Automated SQL Server installation and configuration using PowerShell

Jamie Wick

T: @Jamie_Wick

E: sql@wicktech.net

#470 | WASHINGTON DC 2015

SQLSaturday RVA

PASS SQL saturday

#486 | RICHMOND 2016

http://www.sqlsaturday.com/486

March 19, 2016 in Richmond, VA

Richmond SQL Server Users Group

http://rva.sqlpass.org/

Second Thursday at Markel Plaza (Glen Allen)

RichmondSQL™

Everything you may (or may not) want to know about me.

Systems & Database Engineer

The College Of WILLIAM&MARY

Masters of Science in Computer Information Systems

BOSTON UNIVERSITY

SysAdmin in Brisbane, Australia for ~5 years.

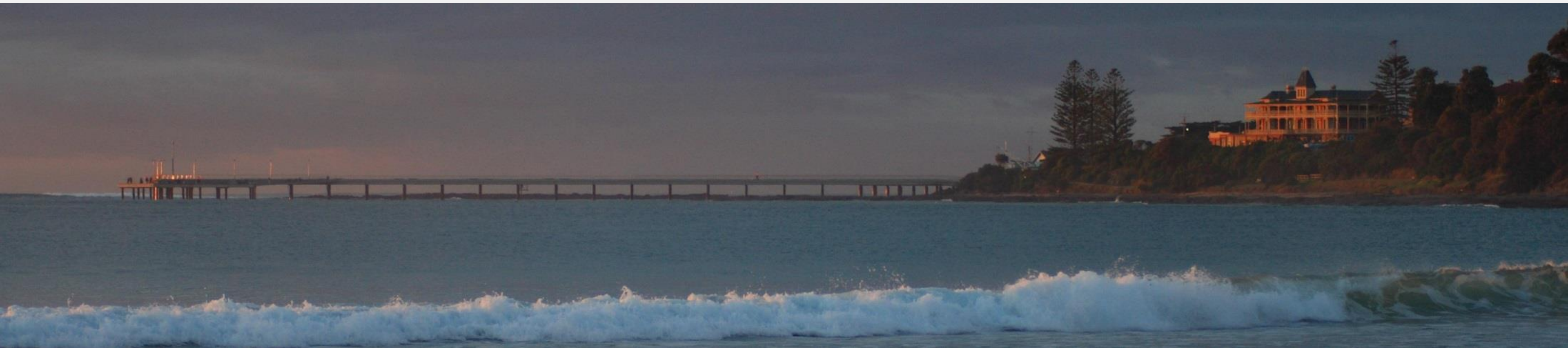
THE UNIVERSITY OF QUEENSLAND AUSTRALIA

Assorted certifications in MS SQL Server 2005 & 2008

Photo Junkie

Agenda

- SQL Server (setup)
- PowerShell Basics
  Syntax
  Commandlets
- Creating PowerShell Scripts
- Scripts 101
  Prepare OS & install SQL
  Post install configurations



Installation & Configuration of SQL Server

(what we’re trying to do…)

SQL Server Setup

Question:

What goes into all of your SQL installs?

or

What is **supposed** to go into all of your SQL installs?

Answer:

Umm….

*Let me go find that **Checklist** from a couple years ago.*

SQL Setup Checklist

This looks familiar…

<u>Requirements for SQL installs</u>

- .NET 3.5 feature
- SQL configuration file (for unattended installs)
  - C:\Program Files\Microsoft SQL Server\...\ConfigurationFile.ini
  - QS-Config
- Firewall exceptions

Port	Use
TCP 1433	Database Engine default install
UDP 1434	SQL Server Browser Service
TCP 1434	Dedicated Admin Connection (DAC)
TCP 80 & 443	HTTP/HTTPS for SSRS
TCP 139 & 445	SQL Filestream & Filetables
TCP 2383	Analysis Services
TCP 135	MS Distributed Transaction Coordinator

