Microsoft

Open CloudServer (OCS) Operations Toolkit User Guide

Modification History

Revision	Date	Comments
1.00	Oct-10-14	Initial Release
1.01	Oct-26-14	Changed name to Open CloudServer (OCS) Operations Toolkit. Changed Update.ps1 filename to WcsUpdate.ps1 for updates to avoid conflicts. Grouped commands into Diagnostics, Stress, Updates, and Misc Added additional descriptions and clarifications to improve readability.

Table of Contents

1 Overview

The Open CloudServer (OCS) Operations Toolkit is a collection of scripts and utilities for updating, diagnosing, and testing OCS servers and chassis managers. This Toolkit provides a one stop shop for utilities, tests and diagnostics that provide:

- Diagnostics
  - Identify defective components such as HDD, DIMM, and processor
  - View, log, and compare configurations
  - Read, clear and log errors
- Stressors
  - System stress tests to identify intermittent problems
  - Component specific stress tests
  - Cycling tests to identify intermittent initialization problems
- Updates
  - Update programmable components such as BIOS and BMC
  - Batch update of all programmable components
- Miscellaneous
  - Debug functions to execute IPMI and REST commands

The Toolkit runs on 64 bit versions of WinPE version 5.1 or later, Windows Server 2012 or later, and Windows 8.1 or later. The Toolkit can be deployed on bootable WinPE USB flash drives, WinPE RAM drives (from PXE Server), and drives with the Windows Server and Desktop Operating Systems.

1.1 Implementation

The Toolkit is implemented using PowerShell scripts. Every command in the Toolkit is a PowerShell function that can be executed within a PowerShell console, from a CMD shell, or from within another PowerShell script. The advantages of using PowerShell are…

- Shared code inside and outside of Microsoft
- Significant experience within Microsoft and the industry
- Readable and writeable with text editor
- Runs under WinPE and Windows
- WMI support
- Expandable with 3rd party utilities (ie: LSI/Mellanox)

2 Quick Start Guide - WinPE

The Toolkit is designed to run under WinPE 5.1 or later. The Toolkit can be added to an existing WinPE image or a new image can be created specifically for the Toolkit.

2.1.1 Create the USB flash drive

To create a Toolkit WinPE image on a USB flash drive see the Installation for WinPE section.

2.1.2 Insert the USB flash drive

Insert the USB flash drive into the server blade's USB connector. The mechanical location of the USB connector is different for each blade and in some cases may require a special converter cable and/or removing brackets to get access.

2.1.3 Boot to the USB flash drive

Power on the blade and boot to the USB drive. In some cases the BIOS boot order may need to be changed to boot to the USB drive. If this is the case enter BIOS setup using the serial connection and change the boot order.

2.1.4 Open a Serial Connection

If running on a headless server then a serial connection is used for video and keyboard. To open a serial connection from a Chassis Manager from WCSCLI command prompt:

```
wcscli -startbladeserialsession -i <bladeslot>
```

When the blade has booted to the WinPE image the serial console will display a SAC> prompt. A short time later a message saying CMD is available is displayed. After the CMD is available start a CMD session from the SAC prompt by typing:

```
cmd
ch -si 1
```

2.1.5 Start the Toolkit

To start the Toolkit from the command prompt type PowerShell and then load the scripts by typing

```
. \WcsTest\Scripts\WcsScripts.ps1
```

2.1.6 Common Commands

The following table lists common commands used for debugging with the WinPE image.

Command	Description
View-WcsHealth	Displays the current health of the system
View-WcsSel -Hardware	Displays hardware error entries in the BMC SEL
Clear-WcsError	Clears Windows event logs and BMC SEL
View-WcsDimm	Displays basic DIMM info for each DIMM in the system
View-WcsDisk	Displays basic disk info and status for each disk in the system
View-WcsFirmware	Displays BIOS and BMC versions
Run-Quickstress	Runs system stress for the time specified

3 Quick Start Guide -Windows Server and Desktop OS

3.1.1 Install the Toolkit

To install the Toolkit see the Installation for Windows Server and Desktop OS section.

3.1.2 Start the Toolkit

Double click the OCS Toolkit shortcut on the desktop. This opens a PowerShell console and automatically loads the scripts.

3.1.3 Common Commands

To display a list of commands type Get-OcsHelp or OcsHelp.

The following table lists common commands used within the Windows Desktop and Server OS. For additional help on any of these commands type Get-Help <command name> -Full

Command	Description
View-WcsHealth	Displays the current health of the system
View-WcsSel -Hardware	Displays hardware error entries in the BMC SEL
Clear-WcsError	Clears Windows event logs and BMC SEL
View-WcsConfig	Displays configuration information on the system
View-WcsVersion	Displays the version of the Toolkit
Run-Quickstress	Runs system stress for the time specified

3.1.4 Documentation

This User Guide can be found in \WcsTest\Scripts\Reference\Documentation directory.

