# Discuss PowerShell basics

#

# Get-PowerGUI -website http://www.powergui.org -now $true

# Get-SCCMcmdlets -webiste http://www.snowland.se/sccm-posh/ -now $true -really YES -whatspowershell Um..back..away..from..the..machine

#

# Demo downloading script. http://powershellzip.codeplex.com/ NEED TO MODIFY ZIP PROPERTIES BEFORE UNZIPPING!

#

# Ready?????

#

# One of the first things I did when I was building the SCCM environment was to ensure that all the

# ADSites that I wanted to manage were included. But, how to get those into a list...

[System.DirectoryServices.ActiveDirectory.Forest]::GetCurrentForest().Sites | **Foreach-Object** {$\_.Subnets} | **Export-Csv** c:\adsites.csx

# OK..onto the cool...."get the cool....get the cool shoe shine"

# Randomize Collection members. What if we have a business unit that only wants a certain amount of machines to run an Advertisement each night?

#

# Find the Top Level Collection that we want to randomize

$var = Get-SCCMCollection *-SccmServer* $SCCM | **Where-Object** {$\_.CollectionID -eq 'CEN004E9'}

$var | **gm**

Get-SCCMCollectionRules *-SccmServer* $SCCM *-CollectionID* 'CEN004E9'

Get-SCCMCollectionMembers *-SccmServer* $SCCM *-CollectionID* 'CEN004E9' | **gm**

Get-SCCMCollectionMembers *-SccmServer* $SCCM *-CollectionID* 'CEN004E9' | **Format-Table** name

# Or, perhaps we want to place the machinenames into a CSV file for review

Get-SCCMCollectionRules *-SccmServer* $SCCM *-CollectionID* 'CEN004E9' | **select-object** name | **Export-Csv** I:\iHeartTechFuseMN.csv

# Remember, if you are going to export stuff, you must use the Select-Object cmdlet (instead of the Foramt cmdlets)

# to specify the properties that you want.

#

# Create 10 subcollections from Parent Collection CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 1" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 2" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 3" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 4" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 5" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 6" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 7" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 8" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 9" *-parentCollectionID* CEN004E9

New-SCCMCollection *-SccmServer* $SCCM *-name* "Push 0" *-parentCollectionID* CEN004E9

# Wow...that was easy..um...duh..PowerShell...WINNING

# You want WINNING? How does 10 lines into one line sound? Does it sound like WINNING

foreach ($i in 0..9) {New-SCCMCollection *-SccmServer* $SCCM *-name* "Push $i" *-parentCollectionID* CEN004EA}

# Create collection rules for each of the above collections

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 1"}).collectionID)*-name* One *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%1'" *-limitToCollectionId* CEN004E9 *-queryRuleName* one

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 2"}).collectionID) *-name* Two *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%2'" *-limitToCollectionId* CEN004E9 *-queryRuleName* two

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 3"}).collectionID)*-name* Three *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%3'" *-limitToCollectionId* CEN004E9 *-queryRuleName* three

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 4"}).collectionID) *-name* four *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%4'" *-limitToCollectionId* CEN004E9 *-queryRuleName* four

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 5"}).collectionID) *-name* five *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%5'" *-limitToCollectionId* CEN004E9 *-queryRuleName* five

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 6"}).collectionID) *-name* six *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%6'" *-limitToCollectionId* CEN004E9 *-queryRuleName* six

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 7"}).collectionID) *-name* seven *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%7'" *-limitToCollectionId* CEN004E9 *-queryRuleName* seven

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 8"}).collectionID) *-name* eight *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%8'" *-limitToCollectionId* CEN004E9 *-queryRuleName* eight

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 9"}).collectionID) *-name* nine *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%9'" *-limitToCollectionId* CEN004E9 *-queryRuleName* nine

Add-SCCMCollectionRule *-SccmServer* $SCCM *-collectionID* ((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 0"}).collectionID) *-name* zero *-queryExpression* "select \* from SMS\_R\_System where SMS\_R\_System.ResourceID like '%0'" *-limitToCollectionId* CEN004E9 *-queryRuleName* zero

# Explain why we had to put double parenthesisisis around the -collectionID parameter

((Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 0"}).collectionID)

Get-SCCMCollection *-SccmServer* $SCCM | **where** {$\_.name -eq "Push 0"} | **select** CollectionID

# Wow, that was simply amazing. What esle can I do with the SCCM cmdlets

**Get-Command** *-Module* SCCM-Commands

# Oh...there is a cmdlet called get-sccmcomands. How about that?

Get-SCCMCommands

# Working with Packages and DP's

# Copy a certain package to a DP group

Add-SCCMDistributionPoint *-SccmServer* $SCCM *-DPGroupName* TEST *-DPPackageID* SMSxxxx

# When you add a new DP you have to copy all of the packages (that you want) INDIVIDUALLY to the DP's

Add-SCCMDistributionPoint *-SccmServer* $SCCM *-DPPackageID* SMSxxxx *-AllDPs*

# Or we could create an array of all of the packages of interest

$Packages = Get-SCCMPackage *-SccmServer* $SCCM | **where** {$\_.name -like '\*CCI\*'}

foreach ($i in $Packages.packageid) {Add-SCCMDistributionPoint *-SccmServer* $SCCM *-DPPackageID* SMSxxxx *-AllDPs*}

# Show how priority and bandwith settings work

# Advertisments

Get-SCCMAdvertisement *-SccmServer* $SCCM | **Select-Object** *-First* 1

# Properties of computer objects

Get-SCCMComputer *-SccmServer* $sccm *-ResourceID* 58581 | **select** \*

# Inboxes. Really? You mean I don't have to drill into each inbox anymore? No, that is not waht I mean. Wait, I guess that is what I mean!

Get-SCCMInboxes *-SccmServer* $SCCM

# list of servers in your SCCM environment

Get-SCCMSite *-SccmServer* $SCCM | **ft** servername, reportingsitecode, sitecode, sitename, version

# Cool. What else can we do with Powershell and SCCM.

# Add MPLS.SCCM.EUC to local administrators group on all SCCM servers

#$ErrorActionPreference = "SilentlyContinue"

$account = "MPLS.SCCM.EUC"

#$SCCMservers = get-wmiobject -namespace "root/SMS/Site\_CEN" -Query "Select ServerName from SMS\_Site"

#$servers = foreach ($i in $SCCMservers) {$i.servername}

$servers = "SWMN00xb03237"

foreach ($i in $servers) {

$ErrorActionPreference = "SilentlyContinue"

$computer = [ADSI]("WinNT://" + $i + ",computer")

$Group = $computer.psbase.children.find("administrators")

$members= $Group.psbase.invoke("Members") | **%**{$\_.GetType().InvokeMember("Name", 'GetProperty', $null, $\_, $null)}

Foreach ($i in $members){if ($i -eq $account) {$AlreadyExist = $True}}

If ($AlreadyExist -ne $true) {$group.add("WinNT://amercarlson/MPLS.SCCM.EUC")}

**Remove-Variable** AlreadyExist

}