join a computer into a domain before running sysprep.exe to capture a custom image because the domain can leave remnants in the image, e.g. policies and other settings that cause issues later. So the MDT wizard has a check in it that says "don't even offer the choice of capturing an image if you specify to join a domain." But people went out of their way to bypass that check by manually settings task sequence variables like DoCapture=YES to force MDT to capture the image anyway. This check was added in MDT 2012 to make MDT more insistent on the recommendation of not capturing an image of a domain-joined machine.
10403	UDIWizard.wsf	Invalid wizard configuration file specified. The deployment will not proceed.	2	8	Either the default UDIWizard_Config.xml file, or a custom file specified by added a /definition switch to the UDIWziard.wsf command line, couldn't be found. Without this file, it's impossible to display the UDI wizard. Make sure the file is present at

					the path specified or in the MDT toolkit files package.
20001	Wizard.hta	(Varies)	3	3	The LTI wizard can run into various errors, but it will report all of them using the same error number: • Definition file not defined. Please call with /Defintion:<file> • Unable to find definition file: <file> • Unable to Create MSXML2.DOMDocument(.6.0) Object. • Unable to load VBScript File: <file> • Unable to load XML file: <file> You typically wouldn't see these unless you were customizing the wizard and broke something, or were trying to launch Wizard.hta manually and not telling it what wizard definition XML file to display.
6501	ZTIBackup.wsf	Computer backup not possible, no network path (BackupShare, BackupDir) specified.	5	9	If you choose "automatically determine the computer backup location" in the Lite Touch wizard, or set ComputerBackupLocation=NETWORK, MDT attempts to build a path to use for the backup by combining the BackupShare and BackupDir task sequence variables. But if those aren't set, that logic won't work, so you'll get this error when those variables aren't specified. (MDT should probably not allow this choice in cases when BackupShare isn't specified, a good feedback item to submit.)
7100	ZTIConfigureADDS.wsf	ERROR - This script should only run in the full OS.	2	9	You would see this error if you added a "Configure ADDS" step to the task sequence to create a new domain controller, but put that step in the wrong place in the task sequence. That task sequence action needs to run in the full OS, not in Windows PE.
7101	ZTIConfigureADDS.wsf	ERROR - Not enough values supplied for generating DCPromo answer file	1	4	Normally you would specify the DCPromo parameters in the "Configure ADDS" step in the

					task sequence editor, but if you tried to run ZTIConfigureADDS.wsf yourself without setting the ReplicaOrNewDomain task sequence variable properly, you would get this error.
7102	ZTIConfigureADDS.wsf	ERROR - Mandatory properties for creating a new replica DC were not specified	1	4	Same as above, but in this case you didn't set one of these variables: • ReplicaDomainDNSName • ADDSUserName • ADDSPassword • ADDSUserDomain
7103	ZTIConfigureADDS.wsf	ERROR - Mandatory properties for creating a new child domain were not specified	1	4	Same as above, but in this case you didn't set one of these variables as needed: • ChildName • ParentDomainDNSName • ADDSUserName • ADDSPassword • ADDSUserDomain
7104	ZTIConfigureADDS.wsf	ERROR - Mandatory properties for creating a new forest were not specified	1	4	Same as above, but in this case the NewDomainDNSName task sequence variable wasn't set.
7105	ZTIConfigureADDS.wsf	ERROR - Mandatory properties for creating a new forest were not specified	1	4	Same as above, but in this case you didn't set one of these variables: • NewDomainDNSName • ADDSUserName • ADDSPassword • ADDSUserDomain
7200	ZTIConfigureDHCP.wsf	Unable to configure DHCP Server because the service is not installed.	2	10	You added a step to "Configure DHCP" to the task sequence, but when the step ran it found that the DHCP Server wasn't installed. You would need to first install it using an "Install Roles and Features" step.
7201	ZTIConfigureDHCP.wsf	Unable to read the Scope Details - GetScopeDetails() failed	1	5	Normally you would configure the DHCP scope details through the task sequence UI, but if you ran ZTIConfigureDHCP.wsf manually and didn't specify

					all the needed task sequence variables, you could get this error.
7202	ZTIConfigureDHCP.wsf	Not enough values specified for scope creation.	1	5	Same as above, but you didn't specify subnet mask, IP, or scope name properties.
7203	ZTIConfigureDHCP.wsf	Not enough values provided to Set the IP range For this scope.	1	5	Same as above, but you didn't specify the starting and ending IP address details for the DHCP scope.
7300	ZTIConfigureDNS.wsf	Unable to issue DNS commands	1	5	When you add a "Configure DNS" task sequence step into the task sequence, you configure the specific DNS settings that should be configured. The ZTIConfigureDNS script builds a DNSCMD.EXE script based on these values and executes it. This error means that DNSCMD.EXE reported a non-zero return code. You can review the BDD.LOG log to see what commands failed.
7711	ZTIDiskpart.wsf	Disk OSDDiskIndex(<Index>) can not be found!	5	7	You told MDT to format a disk with a particular index (by default, index 0) but there was no such disk. Make sure you specify a disk that exists. Assuming you are using the default disk index, this can also mean that you are missing the mass storage driver needed for the disk.
7712	ZTIDiskpart.wsf	Disk (<Path>) can not be formatted in OSD.	5	7	You told MDT to format a disk (typically with index 0) but when MDT checked the disk it found out that it couldn't be used for an OS, typically because it's either a removable disk or too small to hold the OS. Make sure you specified the correct disk.
7820	ZTIDiskpart.wsf	DoNotFormatAndPartition has been set to Yes, however the Target Partition for the OS could not be determined. See BDD.Log for details.	1	8	The DoNotFormatAndPartition variable was set to YES, indicating that no formatting or partitioning should be done. But no operating system volume could be determined, so an error is generated at this point rather than waiting until later when LTIApply tries to figure out where to put the new OS.
7701	ZTIDiskpart.wsf	Disk is not large enough for System and BDE partitions, Required = 300MB	1	1	This is a deceptive message, because what it is really trying to tell you is that you specified a custom value for the BdeDriveSize task sequence

					variable (used to control the size of the boot volume) that is too small. You must specify a size of 300 or higher.
7706	ZTIDiskpart.wsf	ERROR! Drive <dest> was not found!	1	5	The ZTIDiskpart script formatted and partitioned a drive, but then couldn't find the volume that it just created, so something is wrong.
7707	ZTIDiskpart.wsf	ERROR! Drive <dest> is not Ready!	1	5	The ZTIDiskpart script formatted and partitioned a drive, but when checking the volume that it had just created, it found that the volume wasn't ready, so something is wrong.
7708	ZTIDiskpart.wsf	ERROR! Drive <dest> is not a fixed disk: <type>	1	5	The ZTIDiskpart script formatted and partitioned a drive, but when checking the volume that it had just created, it found that the volume wasn't reporting itself as a fixed disk, so something is wrong.
7709	ZTIDiskpart.wsf	ERROR! Files are present on <dest> was not cleaned: <count>	1	5	The ZTIDiskpart script just formatted and partitioned a drive, but when checking the volume it had just created, it found files on the volume. That shouldn't happen, so something is wrong.
7710	ZTIDiskpart.wsf	ERROR! Folders are present on <dest> was not cleaned: <count>	1	5	The ZTIDiskpart script just formatted and partitioned a drive, but when checking the volume it had just created, it found folders on the volume. That shouldn't happen, so something is wrong.
10201	ZTIDomainJoin.wsf	Unable to Join <domain> Stop installation.	1	8	You set task sequence variable DomainErrorRecovery to "FAIL" indicating that the ZTIDomainJoin.wsf script should force a failure if it is unable to join the specified domain.
10801	ZTIExecuteRunbook.wsf	Unable to create Orchestrator job for the specified runbook.	3	8	The ZTIExecuteRunbook script tried to talk to the Orchestrator web service to create a job to run the specified runbook, but this failed.
10802	ZTIExecuteRunbook.wsf	Unable to find job.	1	7	The ZTIExecuteRunbook script was able to create a job to run the specified runbook, but it was then unable to retrieve the job from the Orchestrator web service.
10803	ZTIExecuteRunbook.wsf	Unable to get Orchestrator job status.	3	7	The ZTIExecuteRunbook script was not able to retrieve the status of the job created to execute

