



Migration Automation Toolkit (MAT)

MAT v1.5 Installation and Usage Guide

Table of Contents

Contents

Table of Contents	2
Assumptions	5
1. General Information	6
2. Hardware Requirements	7
2.1. Conversion Servers.....	7
2.2. Hyper-V.....	7
3. Software Requirements.....	8
3.1. Control Server	8
3.2. Helper Server.....	8
3.3. Guest Virtual Machine	8
4. Supported Configurations	9
4.1.1. VMware sources	9
4.1.2. Destination Host Server	9
4.1.3. Guest operating systems supported for conversion.....	9
4.2. Supported Configurations for Disk Conversion	10
4.3. Assumptions.....	10
5. Upgrade Information	11
6. Architecture Diagram	12
7. Install SQL 2008 Express or later	14
8. Install Microsoft Virtual Machine Converter v1.0.....	15
9. Install vSphere PowerCLI 5.1	16
10. Install the Migration Automation Toolkit.....	17
10.1. Create a MAT folder.....	17
10.2. Unpack the toolkit.....	17
10.3. Create a share for the Target Host	17
10.4. Run the <i>Setup MAT.sql</i> script.....	18
11. Configure MAT Settings	20
11.1. General.....	20
11.2. SchedCred	20
11.3. HyperV	21
11.4. VMware.....	21
11.5. RemoteHost.....	22
12. Install MAT on Helper Servers (Optional)	23
13. Collect VM information.....	24
14. List Management	25
15. Conversion	27
16. Monitoring Conversions	28
17. The PrepareVM function	29
17.1. Removing CD\DVD.....	29

Migration Automation Toolkit

17.2. Removing Floppy disks	29
17.3. Network Interface Card information capture and restore	29
17.3.1. How the MigrateNICs function works.....	30
18. Reporting	32
19. Database Maintenance	33
20. Duplicate Conversions.....	34
21. Interrupting a Conversion.....	35
22. Advanced Concepts	36
22.1. Overall PowerShell workflow.....	36
22.1.1. Control Server.....	36
22.1.2. Helper Server	39
22.2. MAT Extensibility	40
22.2.1. Extensions which hold the queue until finished	40
22.3. Shared Logs	41
22.4. Shared XML.....	41
22.5. Balancing the load	42
22.5.1. One Hyper-V host for each Helper.....	42
23. Additional Functions	43
23.1.1. Actions.....	43
Start Delay	44
24. PowerShell Parameter and Variable Reference	45
24.1. ConvertVM.ps1	45
24.1.1. Parameters.....	45
24.1.2. Variables	45
24.2. ConvertVM-Process.ps1	46
24.2.1. Parameters.....	46
24.2.2. Variables	46
24.3. ConvertVM-Logging.ps1	47
24.3.1. Parameters.....	47
24.3.2. Variables	47
25. Permissions	48
25.1. VMware Virtual Infrastructure.....	48
25.2. Guest Operating System (VM being converted)	48
25.3. Control Server and Helper Server(s)	48
26. Getting help	49
Appendix A: Document History	50
Document History	50

Migration Automation Toolkit

Copyright © 2014 Microsoft Corporation. All rights reserved. Complying with the applicable copyright laws is your responsibility. By using or providing feedback on this documentation, you agree to the license agreement below.

If you are using this documentation solely for non-commercial purposes internally within YOUR company or organization, then this documentation is licensed to you under the Creative Commons Attribution-NonCommercial License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/2.5/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.

This documentation is provided to you for informational purposes only, and is provided to you entirely "AS IS". Your use of the documentation cannot be understood as substituting for customized service and information that might be developed by Microsoft Corporation for a particular user based upon that user's particular environment. To the extent permitted by law, MICROSOFT MAKES NO WARRANTY OF ANY KIND, DISCLAIMS ALL EXPRESS, IMPLIED AND STATUTORY WARRANTIES, AND ASSUMES NO LIABILITY TO YOU FOR ANY DAMAGES OF ANY TYPE IN CONNECTION WITH THESE MATERIALS OR ANY INTELLECTUAL PROPERTY IN THEM.

Microsoft may have patents, patent applications, trademarks, or other intellectual property rights covering subject matter within this documentation. Except as provided in a separate agreement from Microsoft, your use of this document does not give you any license to these patents, trademarks or other intellectual property.

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious.

Microsoft, Active Directory, Hyper-V, Windows, Windows PowerShell, Windows Server, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

You have no obligation to give Microsoft any suggestions, comments or other feedback ("Feedback") relating to the documentation. However, if you do provide any Feedback to Microsoft then you provide to Microsoft, without charge, the right to use, share and commercialize your Feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft software or service that includes the Feedback. You will not give Feedback that is subject to a license that requires Microsoft to license its software or documentation to third parties because we include your Feedback in them.

Migration Automation Toolkit

Purpose

The purpose of this document is to:

1. Provide the step-by-step instructions for installing, configuring and using the Migration Automation Toolkit (MAT).
2. Provide additional troubleshooting information for common problems and provide guidance for extending the MAT.

Assumptions

This document was based on these assumptions:

1. All required hardware is configured and already has Microsoft® Windows Server® 2012 Enterprise installed.
2. SQL Server 2008 (or later) Express or other SQL Server Editions is installed.
3. Installation will take place in an existing Active Directory environment
4. Full administrative access is available to the server where the console is being installed upon.

Publication

This document was complete at the time of the version 1.5.0 build of MAT on 1/13/2014. Updated versions of this document will be posted to the Building Clouds site, please verify that you have the latest version before proceeding.

<http://aka.ms/buildingclouds>

Feedback

Please post questions and comments about this guide at <http://aka.ms/buildingclouds> or the TechNet forums where you downloaded this document.

1. General Information

The MAT relies on a number of technologies to work properly. You must have the following installed before the MAT can operate properly:

- A fully licensed VMware environment running one or more of the following:
 - vCenter Server 5.0
 - vCenter Server 4.1
 - ESXi Server 5.0
 - ESXi/ESX Server 4.1
- VMware PowerCLI 5.1
- Microsoft Virtual Machine Converter
- SQL Server 2008 (or later) Express or other SQL Server Editions
- Windows Server 2012 (**This version of the MAT does NOT work with Windows Server 2012 R2**)
- The MAT package

This document will cover the prerequisites and general installation procedure for the MAT. Detailed installation steps for specific applications may be provided in other documents online.

The MAT Package contains eight items:

- **ConvertVM.ps1**
- **ConvertVM-Process.ps1**
- **ConvertVM-Logging.ps1**
- **ConvertVM-Functions.ps1**
- **Variable.XML**
- **Setup MAT.sql**
- **MAT.Readme.1.5.0.txt**
- **MAT FAQ 1.5.0.txt**
- **MAT_1_5_Installation_Guide (this document)**

2. Hardware Requirements

2.1. Conversion Servers

- **Control Server (one required) – controls overall conversion workflow and converts VMs**
 - 4 vCPU
 - 8 GB RAM
 - Application: 250 MB Disk Free Space
 - Conversion Storage: 250GB (Minimum) / 500GB (Preferred) Disk Free Space
- **Helper Server (optional) – remote servers which help with conversions**
 - 2 vCPU
 - 4 GB RAM
 - Application: 250 MB Disk Free Space
 - Conversion Storage: 250GB (Minimum) / 500GB (Preferred) Disk Free Space

*Servers can be either virtual or physical

2.2. Hyper-V

You will need at least one Hyper-V host to accommodate the converted VMs. Right-sizing the Hyper-V environment is something that should be done by MCS or a certified partner in advance of conversion. Generally speaking, you want one Hyper-V host for each ESX server (assuming normal utilization and similar hardware).

Note: Servers are listed here as the minimum requirements. It is possible to increase conversion throughput by having more than one Helper Server installed.

3. Software Requirements

3.1. Control Server

- Windows Server 2012 (earlier versions will also work)
- PowerCLI 5.1
- Microsoft .NET Framework 4.5
- SQL Express 2012 or SQL Server (earlier versions were not tested but should work)
- The Microsoft Virtual Machine Converter (MVMC) v1.0
- The MAT PowerShell scripts
- The VMlist.txt (this is a product of running the MAT and collecting VM data)

3.2. Helper Server

- Windows Server 2012 (earlier versions will also work)
- The Microsoft Virtual Machine Converter (MVMC) v1.0
- Microsoft .NET Framework 4.5
- The MAT PowerShell scripts
- PowerCLI 5.1 (if using MigrateNICs)

3.3. Guest Virtual Machine

- In order to use the NIC features of the MAT, UAC must be disabled in the guest VM.

4. Supported Configurations

Any combination of the following is supported using either the MAT or the MVMC:

4.1.1. VMware sources

- vCenter Server 5.0
- vCenter Server 4.1
- ESXi Server 5.0
- ESXi/ESX Server 4.1

Note: MVMC supports ESXi/ESX 4.0 if the host is managed by vCenter 4.1 or vCenter 5.0. In this case, you must connect to vCenter 4.1 or 5.0 through MVMC to convert virtual machines on the 4.0 hosts.