4 Command List Summary

4.1 Diagnostic Commands

4.1.1 System Health and Error Commands

Command	Description	Limitations
Clear-WcsError	Clears Windows event logs and BMC SEL	
Check-WcsError	Checks the Windows event logs and BMC SEL for suspect errors	
View-WcsHealth	Displays the current health of the system	
Log-WcsHealth	Logs the current health of the system to a file	
Get-WcsHealth	Returns an object containing the health of the system	
View-WcsSel	Displays the BMC SEL	
Log-WcsSel	Logs the BMC SEL to a file	
Get-WcsSel	Returns an object containing the BMC SEL entries	

4.1.2 Configuration Commands

Command	Description	Limitations
View-WcsDimm	Displays DIMM info and status for each DIMM	
View-WcsDisk	Displays disk info and status for each disk	
View-WcsDrive	Displays logical drives in the system	
View-WcsFirmware	Displays BIOS and BMC versions	
View-WcsFru	Displays basic FRU info	
View-WcsHba	Displays basic info for storage adapter	
View-WcsNic	Displays basic info for NIC adapter	
View-WcsProcessor	Displays basic info on processor	
View-WcsUpdate	Displays updates available	
Get-WcsConfig	Reads the system configuration from a file or the current system and returns an XML object	
Log-WcsConfig	Logs a system configuration into an XML file and text readable file	
Compare-WcsConfig	Compares two configurations and returns the number of mismatches	
View-WcsConfig	Displays a system configuration	
Update-WcsConfig	Batch updates system to all of the latest programmable components	
Log-MsInfo32	Logs system information from msinfo32 into a log file	No WinPE

4.2 Update Commands

Command	Description	Limitations
Update-WcsConfig	Batch updates system to all of the latest programmable components	

4.3 Stress Commands

4.3.1 System Stress Commands

Command	Description	Limitations
Run-QuickStress	Runs IO, memory and processor stress using Prime95 and DiskSpeed or IOmeter	
Run-IOmeter	Runs IOmeter	No WinPE
Run-DiskSpeed	Runs DiskSpeed IO stress	
Run-Prime95	Runs Prime95 torture test	

4.3.2 Cycle Test Commands

Command	Description	Limitations
Cycle-OsReboot	Runs IO, memory and processor stress using Prime95 and DiskSpeed or IOmeter	
Cycle-WcsUpdate	Cycles between two versions of updates	No WinPE

Cycle-WcsBladePower	Cycles power to multiple blades within a chassis	No WinPE
Set-Autologin	Setup the OS for autologin	No WinPE

4.3.3 Test Commands

Command	Description	Limitations
Pre-WcsTest	Run before test to clear logs and gather config information	
Post-WcsTest	Run after test to gather log and config information	

4.4 Miscellaneous Commands

4.4.1 Base Commands

Command	Description	Limitations
Get-WcsHelp or Get-OcsHelp	Displays Toolkit commands	
View-WcsVersion or View-OcsVersion	Displays the version of the OCS Operations Toolkit	

4.4.2 Comm Commands

Command	Description	Limitations
Invoke-WcsRest	Executes a REST command on one or more chassis managers and returns result as an XML object	

4.4.3 IPMI Commands

Command	Description	Limitations
Invoke-WcsIpmi	Executes an IPMI command on a WCS blade	
Get-WcsFruData	Direct read of FRU device	
Set-WcsFruData	Direct write of FRU device	
Set-WcsFruChecksum	Writes checksum for a FRU data range	
Log-WcsFru	Logs the FRU device data	
Get-WcsFru	Gets the FRU information	
Update-WcsFru	Updates the FRU with user information such as Asset Tag	

4.4.4 Remote Commands

Command	Description	Limitations
Copy-WcsFile	Copies files to one or more remote systems	
Copy-WcsRemoteFile	Copies files from one or more remote systems	
Invoke-WcsCommand	Runs command on one or more remote systems	
Invoke-WcsScript	Runs script on one or more remote systems	
Set-WcsBladeCredential	Sets default credential for WCS blade accesss	
Set-WcsChassisCredential	Sets default credential for WCS chassis managers	
Reboot-WcsBlade	Reboots one or more remote blade systems	
Reboot-WcsChassis	Reboots one or more remote chassis managers	

5 Requirements to Run the Toolkit

5.1.1 PowerShell ExecutionPolicy must allow script execution

The Toolkit requires the PowerShell execution policy be set to allow the running of scripts. One possible way to enable script execution is to run this command:

```
PowerShell -Command Set-ExecutionPolicy RemoteSigned -Force
```

5.1.2 Run As Administrator

Many of the commands must be run as administrator because they read low level hardware information. If commands are not run as an administrator they may return incomplete or incorrect information.

Note that starting the Toolkit using the desktop shortcut automatically runs the Toolkit as administrator.

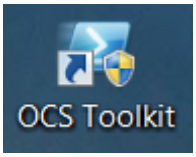
5.1.3 PowerShell Version 3.0 or later

The Toolkit requires PowerShell version 3.0 or later. Windows Server 2012 and WinPE 5.1 (or later) include a PowerShell version that meets this requirement. Older versions of Windows (ie: XP) would require installing version 3.0 of PowerShell separately.

6 Running the Toolkit

6.1 Running in a PowerShell Console - Windows OS

The PowerShell console allows commands to be executed interactively and is the best environment for debugging and developing new scripts. Within Windows Desktop or Server OS the easiest way to run the Toolkit in a PowerShell console is to double-click the OCS Toolkit shortcut on the desktop.



