Get-IISLogFolderMetrics.ps1

```
   <#
.SYNOPSIS
    Get CSV formatted metrics that give details of the IIS Log Folders
    on the targetted machine.
.DESCRIPTION
    This script will enumerate the host to find all websites which are
    present on it.  It then gets details of the size and age of IIS
    log files for each site before presenting the results in a table.
.PARAMETER Format
    Permitted values: text, csv, json
.EXAMPLE
.NOTES
.LINK
#>

[Diagnostics.CodeAnalysis.SuppressMessageAttribute("PSAvoidUsingWriteHost", "")]
Param(
    [ValidateSet("text","csv","csvEx","json","list")]
    [string] $Format = "csv"
)

# if the machine has the web administration module then...
if ((Get-Module -ListAvailable WebAdministration) -eq $null)
{
    # write out an error string
    Write-Host "This task cannot continue because the necessary PowerShell Modules
are not present on this host.\nAre you sure that this host has an implementation of
Internet Information Server installed?"

    # exit now as we can't continue
    exit
}

# import the module
Import-Module WebAdministration

# variable to hold total log size
[double]$totalSize = 0

# create a variable to hold the output
$output = @()

# get all websites
$webSites = (Get-WebSite | Sort-Object id)

# create the header string
$output += 'Website Name,Log Folder,Log Size (MB),Oldest Log File Date,Oldest Log
File Age (Days),Volume Free Space (MB)%EOL%'


# loop through all of the websites
foreach ($site in $webSites)
{
    # get the path where the logs for this website exist
    [string]$logFolder =
[System.Environment]::ExpandEnvironmentVariables($site.logFile.directory)

    # append a slash if required
    if (!($logFolder.EndsWith("\"))) { $logFolder += "\" }

    # append the default folder name and the ID of the current set
    $logFolder += "W3SVC" + $site.id.ToString()

    # if the folder exists then...
    if (Test-Path -Path $logFolder)
    {
        # get the number of bytes in this folder
        $folderMBytes = ((Get-ChildItem $logFolder -Recurse | Measure-Object -
Property Length -Sum -ErrorAction Stop).Sum /1MB)

        # add to the total
        $totalSize += $folderMBytes

        # if the folder bytes is null then set it to zero as there are no longs
        if ($folderMBytes -eq $null) { $folderMBytes = 0 }
```

```
        # get the oldest log file
        $oldestLogFile = (Get-ChildItem $logFolder -Recurse | Sort-Object
CreationTime | Select -First 1).CreationTime

        # if there is an oldest log file then...
        if ($oldestLogFile -ne $null)
        {
            # calculate the age in days
            $oldestLogAgeDays = ([Math]::Floor((New-TimeSpan -Start $oldestLogFile
-End (Get-Date)).TotalDays))
        }
        else
        {
            # no oldest log file because there are no logs so set the oldest log
age in days to a blank string
            $oldestLogAgeDays = ""
        }

        # build the WMI filter to get free space
        $wmiFilter = "deviceid='" + $logFolder.Substring(0,1) + ":'"

        # get the free space on the disk where the logs resize
        $freeSpaceMBytes = ((Get-WMIObject Win32_Logicaldisk -filter
$wmiFilter).FreeSpace / 1MB).ToString("0.00")
    }
    else
    {
        # the folder wasn't found so set the size and log age values appropriately
        $folderMBytes = 0
        $oldestLogFile = $null
        $oldestLogAgeDays = ""
    }

    # create a new line in the output file
    $output += '"{0}","{1}","{2}","{3}","{4}","{5}"%EOL%' -f `
        $site.name,
        $logFolder,
        $folderMBytes.ToString("0.00"),
        $oldestLogFile,
        $oldestLogAgeDays,
        $freeSpaceMBytes

}

# write out the total too
# create a new line in the output file
$output += '"{0}","{1}","{2}","{3}","{4}","{5}"%EOL%' -f `
    "",
    "Total Size:",
    $totalSize.ToString("0.00"),
    "",
    "",
    ""

if ($Format -eq 'text')
{
    ConvertFrom-Csv ($output -replace '%EOL%','') `
        | Format-Table -AutoSize `
        | Out-String -Width 4096 `
        | Write-Host
}
elseif ($Format -eq 'csv')
{
    $output -replace '%EOL%','' `
        | Out-String -Width 4096 `
        | Write-Host
}
elseif ($Format -eq 'csvEx')
{
    $output `
        | Out-String -Width 4096 `
        | Write-Host
}
elseif ($Format -eq 'json')
{
    ConvertFrom-Csv ($output -replace '%EOL%','') `
```

```
        | ConvertTo-Json `
        | Out-String -Width 4096 `
        | Write-Host
}
elseif ($Format -eq 'list')
{
    ConvertFrom-Csv ($output -replace '%EOL%','') `
        | Format-List `
        | Out-String -Width 4096 `
        | Write-Host
}
```