4.1.2. Destination Host Server

- Hyper-V on Windows Server 2008 R2 SP1 Standard
- Hyper-V on Windows Server 2008 R2 SP1 Enterprise
- Hyper-V on Windows Server 2008 R2 SP1 Datacenter
- Hyper-V on Windows Server 2012
- Microsoft Hyper-V Server 2008 R2
- Microsoft Hyper-V Server 2012

4.1.3. Guest operating systems supported for conversion

- Windows Server 2003 Standard Edition with SP2 x86
- Windows Server 2003 Standard Edition with SP2 x64
- Windows Server 2003 Enterprise Edition with SP2 x86
- Windows Server 2003 Enterprise Edition with SP2 x64
- Windows Server 2003 R2 Enterprise Edition with SP2 x86
- Windows Server 2003 R2 Enterprise Edition with SP2 x64
- Windows Server 2003 R2 Standard Edition with SP2 x86
- Windows Server 2003 R2 Standard Edition with SP2 x64
- Windows Vista Enterprise x64
- Windows Vista Enterprise x32
- Windows 7 Enterprise x86
- Windows 7 Enterprise x64
- Windows 7 Professional x86
- Windows 7 Professional x64
- Windows 7 Ultimate x86
- Windows 7 Ultimate x64
- Windows Server 2008 Enterprise x86
- Windows Server 2008 Enterprise x64
- Windows Server 2008 Datacenter x86
- Windows Server 2008 Datacenter x64
- Windows Server 2008 R2 Standard
- Windows Server 2008 R2 Enterprise x64
- Windows Server 2008 R2 Datacenter x64

4.2. Supported Configurations for Disk Conversion

The following VMware virtual disk types are supported for conversion:

- monolithicSparse
- vmfsSparse
- monolithicFlat
- vmfs
- twoGbMaxExtentSparse
- twoGbMaxExtentFlat
- delta disk conversion
- Stream optimized disks

4.3. Assumptions

MVMC will successfully perform virtual machine conversions when the following conditions are met:

- The virtual machine to be converted is in a running state.
- The virtual machine has VMware tools installed.
- The virtual machine is joined to an Active Directory® domain.
- Remote access through Windows Management Instrumentation (WMI) is enabled on the VMware-based virtual machine to be converted and the destination Hyper-V host. See the **“Error! Reference source not found.”** section in this guide for more details.
- The account used for connecting to the VMware-based virtual machine that needs to be converted is part of an Active Directory domain and also a local administrator on that machine.
- You have the correct credentials to connect to the required environments.
- The Windows user account that you are using has write access to the UNC path specified on the destination Hyper-V host for copying the virtual hard disks.
- The Hyper-V host has the required disk space available for the converted virtual hard disks.

The following assumptions are valid after a successful conversion:

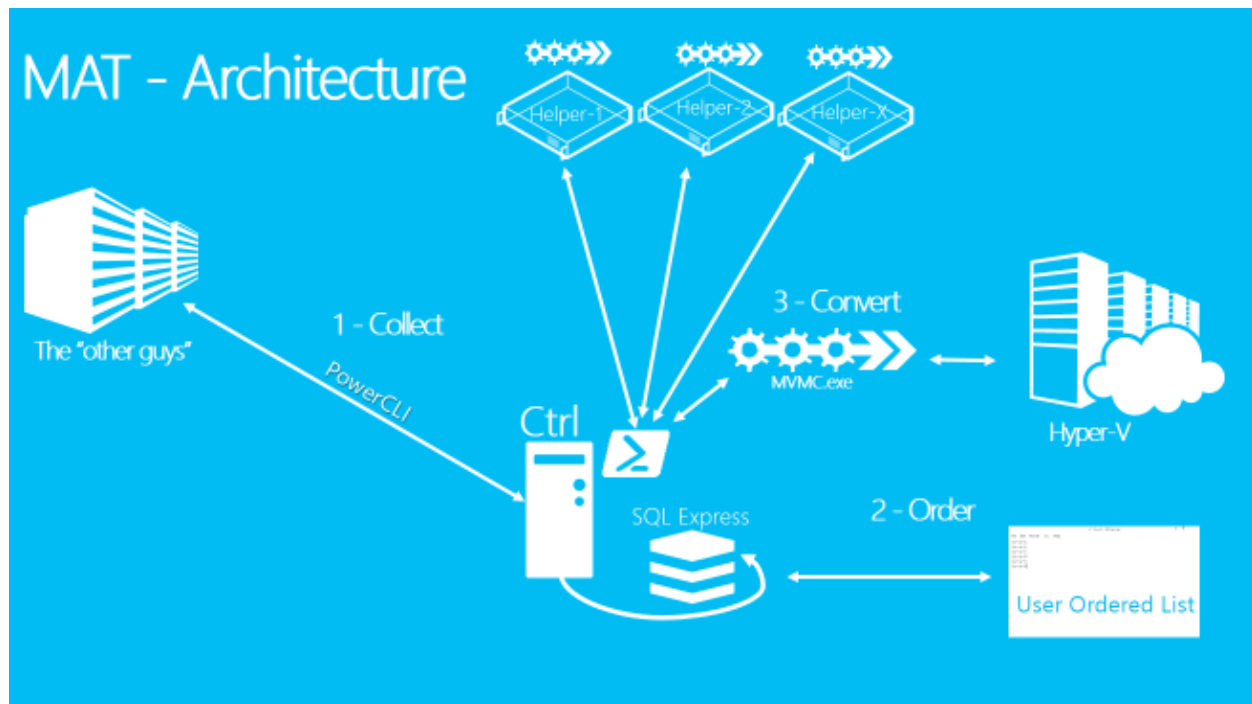
- The destination virtual machine will be in a started or stopped state depending on the settings chosen by the user.
- Once the virtual disks attached to the virtual machine are copied successfully to the converter machine, the source virtual machine will be restored to a started or stopped state, depending on the settings chosen by the user.
- Product activation requires each instance of a Windows operating system installation to be activated with Microsoft. Because conversion creates a second instance of the virtual machine on Hyper-V, this instance needs to be activated.

5. Upgrade Information

Upgrades from earlier versions of the MAT are not supported.

6. Architecture Diagram

The following shows the basic architecture and components for the MAT:



Installing Individual Software Components

7. Install SQL 2008 Express or later

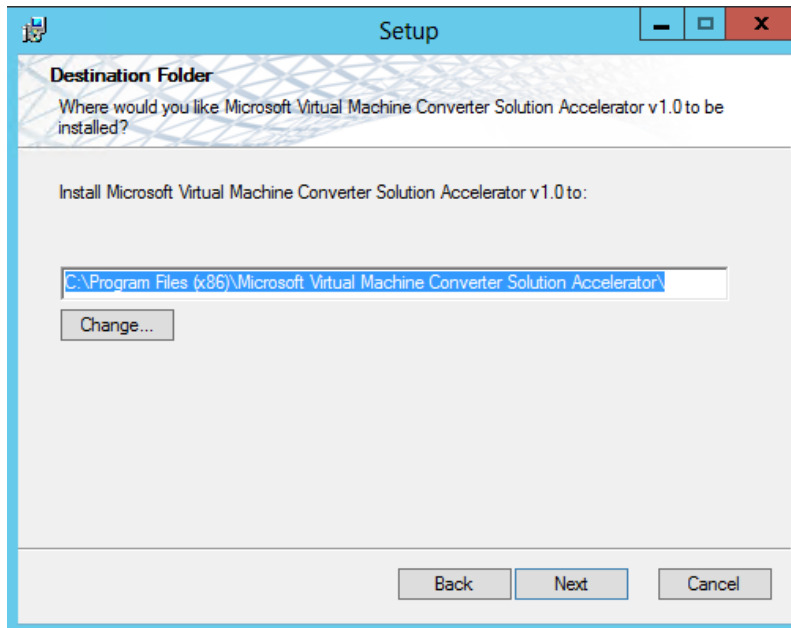
Install SQL 2008 or R2 on *ServerA* following the standard installation procedure

[SQL Install Instructions](#) You can install either SQL 2008 or R2 or 2012.

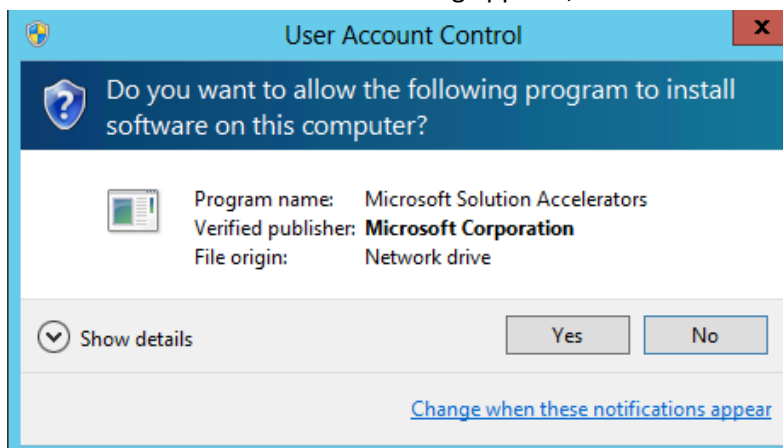
For more information, please see: <http://technet.microsoft.com/en-us/library/hh420342.aspx>

8. Install Microsoft Virtual Machine Converter v1.0

1. Download the Microsoft Virtual Machine Converter Solution v1.0 Accelerator MSI to your local machine. You can [download the MVMC here](#).
2. Launch the “Microsoft Virtual Machine Converter Solution Accelerator.msi” installer package.
3. If you accept the terms of the License Agreement, check the box and click **Next**.
4. Assign a destination folder. The default of *C:\Program Files (x86)\Microsoft Virtual Machine Converter Solution Accelerator* is valid. Click **Next**.