To load the scripts without using the shortcut start a PowerShell console (as Administrator) and dot source the top level script by typing:

```
. \WcsTest\Scripts\WcsScripts.ps1
```

6.2 Running in a PowerShell Console - WinPE

Start a PowerShell console (as Administrator) and dot source the top level script by typing:

```
. \WcsTest\Scripts\WcsScripts.ps1
```

When using the SAC (Special Administration Console) the PowerShell console can be started by typing PowerShell from the CMD session prompt.

Note that the console window is limited to 25 lines when running the SAC within WinPE. This significantly reduces the data that can be displayed.

6.3 Running from the Command Line

A CMD shell can be opened using either SAC or Remote Desktop. From within the CMD shell individual Toolkit commands can be run using the following syntax:

```
PowerShell -Command EXIT (. \WcsTest\Scripts\WcsScripts.ps1; <command>)
```

The Exit() syntax is required so the command's return code is provided to the caller.

The . \WcsTest\Scripts\WcsScripts.ps1 is required to load the scripts and libraries that the command requires. Note the loading of the scripts can be added to a profile so that this is done automatically.

6.4 Running Remotely via PSEXEC64

Commands can be run on one or more remote systems. This is implemented using the PSEXEC64 utility. Remote execution requires:

1. Knowing the IP address or hostname of the targets
2. Network access to the targets
3. Knowing the administrator credentials for the target
4. Remote execution enabled on the target's OS

Because of the above requirements remote execution is typically only used in a lab or preproduction environment.

7 Documentation and Help

7.1 User Guide

This User Guide can be found in the \WcsTest\Scripts\References\Documentation directory. This User Guide provides general information on the Toolkit but for more detailed information refer to the Online Help and PowerShell scripts.

7.2 Online Help

All commands include standard PowerShell help information that can be viewed using the built-in PowerShell Get-Help command. To display a list of commands type one of the following commands:

```
Get-WcsHelp
```

```
WcsHelp
OcsHelp
Get-OcsHelp
```

To display help information on a command in the above list use the Get-Help commands like:

```
Get-Help  Run-QuickStress
```

Displays help information on the Run-QuickStress command. Use the standard PowerShell switches -Full and -Examples to get additional help information.

7.3 PowerShell Scripts

Commands are implemented as PowerShell functions and can be read with a text editor or the PowerShell ISE. Reading the scripts is the best way to determine what the commands actually do and why they do it.

The scripts can be found in the \WcsTest\ Scripts and \WcsTest\Scripts\Library directories. The top level script is \WcsTest\Scripts\WcsSripts.ps1. This script loads the rest of the library scripts.

To find the source for a specific command look in the library file for that command's group. The command's group can be found in the Command List Summary section. For example:

- Base commands are in \WcsTest\Scripts\Library\BaseLibrary.ps1
- Configuration commands are in \WcsTest\Scripts\Library\ConfigLibrary.ps1
- Comm commands are in \WcsTest\Scripts\Library\CommLibrary.ps1
- System stress commands are in \WcsTest\Scripts\Library\StressLibrary.ps1
- Cycle test commands are in \WcsTest\Scripts\Library\CycleLibrary.ps1
- Test commands are in \WcsTest\Scripts\Library\TestLibrary.ps1
- Remote commands are in \WcsTest\Scripts\Library\RemoteLibrary.ps1
- IPMI commands are in \WcsTest\Scripts\Library\IpmiLibrary.ps1

7.4 Logging Results

By default all commands log results in the \WcsTest\Results directory. This can be changed using input parameters on most Toolkit commands. However, the idea is to place all commands in the same area so they are easy to find.

8 Diagnostic Commands

8.1 Health and Error Commands

8.1.1 Clear-WcsError

Command clears the Windows Event Logs and the BMC SEL.

8.1.2 Check-WcsError

Command backs up and checks the Windows Event Logs and BMC SEL for suspect errors. Suspect errors are defined as entries likely to be caused by hardware issues. By default the command considers the following entries as suspect errors:

- BMC SEL entries for hardware errors
- Windows System Event Log entries from WHEA, bug checks, and critical errors

8.1.3 View-WcsHealth

Command displays the overall health of the System. The commands includes switches to ignore errors in the Windows Device Manager, FRU, and hardware error logs. The below example shows a system with ECC errors and missing device drivers. Note there is no device manager in WinPE.

```
PS C:\WcsTest> view-wcshealth

  SYSTEM HEALTH DEGRADED - FOUND ERRORS...

-------------------------------------------------------------------------
 Location        # Errors   Last Error
-------------------------------------------------------------------------
 DIMM A1                2   Uncorrectable ECC -  8/26/2014  10:18:53 PM
 DEVICE MANAGER         3   Device drivers are not installed. (0x1C) PCI Simple Communications Controller
PS C:\WcsTest>
```

8.1.4 Log-WcsHealth

Command logs the current health to a file.

8.1.5 View-WcsSel

Displays the BMC SEL entries. The command has a switch to display decoded or undecoded entries. It also has a switch to only display hardware error entries. It can also filter by sensor and record types.

8.1.6 Log-WcsSel

Command logs the current BMC SEL entries to a file.