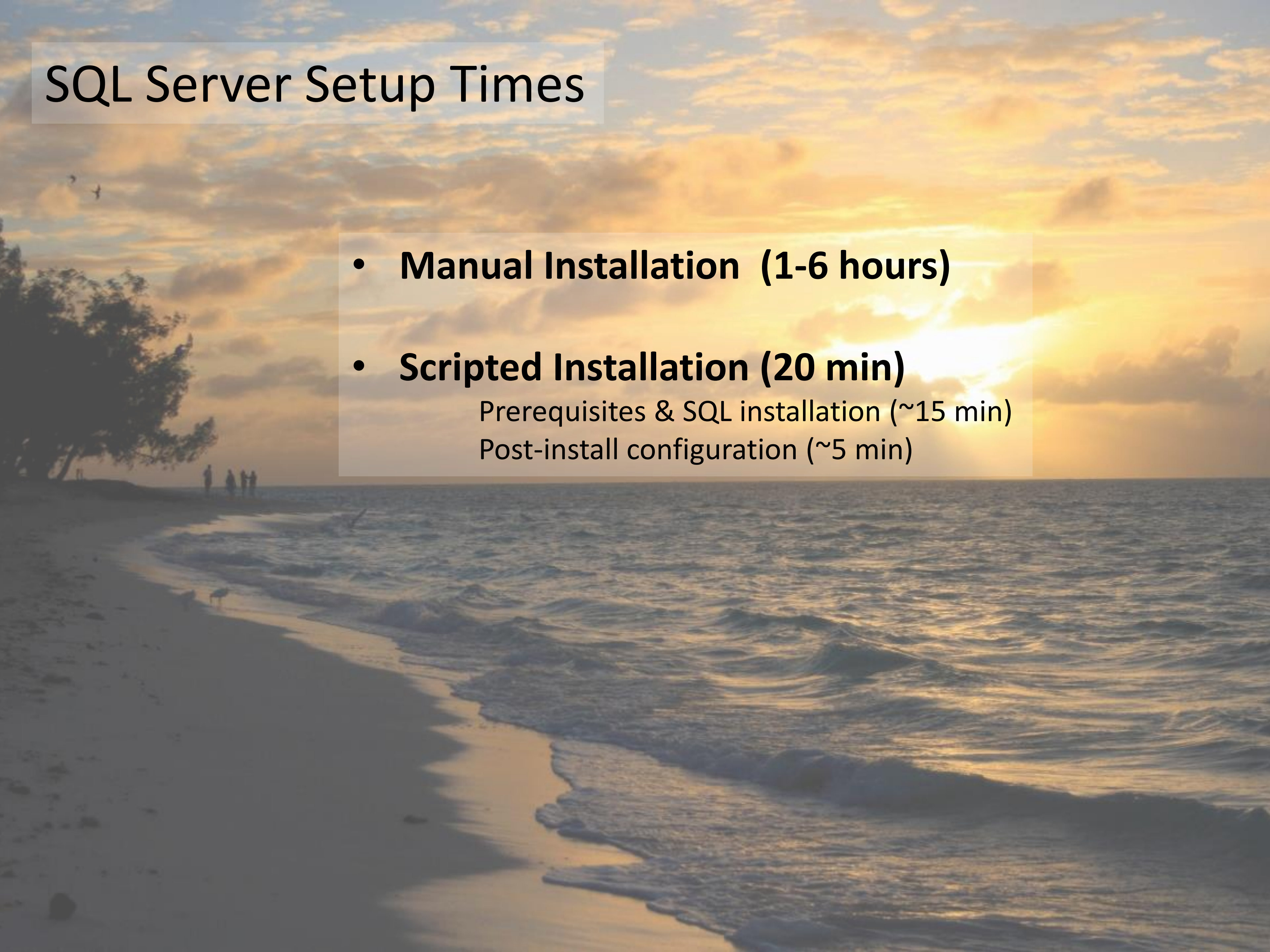
SQL Setup Checklist cont.

