```
<#
Disclaimer!!!
The sample scripts are not supported under any Microsoft standard support program
or service.
The sample scripts are provided AS IS without warranty of any kind. Microsoft
further disclaims all implied warranties including,
without limitation, any implied warranties of merchantability or of fitness for a
particular purpose.
The entire risk arising out of the use or performance of the sample scripts and
documentation remains with you.
In no event shall Microsoft, its authors, or anyone else involved in the creation,
production, or delivery of the scripts be liable for any damages
whatsoever (including, without limitation, damages for loss of business profits,
business interruption, loss of business information, or other pecuniary loss)
arising out of the use of or inability to use the sample scripts or documentation,
even if Microsoft has been advised of the possibility of such damages.
#>

#function to log
function Write-Information([Boolean] $isHost, [String] $message) {
    try {
         # Write the error message to the host also.
         if ($isHost -eq $true) {
            Write-Host $message
        }

        if ( $logpath -eq "" ) { return }
 (Get-Date).ToString()+" "+ $message | Out-File $logpath -Append
        # Log all the messages to the log file.
        #if ($null -ne $logger) {
         #   $logger.WriteLine();
         #   $logger.WriteLine($message)
       # }

    }
    catch {
        Write-Host "Not able to initialize logger"
    }

}
try {
    $timestamp = Get-Date -Format 'yyyy-MM-dd HH:mm' | ForEach-Object { $_ -replace
":", "." }
    $logpath ="SiteMailboxesLogFile $timestamp.log"
    #$script:logfilePath = $logpath
    #$script:logger = new-object System.IO.StreamWriter ($logfilePath , $true)
}
catch {
    Write-Host "Not able to initialize logger"
}
```

```
try {
    Write-Host  "Connecting to Exchange Online. Enter your admin credentials"
    $UserCredential = Get-Credential
    $Session = New-PSSession -ConfigurationName Microsoft.Exchange -ConnectionUri
https://outlook.office365.com/powershell-liveid/ -Credential $UserCredential
-Authentication Basic -AllowRedirection
    Import-PSSession $Session -DisableNameChecking | Out-Null


#https://docs.microsoft.com/en-us/powershell/exchange/connect-to-exchange-online-po
wershell?view=exchange-ps
    #For Office 365 operated by 21Vianet, use the ConnectionUri value:
https://partner.outlook.cn/PowerShell
    #For Office 365 Germany, use the ConnectionUri value:
https://outlook.office.de/powershell-liveid/
    #For Microsoft 365 GCC High, use the ConnectionUri value:
https://outlook.office365.us/powershell-liveid/
    #For Microsoft 365 DoD, use the ConnectionUri value:
https://webmail.apps.mil/powershell-liveid
    #If you're behind a proxy server, run this command first: $ProxyOptions =
New-PSSessionOption -ProxyAccessType <Value>, where <Value> is IEConfig,
WinHttpConfig, or AutoDetect.
    #Then, add the following parameter and value to the end of the $Session = ...
command: -SessionOption $ProxyOptions.
    #For more information, see New-PSSessionOption.

}
catch {
    Write-Information $true  "Issue Connecting to Exchange Online."
    Write-Information $true $PSItem.Exception.Message
    exit
}

try {
    #Get sitemailbox list for a give tenant
    Get-SiteMailbox -BypassOwnerCheck -ResultSize Unlimited | Select-Object Name,
PrimarySmtpAddress, WhenCreated, ClosedTime, SharePointUrl | Export-Csv
SiteMailboxesList.csv
}
catch {
    Write-Information $true  "Issue fetching SiteMailboxes list."