8.2 Configuration Commands

These commands get, view, log, and compare system configurations. There are several configuration commands that display a subset of the information displayed by the View-WcsConfig command. These commands limit their output to less than 25 lines and 80 characters so they work well when using the SAC console within WinPE.

8.2.1 View-WcsFirmware

Displays the BIOS and BMC version of the system.

8.2.2 View-WcsDisk

Command displays the disk location, status, serial number, and firmware version. Disk location is displayed as <BladeType>-<LabelNumber> where:

< BladeType > is SB for Server Blade or DB for a Disk/Data Blade

< LabelNumber > is the number on the blade disk label in the front/top of blade

Example output from View-WcsDisk showing HDD on the data blade (JBOD).

```
PS C:\WcsTest> view-wcsdisk
-------------------------------------------------------------------------
Location            Status      Serial                Firmware      Size
-------------------------------------------------------------------------
DB-0                OK          WD-WCC131881809       1K02          4.0 TB
DB-1                OK          WD-WCC131863082       1K02          4.0 TB
DB-2                OK          WD-WCC131865500       1K02          4.0 TB
DB-3                OK          WD-WCC131906868       1K02          4.0 TB
DB-4                OK          WD-WCC131895120       1K02          4.0 TB
DB-5                OK          WD-WCC131896993       1K02          4.0 TB
DB-6                OK          WD-WCC131917154       1K02          4.0 TB
DB-7                OK          WD-WCC131891865       1K02          4.0 TB
DB-8                OK          WD-WCC131884411       1K02          4.0 TB
```

8.2.3 View-WcsDimm

Command displays the DIMM location, status, serial number, model, and size. DIMM location is the silkscreen label on the board next to the connector. If the DIMM has reported ECC errors then the status will indicate an error as show on DIMM A1 below.

```
PS C:\WcsTest> view-wcsdimm
-----------------------------------------------------------------------------
Location        Status      Serial          Model                    Size
-----------------------------------------------------------------------------
DIMM A1         ERROR       213E702C        M393B1G73BH0-YH9         8.0 GiB
DIMM A2         OK          213E7052        M393B1G73BH0-YH9         8.0 GiB
DIMM B1         OK          213E7033        M393B1G73BH0-YH9         8.0 GiB
DIMM B2         OK          213E702D        M393B1G73BH0-YH9         8.0 GiB
DIMM C1         OK          213E709B        M393B1G73BH0-YH9         8.0 GiB
DIMM C2         OK          213E6FF3        M393B1G73BH0-YH9         8.0 GiB
DIMM D1         OK          213E70A1        M393B1G73BH0-YH9         8.0 GiB
DIMM D2         OK          213E7023        M393B1G73BH0-YH9         8.0 GiB
DIMM E1         OK          213E6FEF        M393B1G73BH0-YH9         8.0 GiB
DIMM E2         OK          213E70A4        M393B1G73BH0-YH9         8.0 GiB
DIMM F1         OK          213E70AB        M393B1G73BH0-YH9         8.0 GiB
DIMM F2         OK          213E70A5        M393B1G73BH0-YH9         8.0 GiB
```

8.2.4 View-WcsDrive

Displays logical drives mapped by the OS.

```
PS C:\WcsTest> View-WcsDrive
-------------------------------------------------------------------------------
 Drive   Description          Size
-------------------------------------------------------------------------------
 C:      Local Fixed Disk     107374178304 (107.4 GB)

PS C:\WcsTest> _
```

8.2.5 View-WcsFru

Displays blade FRU information. Example output showing a blade with missing Asset information.

```
PS C:\WcsTest> View-WcsFru
-------------------------------------------------------------------------------
 FRU Field                    Value
-------------------------------------------------------------------------------
 ProductName                  C1000
 ProductModel                 X873095-001
 ProductAsset
 ProductSerial                QTFCIM3240003
 Version                      FRU v0.01

PS C:\WcsTest> _
```

8.2.6 View-WcsHba

Displays blade HBA model, serial number, and firmware.

```
PS C:\WcsTest> View-WcsHba
-------------------------------------------------------------------------------
 HBA                          Serial          Firwmare        FW Package
-------------------------------------------------------------------------------
 LSI MegaRAID SAS 9270CV-8i   SV31732362      3.340.55-3173   23.22.0-0020

PS C:\WcsTest> _
```

8.2.7 View-WcsNic

Displays blade NIC model, MAC, and firmware.

```
PS C:\WcsTest> View-Wcsnic
-------------------------------------------------------------------------------
 NIC                                                  Mac
-------------------------------------------------------------------------------
 Mellanox ConnectX-3 Ethernet Adapter                 08:9E:01:93:CF:BD
 Mellanox ConnectX-3 Ethernet Adapter                 08:9E:01:93:CF:BC

-------------------------------------------------------------------------------
 Mellanox ID          Firwmare        PXE             UEFI
-------------------------------------------------------------------------------
 4099                 2.31.5080       3.4.225         10.4.18

PS C:\WcsTest> _
```

8.2.8 View-WcsProcessor

Displays processor information.

```
PS C:\WcsTest> View-Wcsprocessor
-------------------------------------------------------------------------------
 Processor  Model
-------------------------------------------------------------------------------
 SOCKET 0   Intel(R) Xeon(R) CPU E5-2450 0 @ 2.10GHz
            Intel64 Family 6 Model 45 Stepping 7

 SOCKET 1   Intel(R) Xeon(R) CPU E5-2450 0 @ 2.10GHz
            Intel64 Family 6 Model 45 Stepping 7


PS C:\WcsTest> _
```

8.2.9 Full Configuration Commands

The rest of the configuration commands view, log, and compare the full configuration of systems. Because these commands display a lot of information it is not recommended to run them within the SAC.