I remember some of this…

Some possible configurations

- Add admins group
- Rename the SA account
- Register Service Provider Name (SPN)
- Operators
- Enable the SQL Agent service
- DBMail
- Contained Database Authentication
- Enable CLR
- Import Custom SQL procedures
- Create Linked Servers
- Send audits to Server Security Log
- Add TempDB files
- Set Server Memory usage
- Set default backup path
- Backup Jobs
- Stats updates
- Index maintenance (Ola Hallengren)
- Cleanup Jobs

- Alerts
  - Severity 19-25
  - ID 18456 - Login Error
  - ID 15247 - Permission Error
  - ID 9002 - Log Full
  - ID 825 - Disk I/O Error
  - New db email notification
  - Low disk space
  - High VLF count
- Management Policies
  - Last Backup
  - Permissions
  - Data/Log File Locations

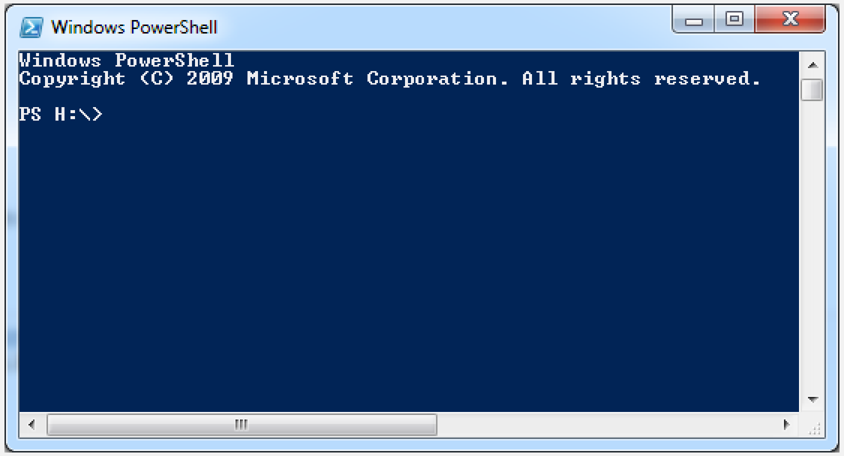


SQL Server Setup Times

- **Manual Installation (1-6 hours)**
- **Scripted Installation (20 min)**
  - Prerequisites & SQL installation (~15 min)
  - Post-install configuration (~5 min)

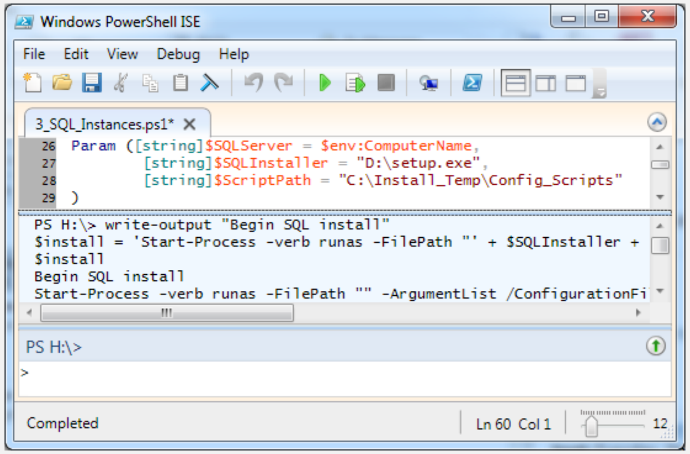
about PowerShell…

- aka PoSh
- Admin tool with Shell and Integrated Scripting Environment (ISE)



```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS H:\>
```



```
Param ([string]$SQLServer = $env:ComputerName,
        [string]$SQLInstaller = "D:\setup.exe",
        [string]$ScriptPath = "C:\Install_Temp\Config_Scripts"
)
```

```
PS H:\> write-output "Begin SQL install"
$install = 'Start-Process -verb runas -FilePath "' + $SQLInstaller +
$install
Begin SQL install
Start-Process -verb runas -FilePath "" -ArgumentList /ConfigurationFi
```

```
PS H:\>
>
```