					the specified runbook. Without this, MDT is unable to determine when the runbook execution has completed.
10804	ZTIExecuteRunbook.wsf	Unable to get Orchestrator job runbook instance.	3	8	The ZTIExecuteRunbook script was not able to retrieve the Orchestrator job runbook instance via the Orchestrator web service. Without this, MDT is unable to get the results from the job.
10805	ZTIExecuteRunbook.wsf	Unable to get Orchestrator job runbook instance parameters.	3	8	The ZTIExecuteRunbook script was not able to retrieve the Orchestrator job runbook instance via the Orchestrator web service. Without this, MDT is unable to set task sequence variables with the parameters returned by the job.
10806	ZTIExecuteRunbook.wsf	Runbook did not complete successfully, final status = <status>	7	9	Orchestrator reported that the runbook job failed. To determine the cause, check on the Orchestrator server to see what happened.
9002	ZTIOSRole.wsf	Unable to located the selected set of roles in ServerManager.xml (OSRoleIndex = <index>)	1	7	The operating system selected in the “Install Roles and Features” couldn’t be found in the ServerManager.xml file. That should never happen.
9003	ZTIOSRole.wsf	Unknown OS current version value, unable to install roles. (OSCurrentVersion = %OSCurrentVersion%)	5	9	In order for the “Install Roles and Features” step to know what operating system it is working with, it looks at the OSCurrentVersion task sequence variable. If this variable isn’t set (maybe because you aren’t using an MDT task sequence and didn’t precede this step with the required “Use Toolkit Package” and “Gather” steps), or if it is set to something invalid (e.g. “WinPE” if you are incorrectly running this step in Windows PE) you’ll get this error.
9601	ZTITatoo.wsf	ERROR - ZTITatoo state restore task should be running in the full OS, aborting.	3	9	The ZTITatoo script found that it was running in Windows PE, when it expected to be running in the new operating system. This might happen if the boot order on the system was wrong and the computer accidentally booted back into Windows PE.

9701	ZTIUserState.wsf	Unable to find any version of USMT, unable to perform user state migration.	5	8	The ZTIUserState script tried to find USMT 3, 4, and 5, but none of them could be found. Make sure the deployment share has the needed files in the Tools\<arch>\USMT<3,4,5> folder.
9702	ZTIUserState.wsf	USMT 4 or higher must be used to perform an offline USMT migration	3	8	The ZTIUserState script found itself running in Windows PE, but only found USMT 3. Since USMT 3 does not support offline state capture from within Windows PE, an error was reported. Make sure USMT 4 or USMT 5 are available since these versions support this.
9703	ZTIUserState.wsf	USMT 3 cannot be used with destination OS %ImageBuild%	3	8	USMT 3 does not support restoring user state to OSes newer than Windows XP. As the destination OS was newer than that, an error was generated before the state capture happened (before the "point of no return").
9704	ZTIUserState.wsf	USMT 4 cannot be used with destination OS %ImageBuild%	6	9	USMT 4 does not support Windows 8 for restoring user state, so it generated an error. This would likely only happen if you were using Windows AIK with MDT, as MDT would typically put USMT 5 on the deployment share automatically if Windows ADK were being used. Either install ADK or manually copy the USMT 5 files to the needed Tools\<arch>\USMT5 folders.
9705	ZTIUserState.wsf	Unable to find USMT <version> files, cannot capture/restore user state.	4	7	The ZTIUserState script was unable to find scanstate.exe in the chosen USMT folder, so it generated an error. Make sure all the needed USMT files were copied to the needed location.
9706	ZTIUserState.wsf	The offline Windows directory could not be determined, offline USMT migration cannot be performed	6	9	In order to capture user state offline, the ZTIUserState script has to be find the Windows folder as that contains the Windows settings. This is then provided to scanstate.exe on the command line. But the script wasn't able to find this, so it generated an error. Make sure the Windows folder is named appropriately, as MDT will only look for "Windows" and "WINNT".

9707	ZTIUserState.wsf	Non-zero return code from USMT capture, rc = <return code>	8	7	Scanstate.exe executed, but returned a non-zero return code indicating that it failed. Check the USMTCapture.log file to see why it failed.
9708	ZTIUserState.wsf	No valid command line option was specified	1	6	The ZTIUserState script expects to be called with /ESTIMATE, /CAPTURE, or /RESTORE switches, but none of those command line options was found. Make sure the command line being used is correct.
9709	ZTIUserState.wsf	Architecture of the original Operating System could not be determined	1	8	The ZTIUserState script looks at the OriginalArchitecture task sequence variable to determine whether the original OS was x86 or x64. This variable is normally set earlier in the process by ZTIGather or UDIWizard, but in this case it looks like that didn't happen.
9801	ZTIValidate.wsf	ERROR - Attempting to deploy a client operating system to a machine running a server operating system.	4	9	The task sequence is preconfigured with the type of OS that it is expecting to deploy, either client or server. If it finds that doesn't match what the currently-running OS is, an error is generated. This is done to prevent the accidental deployment of a client OS to servers (e.g. deploying to the "All Systems" collection in ConfigMgr) or vice versa.
9802	ZTIValidate.wsf	ERROR - Attempting to deploy a server operating system to a machine running a client operating system.	4	9	The task sequence is preconfigured with the type of OS that it is expecting to deploy, either client or server. If it finds that doesn't match what the currently-running OS is, an error is generated. This is done to prevent the accidental deployment of a server OS to clients (e.g. deploying to the "All Systems" collection in ConfigMgr) or vice versa.
9803	ZTIValidate.wsf	ERROR - Machine is not authorized for upgrading (OSInstall=%OSInstall%), aborting.	5	8	The ZTIValidate script will purposely fail any time the OSInstall task sequence variable is not set to blank, Y, or YES. Generally this error happens if you force the value to N or NO, for example in the [Default] section of CustomSettings.ini for computers that aren't defined in the MDT database.
9808	ZTIValidate.wsf	Error - Performing a Refresh from a newer OS Version to an older OS	7	9	You can't perform a refresh from a newer OS to an older version, as there are complications in these

		Version is not supported. You must perform a NewComputer deployment, aborting			scenarios. To work around this, perform a new computer deployment instead.
9809	ZTIValidate.wsf	Error - Performing a Refresh to a partition that does not match the current OS partition is not supported. You must perform a NewComputer deployment, aborting	4	8	You are trying to force the deployment of an OS to a different partition from where it currently resides, which isn't supported in a refresh. If you want to do this, perform a new computer deployment instead.
9804	ZTIValidate.wsf	ERROR - %Memory%MB of memory is insufficient. At least <minmem>MB of memory is required.	6	9	The computer doesn't have enough RAM (based on the value configured in the task sequence "Validate" task).
9805	ZTIValidate.wsf	ERROR - Processor speed of %ProcessorSpeed%MHz is insufficient. At least a <minspeed>MHz processor is required.	5	9	The computer doesn't have a fast enough processor to perform the deployment (based on the value configured in the task sequence "Validate" task).
9807	ZTIValidate.wsf	ERROR - insufficient space is available on <target>. An additional <size>MB is required.	5	9	Based on the total amount of disk space on the system and the expanded image size (plus a 3150MB "fudge factor"), there isn't enough disk space to hold the OS. Either make the image smaller or get a bigger disk.
9902	ZTIWindowsUpdate.wsf	ZTIWindowsUpdate has run and failed too many times. Count = <count>	5	8	The Windows Update process might need to reboot multiple times during the task sequence. In this case, there were more than the default number of 7 reboots, so the script is aborting. Review the log to see if there is an update that is not installing properly.
9903	ZTIWindowsUpdate.wsf	Unexpected issue installing the updated Windows Update Agent, rc = <retval>	3	8	The ZTIWindowsUpdate script is attempting to install the latest version of the Windows Update agent, but it is unable to do that. That might occur because the computer doesn't have internet access to download the agent. Try downloading the agent manually and place it into the needed Tools\<platform> folder on the deployment share so that the client doesn't need to download it.
9904	ZTIWindowsUpdate.wsf	Failed to Create Object: Microsoft.Update.Session.	1	6	The ZTIWindowsUpdate script was unable to create an instance of the Microsoft.Update.Session

