Leveraging PowerShell 2.0 with Dell OpenManage Essentials

This Dell Technical White Paper provides step-by-step instructions to configure OpenManage Essentials and PowerShell 2.0 for running scripts against remote hosts.

Benjamin Villanueva
Dell Enterprise Product Group
Contents
Executive summary ........................................................................................................................................... 4
Introduction ................................................................................................................................................ 4
Prerequisites .............................................................................................................................................. 5
  Install OpenManage Essentials .............................................................................................................. 5
  Install Windows PowerShell 2.0 ................................................................................................................ 5
  Discover and Inventory your Environment ............................................................................................. 5
  Configure PowerShell 2.0 ....................................................................................................................... 5
  PowerShell Execution policy .................................................................................................................. 6
  Enable-PSRemoting .................................................................................................................................. 6
Creating Custom Device Groups ................................................................................................................ 7
PowerShell 2.0 Scripts ................................................................................................................................ 9
  Configuring iDRAC for Active Directory Authentication ....................................................................... 9
  Installing third-party applications remotely ........................................................................................ 14
Conclusion .............................................................................................................................................. 17

Figures
Figure 1. iDRAC Query ................................................................................................................................. 8
Figure 2. Windows 2008 Query .................................................................................................................... 9
Figure 3. Create a Command Line Task ..................................................................................................... 12
Figure 4. Execution Details ....................................................................................................................... 13
Figure 5. System Summary ........................................................................................................................ 13
Figure 6. Create a Command Line Task - General tab ........................................................................... 16
Figure 7. Deploy MSI task .......................................................................................................................... 17
Executive summary

Dell OpenManage Essentials (OME) is an essential tool for administrators to assist in managing their environment. The benefits of using OME against your environment include:

- Leverages the scripting capabilities within the product
- Targets one-to-many configurations
- Targets only specific hosts or devices groups

This white paper outlines the prerequisites to configure and leverage OME in order to take advantage of Microsoft PowerShell 2.0.

Example scripts that demonstrate the capabilities of OME and PowerShell 2.0 include:

- Configuring iDRAC for Active Directory authentication
- Deploying software packages (MSI) to Windows Servers

Introduction

Systems Management is a time-consuming and complex process, especially when dealing with multiple different generations of servers and multiple remote sites. Leveraging OME to facilitate some of the processes can increase the productivity and flexibility offered by having your entire environment in a single place. The ability to target and select only the devices you want to manage are as simple as checking a box.

This white paper demonstrates how you can leverage this ability and demonstrate two possible use cases:

1. Automate the configuration of your iDRAC6 and above devices for Active Directory authentication using a PowerShell script
2. Deploy third-party applications MSI using PowerShell

Combine these two items into common administrative task reduces the time and complexity of configuring and deploying elements to your network.

This white paper focuses on the new features that are introduced in Windows PowerShell 2.0. It focuses on the benefits of running PowerShell’s Remoting cmdlet, which lets you run commands on one or more remote computers from a single computer; in this case, OME server.
Prerequisites

Windows PowerShell 2.0 is a command-line shell and scripting language that is designed for system administration and Automation. Built on the Microsoft .NET Framework, Windows PowerShell enables IT professionals and developers to control and automate the administration of Windows and applications.

The Windows PowerShell includes an interactive prompt and a scripting environment that can be used independently or in combination. This fundamental change in the environment brings entirely new tools and methods to the management and configuration of Windows.

By combining the advantages of Windows PowerShell 2.0 and OpenManage Essentials (OME), Dell created an essential tool for administrators to assist in managing their environment.

Install OpenManage Essentials

OpenManage Essentials is a lightweight, Web-based, one-to many systems management solution that provides a comprehensive view of Dell systems, devices, and components in an enterprise network.