- SQL Server module for PowerShell (SQLPS)
  - Invoke-SQLcmd
  - SQL Management Objects (SMO)
- Supported in many products by Microsoft and other vendors:

Windows Server	Cisco
Exchange	VMWare
Active Directory	Citrix
Sharepoint	Quest (Dell Software)
SQL	

Powershell Basics:

Syntax:

$xxxxxxx – variable name

[string]$xxxx - variable with string data type

“ (double quote) - string expands/replaces variables

‘ (single quote) - literal string (does not expand variables)

@” - here string (multi-line string w/ variable replacement)

# - comment out remainder of line

` (back tick) - escape character (interprets next character as literal)

{} - script block

$_ - current object (used in iterative loops, eg. For Each)

Powershell Basics: (cont.)

Commandlets: (*Verb-noun naming convention)*

Set-ExecutionPolicy [*Restricted, AllSigned, RemoteSigned, Unrestricted]*

Enable-PSRemoting –force

Import-Module *[ServerManager, SQLPS]*

Invoke-Command -Computername -ScriptBlock {}

Invoke-Sqlcmd -ServerInstance *[-Query or -InputFile]*

Invoke-Expression *[$variable]*

Start-Process

Write-Output

Powershell Scripts:

- .ps1 (script)
  PowerShell equivalent of a .bat or .cmd file
- .psm1 (script module), .psd1 (manifest file), .ps1xml (formatting file)
  (Advanced topics we aren't covering)
- Double-click <> Execute
- Set-ExecutionPolicy to allow unsigned scripts to be run
- Parameters

```
Param ([string]$ServerName = ".",
       [string]$Path = "C:\")

.\script.ps1 -ServerName "SQL01" -Path "C:\temp"
```

Creating PowerShell Scripts:

Design Considerations

- Execute script(s) locally or remote?
  - Locally – some Windows configs cannot be changed remotely (*secpol.msc*)
  - Remote – run script(s) from a single repository
- Single Script or Script Set?
  - Single - 1 big script containing all configuration items
  - Set - 1 primary script that calls sub scripts
- Script Users
  - Internal use – doesn't have to be PERFECT, document for co-workers usability
  - External use – better documentation, allow for environment variablility

Script Examples

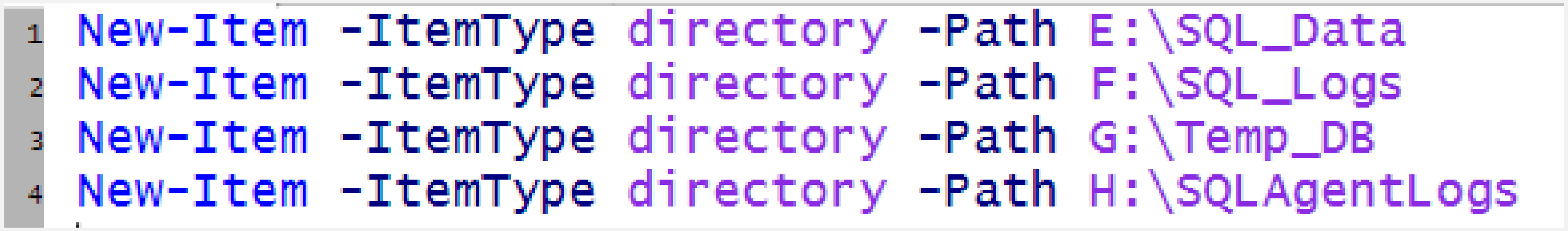
Warning:

The following 38 pages are a little dry.

