# PowerShell Systems Report

# Example usage: .\SystemsReport.ps1 .\list.txt

# Remember that list.txt is the file containing a list of Server names to run this against

#region Variables and Arguments

$users = "youremail@yourcompany.com" # List of users to email your report to (separate by comma)

$fromemail = "youremail@yourcompany.com"

$server = "yourmailserver.yourcompany.com" #enter your own SMTP server DNS name / IP address here

$list = $args[0] #This accepts the argument you add to your scheduled task for the list of servers. i.e. list.txt

$computers = get-content $list #grab the names of the servers/computers to check from the list.txt file.

# Set free disk space threshold below in percent (default at 10%)

$thresholdspace = 20

[int]$EventNum = 3

[int]$ProccessNumToFetch = 10

$ListOfAttachments = @()

$Report = @()

$CurrentTime = Get-Date

#endregion

Function Create-PieChart() {

param([string]$FileName)

[void][Reflection.Assembly]::LoadWithPartialName("System.Windows.Forms")

[void][Reflection.Assembly]::LoadWithPartialName("System.Windows.Forms.DataVisualization")

#Create our chart object

$Chart = New-object System.Windows.Forms.DataVisualization.Charting.Chart

$Chart.Width = 300

$Chart.Height = 290

$Chart.Left = 10

$Chart.Top = 10

#Create a chartarea to draw on and add this to the chart

$ChartArea = New-Object System.Windows.Forms.DataVisualization.Charting.ChartArea

$Chart.ChartAreas.Add($ChartArea)

[void]$Chart.Series.Add("Data")

#Add a datapoint for each value specified in the arguments (args)

foreach ($value in $args[0]) {

Write-Host "Now processing chart value: " + $value

$datapoint = new-object System.Windows.Forms.DataVisualization.Charting.DataPoint(0, $value)

$datapoint.AxisLabel = "Value" + "(" + $value + " GB)"

$Chart.Series["Data"].Points.Add($datapoint)

}

$Chart.Series["Data"].ChartType = [System.Windows.Forms.DataVisualization.Charting.SeriesChartType]::Pie

$Chart.Series["Data"]["PieLabelStyle"] = "Outside"

$Chart.Series["Data"]["PieLineColor"] = "Black"

$Chart.Series["Data"]["PieDrawingStyle"] = "Concave"

($Chart.Series["Data"].Points.FindMaxByValue())["Exploded"] = $true

#Set the title of the Chart to the current date and time

$Title = new-object System.Windows.Forms.DataVisualization.Charting.Title

$Chart.Titles.Add($Title)

$Chart.Titles[0].Text = "RAM Usage Chart (Used/Free)"

#Save the chart to a file

$Chart.SaveImage($FileName + ".png","png")

}

Function Get-HostUptime {

param ([string]$ComputerName)

$Uptime = Get-WmiObject -Class Win32\_OperatingSystem -ComputerName $ComputerName

$LastBootUpTime = $Uptime.ConvertToDateTime($Uptime.LastBootUpTime)

$Time = (Get-Date) - $LastBootUpTime

Return '{0:00} Days, {1:00} Hours, {2:00} Minutes, {3:00} Seconds' -f $Time.Days, $Time.Hours, $Time.Minutes, $Time.Seconds

}

#region HTML

# Assemble the HTML Header and CSS for our Report

$HTMLHeader = @"

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<html><head><title>My Systems Report</title>

<style type="text/css">

<!--

body {

font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;

}

#report { width: 835px; }

table{

border-collapse: collapse;

border: none;

font: 10pt Verdana, Geneva, Arial, Helvetica, sans-serif;

color: black;

margin-bottom: 10px;

}

table td{

font-size: 12px;

padding-left: 0px;

padding-right: 20px;

text-align: left;

}

table th {

font-size: 12px;

font-weight: bold;

padding-left: 0px;

padding-right: 20px;

text-align: left;

}

h2{ clear: both; font-size: 130%; }

h3{

clear: both;

font-size: 115%;

margin-left: 20px;

margin-top: 30px;