8.2.10 View-WcsConfig

To view the current configuration:

```
View-WcsConfig
```

By default only displays a summary of the config. To see the full summary add the -Full switch.

```
View-WcsConfig -Full
```

Partial example of View-WcsConfig output:

```
-------------------------------------------------------------------------------
System Info
-------------------------------------------------------------------------------
   Computer                    WCSEXAD001
   TotalMemmory                68683476992 (64.0 GiB)
   TotalProcessors             16
-------------------------------------------------------------------------------
Software Info
-------------------------------------------------------------------------------
   BIOS Version                T6M_3B07
   BMC Version                 4.05
   OS Name                     Microsoft Windows Server 2012 Datacenter (Version 6.2.9200)
-------------------------------------------------------------------------------
FRU Info
-------------------------------------------------------------------------------
   Chassis Part Number         X873021-001
   Chassis Serial Number
   Board Manufacturer          Microsoft
   Board Name                  C1000
   Board Part Number           X873096-001
   Board Serial Number         MH832100310
   Board FRU                   FRU v0.01
-------------------------------------------------------------------------------
Processor Info
-------------------------------------------------------------------------------
   SOCKET 0                    Intel(R) Xeon(R) CPU E5-2450 0 @ 2.10GHz (Cores: 8 LogicalCores: 8
   SOCKET 1                    Intel(R) Xeon(R) CPU E5-2450 0 @ 2.10GHz (Cores: 8 LogicalCores: 8
-------------------------------------------------------------------------------
DIMM Info
-------------------------------------------------------------------------------
   ChannelB_Dimm1              Micron                    36KSF2G72PZ-1G6E1      Speed: 1333   S
   ChannelC_Dimm1              Micron                    36KSF2G72PZ-1G6E1      Speed: 1333   S
   ChannelE_Dimm1              Micron                    36KSF2G72PZ-1G6E1      Speed: 1333   S
   ChannelF_Dimm1              Micron                    36KSF2G72PZ-1G6E1      Speed: 1333   S
-------------------------------------------------------------------------------
Disk Info
```

8.2.11 Get-WcsConfig

To store the current configuration in a PowerShell variable:

```
$NewConfig  = Get-WcsConfig
```

8.2.12 Log-WcsConfig

To log the configuration stored in the variable $NewConfig to the file MyConfigFile:

```
Log-WcsConfig -Config $NewConfig -File MyConfigFile
```

Configurations are logged in an XML file and a more readable text file. The resulting file names are <filename>.Config.XML and <filename>.Config.Log. In the above example they would be MyConfigFile.Config.XML and MyConfigFile.Config.Log.

By default all configurations are logged to and read from \WcsTest\Configurations directory.

By default the command will log the current configuration. For example, to log the current configuration to \ WcsTest\Configurations\CurrentFile.*

```
Log-WcsConfig  CurrentFile
```

8.2.13 Compare-WcsConfig

Configurations can be compared with the Compare-WcsConfig command. By default the Compare-WcsConfig does not compare unique fields such as MAC address and serial numbers. To include the unique fields in the comparison use the -Exact switch. Typically the -Exact switch is used when comparing against a previous configuration where the components should be exactly the same.

To compare against a previous configuration stored in a file:

```
$Recipe = Get-WcsConfig  <Name of previous configuration file>
Compare-WcsConfig -RefConfig  $Recipe  -Exact
```

To compare against a recipe stored in a file:

```
$Recipe = Get-WcsConfig <Name of recipe file>
Compare-WcsConfig -RefConfig  $Recipe -OnlyRefDevices
```

The results of the comparison can be saved into a variable. Simply pass a reference to the variable to store the results. The below examples shows the comparison results being stored in the $Results variable and then logged to the CompareResultsFile.

```
Compare-WcsConfig -RefConfig  $Recipe -RefToResults ([ref] $Results)
Log-WcsConfig -Config $Results -File CompareResultsFile
```

9 Configuration Updates

To update all system components use the Update-WcsConfig command. To update an individual programmable component within the console type the full path to the WcsUpdate.ps1 file. To determine which updates are available for updating run the View-WcsUpdate command.

9.1.1 Update-WcsConfig

The Update-WcsConfig command updates all the programmables for any WCS blade or chassis manager. The command works as follows:

1. Reads the BIOS version and FRU to determine the system type
2. Reads current versions of all programmables
3. Updates only the down-rev programmables in the proper sequence. This may require reboots or power cycles
4. When all programmables have been updated returns code 0 and displays a message

This command requires the latest programmables to update to the latest configurations. Before updating any programmables always ensure using the latest release.

10 System Stress Commands

The system stress commands provide functional stress at the system or subsystem level. These commands configure the stress applications, run them the time specified, and then parse their log files for errors.