5. Click **Install** to begin the installation.
6. When the User Account Control dialog appears, select **Yes**.



7. Click **Finish** to complete the installation

9. Install vSphere PowerCLI 5.1

Install **the vSphere PowerCLI 5.1** on the Control Server **following** the standard installation procedure.

[PowerCLI Install Instructions](#)

10. Install the Migration Automation Toolkit

10.1. Create a MAT folder

1. Create a folder named **C:\MAT**

NOTE: You can choose any name or folder destination *provided it does not contain any spaces*.

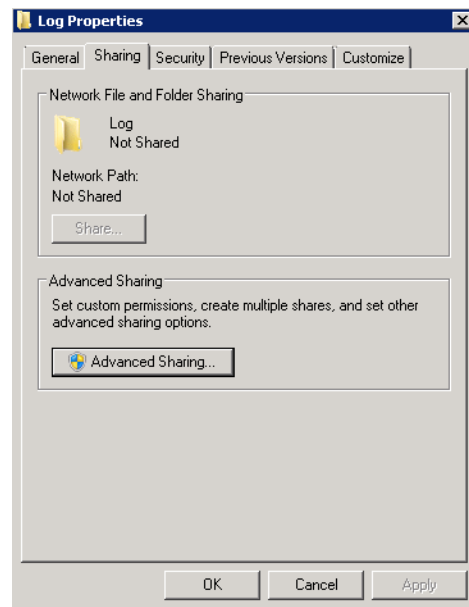
10.2. Unpack the toolkit

The installer should contain eight files

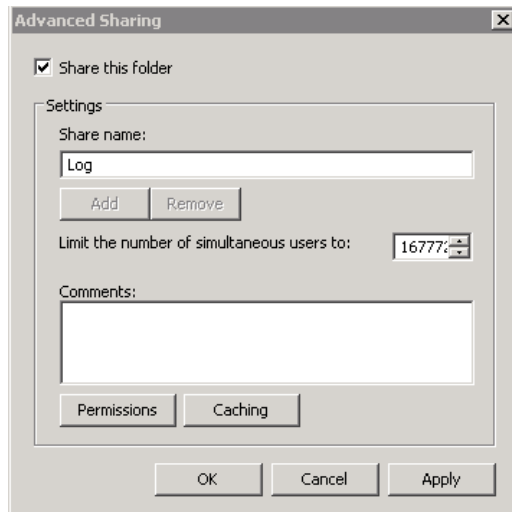
- ConvertVM.ps1
- ConvertVM-Process.ps1
- ConvertVM-Logging.ps1
- ConvertVM-Functions.ps1
- Variable.XML
- MAT.Readme.1.5.txt
- MAT FAQ 1.5.txt
- MAT_1_5_Installation_Guide (this document)

10.3. Create a share for the Target Host

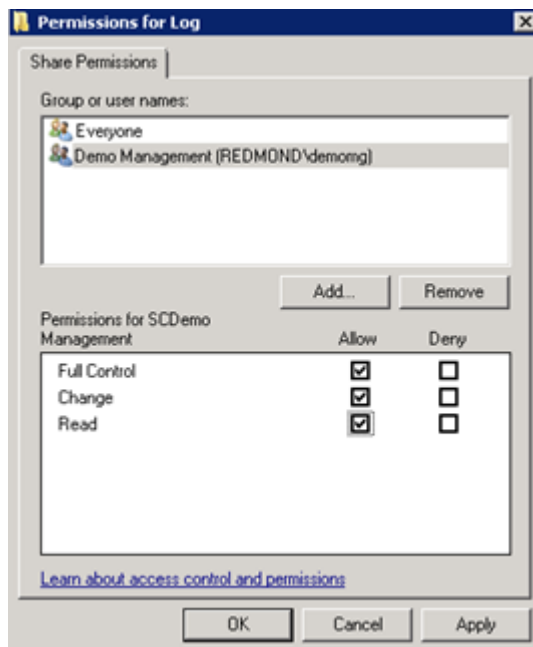
1. On the Hyper-V host you have selected to accept the converted VMs, select or create a folder to house the vhd files that are created by MVMC and copied to the target host.
2. Right-click the target folder then select **Share with** and **Advanced Sharing...**



3. Select **Advanced Sharing...**



4. Check the box to **Share this folder** (this document assumed you used the default Share name **Share**)
5. Click **Permissions**



6. Add the account you use for conversion and give it **Change** rights

10.4. Run the *Setup MAT.sql* script

1. Launch SQL Server Management Studio



2. Set **Server Name** to **localhost** and click **Connect**
3. From the **File** menu, select **Open** and then **File** (or press Ctrl-O)
4. Navigate to **C:\MAT**
5. Select **Setup MAT.sql** and click **Open**.
6. Verify that the current context is **master**, then Execute the script by clicking the Execute button.



7. The script should produce the following output:

```
Creating the MAT Database v1.5.0
MAT Database created.

(1 row(s) affected) Version Table created.
VMQueue Table created.
GlobalSettings Table created.
Populating GlobalSettings Table...

(1 row(s) affected)
VMConversionData_PS
Stored Procedure sp_TransferVMConversionData_PS created.
MAT Database Setup Complete.
```

NOTE: This script only needs to be run once. The script **is destructive** and will drop any tables that exist and recreate them, thereby removing **all** conversion related data. This will reset your environment to a clean install state.

11. Configure MAT Settings

1. Open the **Variable.xml** file and supply the correct parameters for all the required XML values. You should supply values in the following sections:

NOTE: Take great care when editing the XML file that you do not alter the XML formatting. Ideally use an XML editor.

11.1. General

2. Set the value **Logpath** to the temp path for the user which will be running the conversions. Replace the value (username) with the appropriate value. For example:
C:\Users\administrator\AppData\Local\Temp
3. Set the value **Datasource** to the name of the SQL Server you are using or the name of the default instance in you are using names instances.
4. Set the MigrateNICs value to either 1 or 0. 1 = Migrate NICs to Hyper-V, 0 = do not migrate nics. If you enable this option, you must set sPower = 0.
5. If you wish to alter the behavior of the actions Show-Status or Show-StatusLoop you can change the value of **Showrows**. This value limits how many rows are returned by these two actions.

```
<General>

  <Variable Name="XMLVersion" Value="1.5.0" />

  <Variable Name="Logpath" Value="C:\Users\username\AppData\Local\Temp\" />

  <Variable Name="Datasource" Value="SERVER1"/>

  <Variable Name="MigrateNICs" Value="1"/>

  <Variable Name="Showrows " Value="30"/>

</General>
```

11.2. SchedCred

(If you are not using Helper Servers you can ignore this section)

6. Set the value **WinAccount** to an account which has the rights to schedule and execute tasks as an admin on the remote system.
7. Set the value **WinPassword** to the password of the account used in WinAccount.

```
<SchedCreds>

  <Variable Name="WinAccount" Value="contoso\administrator" />
```

```
<Variable Name="WinPassword" Value="password" />

</SchedCreds>
```

11.3. HyperV

8. Set the value **tpath** to the share you created in section 10.3
9. Set the value **thost** to name of the Hyper-V server which will accept and host (at least temporarily) the converted VMs.
10. Set the value **sPower** to either 1 or 0. The value controls the final power state of the Source VM. 1 = on 0 = off
11. Set the value **tPower** to either 1 or 0. The value controls the final power state of the Target VM. 1 = on 0 = off
12. Set the value **DyMac** to either 1 or 0. The value controls the whether or not Hyper-V dynamically assigns a MAC address or if it uses the one assigned by VMware. 1 = Use a dynamic address, 0 = use the VMware MAC address (This setting only applies if MigrateNICs = 1)
13. Set the value **dynamicdisk** to either 1 or 0. The value whether the Hyper-V VM will use dynamic or fixed disks. 1 = dynamic 0 = fixed

```
<HyperV>

<Variable Name="tpath" Value="\\SERVER1\Share" />

<Variable Name="thost" Value="SERVER1" />

<Variable Name="sPower" Value="1" />

<Variable Name="tPower" Value="0" />

<Variable Name="DynMac" Value="0" />

<Variable Name="dynamicdisk" Value="1" />

</HyperV>
```

11.4. VMware

14. Set the value **sHost** to the FQDN or IP address of your VMware source host (ESX or vSphere)
15. Set the value **shusername** to an account which has admin rights to the source host
16. Set the value **shpwd** to the password of the shusername account
17. Set the value **gUser** to an account which has admin rights to the Guest VM. This account is used to remove VMware Tools. This can be a domain or local account

18. Set the value **gPwd** to the password of the gUser account

```
<VMware>

<Variable Name="shost" Value="10.10.10.1" />

<Variable Name="shusername" Value="contoso\administrator" />

<Variable Name="shpwd" Value="password" />

<Variable Name="gUser" Value="contoso\administrator" />

<Variable Name="gPwd" Value="password" />

</VMware>
```

11.5. RemoteHost

19. If you plan to use Helper Servers, add a line for each server using the following format. The following example shows three Helper Servers named Server1, Server2 and Server3.

```
<RemoteHost>

<ConvertServer>SERVER1</ConvertServer>

<ConvertServer>SERVER2</ConvertServer>

<ConvertServer>SERVER3</ConvertServer>

</RemoteHost>
```

20. Review the value for **Queuelength** which is stored near the top of *ConvertVM.ps1* in the Variables section. The default is set to 2, which limits the total number of concurrent conversions per server to 2. If you wish, you can raise this value to 3 or lower it to 1. Raising the value above 3 is not supported as the MAT only allows for three logs to be created simultaneously.

```
##### Variables #####

$Queuelength = 3 ### Do not set Queuelength above 3 ###

$CurrentPath = "C:\MAT"

$XMLPath = "C:\MAT\Variable.xml"

$VMList = "$CurrentPath\VMList.txt"

$VerboseLogging = 3

$MaxCapacityLoopTimer = 60

$Lookback = 1
```

NOTE: If you altered the default path from C:\MAT during installation, you need to update the values for \$CurrentPath and \$XMLPath in the Variables section of *ConvertVM.ps1*.