}

p{ margin-left: 20px; font-size: 12px; }

table.list{ float: left; }

table.list td:nth-child(1){

font-weight: bold;

border-right: 1px grey solid;

text-align: right;

}

table.list td:nth-child(2){ padding-left: 7px; }

table tr:nth-child(even) td:nth-child(even){ background: #CCCCCC; }

table tr:nth-child(odd) td:nth-child(odd){ background: #F2F2F2; }

table tr:nth-child(even) td:nth-child(odd){ background: #DDDDDD; }

table tr:nth-child(odd) td:nth-child(even){ background: #E5E5E5; }

div.column { width: 320px; float: left; }

div.first{ padding-right: 20px; border-right: 1px grey solid; }

div.second{ margin-left: 30px; }

table{ margin-left: 20px; }

-->

</style>

</head>

<body>

"@

#endregion

#region System Info

$OS = (Get-WmiObject Win32\_OperatingSystem -computername $computer).caption

$SystemInfo = Get-WmiObject -Class Win32\_OperatingSystem -computername $computer | Select-Object Name, TotalVisibleMemorySize, FreePhysicalMemory

$TotalRAM = $SystemInfo.TotalVisibleMemorySize/1MB

$FreeRAM = $SystemInfo.FreePhysicalMemory/1MB

$UsedRAM = $TotalRAM - $FreeRAM

$RAMPercentFree = ($FreeRAM / $TotalRAM) \* 100

$TotalRAM = [Math]::Round($TotalRAM, 2)

$FreeRAM = [Math]::Round($FreeRAM, 2)

$UsedRAM = [Math]::Round($UsedRAM, 2)

$RAMPercentFree = [Math]::Round($RAMPercentFree, 2)

#endregion

#region The\_Gathering

#Listing the top ten system processes in terms of memory used

$TopProcesses = Get-Process -ComputerName $computer | Sort WS -Descending | Select ProcessName, Id, WS -First $ProccessNumToFetch | ConvertTo-Html -Fragment

# Fetch the Uptime of the current system using our Get-HostUptime Function.

$SystemUptime = Get-HostUptime -ComputerName $computer

#endregion

#region Services Report

#Look for services with run states that conflict with startup settings (Automatic starts that aren't running, etc)

$ServicesReport = @()

$Services = Get-WmiObject -Class Win32\_Service -ComputerName $computer | Where {($\_.StartMode -eq "Auto") -and ($\_.State -eq "Stopped")}

foreach ($Service in $Services) {

$row = New-Object -Type PSObject -Property @{

Name = $Service.Name

Status = $Service.State

StartMode = $Service.StartMode

}

$ServicesReport += $row

}

$ServicesReport = $ServicesReport | ConvertTo-Html -Fragment

#endregion

#region Event Logs Report

#this is what feeds the pie chart

$SystemEventsReport = @()

$SystemEvents = Get-EventLog -ComputerName $computer -LogName System -EntryType Error,Warning -Newest $EventNum

foreach ($event in $SystemEvents) {

$row = New-Object -Type PSObject -Property @{

TimeGenerated = $event.TimeGenerated

EntryType = $event.EntryType

Source = $event.Source

Message = $event.Message

}

$SystemEventsReport += $row

}

$SystemEventsReport = $SystemEventsReport | ConvertTo-Html -Fragment

$ApplicationEventsReport = @()

$ApplicationEvents = Get-EventLog -ComputerName $computer -LogName Application -EntryType Error,Warning -Newest $EventNum

foreach ($event in $ApplicationEvents) {

$row = New-Object -Type PSObject -Property @{

TimeGenerated = $event.TimeGenerated

EntryType = $event.EntryType

Source = $event.Source

Message = $event.Message

}

$ApplicationEventsReport += $row

}

$ApplicationEventsReport = $ApplicationEventsReport | ConvertTo-Html -Fragment

#endregion

# Create the chart using our Chart Function

Create-PieChart -FileName ((Get-Location).Path + "\chart-$computer") $FreeRAM, $UsedRAM

$ListOfAttachments += "chart-$computer.png"