Relax there are only 6…

Installing SQL Server:

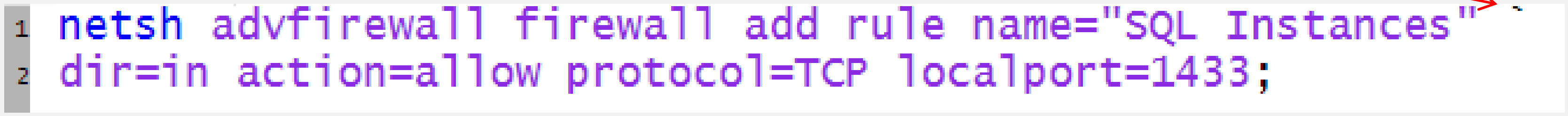
Create Directories



```
New-Item -ItemType directory -Path E:\SQL_Data
New-Item -ItemType directory -Path F:\SQL_Logs
New-Item -ItemType directory -Path G:\Temp_DB
New-Item -ItemType directory -Path H:\SQLAgentLogs
```

Open Firewall Ports

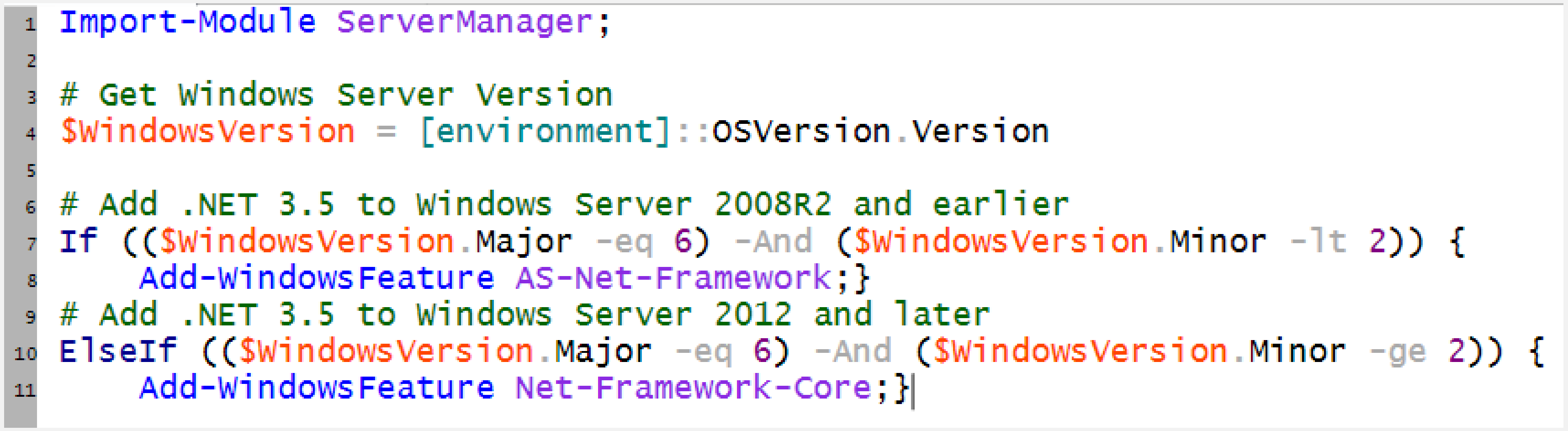
Back-tick for line continuation



```
netsh advfirewall firewall add rule name="SQL Instances" `
dir=in action=allow protocol=TCP localport=1433;
```

Installing SQL Server: (cont.)

