```powershell
#-------------------------------------------------------------------------------
---------------
#SCCM-EmailLogOSD.ps1
#
#Written by: Matt Logue
#
#Last Modified by: Matt Logue
#
#Last Modified: August 8, 2020
#
#Version: 1.0
#
#Purpose: The script looks for in a shared log directory under
\\HSSCCM\Packages\Logs for machines that have written their
# log files to the path.  Then it emails the log file to the helpdesk that contains
information such as MAC address, machine name, etc
#
#-------------------------------------------------------------------------------
---------------

<#
.SYNOPSIS
The script looks for in a shared log directory under \\HSSCCM\Packages\Logs for
machines that have written their
log files to the path.  Then it emails the log file to the helpdesk that contains
information such as MAC address, machine name, etc

.DESCRIPTION
  Requires
  1. Connection to the shared folder path and read permissions
    2. Mailbox permissions - SendAs hssccm@hsc.wvu.edu

.PARAMETER
  No required parameters

.NOTES
  Author: Matt Logue
      Last Updated by: Matt Logue
    Last Updated: August 8, 2020
#>


Param (
[string]$Date = (Get-Date),
[string]$LogDir = "\\hssccm\packages\logs\sccmimagetranscripts\machines",
[string]$ArchiveDir =
"\\hssccm\packages\logs\sccmimagetranscripts\machines\Archive",
[string]$ScriptFolder = $PSScriptRoot,
[string]$EmailUser = "microsoft@hsc.wvu.edu",
[string]$EmailFrom = "hssccm@hsc.wvu.edu",
[string]$EmailTo = "hscitdss@hsc.wvu.edu",
[string]$DaysToKeepLogFiles = 5

)

$inst = ""

Set-Location $ScriptFolder
$TranscriptLogFile = "$ScriptFolder\Logs\emaillogtranscript-$(Get-date -Format
yyyMMdd-HHmm).txt"
```

```powershell

$Host.UI.RawUI.WindowTitle = 'SCCM - Windows Imaging Task Sequence Log Generation'
$error.Clear()

#Add references to file containing needed functions
Import-module $env:userprofile\Documents\Github\HSC-PowerShell-Repository\1HSC-PowerShell-Modules\HSC-CommonCodeModule.psm1

#Starting Transcript File
try
{
  Stop-Transcript -ea "Stop"
}
catch
{
}

"TranscriptLogFile: " + $TranscriptLogFile
Start-Transcript -Path $TranscriptLogFile -Force
"Transcript log file started"

#Cleaningup old log files
Write-Verbose "Removing Old Log Files"
Remove-OldLogFiles -Days $DaysToKeepLogFiles -Path $ScriptFolder\Logs
Remove-OldLogFiles -Days $DaysToKeepLogFiles -Path $ArchiveDir
Remove-OldLogFiles -Days $DaysToKeepLogFiles -Path $TranscriptLogFile

####################Credentials for sending mail message############################
$UserName = $EmailUser
$User = ($username -split "@")
$hashpassword = "$ScriptFolder\"+$user[0]+".txt"


If (Test-Path $hashpassword) {  #If the hashedpassword file exist, get-content
    $SecurePassword = Get-Content $hashpassword | ConvertTo-SecureString
}
If (!(Test-Path $hashpassword)) {  #If the hashedpassword file doesn't exist, read host for input
    Read-Host -Prompt "Enter Password" -AsSecureString | ConvertFrom-SecureString | Out-File $hashpassword
}

$Credentials = New-Object System.Management.Automation.PSCredential -ArgumentList $UserName, $SecurePassword

#################################
#         MAIN PROGRAM          #
#################################

While ($true){

#Set body variable to empty string in case it is set to create message
$body = ""

#Checking for existing log files if not, it waits 10 minutes
while ((Get-ChildItem $LogDir -File | Measure-Object).Count -le 0) {
    Write-Output "No Log Files Found - Waiting 5 Minutes";
    Start-Sleep 300;
}
```

```powershell
#Finding files in $LogDir
$LogFiles = Get-ChildItem -Path $LogDir -file | select *

#Reading each log files and creating the body for the message for each one it finds
foreach ($file in $LogFiles.FullName) {
$body = Get-Content $file
$computername = ($(Get-content $file | Select-string "ComputerName") -replace "ComputerName:").Trim()
  # ForEach ($line in $content){
   #   $body += "$line`n"
    #}

$body = $body | Out-string

$body += "Previous machine images can be found at: \\hssccm\Packages\Logs\SCCMImageTranscripts\Machines\Archive"


#Set email fields based on the $inst value

$From = $EmailFrom

If (($inst.ToLower()) -eq "test")
{
    $To = $EmailUser
}
Else
{
    $To = $EmailTo
}

$Bcc = $UserName
$Subject = "Machine Imaged by SCCM: $computername"
$SMTPServer = "smtp.office365.com"
$SMTPPort = "587"

Write-Output "From: "$From
Write-Output "To: "$To
Write-Output "Subject: "$Subject
Write-Output "Body: `n"$body

Write-Host "Sending Email...`t $subject" -Foregroundcolor Yellow

#Trying to send message if any error the script breaks so the log file doesn't get moved
try {

Send-MailMessage -From $From -To $To -Bcc $Bcc -Subject $Subject -Body $Body -SmtpServer $SMTPServer -port $SMTPPort -UseSsl -Credential $Credentials
#Moving log file to $ArchiveDir
Move-Item $file $ArchiveDir -Force
}
catch {

for ($i=0 ; $i -lt $Error.count ; $i++)
{
     Write-Host $Error[$i] -Foregroundcolor Red
}
```

```
break

}

#$body | Add-Content "$ScriptFolder\emaillog.txt"

Start-sleep 3

Write-Host "Email Sent...`t $subject" -Foregroundcolor Green
}

}
Stop-Transcript


```