Prior to running these example scripts, you need to install OME on Windows 2008 R2. You can find a detailed white paper with the steps to install OME at the following location:

http://en.community.dell.com/techcenter/extras/m/white_papers/19998614.aspx

Install Windows PowerShell 2.0

Windows 2008 R2 has PowerShell 2.0 built-in by default. There are no additional tasks needed on the 2008 R2 operating system, but you need to download and install prior versions of Windows and their corresponding package from http://support.microsoft.com/kb/968930.

Discover and Inventory your Environment

A fundamental element of any Device Management is discovery and inventory of the devices an organization is looking to manage. Discovery needs to be non-invasive, easy to administer, efficient, thorough, accurate, broad in scope and responsive to network changes. Discovery is a process for identifying all Dell hardware devices in your network, such as Dell PowerEdge Servers and Dell Out-of-Band devices (iDRAC). To understand and configure OME to discover and inventory your environment, review the following white paper:

http://en.community.dell.com/techcenter/extras/m/white_papers/19998626.aspx

Configure PowerShell 2.0

There are two prerequisite configuration tasks that are needed on Windows 2008 R2, OME and the Remote Hosts in order to leverage PowerShell 2.0.
**PowerShell Execution policy**

The Set-ExecutionPolicy cmdlet changes the user preference for the Windows PowerShell execution policy.

The execution policy is part of the security strategy of Windows PowerShell. It determines whether you can load configuration files (including your Windows PowerShell profile) and run scripts, and it determines which scripts, if any, must be digitally signed before they will run. For more information, see about_Execution_Policies.

**NOTE:** In Windows Vista, Windows Server 2008, and later versions of Windows, to change the execution policy for the default (LocalMachine) scope, start Windows PowerShell with the “Run as administrator” option.

**Parameters**
- `-ExecutionPolicy <ExecutionPolicy>`
  Specifies the new execution policy. Valid values are:

  - **Restricted**: Does not load configuration files or run scripts. "Restricted" is the default execution policy.
  - **AllSigned**: Requires that all scripts and configuration files be signed by a trusted publisher, including scripts that you write on the local computer.
  - **RemoteSigned**: Requires that all scripts and configuration files downloaded from the Internet be signed by a trusted publisher.
  - **Unrestricted**: Loads all configuration files and runs all scripts. If you run an unsigned script that was downloaded from the Internet, you are prompted for permission before it runs.
  - **Bypass**: Nothing is blocked and there are no warnings or prompts.
  - **Undefined**: Removes the currently assigned execution policy from the current scope. This parameter does not remove an execution policy that is set in a Group Policy scope.

For the purpose of this white paper, the execution policy is set to **RemoteSigned**.

**Enable-PSRemoting**

The Enable-PSRemoting cmdlet configures the computer to receive Windows PowerShell remote commands that are sent by using the WS-Management technology.

You only need to run the command “Enable-PSRemoting” once on each computer that receives commands. You do not need to run it on the OME server because it only sends commands. It is best to run it only where it is needed because the configuration activates listeners.
The Enable-PSRemoting cmdlet performs the following operations:

- Runs the Set-WSManQuickConfig cmdlet, which performs the following tasks:
  - Starts the WinRM service.
  - Sets the startup type on the WinRM service to Automatic.
  - Creates a listener to accept requests on any IP address.
  - Enables a firewall exception for WS-Management communications.
- Enables all registered Windows PowerShell session configurations to receive instructions from a remote computer.
  - Registers the "Microsoft.PowerShell" session configuration, if it is not already registered.
  - Registers the "Microsoft.PowerShell32" session configuration on 64-bit computers, if it is not already registered.
  - Removes the "Deny Everyone" setting from the security descriptor for all the registered session configurations.
  - Restarts the WinRM service to make the preceding changes effective.

Note: For more information please visit about Set-ExecutionPolicy and Enable-PSRemoting, go to:

Creating Custom Device Groups

This example creates two customer device groups within OME.

Figure 1 is a custom group of only iDRAC devices that you can configure for Active Directory authentication.