					COM object. Without that, it can't install Windows Updates.
9905	ZTIWindowsUpdate.wsf	Failed to Create Object: Microsoft.Update.UpdateColl.	1	6	The ZTIWindowsUpdate script was unable to create an instance of the Microsoft.Update.UpdateColl COM object. Without that, it can't install Windows Updates.
9906	ZTIWindowsUpdate.wsf	Critical file <filename> was not found, aborting	3	7	One of the files needed to install and configure the Windows Update agent couldn't be found.
9000	DeployWiz_Roles.VBS	FAILURE: Did not find ServerManager.xml	1	7	The ServerManager.xml file contains the of roles and features. Without this file, the deployment wizard can't display the "Install roles" wizard pane. Make sure this file is present in the expected location in the Scripts folder of the deployment share.
10203	LiteTouch.wsf	FAILURE: FindFile(LTISuspend.wsf)	0	1	See the description for message 5211 above.
10204	LiteTouch.wsf	FAILURE: Run Program <LTISuspend>	0	1	See the description for message 5211 above.
5601	LTIApply.wsf	FAILURE: Verify OS guid: %OSGUID% exists.	5	5	The LTIApply script tried to find the operating system configured in the task sequence, but it didn't exist. This typically happens when the OS has been deleted (maybe when creating a new version of the image) without updating the corresponding task sequence. To fix this, edit the task sequence to select a valid OS on the "Install Operating System" step.
5602	LTIApply.wsf	FAILURE: Open XML with OSGUID: %OSGUID%	0	3	In this case, the script determined that the OS existed, but it couldn't retrieve it from the XML file. Since the existence check also looks at the XML file, this should never happen.
5609	LTIApply.wsf	FAILURE: Boot Drive was not found.	1	5	The LTIApply.wsf tried to find the boot volume as that's where it wants to put the Windows PE boot image. In some cases, this drive is hidden and needs to have a drive letter assigned. But even after automatically trying this, the boot drive still could not be found. This should not happen.
5610	LTIApply.wsf	FAILURE: Verify File: <file>	8	4	The LTIApply script tried to find the LiteTouchPE_<architecture>.wim file on the

					deployment share, but it wasn't there. That typically happens when you haven't yet updated the deployment share to generate this WIM file (or when that architecture isn't enabled).
5630	LTIApply.wsf	FAILURE: Verify File: <image path>	3	3	The LTIApply script is trying to find the operating system WIM file so that it can provide it to SETUP.EXE for installation, but the file does not exist at the path specified in the deployment share's XML configuration files. Maybe the file was deleted or not copied?
5604	LTIApply.wsf	FAILURE: Verify Destination Drive is defined(1)	2	3	The LTIApply script is attempting to figure out the destination drive for the OS, but it can't find it. Maybe it doesn't exist due to a missing driver?
5605	LTIApply.wsf	FAILURE: Verify Destination Drive is defined(2)	2	3	The LTIApply script is attempting to figure out the destination drive for the OS, but it can't find it. Maybe it doesn't exist due to a missing driver?
5640	LTIApply.wsf	FAILURE: Verify File: <image path>	3	3	The LTIApply script is trying to find the operating system WIM file so that it can be applied using ImageX, but the file does not exist at the path specified in the deployment share's XML configuration files. Maybe the file was deleted or not copied?
5606	LTIApply.wsf	FAILURE: Verify Destination Drive is defined(1)	2	3	The LTIApply script is attempting to figure out the destination drive for the OS, but it can't find it. Maybe it doesn't exist due to a missing driver?
5607	LTIApply.wsf	FAILURE: Verify Destination Drive is defined(2)"	2	3	The LTIApply script is attempting to figure out the destination drive for the OS, but it can't find it. Maybe it doesn't exist due to a missing driver?
5624	LTIApply.wsf	FAILURE: Run ImageX: <command>	8	4	While applying the operating system image to the disk using ImageX, a non-zero return code was returned. To troubleshoot, look at the BDD.LOG to see what specific errors were returned by ImageX.
5626	LTIApply.wsf	FAILURE: Verify BootSect.exe returned Successfully.	1	3	After the new OS image was applied, BootSect.exe is executed to ensure that the disk has the right boot sector. This should return a return code of 0,

					but in this case a non-zero return code was found. This shouldn't happen.
5615	LTIApply.wsf	FAILURE: Boot Drive was not found, required?	4	3	After the new OS image was applied and BootSect.exe was executed, the LTIApply script wants to use BCDBOOT.exe to create a new BCD entry for the new OS. In order to do this, it needs to find the boot volume (adding a drive letter to if when required) as that needs to be specified on the BCDBOOT.exe command line. In this case, the boot drive could not be found.
5616	LTIApply.wsf	FAILURE: Verify BCDBootEx	2	4	The BCDBOOT.exe command reported a non-zero return code. Check the BDD.LOG to see if any meaningful errors were logged by the utility.
5627	LTIApply.wsf	FAILURE: Run DISM.exe	9	2	When applying an image using ImageX, the LTIApply script applies the unattend.xml file using DISM.EXE so that the servicing operations (drivers, patches, language packs) and settings are processed as expected. But in this case, DISM.EXE reported a non-zero return code. Typically this means that the unattend.xml was invalid or contained invalid settings. Check the DISM.LOG to see what the actual failure was.
5650	LTIApply.wsf	FAILURE: Verify Directory: <source path>	3	5	The LTIApply script is attempting to find the folder containing the files needed to perform an unattended install of Windows XP or Windows Server 2003, but the path specified on the operating system entry cannot be found. Maybe the OS has been deleted?
5651	LTIApply.wsf	FAILURE: Verify Directory: <source path>\<platform>	3	5	The source folder was found, but the platform-specific subdirectory (x86, amd64) that is required for performing an unattended install of Windows XP and Windows Server 2003 was not found. Maybe the OS source files are not complete?
5628	LTIApply.wsf	FAILURE: Boot Drive was not found, required?	4	3	After the new OS image was applied, the LTIApply script wants to use BootSect.exe to install a new

					boot sector on the boot drive. In this case, the boot drive could not be found.
5629	LTIApply.wsf	FAILURE: Verify BootSect.exe returned Successfully.	1	3	After the new OS image was applied, BootSect.exe is executed to ensure that the disk has the right boot sector. This should return a return code of 0, but in this case a non-zero return code was found. This shouldn't happen.
6001	LTIOEM.wsf	FAILURE: Verify Drive <command>	1	3	The LTIOEM script wants to verify that it knows where the OS should be deployed. In this case, it can't determine that.
6002	LTIOEM.wsf	FAILURE: Verify Drive <command>	1	3	The LTIOEM script wants to verify that the DeployDrive task sequence variable points to a valid location. This should be automatically set to the correct location.
6020	LTIOEM.wsf	FAILURE: Robocopy returned value: <retval>	3	5	The OEM staging process uses ROBOCOPY to copy the media to the hard drive. In this case, ROBOCOPY reported an unexpected return code indicating that the copy was unsuccessful. Maybe the media is corrupt?
6021	LTIOEM.wsf	FAILURE: Robocopy returned value: <retval>	3	5	Same as above.
6010		FAILURE: Test for TSGUID	1	5	The LTIOEM script wants to remove the "staging" task sequence (the one currently running) from the TaskSequences.xml file so that it doesn't show up when booting the OEM machine. It finds the existing task sequence using the TSGUID task sequence variable. This error is returned when the TSGUID variable is blank, which should never happen.
7001	LTISysprep.wsf	FAILURE: Looking for unattend AnswerFile	1	5	When deploying Windows Vista and above, MDT wants to place an unattend.xml file in the correct location for sysprep.exe to find it and process it (processing any specified "generalize" settings). This error indicates that LTISysprep was unable to find the unattend.xml associated with the current task sequence. This should never happen.

5601	LTISysprep.wsf	FAILURE: Verify OS guid: %OSGUID% exists.	1	5	The LTISysprep script tried to find the operating system configured in the task sequence, but it didn't exist. This typically happens when the OS has been deleted (maybe when creating a new version of the image) without updating the corresponding task sequence. To fix this, edit the task sequence to select a valid OS on the "Install Operating System" step. (Typically this would result in an error much earlier in the process.)
5602	LTISysprep.wsf	FAILURE: Open XML with OSGUID: %OSGUID%	0	3	In this case, the script determined that the OS existed, but it couldn't retrieve it from the XML file. Since the existence check also looks at the XML file, this should never happen.
6101	LTISysprep.wsf	FAILURE: Check for file: <cab file>	7	4	In order to sysprep and capture a Windows XP or Windows Server 2003 system, the sysprep files are needed. If these aren't available in the OS source folder, they can be extracted from the DEPLOY.CAB file. In this case though, the DEPLOY.CAB was not available. Without the sysprep files, the script cannot continue. Make sure that the sysprep files (sysprep.exe, setupcl.exe, factory.exe) or the DEPLOY.CAB is available in the OS source folder.
6102	LITSysprep.wsf	FAILURE: expand Sysprep files from DEPLOY.CAB.	3	4	The LTISysprep script was able to find the DEPLOY.CAB, but it wasn't able to extract the files from the CAB. Maybe the CAB is corrupt?
6111	LTISysprep.wsf	FAILURE: Run Sysprep.exe.	4	4	The LTISysprep script ran sysprep.exe to prepare the OS (Windows Vista or above) for capture, but it got a non-zero return code from sysprep.exe indicating that sysprep failed. Check the C:\Windows\system32\sysprep\panther\setupact.log to see what happened.
6121	LTISysprep.wsf	FAILURE: Run Sysprep.	3	4	The LTISysprep script ran sysprep.exe to prepare the OS (Windows XP or Windows Server 2003) for capture, but it got a non-zero return code from sysprep.exe indicating that sysprep failed. As