12. Install MAT on Helper Servers (Optional)

If you intend to run conversions on more than one server, each Helper server needs to be configured to participate in the MAT conversions

1. Install the MVMC package on your Helper(s) following the same steps as in Section 8.
2. Verify your credentials in the `<SchedCreds>` section in Variable.xml
3. Add one `<ConvertServer>HELPERx</ConvertServer>` to the `<RemoteHost>` section of the Variable.xml file for each Helper you wish to use.
4. Copy the MAT files to your Helper(s) in C:\MAT (assuming you haven't edited that variable in ConvertVM.ps1)
5. Verify your SQL server is configured to accept remote connections
6. Log into each Helper using the account assigned in `<SchedCreds>`.

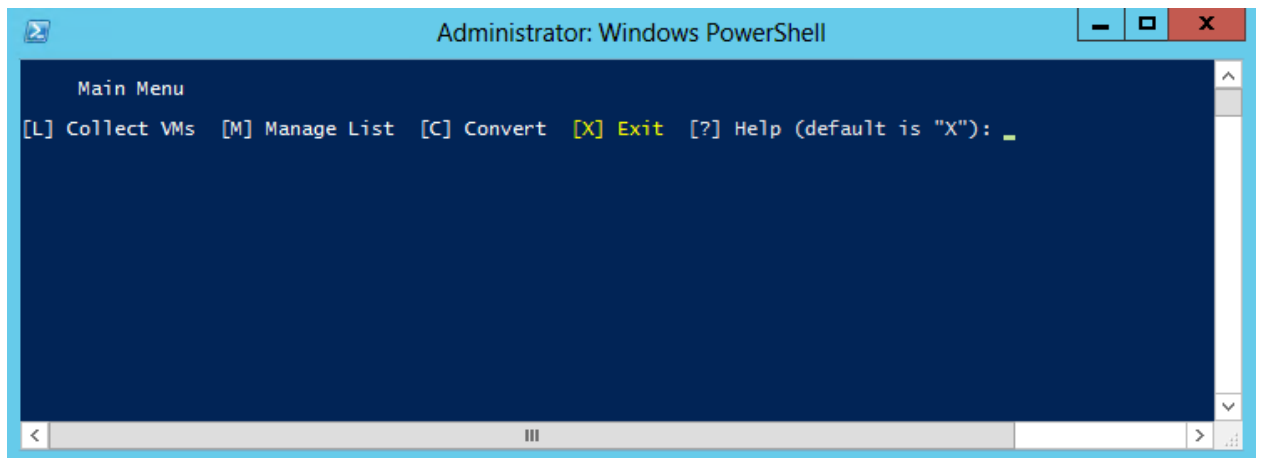
NOTE: You must be logged into each Helper Server in order for the remote to execute properly. The easiest way to do this is to connect to the remote via RDP and then disconnect your session. Any remote session, even a disconnected one will allow the MAT to run properly. The MVMC.exe will not run via Task Manager if the user profile is not loaded, since it requires access to the user's temp folder.

7. If you plan to use the RemoveCD, RemoveFloppy or MigrateNICs function, you will need to install PowerCLI on Helpers.

13. Collect VM information

NOTE: All MAT functions are exposed as parameters and can be viewed using tab-complete within PowerShell

1. From a PowerShell prompt run **.\ConvertVM.ps1 Collect**
- OR
2. To start the MAT run **.\ConvertVM.ps1** in PowerShell. *Run this as an administrator.*
3. On first use, run a collection cycle using the **[L] Collect VMs** option.



In large environments collection may takes minutes or even hours. All VMs collected will be displayed briefly as they are written to the database.

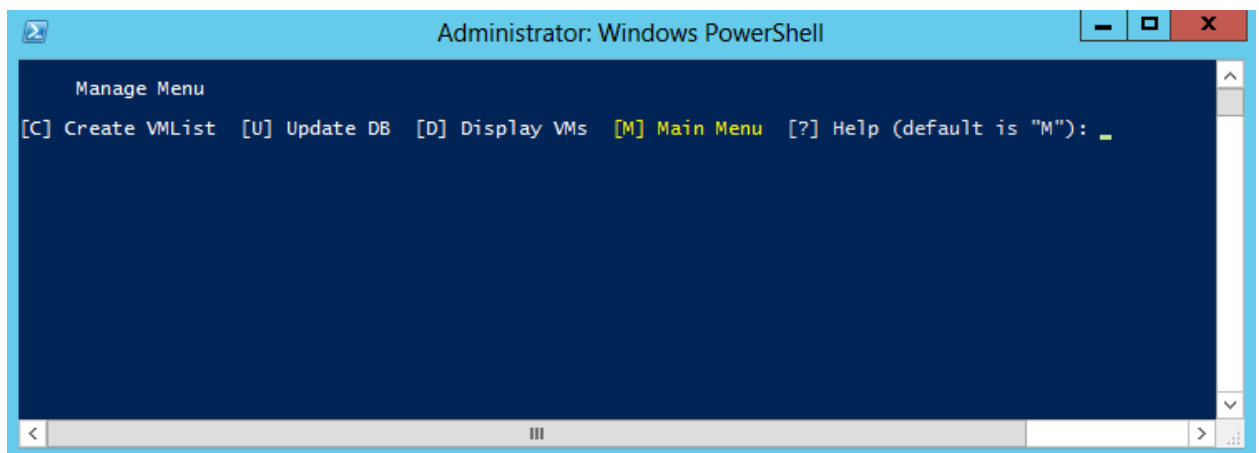
14. List Management

1. From a PowerShell prompt run **.\ConvertVM.ps1 Create-List**
2. Edit the list as described below
3. Run **.\ConvertVM.ps1 Update**

OR

4. After you have completed collection, select **[M]anage List**
5. Select **[C]reate VMList** and the script will create a *VMList.txt* file in your MAT folder.

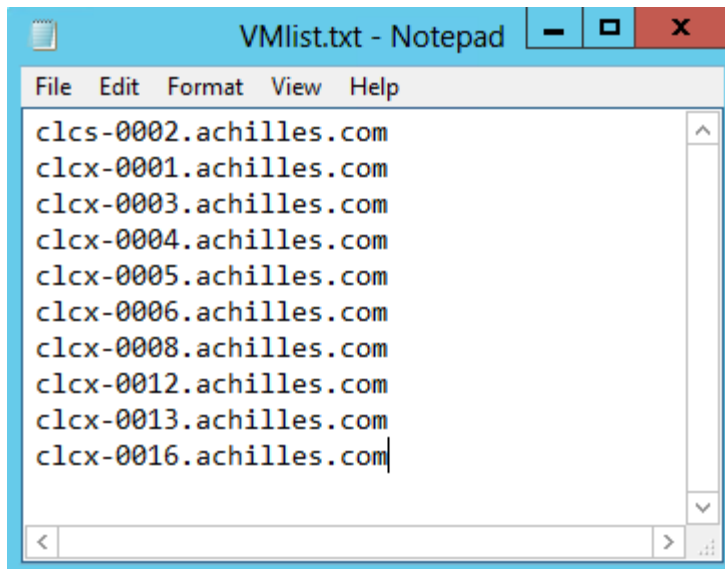
All available VMs which are valid for conversion will be added to this file. Notepad will automatically open this file.



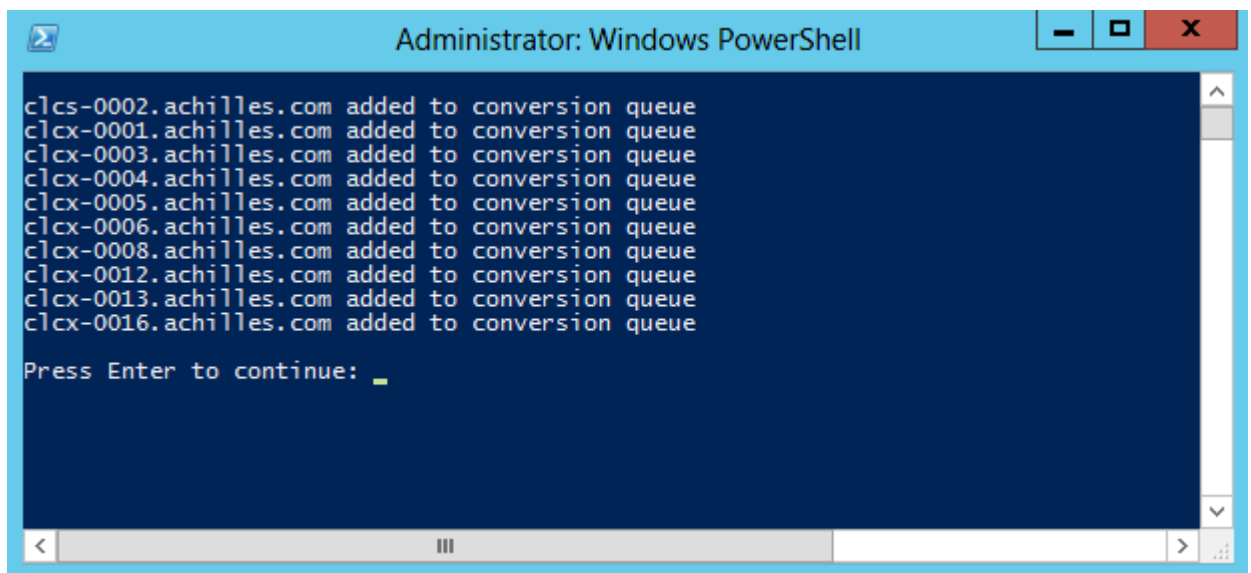
6. Edit the list so that it only contains the names of VMs you want to convert in one batch. A batch is a series of conversions that will run continuously until all the VMs have been converted (or failed). It is best to limit this to a reasonable value of no more than 30 VMs. If you are just getting started, begin with a small list. You can re-create the VMlist as often as you wish.