For the iDRAC-ADConfig groups, you need to navigate from the OME home portal, click on Manage > Device > Device Search.

To configure the device search:

1. Create query to look for Device Type is RAC and Device Model contains iDRAC.
2. Click Run Query to display results.
3. Name query: iDRAC-ADConfig
4. Click on Save Query for future reference.
Figure 2 is a customer group of select Windows servers to install a specific third-party application using an MSI file.
5. Create query to look for **OS Name contains 2008 R2** and **Device Type is Server**.
6. Click **Run Query** to display results.
7. Name query: **Win2k8 Servers**
8. Click on **Save Query** for future reference.

You can create the needed scripts now that that the two queries are configured.

**PowerShell 2.0 Scripts**

Remote management typically uses either a fan-out (one-to-many) or fan-in (many-to-one) configurations. In the fan-out configuration (most common) administrators on a single local management console OME can run Windows PowerShell 2.0 commands to manage many remote systems simultaneously.

The remote management features are based on Windows Remote Management (WinRM), the Microsoft implementation of the Web Services for Management (WS-Man) protocol, which is designed to provide secure encrypted communications between the management console application and the managed system. This Web services basis for the WS-Man protocol enables remote sessions to work through firewalls that pass standard HTTP over Secure Sockets Layer (HTTPS) protocols, allowing administrators to manage systems across the Internet.

**Configuring iDRAC for Active Directory Authentication**

Integrated Dell Remote Access Controller6 (iDRAC6) is a systems management hardware and software solution that provides remote management capabilities, crashed system recovery, and power control functions for the Dell PowerEdge systems.
The iDRAC6 uses an integrated System-on-Chip microprocessor for the remote monitor/control system. The iDRAC6 co-exists on the system board with the managed PowerEdge server. The server operating system is concerned with executing applications; the iDRAC6 is concerned with monitoring and managing the server’s environment and state outside of the operating system.

The iDRAC6 network interface is enabled with a static IP address of 192.168.0.120 by default. It must be configured before the iDRAC6 is accessible. After the iDRAC6 is configured on the network, it can be setup for Active Directory authentication.

Each iDRAC requires a Root CA certificate for authentication. The following steps allow you to export the Root CA certificate:

1. Locate the domain controller that is running the Microsoft Enterprise CA service.
2. Click Start > Run.
3. In the Run field, type `mmc` and click OK.
4. In the Console 1 (MMC) window, click File (or Console on Windows 2000 machines) and select Add/Remove Snap-in.
5. In the Add/Remove Snap-in window, click Add.
6. In the Standalone Snap-In window, select Certificates and click Add.
7. Select Computer account and click Next.
8. Select Local Computer and click Finish.
9. Click OK.
10. In the Console 1 window, expand the Certificates folder, expand the Personal folder, and click the Certificates folder.
11. Locate and right-click the root CA certificate, select All Tasks, and click Export...
12. In the Certificate Export Wizard, click Next, and select No do not export the private key.
13. Click Next and select Base-64 encoded X.509 (.cer) as the format.
14. Click Next and save the certificate to a directory on your system (c:\temp).

In addition to the Root CA certificate, make sure that your domain controllers are set up for LDAPS. If you are using Microsoft Enterprise Root CA, all your domain controllers get assigned a “Domain Controller certificate”.

The following steps are needed to run the PowerShell script automatically to configure the iDRAC device group created earlier for Active Directory authentication.