					Windows XP and Windows Server 2003 don't write a log during the sysprep process, figuring out the cause is a trial-and-error process. Typically this is caused by invalid [SysprepMassStorage] entries in the sysprep.inf.
6191	LTISysprep.wsf	FAILURE: Test for CloneTag in registry to verify Sysprep completed.	6	4	The LTISysprep script ran sysprep.exe to prepare the OS (Windows XP or Windows Server 2003) for capture, but even though it returned a zero return code the process did not appear to work. As Windows XP and Windows Server 2003 don't write a log during the sysprep process, figuring out the cause is a trial-and-error process. Typically this is caused by invalid [SysprepMassStorage] entries in the sysprep.inf. Do not ignore this error and think that it is OK because sysprep.exe reported a return code of zero; the operating system image is corrupt and needs to be recreated.
6192	LTISysprep.wsf	FAILURE: Test for SystemSetupInProgress in registry to verify Sysprep completed.	2	4	See above. Generally if this test fails, so does the previous one, so this error is not typically seen.
10401	UDIWizard.wsf	FAILURE: Download configuration file <configFile>	3	5	The UDIWizard script enables the specification of a URL for the wizard configuration XML file. In this case, an attempt was made to download that configuration file, but that download failed. Without this file, the wizard cannot be displayed. Make sure the URL specified on the UDIWizard command line was correct.
10402	UDIWizard.wsf	FAILURE: Find configuration file <configFile>	3	5	The UDIWizard tried to find the wizard configuration file either at the path specified on the command line or in a "well-known location" in the MDT toolkit files package, but it was unable to find the file. Without this file, the wizard cannot be displayed. Make sure the URL specified on the UDIWizard command line was correct or that the file is present in the MDT toolkit files package.

6402	ZTIAuthorizeDHCP.wsf	FAILURE: Locate IPv4 address for authorization	2	5	The ZTIAuthorizeDHCP script authorizes the current computer in Active Directory so that it can serve out DHCP addresses to clients on the network. This authorization is done using the IPv4 address assigned to the computer. But in this case, no valid IPv4 addresses were found on the computer. Make sure the computer has at least one valid DHCP address (preferably a static one, as that is needed by the DHCP service).
6401	ZTIAuthorizeDHCP.wsf	FAILURE: Authorized DHCP Server	2	5	The NETSH command to authorize the current server for DHCP failed. Make sure the command ran as an account that is a member of the Enterprise Administrators group in Active Directory.
6601	ZTIBCDUtility.vbs	FAILURE: GetObject(... root/wmi:BCDStore)	2	4	This MDT utility script attempted to get the WMI provider for querying the BCD store, but it was unable to do so. This might mean WMI is corrupt or running on an older OS (requires Windows Vista or above).
6602	ZTIBCDUtility.vbs	FAILURE: BCD.OpenStore (" & g_sBCDStore & ")	2	4	This MDT utility script attempted to open the BCD store via WMI but that failed. This could happen if the WMI provider is corrupt or if there is no BCD (e.g. from Windows XP).
6702	ZTIBDE.wsf	FAILURE: Moved boot files	3	1	If the boot files are presently on the OS volume, ZTIBde tries to create a new volume and then recreate the boot files on that new volume using BCDBoot. (Contrary to what the message says, it doesn't try to move the boot files. Instead, it creates new ones using BCDBoot.) This error happens when BCDBoot returns a non-zero return code. Check the BDD.LOG to see what happened.
6703	ZTIBDE.wsf	FAILURE: Create BDE Partition	1	3	ZTIBde found that the drive contained more than 2GB of unallocated (free) disk space, so it decided to create a new boot volume from that. (This typically wouldn't happen unless you preconfigured "empty" disk space for "future

					use.") Then BCDBoot was used to create new boot files on that volume, but it returned a non-zero return code. Check the BDD.LOG to see what happened.
6704	ZTIBDE.wsf	FAILURE: Degragment Drive	0	3	This error can never be reported, see error 6726 below.
6705	ZTIBDE.wsf	FAILURE: Shrink Drive	6	5	The ZTIBde script has detected that it needs to shrink the existing volume to create a new boot volume. But the DISKPART commands to do the shrink failed. Check the BDD.LOG to see what happened. Normally, this means that there wasn't enough contiguous disk space at the end of the drive to shrink the volume the specified amount.
6706	ZTIBDE.wsf	FAILURE: Testing for more than 1 partition	1	3	After creating a new boot partition (which ZTIBde will only do if there is at most one existing partition on the disk), it checks to see if there are at least two partitions now on the disk. If not, this error will be generated. This should not be possible.
6707	ZTIBDE.wsf	FAILURE: Create boot files	1	3	After creating a new volume, ZTIBde recreates the boot files on that new volume using BCDBoot. This error happens when BCDBoot returns a non-zero return code. Check the BDD.LOG to see what happened.
6701	ZTIBDE.wsf	FAILURE: Configure Protectors	6	7	As part of the configuration process, the ZTIBde script configures the BitLocker protectors that were specified. This error means that one or more of the protectors couldn't be configured. Check the BDD.LOG to see which one couldn't be configured.
6708	ZTIBDE.wsf	FAILURE: Encrypt the disk.	0	7	The ZTIBde script instructed BitLocker to begin the encryption process, but this request failed. This error is impossible because the script function called for this will signal error 6710 (below) before the script has a chance to report this one.

6769	ZTIBDE.wsf	FAILURE: Enable Protectors	6	7	As part of the configuration process, the ZTIBde script enables the BitLocker protectors that were previously configured. This error means that one or more of the protectors couldn't be enabled. Check the BDD.LOG to see which one couldn't be enabled.
6709	ZTIBDE.wsf	FAILURE: Connect to MicrosoftVolumeEncryption WMI provider	1	7	The ZTIBde script was unable to retrieve the necessary WMI object for BitLocker. This typically means that the BitLocker feature isn't installed or available in the current OS.
6710	ZTIBDE.wsf	FAILURE: Encrypting the disk	5	7	The ZTIBde script instructed BitLocker to begin the encryption process on a data drive, but this request failed. Check the BDD.LOG to see what error was returned.
6711	ZTIBDE.wsf	FAILURE: ProtectKeyWithTPM	7	7	The ZTIBde script was unable to add the TPM protector. This typically occurs on machines where a TPM isn't present or hasn't been enabled in the BIOS.
6712	ZTIBDE.wsf	FAILURE: ProtectKeyWithTPMAndPIN	7	7	The ZTIBde script was unable to add the TPM and PIN protectors. This typically occurs on machines where a TPM isn't present or hasn't been enabled in the BIOS, or when the PIN specified doesn't meet the requirements specified via Group Policy.
6713	ZTIBDE.wsf	FAILURE: ProtectKeyWithTPMAndStartupKey	4	7	The ZTIBde script was unable to add the TPM and startup key protectors. This typically occurs on machines where a TPM isn't present or hasn't been enabled in the BIOS.
6714	ZTIBDE.wsf	FAILURE: Save External Key to File	7	7	After adding the startup key, the ZTIBde script attempts to back up the key to the specified removable disk or network path. In this case, that process failed to save the key.
6715	ZTIBDE.wsf	FAILURE: Protect with External Key	2	7	The ZTIBde script was unable to add the external key protector.
6716	ZTIBDE.wsf	FAILURE: Save external key to file	7	7	After adding the external key, the ZTIBde script attempts to back up the key to the specified

					removable disk or network path. In this case, that process failed to save the key.
6717	ZTIBDE.wsf	FAILURE: Protect Key with Numerical Password	3	7	The ZTIBde script was unable to add the specified numerical password protector.
6718	ZTIBDE.wsf	FAILURE: GetKeyProtectorNumberialP@ssword:	3	7	The ZTIBde script was unable to retrieve the numerical password that was generated for the volume.
6718	ZTIBDE.wsf	FAILURE: Save Password to File	3	7	The ZTIBde script was unable to save the numerical password to the specified removable disk or network path.
6719	ZTIBDE.wsf	FAILURE: Open <password file>	2	5	The password file specified could not be opened so the password could not be saved into it.
6720	ZTIBDE.wsf	FAILURE: Encrypt the drive	6	7	The ZTIBde script instructed BitLocker to begin the encryption process on the OS drive, but this request failed. Check the BDD.LOG to see what error was returned. Often this happens because the required protectors specified via Group Policy weren't configured.
6766	ZTIBDE.wsf	FAILURE: Get BDE Instance	1	7	The ZTIBde script was unable to retrieve the necessary WMI object for BitLocker. This typically means that the BitLocker feature isn't installed or available in the current OS.
6767	ZTIBDE.wsf	FAILURE: Enable BDE Protectors	4	5	The ZTIBde script wants to enable the BitLocker protectors that have already been configured. But this process failed, so the error was reported.
6721	ZTIBDE.wsf	FAILURE: Open<script file>	1	3	The ZTIBde script wants to shrink the current OS volume. To do this, it is trying to create a file to hold a DISKPART script with the needed commands, but the attempt to create the file failed.
6722	ZTIBDE.wsf	FAILURE: Create partition	4	2	The DISKPART script to shrink the current OS volume failed, causing DISKPART to return a non-zero return code. Check the ZTIShrinkDrive_diskpart.log file to see what happened.