As you successfully convert VMs they will no longer appear in the VMList, so there is not risk of converting a VM that is already done. If you want, you can reorder this list and this will affect the order in which VMs are converted. The first VM listed will be the first VM converted.

NOTE: If you use Helper Servers, they will take VMs from the list in the order they appear and Helper Servers access the list before the local machine (Control Server) does.



7. Save the list.
8. Select the **[U]pdate DB** option. This will update the records in the database to reflect the current list.



9. If you wish, you can use the **[D]isplay VMs** option to confirm the changes were updated.

15. Conversion

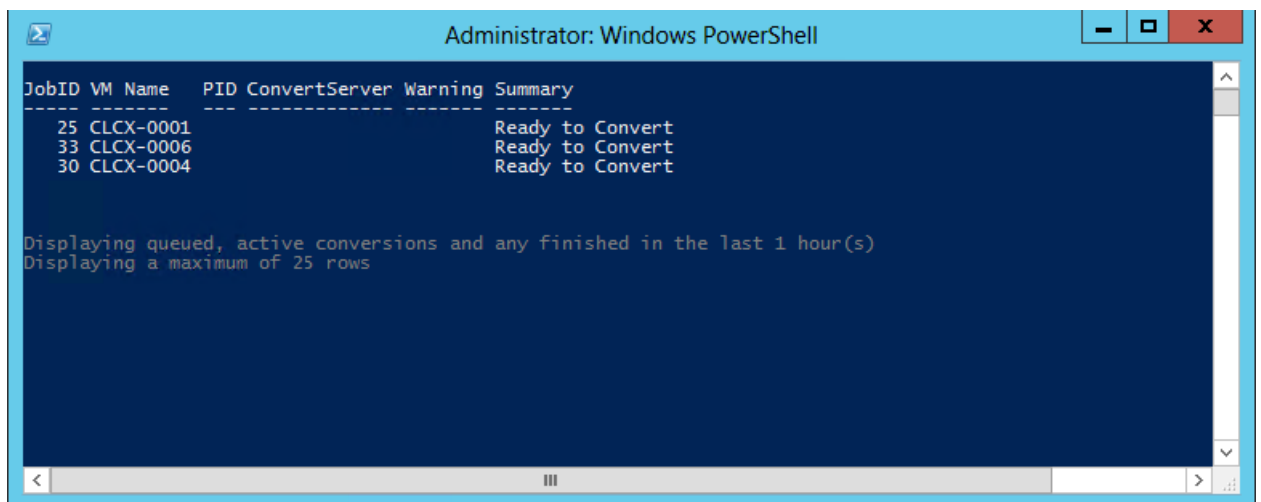
Once the VMList has been prepared and the database has been updated, you can begin your conversion. The VMs listed will be the only ones converted by this 'batch'.

1. From a PowerShell prompt run **.\ConvertVM.ps1 Convert**

OR

2. To start the MAT run ConvertVM.ps1 in PowerShell. *Run this as an administrator.*
3. At the **[M]ain menu** select **[C]onvert**.

As the conversion begins, all the VMs that will be converted in this batch will be listed. If Helper Servers are used, they will start after the list of VMs is displayed. Once all Helper Servers have been serviced, the local conversion will start on the Control Server.



```

Administrator: Windows PowerShell

JobID VM Name PID ConvertServer Warning Summary
-----
25 CLCX-0001 Ready to Convert
33 CLCX-0006 Ready to Convert
30 CLCX-0004 Ready to Convert

Displaying queued, active conversions and any finished in the last 1 hour(s)
Displaying a maximum of 25 rows
  
```

16. Monitoring Conversions

Conversions happen without interaction on both the Control Server and any Helper Servers. This can make it difficult to assess the state of current conversions. For this reason, two monitoring functions are included with the MAT. One function will to give an immediate status and exit; the other will run continuously until the user stops it by issuing *Control-C*

The following Actions can be run at the MAT command line by passing the Action as the first parameter.

Syntax: PS> **.\ConvertVM.ps1 <action> (optional)**

1. **Show-Status** - Displays the Status table once and exits
2. **Show-StatusLoop** - Displays the Status table in an endless loop refreshing every 5 seconds, and then supplying the instructions to exit after another 5 seconds. The total refresh duration for data is 10 seconds.

```

Administrator: Windows PowerShell

JobID VM Name      PID ConvertServer Warning Summary
-----
25 CLCX-0001 23545 Server1      True Stage 4 - Converting Disks
33 CLCX-0006 12385 Server1      Stage 6 - Creating virtual machine on the Hyper-V host
30 CLCX-0004                      Ready to Convert

Displaying queued, active conversions and any finished in the last 1 hour(s)
Displaying a maximum of 25 rows

Press 'Ctrl + C' to terminate script

```

The total number of records displayed in the view will be 30 by default. If you wish to change this value, alter the **\$Showrows** value in Variable.xml.

Items are collected from the SQL table using a TOP function, ordered by Position (which was set when you updated the database from the VMList.txt file.

By default the Show-Status Actions will show data for any active conversion and any conversion that completed within the last hour. If you want to alter this behavior you can change the value of **\$Lookback** in ConvertVM.ps1.

If a VM shows Warning = True, then some part of the conversion did not work properly. See the log for more details.

For more information, see the **PowerShell Parameter and Variable Reference** section.

17. The PrepareVM function

A new function in version 1.5, the PrepareVM function prepares the VM for conversion and provides the following services (all of which are optional)

- Removing or unmounting all CD\DVDs from the VM
- Removing all floppy drives from the VM
- Capturing and recreating the network interface settings for the VM

17.1. Removing CD\DVD

By default the MAT will unmounts any optical media from the VM before conversion.

If you wish to disable this function, edit line 38 of ConvertVM.ps1 to match the following:

```
$RemoveCD = 0
```

17.2. Removing Floppy disks

By default the MAT will remove any floppy disks from the VM before conversion.

If you wish to disable this function, edit line 39 of ConvertVM.ps1 to match the following:

```
$RemoveFloppy = 0
```

17.3. Network Interface Card information capture and restore

By default, the MAT will store and recreate the information from *up to three (3) network cards* on a VM. The information captured and restored is:

- Static IP addresses
- DNS information
- MAC addresses
- VLAN Tagging

If you wish to disable this function, edit line 9 of Variable.XML to match the following:

```
<Variable Name="MigrateNICs" Value="0"/>
```

NOTE: In order to use the NIC features of the MAT, User Account Control (UAC) must be disabled in the guest VM.

17.3.1. How the MigrateNICs function works

The MAT uses PowerCLI to send scripts to the target VM which enable the VM to collect it's own networking information and store this until after the migration.

Because of limits in the size of script blocks that can be sent to the VM, this is done in five steps. These steps create the following files which are run during the final pre-conversion shutdown and at the first post-conversion reboot.

Step 1

- Creates C:\Windows\Temp\nicCreds.txt (deleted in Step 2)

Step 2

- Sets HKLM:\Software\Microsoft\Windows\CurrentVersion\RunOnce\ConfigureServer = C:\Windows\Temp\nicRestore.cmd
- Sets HKLM:\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\DefaultUserName = (defined username)
- Sets HKLM:\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\DefaultPassword = (defined password)
- Sets HKLM:\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\AutoAdminLogon = 1
- Sets HKLM:\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\DefaultDomainName = (defined domain)
- Delete C:\Windows\Temp\nicCreds.txt
- Creates C:\Windows\Temp\nicRestore.cmd (encoded)
- Creates C:\Windows\Temp\nicConfig.xml
- Creates C:\Windows\Temp\nicDNS.xml
- Creates C:\Windows\Temp\nicConfig.txt

Step 3

- Backs up existing shutdown scripts (if they exist)
- Creates C:\Windows\System32\GroupPolicy\Machine\Scripts\Shutdown\MATcleanup.ps1, which sets nics to DHCP and appends _VMware to the name of each and restores any shutdown scripts that were backed up. This file contains the following code:

```
# You should not see this file. It is used during virtual machine migration.
# Consult the MAT Installation and Usage guide for more info.

Remove-Item 'HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Shutdown' -Recurse
Remove-Item 'HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown' -Recurse
Reg Import C:\Windows\Temp\gposcriptsShutdown.reg
Reg Import C:\Windows\Temp\gpostateShutdown.reg
Remove-Item C:\Windows\Temp\gposcriptsShutdown.reg
Remove-Item C:\Windows\Temp\gpostateShutdown.reg
Remove-Item C:\Windows\System32\GroupPolicy\Machine\Scripts\Shutdown\MATcleanup.ps1

gwmi win32_NetworkAdapterConfiguration | ? { !$_.DHCPEnabled -and $_.IPAddress } | %{ $_.EnabledDHCP() }
$NetworkConnections = (New-Object -com shell.application).Namespace(0x31)
Foreach ($NIC in (gwmi win32_NetworkAdapter | where { $_.NetConnectionID }) | Select-Object -
ExpandProperty NetConnectionID)
{ $NetworkConnections.Items() | Where-Object { $_.Name -eq $NIC } | ForEach-Object { $_.Name="$($_.Name)_VMware" }
```

```
}
```

Step 4

- Creates and populates HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Shutdown\0
- Creates and populates HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Shutdown\0\0

Step 5

- Creates and populates HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0
- Creates and populates HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0\0

NOTE: Only the files created in C:\Windows\Temp\ should persist after reboot. These files contain your network information and are not deleted in the event that you need this information. If you need to restore this information because of a problem with the MAT (on either the Source or Target VM) run C:\Windows\Temp\nicRestore.cmd . Otherwise these files can be deleted safely.