The script:

--------Begin Script---------
function RACActiveDirectory ($Hosts)
{
    #Change to path where racadm.exe is located, default location in Windows 2008 R2 with OME
    $RacDir = "&`"C:\Program Files (x86)\Dell\SysMgt\rac\Sracedm.exe`"

--------End Script---------
# Change username and password for the ones configured on your iDRAC
$UserPass = "-u root -p calvin"

# Change your Domain to match your environment
$MyDomain = "delldemo.local"

# Change the AD group that will have admin rights on iDRAC
$RacGroup = "RACAdmins"

# Commands to get all AD information from DHCP
$RacCommands = @(
    "-g cfgActiveDirectory -o cfgADEnable 1",
    "-g cfgActiveDirectory -o cfgADType 2",
    "-g cfgActiveDirectory -o cfgADDcSRVLookupEnable 1",
    "-g cfgActiveDirectory -o cfgADDcSRVLookupDomainName $MyDomain",
    "-g cfgActiveDirectory -o cfgADGsRcSRVLookupEnable 1",
    "-g cfgActiveDirectory -o cfgADGsRcRootDomain $MyDomain",
    "-g cfgActiveDirectory -o cfgADCertValidationEnable 1",
    "-g cfgStandardSchema -i 1 -o cfgSSADRoleGroupName $RacGroup",
    "-g cfgStandardSchema -i 1 -o cfgSSADRoleGroupDomain $MyDomain",
    "-g cfgStandardSchema -i 1 -o cfgSSADRoleGroupPrivilege 0x000001ff",
    "-g cfgLANNetworking -o cfgDNSServersFromDHCP 1",
    "-g cfgLANNetworking -o cfgDNSDomainNameFromDHCP 1",
    "-g cfgUserDomain -o cfgUserDomainName -i 1 $MyDomain"
)

# Export Root Certificate as Base-64 place in directory below
$RootCert = "C:\temp\Root.cer"

$CertCommand = "-t 2 -f c:\temp\Root.cer"

$FullCommand = $RacDir + " -r " + $Hosts + " -u $UserPass"

foreach ($element in $RacCommands) {
    $Run = $FullCommand + " config " + $element
    Write-Host $Run
    Invoke-Expression $Run
}

$UploadCert = $FullCommand + " sslcertupload " + $CertCommand
Write-Host $UploadCert
Invoke-Expression $UploadCert

RACActiveDirectory ($args[0]);

---------End Script---------

1. Save the file as `idrac-ad.ps1` to the `c:\temp` directory.
2. Start the OME console and navigate to Manage > Remote Tasks.
3. Right click on the Command Line and select Create Command Line Task.
5. Name the task: `iDRAC AD`
6. In the Command, type: `powershell.exe`
7. In the arguments, type: `c:\temp\idrac-ad.ps1 $IP`
   **Note:** `$IP` is used to be able to select a Task Target.
8. Check Output to file.
9. Name the file: `c:\temp\idrac-ad.txt` and check Append and Include errors.

Figure 3 demonstrates how the window should appear.
10. Click **Next**.
11. Choose **Select a query** and select **iDRAC-ADConfig**.
12. Click **Next**.
13. Select **Run now**.
14. Enter the Username and Password for the OME Administrator.
15. Click **Finish**.
16. Verify that the task completed successfully by right clicking the task under **Task Execution History** and selecting **Details**

The successful results are displayed below:
17. You can also review the `c:\temp\idrac-ad.txt` file for the results.

18. Verify that you can log in to your iDRAC with your Active Directory credentials.

19. Verify that you are an Admin on the iDRAC.

For more information on the capabilities of the iDRAC and additional information, go to:

http://support.dell.com/support/systemsinfo/documentation.aspx
Installing third-party applications remotely

One of the most exciting and important features of PowerShell 2.0 is the remoting capability. PowerShell remoting enables management of computers from a remote location.

To be able to run scripts and commands on remote computers, the user performing remote scripts must be a member of the administrator group on the remote machine OR should be able to provide administrator credentials at the time of remote execution.

The following example demonstrates how to install third-party MSI applications to a remote collection of Windows 2008 servers.

Within remoting, there are a couple of ways to run commands or scripts on a remote machine. This includes the “Invoke-Command” cmdlet and interactive remoting sessions. This white paper focuses on the “Invoke-Command”.