10.1.1 Run-IoMeter and Run-DiskSpeed (IO Stress)

There are two applications used for IO stress: IOmeter and DiskSpd. To run IOmeter for 60 minutes:

```
Run-IOmeter -TimeInMin 60
```

By default the IO stress test is run on all logical drives, except C:, with at least 100MB free space and all physical drives without a partition. To test the C: use the -Full switch.

10.1.2 Run-Prime95 (Memory and Processor Stress)

The Run-Prime95 command runs the Prime95 torture test concurrently as a memory and processor stress test. To run the stress for 60 minutes:

```
Run-Prime95 -TimeInMin 60
```

10.1.3 Run-Quickstress (Concurrent System Stress)

The Run-QuickStress command runs the Prime95 torture test concurrently with IO stress. The default IO stress is DiskSpd but IOmeter can also be run. To run concurrent systems stress for 60 minutes:

```
Run-QuickStress -TimeInMin 60
```

By default the IO stress test is run on all logical drives, except C:, with at least 100MB free space and all physical drives without a partition. To test the C: use the -Full switch. If no drives are available then only Prime95 is run.

10.2 Cycle Test Commands

The Cycle Test Commands provide a way to continuously cycle systems. During each cycle the configuration is checked against a reference configuration and Check-WcsError is run to find any suspect errors.

The Cycle Test Commands require systems to be setup for auto-login. Follow the help information in the Set-Autologin script help to accomplish this.

10.2.1 Cycle-OsReboot

This command reboots the system using the OS reboot command. This command requires some setup before execution. Refer to the command's help information for what setup is required. To run 100 reboots on a system:

```
Cycle -OsReboot  -NumberOfCycles 100
```

10.2.2 Cycle-WcsUpdate

It is also possible to cycle test the updating of the BIOS or other firmware using the Cycle-WcsUpdate command. This command requires some setup before execution. Refer to the command's help information for what setup is required. This command is targeted for internal validation of server blade systems.

10.2.3 Cycle-WcsBladePower

It is also possible to power cycle all blades in a chassis using the Cycle-WcsBladePower command. This command requires some setup before execution. Refer to the command's help information for what setup is required. This command is targeted for internal validation of server blade systems.

10.3 Test Commands

10.3.1 Pre-WcsTest

Command to be run prior to a test that logs the system configuration and then clears the error logs.

10.3.2 Post-WcsTest

Command to be run after a test that reports the number of suspect errors in the Windows System Event Log and the BMC SEL and then backs up the error logs and saves the system configuration.

11 Miscellaneous Commands

11.1 Communication (Comm) Commands

11.1.1 Invoke-WcsRest

This commands allows REST commands to be sent to one or more chassis managers concurrently. The responses are returned as an array of XML objects. The below examples read the chassis manager service version for all the targets specified in $ipList

```
$Resp = Invoke-WcsRest -TargetList $IpList -Command "GetServiceVersion"
```

11.2 IPMI Commands

11.2.1 Invoke-WcsIpmi

This commands allows in band access to the server BMC using IPMI commands. This command requires knowing the low level byte inputs and outputs for IPMI. Refer to the IPMI specification and this command's help for details on how to run.

11.3 Remote Commands

Commands can be run on one or more remote systems. Remote execution requires:

- Knowing the IP address or hostname of the targets
- Network access to the targets
- Knowing the administrator credentials for the target
- Remote execution enabled on the target's OS

Because of the above requirements remote execution is typically only used in a lab or preproduction environment. There is significant delay with running remote commands so it only make sense to use them when running commands on more than one system or when running commands unattended.

To run a command remotely follow this sequence:

1. By default the remote commands use the credentials in the \WcsTest\Scripts\Library\CredentialLibrary.ps1 file. To use different credentials modify that file or run the following commands:

   ```
   Set-WcsBladeCredential -User <user> -Password <password>
   Set-WcsChassisCredential -User <user> -Password <password>
   ```

2. Copy the Toolkit to the targets. If the targets already have the latest Toolkit installed this step can be skipped.

   ```
   Copy-WcsFile -Target $IpList -LocalDirectory \Wcstest\* [-Chassis]
   ```

Specifying -Chassis uses the chassis credentials. If not specified then uses the blade credentials.

The $IpList variable is an array of IP addresses for the targets. For example:

```
$IpList = @("192.168.200.10","192.168.200.11")
```

3. Run one or more commands using the Invoke-WcsCommand command. This example shows running QuickStress on all the targets in the $ipList.

```
Invoke-WcsCommand -Target $IpList -Command 'Run-Quickstress -Time 1'
```

By default the Invoke-WcsCommand only waits 5 minutes for the remote command to complete. If the commands takes longer than 5 minutes a timeout error occurs. To increase this wait time use the WaitTimeInSeconds parameters.

The command returns the number of targets that did not return an exit code of 0. If all targets returned 0 then the commands returns 0. The command also displays the return code information for each target if not 0.

A common return code is 1326 which indicates the credentials are not valid.

4. [Optional] Copy the log files back. To copy all the results files from all the remote system use the following command.

```
Copy-WcsRemoteFile -Target $IpList -RemoteDirectory \WcsTest\Results
```

The target's files are copied to \WcsTest\RemoteFiles\<target>\Results. For example:

\WcsTest\RemoteFiles\192.168.200.10\Results\...