6723	ZTIBDE.wsf	FAILURE: Get existing BDE Drive	4	3	The ZTIBde script attempted to retrieve the existing boot volume, but was unable to do so.
6724	ZTIBDE.wsf	FAILURE: Open<script file>	1	2	The ZTIBde script wants to change the drive letter of the current boot drive. To do this, it is trying to create a file to hold a DISKPART script with the needed commands, but the attempt to create the file failed.
6725	ZTIBDE.wsf	FAILURE: Execute cmd /c "DISKPART.EXE /s "…	4	2	The ZTIBde script ran DISKPART to change the drive letter, but it reported a non-zero return code. Check the ZTIBdeFix_diskpart.log file to see what happened.
6726	ZTIBDE.wsf	FAILURE: Execute cmd /c DEFRAG.EXE …	3	3	The ZTIBde script has detected that it needs to shrink the existing volume to create a new boot volume. To ensure that this shrink is likely to succeed, it first runs DEFRAG.EXE to try to free up contiguous disk space at the end of the volume. In this case, DEFRAG.EXE returned a non-zero return code. This should not normally occur; check the ZTIDefrag.log to see what happened.
6728	ZTIBDE.wsf	FAILURE: Execute cmd /c ""DISKPART.EXE /s ...	6	4	The ZTIBde script ran DISKPART to shrink the OS volume, but DISKPART returned a non-zero return code. Check the ZTIShrinkDrive_diskpart.log to see what happened.
6734	ZTIBDE.wsf	FAILURE: Get TPM Instance	4	6	The ZTIBde script is attempting to validate the TPM, but can't retrieve a TPM instance from WMI. Maybe the TPM isn't enabled in the BIOS?
6735	ZTIBDE.wsf	FAILURE: Check to see if TPM is enabled	6	7	The TPM can be seen by the OS, but the ZTIBde script wasn't able to verify that it was enabled.
6736	ZTIBDE.wsf	FAILURE: Check to see if TPM is activated	6	7	The TPM was enabled, but the ZTIBde script wasn't able to verify that it was activated.
6737	ZTIBDE.wsf	FAILURE: Check to see if TPM is owned	6	7	The TPM was activated, but the ZTIBde script wasn't able to verify that it was owned by the OS.
6738	ZTIBDE.wsf	FAILURE: Check to see if TPM Ownership is allowed	6	7	The TPM was not owned, but the ZTIBde script couldn't tell if ownership was allowed.
6739	ZTIBDE.wsf	FAILURE: Check to see if TPM is enabled	6	7	The TPM can be seen by the OS, but the ZTIBde script wasn't able to verify that it was enabled.

6740	ZTIBDE.wsf	FAILURE: Check to see if TPM is activated	6	7	The TPM was enabled, but the ZTIBde script wasn't able to verify that it was activated.
6741	ZTIBDE.wsf	FAILURE: Check to see if TPM is owned and ownership is allowed	6	7	The TPM was not owned, but the ZTIBde script couldn't tell if ownership was allowed.
6741	ZTIBDE.wsf	FAILURE: TPM Owner Password set	6	7	The ZTIBde script was unable to set the TPM owner password.
6742	ZTIBDE.wsf	FAILURE: TPM Owner P@ssword set to AdminP@ssword	6	7	The ZTIBde script was unable to set the TPM owner password to the value specified in the AdminPassword task sequence variable.
6743	ZTIBDE.wsf	FAILURE: Set TPM Owner P@ssword to value	6	7	The ZTIBDe script was unable to set the TPM owner password to the default value of "M0nksH00d!4T3al".
6744	ZTIBDE.wsf	FAILURE: Check to see if TPM is enabled	6	7	The TPM can be seen by the OS, but the ZTIBde script wasn't able to verify that it was enabled.
6745	ZTIBDE.wsf	FAILURE: Check TPM Owner	6	7	The TPM was activated, but the ZTIBde script wasn't able to verify that it was owned by the OS.
6747	ZTIBDE.wsf	FAILURE: Check to see if TPM is activated	6	7	The TPM was enabled, but the ZTIBde script wasn't able to verify that it was activated.
6748	ZTIBDE.wsf	FAILURE: Check to see if TPM Ownership is allowed	6	7	The TPM was not owned, but the ZTIBde script couldn't tell if ownership was allowed.
6749	ZTIBDE.wsf	FAILURE: Convert owner p@ssword to owner authorization	6	7	The specified TPM owner password could not be converted to an owner authorization string.
6750	ZTIBDE.wsf	FAILURE: Create endorsement key pair	6	7	An endorsement key pair could not be created for the TPM.
6751	ZTIBDE.wsf	FAILURE: Change owner authorization	6	7	The owner authorization for the TPM could not be changed.
6753	ZTIBDE.wsf	FAILURE: Validate TPM	6	7	The ZTIBde script could not valid that the TPM was ready for BitLocker.
6754	ZTIBDE.wsf	FAILURE: Get BDE Instance	6	7	The ZTIBde script was not able to retrieve the BitLocker WMI object.
6755	ZTIBDE.wsf	FAILURE: Protect Key with TPM	6	7	The ZTIBde script was not able to add the TPM protector.

6764	ZTIBDE.wsf	FAILURE: Configure Bitlocker Policy	6	7	The ZTIBde script was not able to set the required policy for UseAdvancedStartup and EnableBDEWithNoTPM.
6756	ZTIBDE.wsf	FAILURE: Check for removable media to configure ProtectKeyWithTpmAndStartupKey	6	7	No removable media was found for saving the startup key.
6757	ZTIBDE.wsf	FAILURE: Protect key with TPM and statup key	6	7	The attempt to enable the TPM and startup key protectors failed.
6764	ZTIBDE.wsf	FAILURE: Configure Bitlocker Policy	6	7	The ZTIBde script was not able to set the required policy for UseAdvancedStartup and EnableBDEWithNoTPM.
6758	ZTIBDE.wsf	FAILURE: Look for BDE Pin	6	7	The ZTIBde script wanted to enable a PIN protector, but no PIN was specified in the required task sequence variable.
6759	ZTIBDE.wsf	FAILURE: Protect key with TPM and Pin	6	7	The attempt to enable the TPM and PIN protectors failed.
6764	ZTIBDE.wsf	FAILURE: Configure Bitlocker Policy	6	7	The ZTIBde script was not able to set the required policy for UseAdvancedStartup and EnableBDEWithNoTPM.
6760	ZTIBDE.wsf	FAILURE: Find removable media for BDEKeyLocation	6	7	No removable media was found for saving the external key.
6761	ZTIBDE.wsf	FAILURE: Protect with External Key	6	7	The attempt to add the external key protector failed.
6762	ZTIBDE.wsf	FAILURE: Recovery P@ssword being saved to <password file>	6	7	The external key could not be saved to the external file.
7000	ZTIConfigure.wsf	FAILURE: Unable to locate ZTIConfigure.xml, aborting	1	7	The ZTIConfigure script injects settings into the sysprep.inf, unattend.txt, or unattend.xml as needed. It needs the ZTIConfigure.xml file to figure out what needs to be injected where. This error means that it wasn't able to find that file. Make sure the file is present in the Scripts folder of the deployment share.
7001	ZTIConfigure.wsf	FAILURE: Looking for unattend AnswerFile	1	5	The ZTIConfigure script injects settings into the sysprep.inf, unattend.txt or unattend.xml files as needed. But an appropriate file could not be located. Typically this won't happen.