    Write-Information $true $PSItem.Exception.Message
    exit
}
finally {
    #Close ExO session
    Remove-PSSession $Session
}
```

```
Write-Host  "List of existing SiteMailboxes in your tenant is exported to
SiteMailboxesList.csv"

Write-Host  "Please review SiteMailboxesList.csv file and remove mailboxes from it
which you do not like to setup for backup. `r`nSave file after you have removed the
mailboxes which you would not like to get processed further." -ForegroundColor Red
Write-Host ""

$choice = ""
while ($choice -notmatch "[y|n]") {
    Write-Host  "Have you reviewed SiteMailboxesList.csv file and removed mailboxes
not to processed from it.`r`n"
    $choice = read-host "Do you like to proceed further? (Y/N)"
}
try {
    if ($choice -eq "y") {
        $csvpath = Get-Location
        $csvfilename = "SiteMailboxesList.csv"
        $csvpath = "$csvpath\$csvfilename"

        Write-Information $true  "Reading Sitemailboxes list to process "
        $mailboxlistpath = Test-Path $csvpath
    }elseif ($choice -eq "n") {
        exit
    }
}
catch {
    Write-Information $true  "Error reading SiteMailboxesList.csv "
    Write-Information $true $PSItem.Exception.Message
    exit
}

function compSearch {
    [CmdletBinding()]
    Param(
        [Parameter(Mandatory = $True, HelpMessage = 'Enter the email address that
you want to export')]
        $Mailbox,
        [Parameter(Mandatory = $True, HelpMessage = 'Enter the URL for the user''s
SharePoint site here. If you don''t enter one, this will be skipped.')]
        $SharepointURL
    )

    # Create a search name. You can change this to suit your preference
    $SearchName = "$Mailbox PST"

    Write-Information $true  "Creating content search... "
    New-ComplianceSearch -Name $SearchName -ExchangeLocation $Mailbox
-SharePointLocation $SharepointURL -AllowNotFoundExchangeLocationsEnabled $true |
```

```
Out-Null #Create a content search, including the the entire contents of the user's
email and onedrive. If you didn't provide a OneDrive URL, or it wasn't valid, it
will be ignored.
    Write-Information $true  "Starting content search... "
    Start-ComplianceSearch -Identity $SearchName #Start the search created above
    Write-Information $true  "Waiting for content search to complete... "
    for ($SearchStatus; $SearchStatus -notlike "Completed"; ) {
        #Wait then check if the search is complete, loop until complete
        Start-Sleep -s 2
        $SearchStatus = Get-ComplianceSearch $SearchName | Select-Object
-ExpandProperty Status #Get the status of the search
        Write-Host  -NoNewline "." # Show some sort of status change in the
terminal
    }
    Write-Information $true  "Content search is completed"
    Write-Information $true  "Setting up export options for the content search
results"
    New-ComplianceSearchAction -SearchName $SearchName -Export -Format FxStream
-ExchangeArchiveFormat PerUserPst -Scope BothIndexedAndUnindexedItems -EnableDedupe
$true -SharePointArchiveFormat IndividualMessage -IncludeSharePointDocumentVersions
$true | Out-Null
    Start-Sleep -s 5 # Let's wait 5 seconds to create the SearchAction before the
next commands try to run against it.
}

if ($mailboxlistpath -eq $true) {
    try {
        $mailboxlist = Import-Csv $csvpath
        $str = "Number(s) of SiteMailbox(es) to be processed " + $mailboxlist.Count
        Write-Information $true  $str
        if ($mailboxlist.Count -gt 0) {
            try {

#https://docs.microsoft.com/en-us/powershell/exchange/connect-to-scc-powershell?vie
w=exchange-ps
                Write-Information $true  "Connecting to Exchange Compliance
Center."
                $SessionCompliance = New-PSSession -ConfigurationName
Microsoft.Exchange -ConnectionUri
https://ps.compliance.protection.outlook.com/powershell-liveid/ -Credential
$UserCredential -Authentication Basic -AllowRedirection
                Import-PSSession $SessionCompliance -DisableNameChecking  |
Out-Null
            }
            catch {
                Write-Information $true  "Error connecting to Exchange Compliance
Center."
                Write-Information $true $PSItem.Exception.Message
                exit
            }
```

```
            foreach ($mbx in $mailboxlist) {
                $strmsglog = "Processing... " + $mbx.PrimarySmtpAddress
                Write-Information $true  $strmsglog
                compSearch -Mailbox $mbx.PrimarySmtpAddress -SharepointURL
$mbx.SharePointUrl
            }
            Remove-PSSession $SessionCompliance
            Write-Information $true  "Mailboxes content search are complete. Please
use Compliance Center dashboard to download PSTs"
        }
    }
    catch {
        Write-Information $true $PSItem.Exception.Message
        Write-Information $true  "Error processing mailboxes"
    }
    finally{
        Remove-PSSession $SessionCompliance
    }
}
else {
    Write-Information $true  "Please make sure SiteMailboxesList.csv exist on
current directory path"
}
Write-Information $true  "Processing finished"
```