If you find that C:\Windows\System32\GroupPolicy\Machine\Scripts\Shutdown\MATcleanup.ps1 exists after conversion, you can run this file manually to reset any GPO-based shutdown scripts and remove any other artifacts of the conversion.

18. Reporting

The MAT provides reporting in the form of Comma Separated Values (CSV) files. These reports are designed to be used by someone with no SQL experience. Additional reports may be provided in the future or end users can generate their own based on the SQL schema.

The following Actions can be run at the MAT command line by passing the Action as the first parameter.

NOTE: All Reports will be saved in C:\MAT\Reports (unless you have modified the location of MAT)

Syntax: PS> **.\ConvertVM.ps1 <action> (optional)**

1. **Report-All** - Generates a CSV file with all records from the database. The file will be named *Alldata.csv*
2. **Report-Complete** - Generates a CSV file with all completed VMs from the database. The file will be named *Completed.csv*
3. **Report-Incomplete** - Generates a CSV file with all incomplete Vms from the database. The file will be named *Incomplete.csv*
4. **Report-Unsupported** - Generates a CSV file with all unsupported VMs from the database. The file will be named *Notsupported.csv*

Report output will look like this:

```
"Summary","ID","ConvertServer","Supported","Notes","ReadytoConvert","Completed","Status",
"VMName","StartTime","PID","EndTime"
"Successful Conversion","85","","1","VM automatically collected by
script","False","True","1","clcs-0002.achilles.com","4/9/2013 9:20:21 AM","","4/9/2013
9:43:06 AM"
"Ready to Convert","86","SCDEMO099","1","VM automatically collected by
script","True","False","0","clcx-0001.achilles.com","4/9/2013 8:56:07 AM","",""
"Successful Conversion","87","","1","VM automatically collected by
script","False","True","1","clcx-0003.achilles.com","4/9/2013 9:04:24 AM","","4/9/2013
10:03:12 AM"
"Successful Conversion","88","","1","VM automatically collected by
script","False","True","1","clcx-0004.achilles.com","4/9/2013 9:04:34 AM","","4/9/2013
10:03:11 AM"
```

Each report type will follow the same field structure (found on line 1) which is:

```
"Summary","ID","ConvertServer","Supported","Notes","ReadytoConvert","Completed","Status",
"VMName","StartTime","PID","EndTime"
```


19. Database Maintenance

The MAT provides some tools to manage and maintain the MAT database. These tools are designed to be used by someone with no SQL experience. These tasks can also be performed with SQL but direct manipulation of the database can lead to corrupted data and is not recommended.

The following Actions can be run at the MAT command line by passing the Action as the first parameter.

Syntax: PS> **.\ConvertVM.ps1 <action> (optional)**

1. **Reset-List** – This will reset the VM record for any VM listed in C:\MAT\VMlist.txt. Reset records will have the following fields updated: ReadytoConvert = 0, ConvertServer = NULL, InUse = NULL, Position = NULL, Starttime = NULL, EndTime = NULL, Status = 0, Summary = NULL, Notes = 'Record reset by Manage.ps1'
2. **Reset-Incomplete** - Resets every incomplete VM record in the database. Reset records will have the following fields updated: ReadytoConvert = 0, ConvertServer = NULL, InUse = NULL, Position = NULL, Starttime = NULL, EndTime = NULL, Status = 0, Summary = NULL, Notes = 'Record reset by Manage.ps1'
3. **Reset-All** - Resets **EVERY VM record** in the database. Reset records will have the following fields updated: ReadytoConvert = 0, ConvertServer = NULL, InUse = NULL, Position = NULL, Starttime = NULL, EndTime = NULL, Status = 0, Summary = NULL, Notes = 'Record reset by Manage.ps1'
4. **Purge** – This option will **delete ALL records** from database. Use caution when issuing this command as it is irrevocable. You will be prompted to confirm deletion before continuing.

20. Duplicate Conversions

The MAT provides checks to keep from duplicating a conversion, which is to say converting a VM that has already been converted.

However if you encounter a situation where the same VM is converted twice, the situation is not destructive. The second VM will be created with an incremental value appended to the end of the name. You will see this on the file system and in Hyper-V Manager. Simply delete one of the VMs, presumably the newer conversion, and continue with your migration.

21. Interrupting a Conversion

In some situations, you may need to interrupt a conversion. Either because something has gone wrong, or the conversion is taking an unexpectedly long time and may exceed your service window.

To interrupt a running conversion take the following steps.

- Run **.\ConvertVM.ps1 Show-Status** – Review the output to see which conversion(s) you wish to stop. Locate the PID value and the ConvertServer value for that record.

```

Administrator: Windows PowerShell

JobID VM Name      PID ConvertServer Warning Summary
-----
25 CLCX-0001 23545 Server1      True Stage 4 - Converting Disks
33 CLCX-0006 12385 Server1      True Stage 6 - Creating virtual machine on the Hyper-V host
30 CLCX-0004                Ready to Convert

Displaying queued, active conversions and any finished in the last 1 hour(s)
Displaying a maximum of 25 rows

Press 'Ctrl + C' to terminate script
  
```

- Logon to the appropriate machine listed in the ConvertServer field.
 - Kill the PID (Process ID) for the conversion you wish to terminate. From a command prompt type **"taskkill /PID 000"** where **000** is the value from the PID field in the desired record.
- NOTE:** Interrupting a conversion using this method will leave the VM record in an invalid state.
- Restart the VM source as required.
 - Update the VMList.txt so that it only contains the names of the VMs you have interrupted.
 - Run **.\ConvertVM.ps1 Reset-List** which will reset the VM records for all the VMs you listed in VMList.txt.
 - Locate any temporary files that were created by the MVMC process and delete them. They are found in C:\MAT\MVMC and may be in a subdirectory under the user's temp folder.
 - Delete the log files associated with the VM conversion.

22. Advanced Concepts

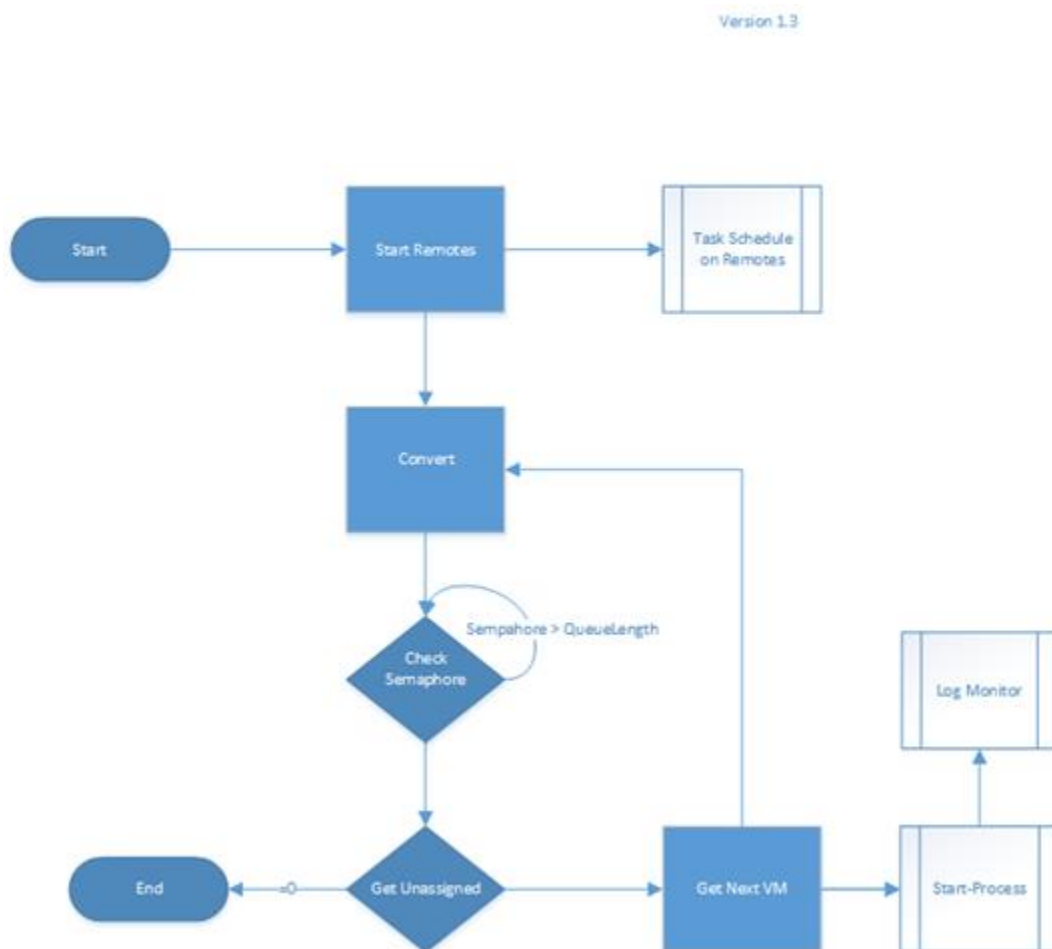
22.1. Overall PowerShell workflow

When you start a set of conversions from the `.\ConvertVM.ps1` menu (and then select Convert) or if you launch the script with the **Convert-Global** action the process will read the `Variable.xml` file and load all the `<ConvertServer>` values you have added. The processes run by the Control Server and the Helper Servers are listed below:

22.1.1. Control Server

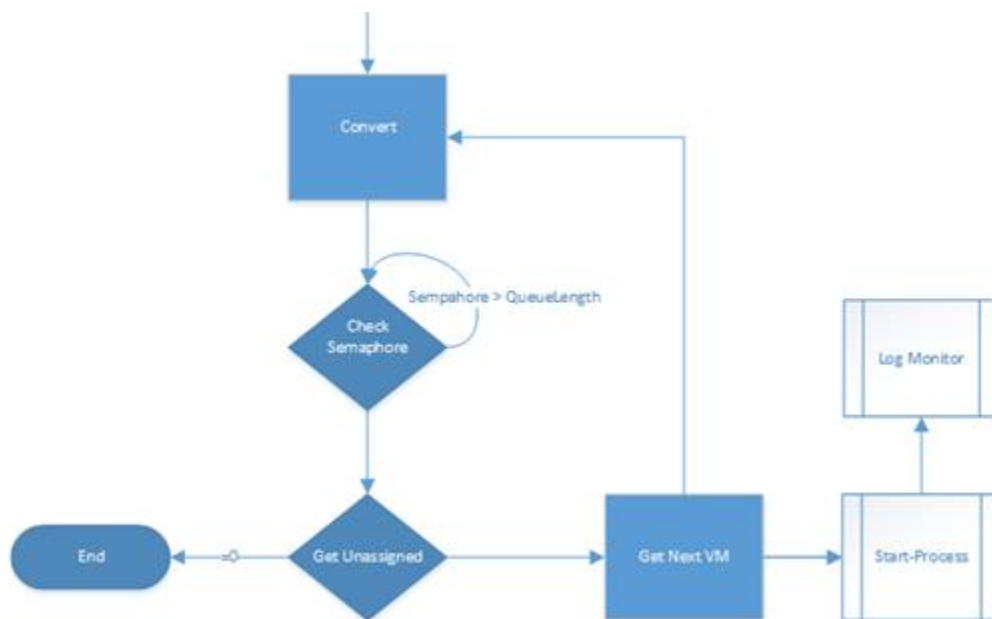
1. On the Control Server, the process starts by validating the format of the `Variable.xml` file. Then it will check to see if the `Variable.xml` contains entries for Helper Servers (i.e. if you added them). If Helpers are found, it will create a list of the servers and iterate through them in the order they appear in the xml file. The process will create a new job on the Helper server(s) using the task scheduler and immediately start that job. Once all the remotes have been serviced, the local Convert function will be triggered.
2. The Convert function will first call `CheckSemaphore`. `CheckSemaphore` will check the server's semaphore counter. It does this by calling SQL to see how many conversions are listed for that machine (fieldname = `QueueLength`). As long as the value of the semaphore is greater than the `QueueLength` the process continues. If it is not, the server is considered to have reached capacity and no more conversions will be launched until at least one of the existing conversions completes (successfully or otherwise). The process will loop for the number of seconds specified by the `$MaxCapacityLoopTimer` variable, doing so until one of the current conversions finishes and thereby freeing up a 'slot'.
3. Next the process checks for unassigned VMs (`ConvertServer = NULL`) from the database, which are marked for conversion (fieldname = `ReadyToConvert`). If the query returns one or more, it assigns one VM (using `GetNextVM`) to itself and thereby increments the `QueueLength` by one. VMs are taken using a TOP 1 SQL call but they are ordered by 'Priority' which is established when (or if) you re-order the `VMlist.txt`. If no more VMs are found in the table, the batch is complete and the script will exit.
4. The next stage is Convert. If you have set the values to remove CD, floppies and to MigrateNICs, these steps will happen before the conversion is started. The script will call the `ConvertVM-Process.ps1` using the `Start-Process` cmdlet. As each server starts to process a VM for conversion it updates the record's `ConvertServer` field with its localhost name, so that no other server will attempt to convert the same machine. The time between the two steps is usually less than one second and delays exist in other locations to minimize the possibility of a record collision. It may seem odd that the script will start another process but this is done for several reasons. First is to let one `ConvertVM-Process.ps1` run for every `MVMC.exe` that runs. Ideally, this would have been done with PowerShell v3 workflow but some of the vagaries of the `MVMC.exe` made this approach more practical. This approach also allows you to extend the `ConvertVM-Process.ps1` without messing with the overall flow.

5. Once ConvertVM-Process.ps1 is called, the script starts back over at the Convert function (admittedly somewhat confusing naming) and the process continues anew.
6. While the MAT workflow has looped and moved one, the ConvertVM-Process.ps1 will call ConvertVM-Logging.ps1 (assuming `$LogOption = 1`). One ConvertVM-Logging.ps1 is called for every MVMC.exe (both tracking the conversion by its Process ID) and the job of ConvertVM-Logging.ps1 is to watch the contents of mvmc.log (or mvmc1.log, or mvmc2.log...) and 'translate' the messages there into more human readable information. It will then write that information to the main log (and the database in case you are watching using the Show-Status functions). Ideally the process would capture the stdout messages from the MVMC.exe, but it doesn't actually make any, so we must rely on ConvertVM-Logging.ps1 to parse the log.
7. When ConvertVM-Logging.ps1 sees that the PID for its particular MVMC.exe exits, it will spin down and exit too. Shortly thereafter ConvertVM-Process.ps1 will exit too. ConvertVM-Process.ps1 sees that the monitored PID exits and then it 'waits' for ConvertVM-Logging.ps1 before closing. It does this by waiting X seconds, where X is the value of `$LogMonitorDelay` multiplied by the value of the `$SleepMultiplier` (both set in ConvertVM-Process.ps1). The default delay is 39 seconds ($30 \times 1.5.0$).



22.1.2. Helper Server

The helper servers are started via a Task Scheduler Job using the command “.\ConvertVM.ps1 Convert”. The Convert parameter bypasses the remote logic and jumps directly into the Convert function where it checks the \$QueueLength and starts the process identically to the Control Server at step 2.



22.2. MAT Extensibility

MAT was designed to be fully extensible for users. If you want to extend the functionality of the MAT for any post-conversion functions, the right place to do this is at the end of the ConvertVM-Process.ps1 script. Because ConvertVM-Process.ps1 runs in parallel with the conversion as a whole, any commands would be run specifically against the target conversion and would not delay the overall process.

NOTE: Extensions could also be added between the GetNextVM and Convert functions but this isn't as desirable as it will slow your conversion flow down. Adding it to ConvertVM-Process.ps1 leaves the overall workflow unchanged and is least likely to be disrupted by future versions of the MAT.

Below are the final lines of the ConvertVM-Process.ps1 script, ideal placement of additional tasks is listed.

```
##### Start #####
write-log 3 "-----Starting Conversion"
$time = get-date
SQLWrite "UPDATE VMQueue SET [Summary] = 'Starting Conversion',
[StartTime] = '$time' WHERE JobID = $ID"
DoConvert
Your code goes here
```

22.2.1. Extensions which hold the queue until finished

If you want to hold execution of additional VMs while your post-processing is running, you should **move** the following two lines from the end of the DoConvert function and place them at the end of your additions. This will keep the \$QueueLength from decrementing before your tasks complete.

```
##### Start #####
write-log 3 "-----Starting Conversion"
$time = get-date
SQLWrite "UPDATE VMQueue SET [Summary] = 'Starting Conversion',
[StartTime] = '$time' WHERE JobID = $ID"
DoConvert
Your code goes here
Write-log 1 "Done with Conversion."
SQLWrite "UPDATE VMQueue SET InUse = NULL, ConvertServer = NULL WHERE
JobID = $ID"
```


22.3. Shared Logs

It is possible to alter the MAT so that in situation where multiple Helper Servers are in use, they will produce a single MAT.log file.

To do this, edit the value for <Variable Name="Logpath"> (found in the general section of variable.xml) and point it to a shared folder.

WARNING: Using a single log file in a multi-server environment can produce a number of issues. First, if you have a large number of conversions running, the log can be a challenge to read. Although each log entry will identify the convert server and VM, the data can be overwhelming.

Additionally there is a chance you will have file contention issues when the Helpers write to the log file. If you plan to use more than two Helper servers to the same log file, test it carefully beforehand.

For the reasons stated above, it is recommended that you leave the MAT.log on each Helper.