7813	ZTIDiskpart.wsf	FAILURE: Verify there are partitions defined in this Task Sequence Step.	1	6	The ZTIDiskpart step is running to format and partition a disk, but not partitions were defined. Make sure you define at least one partition.
7714	ZTIDiskpart.wsf	FAILURE: Verify that any Bitlocker implementation does not have EXTENDED or LOGICAL Drives	2	9	You have specified to create extended or logical partitions as well as to enable BitLocker. BitLocker doesn't support extended or logical partitions. The simple solution is to not use extended or logical partitions.
7718	ZTIDiskpart.wsf	FAILURE: Verify drive object is created	1	2	This should not happen as this is related to internal script logic.
7815	ZTIDiskpart.wsf	FAILURE: Verify Partition size is Numeric: <size string>	2	4	It is possible to override the partition size via a task sequence variable, e.g. OSDPartitions1Size, to specify a different size than what was configured in the "Format and Partition Disk" UI. But in this case, the variable value is not a valid number, hence the error.
7817	ZTIDiskpart.wsf	FAILURE: Verify There are Free drives available.	3	4	Each volume that is formatted by ZTIDiskpart needs to have a drive letter assigned to it. But there are only 26 letters available, so when you run out you will see this error. (The letter can be removed after the MDT "Format and Partition Disk" step runs.)
7801	ZTIDiskUtility.vbs	FAILURE: Verify iDisk index is numeric: <disk index string>	3	2	You can specify the index of the disk to be formatted by setting the OSDDiskIndex variable. In this case, the variable was set to an invalid non-numeric value, hence the error.
7802	ZTIDiskUtility.vbs	FAILURE: Verify oDisk index is an object. <object>	1	2	This should not happen as this is related to internal script logic.
7803	ZTIDiskUtility.vbs	FAILURE: Verify Disk is correct WMI Object. <wmiPath>	1	2	This should not happen as this is related to internal script logic.
7804	ZTIDiskUtility.vbs	FAILURE: Verify First object is set.	1	2	This should not happen as this is related to internal script logic.
7805	ZTIDiskUtility.vbs	FAILURE: Verify oDisk index is an object. <object>	1	2	This should not happen as this is related to internal script logic.
7806	ZTIDiskUtility.vbs	FAILURE: Verify Drive is correct WMI Object. <wmiPath>	1	2	This should not happen as this is related to internal script logic.

7807	ZTIDiskUtility.vbs	FAILURE: Verify Disk has been mapped.	1	2	This should not happen as this is related to internal script logic.
7808	ZTIDiskUtility.vbs	FAILURE: Verify oDisk index is an object. <object>	1	2	This should not happen as this is related to internal script logic.
7809	ZTIDiskUtility.vbs	FAILURE: Verify DiskPart is correct WMI Object. <wmiPath>	1	2	This should not happen as this is related to internal script logic.
7811	ZTIDiskUtility.vbs	FAILURE: FindFile: BCDBoot.exe	1	3	The BCDBOOT program could not be found on the deployment share. This is typically put there automatically by MDT, so this typically wouldn't happen.
7810	ZTIDiskUtility.vbs	FAILURE: Verify UILanguage is set.	2	4	To run BCDBOOT from Windows PE, as MDT typically does when deploying a new OS using ImageX (for Windows Vista and above), you need to specify the language of the new OS. Typically this is done automatically, so you wouldn't typically see this error.
7814	ZTIDiskUtility.vbs	FAILURE: Verify class created: <wmiPath>	1	2	This should not happen as this is related to internal script logic.
7815	ZTIDiskUtility.vbs	FAILURE: Verify WMI Object was accepted by ZTIDiskPartition	1	2	This should not happen as this is related to internal script logic.
10203	ZTIDomainJoin.wsf	FAILURE: FindFile(LTISuspend.wsf)	1	3	If you set task sequence variable DomainErrorRecovery to "MANUAL" and the ZTIDomainJoin script fails to join the specified domain, it will suspend the task sequence. (By default, it reboots and retries automatically.) In order to do this, it needs to find the LTISuspend.wsf script. This error occurs if the script cannot be found, which shouldn't happen.
10204	ZTIDomainJoin.wsf	FAILURE: Run Program <LTISuspend>	1	3	See above. In this case, the script was found but it returned a non-zero return code. Check the BDD.LOG to see what happened.
7001	ZTIDrivers.wsf	FAILURE: Looking for unattend AnswerFile	1	4	On Windows XP and Windows Server 2003, the ZTIDrivers script needs to find the sysprep.inf or unattend.txt file in order to append new folders to the OEMPnpDriversPath. In this case, the error

					means that it wasn't able to find the file. That shouldn't happen.
7901	ZTIDrivers.wsf	FAILURE: AllDrivers.Exists(<GUID>)	1	3	The ZTIDrivers script identified a driver that it wanted to copy to the local machine. But when it went to find the driver in the Drivers.xml file, it wasn't there. That should be impossible.
7904	ZTIDrivers.wsf	FAILURE: AllDrivers.Exists(<GUID>)	1	3	The ZTIDrivers script identified a Windows XP mass storage driver that it wanted to copy to the local machine. But when it went to find the driver in the Drivers.xml file, it wasn't there. That should be impossible.
7908	ZTIDrivers.wsf	FAILURE: Test dFilteredDrivers.Item(<GUID>)	1	2	The ZTIDrivers script identified a Windows XP mass storage driver that it wanted to process. But when it went to find the driver in the Drivers.xml file, it wasn't there. That should be impossible.
7900	ZTIDrivers.wsf	FAILURE: Findfile: Microsoft.BDD.PnpEnum.exe	1	4	The ZTIDrivers script identifies what drivers are needed by scanning the PnP IDs that are present on the computer. The Microsoft.BDD.PnpEnum.exe program generates this list. This error means that the executable couldn't be found. That should not normally happen.
9001	ZTIOSRole.wsf	FAILURE: FindFile ServerManager.xml	1	5	The ZTIOSRole script needs to read the ServerManager.xml file to determine how to install roles and features for various OSes. This error means that it wasn't able to find that file. This should not happen.
7001	ZTIPatches.wsf	FAILURE: Looking for unattend AnswerFile	1	3	The ZTIPatches script makes sure that the unattend.xml contains the needed offline servicing entries. So it needs to find that file. This error means that it wasn't able to find that file. This should not happen.
5400	ZTIUtility.vbs	FAILURE: Create object: Set <classInstance> = New <className>	3	6	The ZTIUtility script actually kicks off the execution of the class defined in the WSF files provided with MDT. Generally, you would only see this error with your own custom scripts. This error would

					happen if the class name specified in the script doesn't match the name of the script. For example, if your script is named "MyScript.wsf" then the class should be declared as "Class MyScript".
5441	ZTIUtility.vbs	FAILURE: FindFile: <cmd>	1	4	This is a generic error that could happen when any MDT script asks ZTIUtility to locate and run a particular executable.
5442	ZTIUtility.vbs	FAILURE: FindFile: <cmd>	1	4	This is a generic error that could happen when any MDT script asks ZTIUtility to locate and run a particular executable.
5490	ZTIUtility.vbs	FAILURE: Create MSXML2.DOMDocument.	2	4	MDT uses MSXML3 to read XML files for a variety of purposes. If for some reason MDT can't create the necessary MSXML3 object, this error will be reported.
5495	ZTIUtility.vbs	FAILURE: Create MSXML2.DOMDocument .ParseErr.ErrCode.	3	2	MDT read an XML file but encountered parsing errors because the file was not valid. Try to figure out what file was being processed and check it to see if there are any obvious errors in the XML structure.
5496	ZTIUtility.vbs	FAILURE: LoadControlFile.FindFile: <configFile>	2	4	MDT uses XML files to track lists of applications, task sequences, drivers, etc. If one of these XML files cannot be found, this error will be logged.
5452	ZTIUtility.vbs	FAILURE: Verify ZTIDiskPArt is loaded	2	4	In some situations, the ZTIUtility script will dynamically load the ZTIDiskUtility script for calculating the destination disk and partition. If for some reason the ZTIDiskUtility script can't be dynamically loaded, you will see this error. This shouldn't normally happen.
5451	ZTIUtility.vbs	FAILURE: Verify oDiskPart.Drive is found!	1	4	MDT was unable to get information about the drive identified as the target for the OS. This shouldn't normally happen.
10301	ZTIVHDCreate.wsf	FAILURE: Verify source disk was found %VHDCreateSource%	1	3	The ZTIVHDCreate script was told to make a copy of the source VHD specified in task sequence variable VHDCreateSource, but it was unable to find the specified VHD file. This variable is not set