To invoke a command on a local or remote computer, you can use the following method:

```
Invoke-Command -ComputerName Server1 -ScriptBlock {Get-Process}
```

All commands and variables within the ScriptBlock are evaluated on the remote computer. In the above command -ScriptBlock {Get-Process}, PowerShell returns the results of the processes running on Server1.

The following example assumes that you have access to remote computer as an administrator. This method works well in a domain environment where the logged on user has administrator credentials to access any computer in the domain.

This Invoke-Command method along with OME will be used to demonstrate how a one-to-many script can be used to deploy a specific MSI to a collection of servers.

The script below should be copied in the c:\temp directory:

```powershell
--------Begin Script---------
function DNSQuery ($IP)
{
    if ($IP -match "^\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}$")
    {
        $result = [System.Net.Dns]::gethostentry($IP).HostName
    } else
    {
        write-host -fore Red "Invalid IP Address"
        break
    }
    $result
}
function DeployMSI
{
    param ([Parameter (Mandatory = $true)] $server)
    try {
        $errorActionPreference = "Stop"
        $stemppath = Test-Path \$server\C$\Sourcepath
        if ($stemppath -eq 'TRUE')
        {
            $sourcepath = ''c:\temp\7zip.msi"
            $destpath = "\$server + "C:\temp"
            copy-item $sourcepath -destination $destpath
        }
        catch {
            write-host -fore Red "Failed to copy MSI to $server, check MSI path"
        return $false
    }
    write-host -fore Green "Successfully copied MSI to $server" $true
}
--------End Script--------
```
Leveraging PowerShell 2.0 with Dell OpenManage Essentials

```powershell
function RunMSI
{
    param([Parameter(Mandatory = $true)] $server)
    try {
        $errorActionPreference = "Stop"
        $script = "{ARGS = "-i c:\temp\7zip.msi /qn /lv c:\temp\OME_MSI.log"
            [diagnostics.process]::start("msiexec.exe", $ARGS).WaitForExit() }"
        Invoke-command -computername $server -scriptblock $script
        Write-host -fore Green "Successfully installed MSI on $server"
    } catch {
        write-host -fore Red "Unable to run MSI check credentials and paramaters"
        return $false
    }
    $true
}
$server = DNSQuery ($args[0])
DeployMSI $server
RunMSI $server

-----------End Script-----------

Steps to implement the above script are:

1. Save the above script as deploy-msi.ps1 to the c:\temp directory.
2. Start the OME console and navigate to Manage > Remote Tasks.
3. Right click on Command Line and select Create Command Line Task.
4. Select Generic Command
5. Name the task: Deploy MSI
6. In the Command, type: powershell.exe
7. In the arguments, type: c:\temp\deploy-msi.ps1 $IP
   Note: $IP is used to be able to select a “Task Target”
8. Check Output to file.
9. Name the file: c:\temp\deploy.txt
10. Check Append and Include Errors
    Figure 6 demonstrates how the window should appear.
11. Click Next.

12. Choose Select a query and select the previously “Saved Query” called Win2k8 Servers.

13. Click Next.

14. Select Run Now.

15. Enter Username and Password for the OME Administrator.

16. Click Finish.

17. Verify that the task completed successfully by right clicking the task under Task Execution History and selecting Details

Figure 7 demonstrates how the window should appear.
18. On the local host, you can review the log file “deploy.txt” that was created in the c:\temp directory.

19. On the remote host, you can review the log file “OME_MSI.log” that was created in the c:\temp directory.

You can also use this example script to leverage any additional administrative task on remote host required in your environment.

Conclusion

The Microsoft Windows PowerShell 2.0 features in Windows 2008 R2 provide a flexible way to access the rich management features of Dell OpenManage Essentials. Combining Windows PowerShell with OME enables administrators to simplify and streamline remote systems management of Dell PowerEdge servers 11th-generation and beyond.