\WcsTest\RemoteFiles\192.168.200.11\Results\...

12 Installation for Windows Server and Desktop OS

To install the Toolkit to Windows 8.1 or Server 2012 or later:

1. If a previous installation exists delete the \WcsTest\Scripts folder

2. Copy all files into \WcsTest

The Toolkit can be installed in any directory. Simply extract the compressed file to the install directory desired. However, examples in this document assume the Toolkit is installed in \WcsTest so if the Toolkit is installed in a different directory replace \WcsTest with the install directory.

3. Enable script execution in PowerShell. One possible way to enable script execution is to run this command:

```
PowerShell -Command Set-ExecutionPolicy RemoteSigned -Force
```

4. Copy the shortcut \WcsTest\OCS Toolkit.lnk to the desktop. If not installing to \WcsTest then modify the shortcut properties to point to the install directory.

5. For Open Source Releases install 3rd party applications according to Appendix: Installing 3rd Party Apps

To uninstall:

1. Delete the install directory (typically c:\WcsTest)

13 Installation for WinPE

The Toolkit is designed to work with WinPE 5.1 and later. To add the Toolkit to an existing WinPE image simply add the \WcsTest folder and enable script execution for PowerShell.

Instructions for creating a new WinPE image can be found here:

\WcsTest\Scripts\References\Documenation\WinPE\Creating_OCS_WinPE_Image.txt

14 Troubleshooting with the Toolkit

14.1 Finding Component's Physical Location

14.1.1 Finding Disks

The View-WcsDisk command reports a disk's location as <BladeType>-<LabelNumber> where:

< BladeType > is SB for Server Blade or DB for a Disk/Data Blade

< LabelNumber > is the number on the blade disk label in the front/top of blade

For example, the Disk SB-3 is server blade disk #3 according to the label on the front/top of the server blade. Disk DB-8 is disk blade disk #8 according to its label.

14.1.2 Finding DIMMs

The View-WcsDimm command reports a DIMM's location as the silkscreen label on the board next to the connector.

14.1.3 Finding Processors

The View-WcsProcessor command reports processors as Socket 0 or Socket 1 which corresponds to CPU 0 and CPU 1 on the board silkscreen.

14.2 Identifying Unresponsive Components

By definition unresponsive components cannot be read therefore the only way to detect them is compare the current components against a list of expected components or a previous list when all components were working. This can be done using the commands below.

14.2.1 Using Process of Elimination

To identify an unresponsive disk list all responding disks then visually inspect the system to identify any slots with disks that did not respond. To list the responding disks use one of:

```
View-WcsDisk
View-WcsConfig
```

The same technique can be used to identify DIMMs, NIC, and HBA.

14.2.2 Comparing against a Previous Configuration

When using a previous configuration the comparison reports the unresponsive components along with unique information for the missing component such as serial number, firmware, and manufacturer. To compare against a previous configuration:

```
$Recipe = Get-WcsConfig  <Name of previous configuration file>
Compare-WcsConfig -RefConfig  $Recipe  -Exact
```

14.2.3 Comparing against a Recipe File

The recipe file lists the expected components at each location but without any knowledge of component specific information such as serial number and firmware version. When the comparison is done it reports any components missing from the recipe file. To compare against a recipe:

```
$Recipe = Get-WcsConfig <Name of recipe file>
Compare-WcsConfig -RefConfig $Recipe -OnlyRefDevices
```

14.3 Identifying Disks with Errors

To identify disks that are responsive but reporting errors run the following command:

```
View-WcsDisk
```

14.4 Identifying DIMM with ECC Errors

To identify DIMMS that are reporting errors run the following command:

```
View-WcsDimm
```

14.5 Identifying Intermittent Boot Problems

Intermittent boot problems can be identified by continuously rebooting the system and looking for errors. To cycle a system for 300 cycles run the following command:

```
Cycle-OsReboot  -NumberOfCycles 300
```

This command requires some setup. Refer to the command's help information for details.

14.6 Identifying Intermittent Problems

In many cases intermittent hardware problems can be found by running functional system stress for an extended period of time. To run system stress for 60 minutes:

```
Run-QuickStress -TimeInMin  60
```

To run the stress for 15 minutes:

```
Run-QuickStress -TimeInMin  15
```

After system stress is completed check for errors in the Windows System Event Log and BMC SEL by running Check-WcsError.

15 Appendix: Additional References

15.1 Special Administration Console (SAC) Reference

Special Administration Console (SAC) is the primary Emergency Management Services command-line environment hosted by Windows Server operating systems. It is separate from the command-line environment and provides different functionality. When Emergency Management Services is enabled, SAC remains active as long as the kernel is running.

SAC provides a set of commands you can use to perform a number of management tasks that help return your system to a normally functioning state. These tasks include:

- Restarting or shutting down the server.
- Viewing a list of processes that are currently active.
- Ending processes.
- Setting or viewing the Internet Protocol (IP) address of the server.
- Generating a Stop error to create a memory dump file.
- Starting and accessing command prompts.

For more information about using SAC see MSDN.