					by default, so if you do set it, make sure the value specified is valid.
10311	ZTIVHDCreate.wsf	FAILURE: Verify Win32_LogicalDrive is available	2	3	The ZTIVHDCreate script will retrieve the drive letter from the variable that the VHDInputVariable task sequence points to. (For example, if VHDInputVariable is set to "VHDDisk" and the VHDDisk variable is set to M:, then the script would use M:.) If that variable is blank, then it will pick the first available OS volume. In this case, the value specified or chosen wasn't valid as it couldn't be retrieved from WMI. Make sure you specify a valid drive letter.
10302	ZTIVHDCreate.wsf	FAILURE: Verify file was created: <vhd file>	3	4	After using DISKPART to create a VHD file, the ZTIVHDCreate script verifies that the file was indeed created. In this case, the file couldn't be found, which probably means that the DISKPART commands failed. Check the BDD.LOG to see what happened.
10303	ZTIVHDCreate.wsf	FAILURE: Verify Diff file is not the same name as the Parent: [<vhd file>] <> [<diff file>]	1	4	The ZTIVHDCreate script can create a differencing disk on the specified parent disk. (By default it doesn't do this.) If you specify the same file name for both, you will get this error, so don't do that.
10303	ZTIVHDCreate.wsf	FAILURE: Verify file was created: <diff file>	1	4	After using DISKPART to create a VHD differencing disk, the ZTIVHDCreate script verifies that the file was indeed created. In this case, the file couldn't be found, which probably means that the DISKPART commands failed. Check the BDD.LOG to see what happened.
10310	ZTIVHDCreate.wsf	FAILURE: Verify that the DIsk COunt increased by 1: <prev count> <new count>	2	4	After creating and attaching a new VHD file, the ZTIVHDCreate script expects the number of disks reported via WMI to increase by one. If this doesn't happen (possibly because DISKPART was unable to attach the disk, maybe due to antivirus software), you will see this error. Check the BDD.LOG to see the output from the DISKPART commands used to attach the disk.

10303	ZTIVHDCreate.wsf	FAILURE: Verify that a new disk was created.	1	4	See above. In this case, a new disk was found because the count increased, but the script couldn’t find it in the list. This should be impossible.
10501	ZTIWinRE.wsf	FAILURE: Verify a boot drive object was returned.	2	4	The ZTIWinRE script wants to copy the Windows RE boot image onto the boot volume. In order to do that, it needs to know the drive letter of that volume. If there is no drive letter assigned, it will assign one. But in the case of this error, no boot drive could be located, or no drive letter could be assigned. Check the BDD.LOG to see what happened.
10502	ZTIWinRE.wsf	FAILURE: Verify a drive was returned: <object>	1	3	In this case, a drive object was found but it had no drive letter. This should be impossible. Check the BDD.LOG to see what happened.
10503	ZTIWinRE.wsf	FAILURE: Verify File: <found path>	8	2	The ZTIWinRE script tried to find the LiteTouchPE_<architecture>.wim file on the deployment share, but it wasn’t there. That typically happens when you haven’t yet updated the deployment share to generate this WIM file (or when that architecture isn’t enabled).
10504	ZTIWinRE.wsf	FAILURE: Verify REAgentC.exe is found	1	6	The ZTIWinRE script tried to find the REAgentC command needed to tell Windows RE about the presence of a new boot WIM file, but it was unable to find the file. This should not happen.

User State Migration Tool Return Codes

The User State Migration Tool can report various errors. While these do vary somewhat for each version of USMT, they are mostly consistent between versions. For the specifics of each version, see the following links:

- USMT 3.0. Not available online.
- USMT 4.0. http://technet.microsoft.com/en-us/library/dd823291(v=WS.10).aspx
- USMT 5.0. http://technet.microsoft.com/en-us/library/hh824897.aspx

The following are the return codes reported by USMT 5.0:

Return code value	Return code	Error message	Troubleshooting, mitigation, workarounds	Category
0	USMT_SUCCESS	Successful run	Not applicable	Success or Cancel
1	USMT_DISPLAY_HELP	Command line help requested	Not applicable	Success or Cancel
2	USMT_STATUS_CANCELED	Gather was aborted because of an EFS file	Not applicable	
		User chose to cancel (such as pressing CTRL+C)	Not applicable	Success or Cancel
3	USMT_WOULD_HAVE_FAILED	At least one error was skipped as a result of /c	Review ScanState, LoadState, or UsmtUtils log for details about command-line errors.	
11	USMT_INVALID_PARAMETERS	/all conflicts with /ui, /ue or /uel	Review ScanState log or LoadState log for details about command-line errors.	
		/auto expects an optional parameter for the script folder	Review ScanState log or LoadState log for details about command-line errors.	
		/encrypt can't be used with /nocompress	Review ScanState log or LoadState log for details about command-line errors.	

		/encrypt requires /key or /keyfile	Review ScanState log or LoadState log for details about command-line errors.	
		/genconfig can't be used with most other options	Review ScanState log or LoadState log for details about command-line errors.	
		/genmigxml can't be used with most other options	Review ScanState log or LoadState log for details about command-line errors.	
		/hardlink requires /nocompress	Review ScanState log or LoadState log for details about command-line errors.	
		/key and /keyfile both specified	Review ScanState log or LoadState log for details about command-line errors.	
		/key or /keyfile used without enabling encryption	Review ScanState log or LoadState log for details about command-line errors.	
		/lae is only used with /lac	Review ScanState log or LoadState log for details about command-line errors.	
		/listfiles cannot be used with /p	Review ScanState log or LoadState log for details about command-line errors.	
		/offline requires a valid path to an XML file describing offline paths	Review ScanState log or LoadState log for details about command-line errors.	
		/offlinewindir requires a valid path to offline windows folder	Review ScanState log or LoadState log for details about command-line errors.	
		/offlinewinold requires a valid path to offline windows folder	Review ScanState log or LoadState log for details about command-line errors.	
		A command was already specified	Verify that the command-line syntax is correct and that there are no duplicate commands.	

		An option argument is missing	Review ScanState log or LoadState log for details about command-line errors.	
		An option is specified more than once and is ambiguous	Review ScanState log or LoadState log for details about command-line errors.	
		By default /auto selects all users and uses the highest log verbosity level. Switches like /all, /ui, /ue, /v are not allowed.	Review ScanState log or LoadState log for details about command-line errors.	
		Command line arguments are required. Specify /? for options.	Review ScanState log or LoadState log for details about command-line errors.	
		Command line option is not valid	Review ScanState log or LoadState log for details about command-line errors.	
		EFS parameter specified is not valid for /efs	Review ScanState log or LoadState log for details about command-line errors.	
		File argument is invalid for /genconfig	Review ScanState log or LoadState log for details about command-line errors.	
		File argument is invalid for /genmigxml	Review ScanState log or LoadState log for details about command-line errors.	
		Invalid space estimate path. Check the parameters and/or file system permissions	Review ScanState log or LoadState log for details about command-line errors.	
		List file path argument is invalid for /listfiles	Review ScanState log or LoadState log for details about command-line errors.	
		Retry argument must be an integer	Review ScanState log or LoadState log for details about command-line errors.	
		Settings store argument specified is invalid	Review ScanState log or LoadState log for details about command-line errors. Make sure that the store	

			path is accessible and that the proper permission levels are set.	
		Specified encryption algorithm is not supported	Review ScanState log or LoadState log for details about command-line errors.	
		The /efs:hardlink requires /hardlink	Review ScanState log or LoadState log for details about command-line errors.	
		The /targetWindows7 option is only available for Windows XP, Windows Vista®, and Windows 7	Review ScanState log or LoadState log for details about command-line errors.	
		The store parameter is required but not specified	Review ScanState log or LoadState log for details about command-line errors.	
		The source-to-target domain mapping is invalid for /md	Review ScanState log or LoadState log for details about command-line errors.	
		The source-to-target user account mapping is invalid for /mu	Review ScanState log or LoadState log for details about command-line errors.	
		Undefined or incomplete command line option	Review ScanState log or LoadState log for details about command-line errors.	Invalid Command Lines
		Use /nocompress, or provide an XML file path with /p"pathtoafile" to get a compressed store size estimate	Review ScanState log or LoadState log for details about command-line errors.	
		User exclusion argument is invalid	Review ScanState log or LoadState log for details about command-line errors.	
		Verbosity level must be specified as a sum of the desired log options: Verbose (0x01), Record Objects (0x04), Echo to debug port (0x08)	Review ScanState log or LoadState log for details about command-line errors.	
		Volume shadow copy feature is not supported with a hardlink store	Review ScanState log or LoadState log for details about command-line errors.	

