```
#Compared-Ext15WithPrimarySMTPAddress.ps1
#Written by: Jeff Brusoe
#Last Updated: May 12, 2020
#
#Compares value pushed to extension15 with primary SMTP address

[CmdletBinding()]
param (
    #Common HSC PowerShell Parameters
    [switch]$NoSessionTranscript,
    [string]$LogFilePath = "$PSScriptRoot\Logs",
    [switch]$StopOnError, #$true is used for testing purposes
    [int]$DaysToKeepLogFiles = 5, #this value used to clean old log files

    #File specific parameters
    [switch]$SkipBlankExt15 #A blank field normally implies that this account isn't
    maintained by SailPoint
)

###############################
#Import HSC PowerShell Modules#
###############################

#Build path to HSC PowerShell Modules
$PathToHSCPowerShellModules = $PSScriptRoot
$PathToHSCPowerShellModules =
$PathToHSCPowerShellModules.substring(0,$PathToHSCPowerShellModules.lastIndexOf("\")+1)
$PathToHSCPowerShellModules += "1HSC-PowerShell-Modules"
Write-Output $PathToHSCPowerShellModules

#Attempt to load common code module
$CommonCodeModule = $PathToHSCPowerShellModules + "\HSC-CommonCodeModule.psm1"
Write-Output "Path to common code module: $CommonCodeModule"
Import-Module $CommonCodeModule -Force -ArgumentList
$NoSessionTranscript,$LogFilePath,$true,$DaysToKeepLogFiles

#Attempt to load HSC Active Directory Module
$ActiveDirectoryModule = $PathToHSCPowerShellModules + "\HSC-ActiveDirectoryModule.psm1"
Write-Output "Path to HSC Active Directory Module: $ActiveDirectoryModule"
Import-Module $ActiveDirectoryModule -Force

if ($Error.Count -gt 0)
{
    #Any errors at this point are from loading modules. Program must stop.
    Write-Warning "There was an error configuring the environment. Program is exiting."
    Exit-Commands
}

######################################
#End of Import HSC PowerShell Modules#
######################################

#############################
#Configure environment block#
#############################
Write-Output "Getting Parameter Information"
Get-Parameter -ParameterList $PSBoundParameters

Set-Environment
Set-WindowTitle

$SummaryFile = "$LogFilePath\" + (Get-Date -format yyyy-MM-dd-HH-mm) +
"-EmailFieldsSummary.csv"
New-Item -type File -Path $SummaryFile  -Force | Out-Null

$NoMatchFile = "$LogFilePath\" + (Get-Date -format yyyy-MM-dd-HH-mm) + "-NoMatchFile.csv"
New-Item -type File -Path $NoMatchFile -Force | Out-Null
```

```powershell

$Error.Clear()

########################################
#End of environment configuration block#
########################################

Write-Output "Generating list of AD users"
try
{
    $ADUsers = Get-ADUser -Filter * -Properties
    proxyAddresses,mail,extensionAttribute15 -ErrorAction Stop -SearchBase
    "OU=HSC,DC=hs,DC=wvu-ad,DC=wvu,DC=edu"
    Write-Output "Successfully generated AD user list"
}
catch
{
    Write-Warning "Unable to get list of AD users. Program is exiting."
    Exit-Command
}

foreach ($ADUser in $ADUsers)
{
    Write-Output $("Current User: " + $ADUser.SamAccountName)

    [string]$ADMail = $ADUser.mail
    if ([string]::IsNullOrEmpty($ADMail))
    {
        Write-Verbose "AD mail field is empty"
        $ADMail = "Mail Attribute Not Present"
    }

    [string]$ext15 = $ADUser.extensionAttribute15
    if ([string]::IsNullOrEmpty($ext15))
    {
        Write-Verbose "extensionAttribute15 is empty"
        $ext15 = "Ext15 Not Present"
    }

    [string[]]$ProxyAddressArray = $ADUser  | select -ExpandProperty proxyAddresses
    $ProxyAddressArray

    $ProxyCount = ($ProxyAddressArray | where {$_ -clike "SMTP:*" } | Measure).Count

    if ($ProxyCount -eq 0)
    {
        Write-Verbose "PrimarySMTPAddress wasn't found"
        $PrimarySMTPAddress = "PrimarySMTP Not Present"
    }
    elseif ($ProxyCount -eq 1)
    {
        Write-Verbose "Single Primary SMTP address was found."
        $PrimarySMTPAddress = $ProxyAddressArray | where {$_ -clike "SMTP:*" }
        $PrimarySMTPAddress = $PrimarySMTPAddress -replace "SMTP:",""
    }
    else
    {
        #This shouldn't be reached, but I have seen a few cases of this.
        Write-Warning "Multiple Primary SMTP addresses were found."

        foreach ($ProxyAddress in ($ProxyAddressArray |  where {$_ -clike "SMTP:*"}))
        {
            Write-Output "Primary SMTP Address: $ProxyAddress"
        }

        Write-Output "***********************"

        continue
    }

    Write-Output "Mail Attribute: $ADMail"
    Write-Output "Primary SMTP Address: $PrimarySMTPAddress"
    Write-Output "extensionAttribute15: $ext15"


    $UserInfo = New-Object -type PSObject

    $UserInfo | Add-Member -MemberTYpe NoteProperty -Name SamAccountName -Value $ADUser.SamAccountName
    $UserInfo | Add-Member -MemberTYpe NoteProperty -Name ADMailAttribute -Value $ADMail
    $UserInfo | Add-Member -MemberTYpe NoteProperty -Name PrimarySMTPAddress -Value $PrimarySMTPAddress
    $UserInfo | Add-Member -MemberTYpe NoteProperty -Name extensionAttribute15 -Value $ext15

    if ($PrimarySMTPAddress.indexOf("@") -gt 0)
    {
        #This is a valid email address
        $EmailPrefix = $PrimarySMTPAddress.substring(0,$PrimarySMTPAddress.indexOf("@"))
        Write-Output "Email Prefix: $EmailPrefix"
    }
    else
    {
        Write-Output "Unable to generate email prefix"
        $EmailPrefix = "No email prefix"
    }

    $UserInfo | Add-Member -MemberType NoteProperty -Name EmailPrefix -Value $EmailPrefix

    if ((!$SkipBlankExt15) -OR (($ext15 -ne "Ext15 Not Present") -AND ($SkipBlankExt15)))
    {
        $UserInfo | Export-Csv $SummaryFile -Append

        if ($EmailPrefix -ne $ext15)
        {
            $UserInfo | Export-Csv $NoMatchFile -Append
        }
    }
    else
    {
        $UserInfo = $null
    }

    if ($Error.Count -gt 0)
    {
        Write-Output "Stopping...."
        $Error
        Stop-Transcript
        return
    }

    Write-Output "***********************"
}

if (!$NoSessionTranscript)
{
    Stop-Transcript
}

#Exit
```