22.4. Shared XML

Like the shared log scenario above, it is possible to alter the MAT so that in situation where multiple Helper Servers are in use, they will read a single XML file.

To do this, share your C:\MAT folder (or whatever location you chose) edit the value for `$XMLPath = "C:\MAT\Variable.xml"` (found at line 21 of ConvertVM.ps1) and point it to your shared XML file.

Although only limited testing was done using a shared XML file, there were no file contention issues detected since the XML file is only read once when the each server begins to process its batch.

NOTE: You can only use the shared XML model if all your accounts are identical, including the account used to remove VMware tools. Also be sure to secure this file if you share it, since it contains sensitive information.

22.5. Balancing the load

As documented above it is possible to use a common XML file and in some situations common logging. However, this approach is not always ideal.

22.5.1. One Hyper-V host for each Helper

If you are running at a large scale you run the risk of overloading your Hyper-V host (as all Helpers will be sending data to it) both in terms on network traffic and storage. A better approach is to customize each Helper server's XML file (*tpath*) so that it has a dedicated Hyper-V server that it uses. For example, "Helper1" points to "Hyper-VHost1", "Helper2" points to "Hyper-VHost2" and so on.

This paired approach limits the total number of VMs being written to each Hyper-V host and will improve your overall performance. Your Hyper-V hosts should be able to easily accommodate the maximum of three VMs being sent to it by its Helper. Following this approach, you should be able to scale out Helpers limited only by your Hyper-V hosts that will accept the VMs.

23. Additional Functions

The ConvertVM.ps1 script also supports a number of additional functions that can be activated by passing the appropriate parameter to the script. These options are below:

Syntax: PS> .\ConvertVM.ps1 <action> (optional) <delay in seconds> (optional)

23.1.1. Actions

- **Convert** - Starts Conversion locally. (Used on Remote servers)
- **Convert-Global** - Starts Conversion on Remotes and locally
- **Create-List** - Creates a VMList.txt based on unconverted VMs in database
- **Help** - Shows the Help menu
- **Menu** - Starts Main Menu <Default>
- **Purge** - Deletes ALL records from database
- **Report-All** - Generates a CSV file with all records from the database
- **Report-Complete** - Generates a CSV file with all completed VMs from the database
- **Report-Incomplete** - Generates a CSV file with all incomplete Vms from the database
- **Report-Unsupported** - Generates a CSV file with all unsupported VMs from the database
- **Report-Warning** - Generates a CSV file with all VMs that produced warnings during conversion
- **Reset-All** - Resets EVERY VM record in the database
- **Reset-Incomplete** - Resets every incomplete VM record in the database
- **Reset-List** - Resets only those VM records listed in C:\MAT\VMList.txt
- **Show-Status** - Displays the Status table once
- **Show-StatusLoop** - Displays the Status table in an endless loop
- **Update** - Updates the records for all VMs in VMList.txt marking them as "Ready to Convert"

Start Delay

- <Delay in seconds> [int] Delays the start time of conversions (default is 0)

These options can also be viewed using **.\ConvertVM.ps1 Help**

24. PowerShell Parameter and Variable Reference

24.1. ConvertVM.ps1

24.1.1. Parameters

- **[string]\$Task**, - option to call any of the Actions supported by MAT. The default action is "Menu"
- **[string]\$DelayTimer**, - Delays the start of the batch of conversions by the specified value (in seconds)

24.1.2. Variables

- **\$QueueLength = 2** - Total number of concurrent conversions allowed on local machine, values above 3 are not supported.
- **\$CurrentPath = "C:\MAT"** – Folder location of the MAT files.
- **\$XMLPath = "C:\MAT\Variable.xml"** – Full path of the variable.xml file.
- **\$VMList = "\$CurrentPath\VMList.txt"** – full path of the VMList.txt file.
- **\$VerboseLogging = 3** – Logging level. 1 = minimum, 3 = normal, 7 = verbose
- **\$MaxCapacityLoopTimer = 60** – Length of sleep delay between checks if \$Semaphore -ge \$QueueLength
- **\$Lookback = 1** – Used by ShowStatus, limits entries displayed to those which have finished in the last (\$Lookback) hours. Default is 1.
- **\$BuildVersion = "1.5.0 "** – Current build version – should be consistent across all files in MAT
- **\$Catalog = "MAT"** – Database catalog for MAT, default = MAT
- **\$RemoteJobName = "MAT"** – Name of Task Scheduler job created on Helper Servers
- **\$TaskArgues = "\$CurrentPath\ConvertVM.ps1 Convert"** – Argument passed to Helper apps to kick off the MAT remotely
- **\$Computername = Get-Childitem env:computername** – Gets the name of the local machine for use with \$localhost
- **\$localhost = \$Computername.Value** – Sets \$localhost name as name of computer running the script.

- **\$Preflight = 1** - Enables or disables preflight checks. 1 = True
- **\$RemoveCD = 1** – Defines whether or not all CD\DVD media is unmounted before conversion. 1 = True
- **\$RemoveFloppy = 1** – Defines whether or not all floppy disks are removed before conversion. 1 = True

24.2. ConvertVM-Process.ps1

24.2.1. Parameters

- **[string]\$ID**, - Record ID from the database table VMQueue, field = JobID
- **[string]\$DataSource**, - location of the database or instance
- **[String]\$Catalog**, - Database catalog, by default = MAT
- **[string]\$MATLog**, - location of the main log produced by the MAT
- **[string]\$VMName**, - Name of the VM being converted, from field = VMName
- **[string]\$Logpath**, - location of folder path for logging
- **[string]\$CurrentPath**, - use to abbreviate execution path of ps1
- **[int]\$VerboseLogging** – Level of logging verbosity, currently 1,3 or 7
- **[string]\$tPower**, - value of Target VM power option
- **[string]\$sPower**, - value of Source VM power option
- **[String]\$XMLPath** – location of the Variable.xml file
- **[String]\$MigrateNICs** – value of MigrateNICs in variable.xml file

24.2.2. Variables

- **\$LogOption = 1** – binary, whether or not ConvertVM-Logging.ps1 is invoked. 1= on.
- **\$LogMonitorDelay = 30** - duration of sleep issued between log checks (passed to ConvertVM-Logging.ps1)

- **\$SleepMultiplier = 1.5.0** – value multiplied by \$LogMonitorDelay to provide a sleep value to allowing ConvertVM-Logging.ps1 to finish running before ConvertVM-Process.ps1 writes final database entry for active VM.
- **\$Computername = Get-Childitem env:computername** – Gets the name of the local machine for use with \$localhost
- **\$localhost = \$Computername.Value** – Sets \$localhost name as name of computer running the script.
- **\$fpath = "C:\Program Files (x86)\Microsoft Virtual Machine Converter Solution Accelerator\MVMC.exe"** – Sets path of MVMC.exe

24.3. ConvertVM-Logging.ps1

24.3.1. Parameters

- **[string]\$ID**, - Record ID from the database table VMQueue, field = JobID
- **[string]\$DataSource**, - location of the database or instance
- **[String]\$Catalog**, - Database catalog, by default = MAT
- **[string]\$mvmclog**, - location of the log produced by the MVMC.exe
- **[string]\$MATLog**, - location of the main log produced by the MAT
- **[string]\$VMName**, - Name of the VM being converted, from field = VMName
- **[int]\$LogMonitorDelay**, - duration of sleep issued between log checks
- **[int]\$VerboseLogging** – Level of logging verbosity, currently 1,3 or 7

24.3.2. Variables

- **\$Status = "zzz"** - An arbitrary value that should not be produced by any log status message. If this value is still set to “zzz” after the logging script has run, it indicates a failure.

25. Permissions

25.1. VMware Virtual Infrastructure

Administrator Rights and Root access to the ESX Sandbox Host

25.2. Guest Operating System (VM being converted)

Local Administrator Rights

25.3. Control Server and Helper Server(s)

Administrator Rights

26. Getting help

The MAT is provided as-is with most of the information you should require available in this document. However if you find a bug or have improvements you would like to share, or if you have questions use the TechNet forum.

If you are reporting an issue with the MAT, please be sure to attach your logs. The logging level of the MAT should be set to '7' in the ConvertVM.ps1 file (line 28 by default)

```
$VerboseLogging = 7
```

Once you have recreated the issue with full logging enabled, please include that log with any communications along with any other information that is relevant including:

- OS and version of Conversion server
- OS and version of Hyper-V host
- OS and version of guest VM (if issue is specific to a VM)
- Version of VMware guest
- Version of VMware ESX host
- Version of vCenter/vSphere (if applicable)

NOTE: The MAT logs will contain the display names and fully qualified domain names of your virtual machines. If your company treats this information as sensitive, you can use a find/replace function to change the names into something simple like "Server1". As long as you do not alter the other contents of the log, this will not affect our ability to review the logging data.

Appendix A: Document History

Document History

Version	Status	Description of changes	Author	Date
1	Draft	Original Document	Mark Gosson	5/26/2013
2	Draft	Updated PS references	Mark Gosson	5/28/2013
3	Final Draft	Version 1	Mark Gosson	5/29/2013
4	Final Draft	Edits	Michael Greene	6/2/2013
5	Final	Published Version	Mark Gosson	6/4/2013
6	Final	Updated MAT acronym	Mark Gosson	7/23/2013
7	Final	Updates for 1.5	Mark Gosson	8/28/2013
9	Final	Updates to Getting Help	Mark Gosson	1/13/2014