		Wait delay argument must be an integer	Review ScanState log or LoadState log for details about command-line errors.	
12	USMT_ERROR_OPTION_PARAM_TOO_LARGE	Command line arguments cannot exceed 256 characters	Review ScanState log or LoadState log for details about command-line errors.	Invalid Command Lines
		Specified settings store path exceeds the maximum allowed length of 256 characters	Review ScanState log or LoadState log for details about command-line errors.	
13	USMT_INIT_LOGFILE_FAILED	Log path argument is invalid for /l	When /l is specified in the ScanState command line, USMT validates the path. Verify that the drive and other information, for example file system characters, are correct.	Invalid Command Lines
14	USMT_ERROR_USE_LAC	Unable to create a local account because /lac was not specified	When creating local accounts, the command-line options /lac and /lae should be used.	Invalid Command Lines
26	USMT_INIT_ERROR	Multiple Windows installations found	Listfiles.txt could not be created. Verify that the location you specified for the creation of this file is valid.	Setup and Initialization
		Software malfunction or unknown exception	Check all loaded .xml files for errors, common error when using /I to load the Config.xml file.	
		Unable to find a valid Windows directory to proceed with requested offline operation; Check if offline input file is present and has valid entries	Verify that the offline input file is present and that it has valid entries. USMT could not find valid offline operating system. Verify your offline directory mapping.	
27	USMT_INVALID_STORE_LOCATION	A store path can't be used because an existing store exists; specify /o to overwrite	Specify /o to overwrite an existing intermediate or migration store.	Setup and Initialization
		A store path is missing or has incomplete data	Make sure that the store path is accessible and that the proper permission levels are set.	

		An error occurred during store creation	Make sure that the store path is accessible and that the proper permission levels are set. Specify /o to overwrite an existing intermediate or migration store.	
		An inappropriate device such as a floppy disk was specified for the store	Make sure that the store path is accessible and that the proper permission levels are set.	
		Invalid store path; check the store parameter and/or file system permissions	Invalid store path; check the store parameter and/or file system permissions	
		The file layout and/or file content is not recognized as a valid store	Make sure that the store path is accessible and that the proper permission levels are set. Specify /o to overwrite an existing intermediate or migration store.	
		The store path holds a store incompatible with the current USMT version	Make sure that the store path is accessible and that the proper permission levels are set.	
		The store save location is read-only or does not support a requested storage option	Make sure that the store path is accessible and that the proper permission levels are set.	
28	USMT_UNABLE_GET_SCRIPTFILES	Script file is invalid for /i	Check all specified migration .xml files for errors. This is a common error when using /i to load the Config.xml file.	Setup and Initialization
		Unable to find a script file specified by /i	Verify the location of your script files, and ensure that the command-line options are correct.	
29	USMT_FAILED_MIGSTARTUP	A minimum of 250 MB of free space is required for temporary files	Verify that the system meets the minimum temporary disk space requirement of 250 MB. As a workaround, you can set the environment variable USMT_WORKING_DIR=<path> to	Setup and Initialization

			redirect the temporary files working directory.	
		Another process is preventing migration; only one migration tool can run at a time	Check the ScanState log file for migration .xml file errors.	
		Failed to start main processing, look in log for system errors or check the installation	Check the ScanState log file for migration .xml file errors.	
		Migration failed because of an XML error; look in the log for specific details	Check the ScanState log file for migration .xml file errors.	
		Unable to automatically map the drive letters to match the online drive letter layout; Use /offline to provide a mapping table	Check the ScanState log file for migration .xml file errors.	
31	USMT_UNABLE_FINDMIGUNITS	An error occurred during the discover phase; the log should have more specific information	Check the ScanState log file for migration .xml file errors.	Setup and Initialization
32	USMT_FAILED_SETMIGRATIONTYP E	An error occurred processing the migration system	Check the ScanState log file for migration .xml file errors, or use online Help by typing /? on the command line.	Setup and Initialization
33	USMT_UNABLE_READKEY	Error accessing the file specified by the /keyfile parameter	Check the ScanState log file for migration .xml file errors, or use online Help by typing /? on the command line.	Setup and Initialization
		The encryption key must have at least one character	Check the ScanState log file for migration .xml file errors, or use online Help by typing /? on the command line.	
34	USMT_ERROR_INSUFFICIENT_RIGH TS	Directory removal requires elevated privileges	Log on as Administrator, and run with elevated privileges.	Setup and Initialization
		No rights to create user profiles; log in as Administrator; run with elevated privileges	Log on as Administrator, and run with elevated privileges.	
		No rights to read or delete user profiles; log in as Administrator, run with elevated privileges	Log on as Administrator, and run with elevated privileges.	
35	USMT_UNABLE_DELETE_STORE	A reboot is required to remove the store	Reboot to delete any files that could not be deleted when the command was executed.	Setup and Initialization

		A store path can't be used because it contains data that could not be overwritten	A migration store could not be deleted. If you are using a hardlink migration store you might have a locked file in it. You should manually delete the store, or use **usmtutils /rd** command to delete the store.	
		There was an error removing the store	Review ScanState log or LoadState log for details about command-line errors.	
36	USMT_ERROR_UNSUPPORTED_PLATFORM	Compliance check failure; please check the logs for details	Investigate whether there is an active temporary profile on the system.	Setup and Initialization
		Use of /offline is not supported during apply	The **/offline** command was not used while running in the Windows Preinstallation Environment (Windows PE).	
		Use /offline to run gather on this platform	The **/offline** command was not used while running in Windows PE.	
37	USMT_ERROR_NO_INVALID_KEY	The store holds encrypted data but the correct encryption key was not provided	Verify that you have included the correct encryption /key or /keyfile.	Setup and Initialization
38	USMT_ERROR_CORRUPTED_NOTENCRYPTED_STORE	An error occurred during store access	Review ScanState log or LoadState log for details about command-line errors. Make sure that the store path is accessible and that the proper permission levels are set.	Setup and Initialization
39	USMT_UNABLE_TO_READ_CONFIG_FILE	Error reading Config.xml	Review ScanState log or LoadState log for details about command-line errors in the Config.xml file.	Setup and Initialization
		File argument is invalid for /config	Check the command line you used to load the Config.xml file. You can use online Help by typing /? on the command line.	
40	USMT_ERROR_UNABLE_CREATE_PROGRESS_LOG	Error writing to the progress log	The Progress log could not be created. Verify that the location is	Setup and Initialization

			valid and that you have write access.	
		Progress log argument is invalid for /progress	The Progress log could not be created. Verify that the location is valid and that you have write access.	
41	USMT_PREFLIGHT_FILE_CREATION_FAILED	Can't overwrite existing file	The Progress log could not be created. Verify that the location is valid and that you have write access.	Setup and Initialization
		Invalid space estimate path. Check the parameters and/or file system permissions	Review ScanState log or LoadState log for details about command-line errors.	
42	USMT_ERROR_CORRUPTED_STORE	The store contains one or more corrupted files	Review UsmtUtils log for details about the corrupted files. For information on how to extract the files that are not corrupted, see [How to Extract Files from a Compressed USMT Migration Store]().	
61	USMT_MIGRATION_STOPPED_NONFATAL	Processing stopped due to an I/O error	USMT exited but can continue with the /c command-line option, with the optional configurable <ErrorControl> section or by using the /vsc command-line option.	Non-fatal Errors
71	USMT_INIT_OPERATING_ENVIRONMENT_FAILED	A Windows Win32 API error occurred	Data transfer has begun, and there was an error during the creation of migration store or during the apply phase. Review the ScanState log or LoadState log for details.	Fatal Errors
		An error occurred when attempting to initialize the diagnostic mechanisms such as the log	Data transfer has begun, and there was an error during the creation of migration store or during the apply phase. Review the ScanState log or LoadState log for details.	

		Failed to record diagnostic information	Data transfer has begun, and there was an error during the creation of migration store or during the apply phase. Review the ScanState log or LoadState log for details.	
		Unable to start. Make sure you are running USMT with elevated privileges	Exit USMT and log in again with elevated privileges.	
72	USMT_UNABLE_DOMIGRATION	An error occurred closing the store	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.	Fatal Errors
		An error occurred in the apply process	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.	
		An error occurred in the gather process	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.	
		Out of disk space while writing the store	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.	
		Out of temporary disk space on the local system	Data transfer has begun, and there was an error during migration-store creation or during the apply phase. Review the ScanState log or LoadState log for details.	