Install .NET Feature



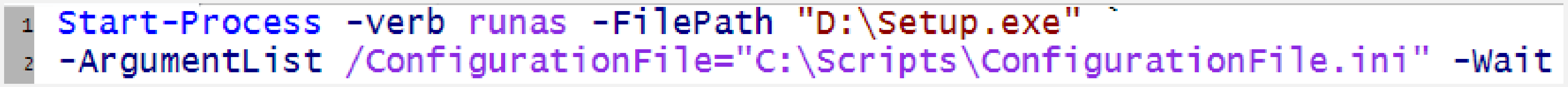
```
Import-Module ServerManager;

# Get Windows Server Version
$WindowsVersion = [environment]::OSVersion.Version

# Add .NET 3.5 to Windows Server 2008R2 and earlier
If (($WindowsVersion.Major -eq 6) -And ($WindowsVersion.Minor -lt 2)) {
    Add-WindowsFeature AS-Net-Framework;}
# Add .NET 3.5 to Windows Server 2012 and later
ElseIf (($WindowsVersion.Major -eq 6) -And ($WindowsVersion.Minor -ge 2)) {
    Add-WindowsFeature Net-Framework-Core;}
```

Note: Install media location option ***-Source D:\Sources\SxS\***

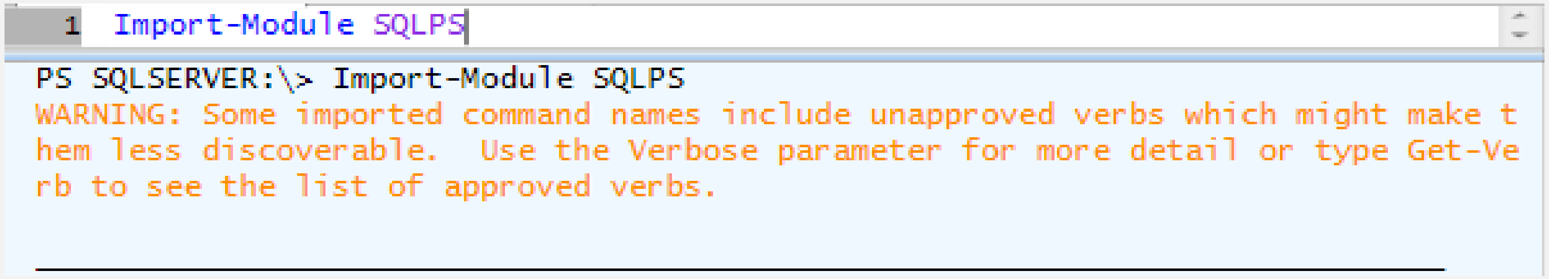
Run SQL install as administrator, specifying config file



```
Start-Process -verb runas -FilePath "D:\Setup.exe" `
-ArgumentList /ConfigurationFile="C:\Scripts\ConfigurationFile.ini" -Wait
```

Configuring SQL Server:

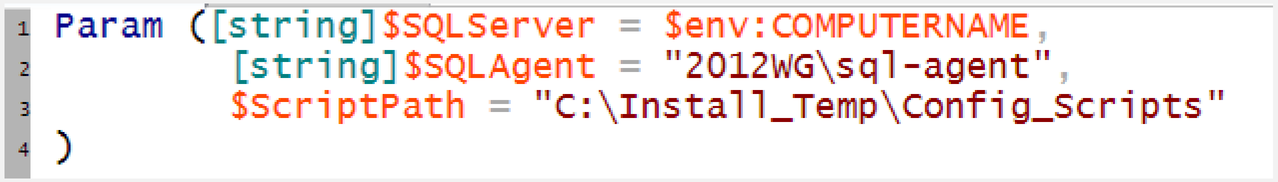
Import SQLPS module



```
Import-Module SQLPS
```

```
PS SQLSERVER:\> Import-Module SQLPS
WARNING: Some imported command names include unapproved verbs which might make t
hem less discoverable.  Use the Verbose parameter for more detail or type Get-Ve
rb to see the list of approved verbs.
```