15.1.1 Using SAC with the Toolkit

Useful SAC commands when running the Toolkit:

cmd	Creates a Windows command-prompt channel
ch -si <n>	Changes to channel <n> where channel 0 is SAC
I	Displays IP parameters
restart	Restarts

Within a CMD window to start Powershell:

Powershell

To enable script execution in PowerShell

Set-ExecutionPolicy -RemoteSigned -Force

To load the OCS Operations Toolkit scripts "dot source" them

. \<installDirectory>\Scripts\wcsScripts.ps1

Where <installDirectory> is the install directory of the scripts, typically \wcsTest

16 Appendix: Installing 3rd Party Apps

This section describes how to install 3rd party applications to be compatible with the OCS Operations Toolkit. It is not necessary to install any or all of these applications.

16.1 Common Binaries

16.1.1 DiskSpd

To integrate DiskSpd copy dskspd.exe to \WcsTest\Scripts\Binaries\Diskspd.

16.1.2 IOMeter

To integrate IOmeter copy dynamo.exe and iometer.exe to \WcsTest\Scripts\Binaries\IOmeter.

16.1.3 LSI

To integrate LSI utilities copy megacli64.exe, storcli64.exe, and sas2flash.exe to \WcsTest\Scripts\Binaries\LSI.

16.1.4 Mellanox

Copy the Mellanox utilities to \WcsTest\Scripts\Binaries\Mellanox.

16.1.5 Quanta

Copy the Quanta utilities to \WcsTest\Scripts\Binaries\Quanta.

16.1.6 Prime95

Copy prime95.exe, prime.txt, and local.txt to\WcsTest\Scripts\Binaries\Prime95

16.1.7 PSEXEC

Copy psexec64.exe to \WcsTest\Scripts\Binaries

16.2 Update Binaries

Every system has a collection of update utilities. Copy these utilities to the directories under \WcsTest\Scripts\Updates.

17 Appendix: Compatibility Requirements

A system is determined to be compatible with the OCS Operations Toolkit when it meets the requirements in this section.

17.1 Generic Requirements

ID	Requirement	Comment
1.01	System BIOS and FRU properties must uniquely identify the system	
1.02	SEL error entries must comply with the IPMI v2.0 and WCS Software Blade API specifications v2.0	
1.03	System FRU must be readable programmatically	
1.04	The mandatory IPMI commands in IPMI v2.0 specification must be supported and accessible using the Windows IPMI driver	

17.2 Update Requirements

The OCS Operations Toolkit command 'Update-WcsConfig' must automatically update all components that are not the correct revision AND not update any components that are the correct version. This requires the update utilities meet the requirements in the table below.

ID	Requirement	Comment

2.01	The OCS Operations Toolkit command 'Update-WcsConfig' must automatically update all components that are not the correct revision AND not update any components that are the correct version	
2.02	All field update-able components (ie: BIOS, BMC) must have update utilities that can be integrated into the OCS Operations Toolkit	
2.03	Update utilities must support command line execution using input arguments and/or configuration files	
2.04	Update utilities must run in the x64 version of WinPE 5.1 (or later) and Windows Server 2012 (or later)	
2.05	Update utilities must allow the ability to query version without update	
2.06	Update utilities must provide error codes upon exit where a code of 0 indicates success	
2.07	Utility size should be reasonable (prefer less than 5MB)	
2.08	Update utilities must provide logs for debug and record keeping via file or standard out/error redirect	Prefer XML
2.09	Update utilities must complete update in a reasonable time (< 2 minute preferred, < 10 minute required)	
2.10	Update utilities must allow upgrade from any previous version with only a reboot required (no power cycles or intermediate versions required)	

17.3 Diagnostic Requirements

The OCS Operations Toolkit command 'View-WcsHealth' must identify defective components (HDD, DIMM, processors, etc.) with a high degree of confidence. This requires meeting the requirements in the table below.

ID	Requirement	Comment
3.01	The OCS Operations Toolkit command 'View-WcsHealth' must identify defective components (HDD, DIMM, processors, etc.) with a high degree of confidence	
3.02	Error information must be available in the BMC SEL as defined in the WCS Software Blade API specification v2.0	
3.03	Details on how to decode OEM specific SEL entries must be provided	

3.04	Mapping of the component physical location to logical location must be provided • For example, HDD slot labels to Win32_DiskDrive SCSI properties • For example, DIMM silkscreen label to Win32_PhysicalMemory device locator	
3.05	Disk SMART information must be accessible via WMI win32_DiskDrive or another 3rd party utility	

17.4 Configuration Requirements

The OCS Operations Toolkit must be able to display, log, and compare the system configuration. This requires meeting the requirements in the table below.

ID	Requirement	Comment
4.01	The OCS Operations Toolkit commands View-WcsConfig, Log-WcsConfig, Compare-WcsConfig must correctly report the system configuration	
4.02	Methods must be available for determining all configuration and version information not available via Windows WMI (such as CPLD version)	

17.5 RAID Controller Requirements

The OCS Operations Toolkit must be able to manage the RAID controller (if used). This section has not been finalized.

ID	Requirement	Comment
5.01	The system must allow the flushing the RAID cache using the 'Flush-WcsRaidCache' command	
5.02	Support additional RAID management commands as they are defined	