Set Script Parameters

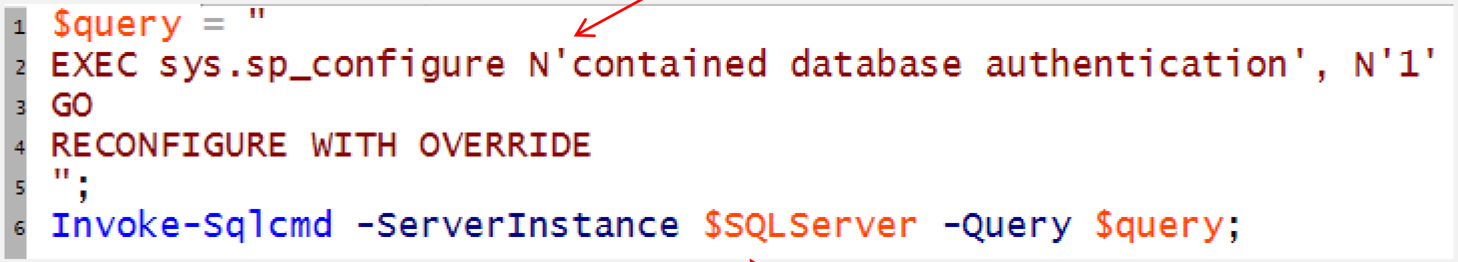


```
Param ([string]$SQLServer = $env:COMPUTERNAME,
       [string]$SQLAgent = "2012WG\sql-agent",
       $ScriptPath = "C:\Install_Temp\Config_Scripts"
)
```

Configuring SQL Server: (cont.)

Execute SQL query

Double-quote to contain single quotes



```
$query = "
EXEC sys.sp_configure N'contained database authentication', N'1'
GO
RECONFIGURE WITH OVERRIDE
";
Invoke-Sqlcmd -ServerInstance $SQLServer -Query $query;
```

Parameter variable for server instance

Configuring SQL Server: (cont.)

Execute SQL script file

Back-tick

```
Invoke-Sqlcmd -ServerInstance $SQLServer `
-InputFile "C:\Install\Alerts_Triggers.sql" -Verbose
```

Execute SQL script file using $ScriptPath parameter

Back-tick

```
$query = 'Invoke-Sqlcmd -ServerInstance ' + $SQLServer + `
' -InputFile "' + $ScriptPath + '\Alerts_Triggers.sql" -Verbose';
Invoke-expression $query;
```

Invoke-expression allows execution of dynamically created parameter

Configuring SQL Server: (cont.)

Execute PowerShell script file

```
. C:\Install\Config_Scripts\Full_Backup_Job.ps1 -SQLServer $SQLServer;
```

Period = “Do This”

Execute PowerShell script file using $ScriptPath parameter

```
$path = Join-Path $ScriptPath "\Full_Backup_Job.ps1";
. $path -SQLServer $SQLServer;
```

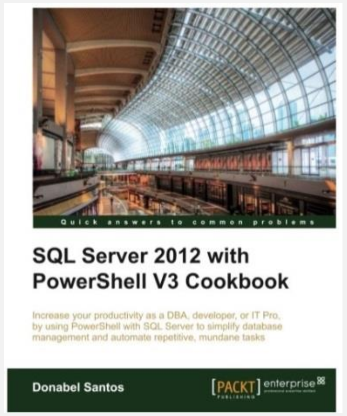
Passes $SQLServer value as parameter to .ps1

Demo Scripts:

Additional Resources:

PASS POWERSHELL VIRTUAL CHAPTER

http://powershell.sqlpass.org/


Quick answers to common problems
SQL Server 2012 with PowerShell V3 Cookbook
Increase your productivity as a DBA, developer, or IT Pro, by using PowerShell with SQL Server to simplify database management and automate repetitive, mundane tasks
Donabel Santos
[PACKT] enterprise

SQL Server 2012 with PowerShell V3 Cookbook

by Donabel Santos

Additional Resources:

QS Config
http://www.sqlhammer.com/blog/qs-config/

Microsoft SMO Programming Guide
http://technet.microsoft.com/en-us/library/ms162169.aspx

Questions?

T: @Jamie_Wick
E: sql@wicktech.net