```powershell
function Add-HSCProxyAddress
{
  <#
    .SYNOPSIS
      This function adds a single (or multiple) proxy addresses to an AD Account.

    .DESCRIPTION
      The purpose of this function is to add a new proxy address to a specified user which
      can be passed into the function as either an ADUser object or a string representing the SAMAccountName. The
      function returns either true or false based on if this operation is successful.

    .PARAMETER ADUser
      Specifies an ADUser object to add proxy address(es) to.

    .PARAMETER SamAccountName
      This parameter is a string representing an AD user's SAM account name
      that will be searched for to add the proxy address.

    .PARAMETER NewProxyAddress
      The new proxy addresses to be added

    .PARAMETER Primary
      A switch parameter to indicate that this is a primary SMTP address

    .EXAMPLE
      PS C:\Users\microsoft\Documents\GitHub\HSC-PowerShell-Repository> Get-ADUser jefftest -Properties * | select proxyAddresses -ExpandProperty proxyAddresses

      PS C:\Users\microsoft\Documents\GitHub\HSC-PowerShell-Repository> Add-HSCProxyAddress jefftest -NewProxyAddress "jefftest@hsc.wvu.edu" -Primary
      Attempting to find user: jefftest
      User found
      Adding: SMTP:jefftest@hsc.wvu.edu
      Successfully added new proxy address
      True

      PS C:\Users\microsoft\Documents\GitHub\HSC-PowerShell-Repository> Get-ADUser jefftest -Properties * | select proxyAddresses -ExpandProperty proxyAddresses
      SMTP:jefftest@hsc.wvu.edu

    .EXAMPLE
      PS C:\Users\microsoft\Documents\GitHub\HSC-PowerShell-Repository> Get-ADUser jefftest -Properties * | select proxyAddresses -ExpandProperty proxyAddresses
      SMTP:jefftest@hsc.wvu.edu

      PS C:\Users\microsoft\Documents\GitHub\HSC-PowerShell-Repository> Add-HSCProxyAddress jefftest -NewProxyAddress "jefftest2@hsc.wvu.edu"
      Attempting to find user: jefftest
      User found
      Current Proxy Address from ADUser: SMTP:jefftest@hsc.wvu.edu
      Adding: smtp:jefftest2@hsc.wvu.edu
      Successfully added new proxy address
      True

      PS C:\Users\microsoft\Documents\GitHub\HSC-PowerShell-Repository> Get-ADUser jefftest -Properties * | select proxyAddresses -ExpandProperty proxyAddresses
      smtp:jefftest2@hsc.wvu.edu
```

```
    SMTP:jefftest@hsc.wvu.edu

  .NOTES
    Written by: Jeff Brusoe
    Last Updated: October 14, 2020
#>

[CmdletBinding(SupportsShouldProcess=$true)]
[OutputType([bool])]

param (
  [Parameter(ValueFromPipeline=$false,
    ParameterSetName="ADUser",
    Mandatory=$true,
    Position=0)]
  [ValidateNotNullOrEmpty()]
  [Microsoft.ActiveDirectory.Management.ADAccount]$ADUser,

  [Parameter(ValueFromPipeline=$false,
    ParameterSetName="SamAccountName",
    Mandatory=$true,
    Position=0)]
  [ValidateNotNullOrEmpty()]
  [string]$SAMAccountName,

  [Parameter(ValueFromPipeline=$true,
    Mandatory=$true,
    Position=1)]
  [Alias("NewProxyAddresses")]
  [ValidateNotNullOrEmpty()]
  [string[]]$NewProxyAddress,

  [switch]$Primary
)

begin
{
  Write-Verbose $("Parameter Set Name: " + $PSCmdlet.ParameterSetName)

  if ($PSCmdlet.ParameterSetName -eq "ADUser") {
    Write-Verbose $("ADUser SamAccountName: " + $ADUser.SamAccountName)
  }
  else {
    Write-Verbose "SamAccountName: $SamAccountName"
  }

  $ProxyAdded = $false
}

process
{
  if (!(Test-HSCADModuleLoaded))
  {
    Write-Verbose "Attempting to load AD Module"

    try {
      Import-Module ActiveDirectory -ErrorAction Stop
      $ProxyAdded = $true
    }
    catch {
```

```
        Write-Warning "Unable to load Active Directory module"
        $ProxyAdded = $false
      }
    }
    else {
      Write-Verbose "AD module is already loaded"
      $ProxyAdded = $true
    }

    Write-Verbose $("In process block - Parameter Set Name: " +
$PSCmdlet.ParameterSetName)

    if ($PSCmdlet.ParameterSetName -eq "SamAccountName" -AND $ProxyAdded)
    {
      try {
        Write-Output "Attempting to find user: $SamAccountName" | Out-Host

        $LDAPFilter = "(&(objectCategory=person)(objectClass=user)
(sAMAccountName=$SamAccountName))"
        Write-Verbose "LDAP Filter: $LDAPFilter"

        $ADUser = Get-ADUser -LDAPFilter $LDAPFilter -properties proxyAddresses -
ErrorAction Stop

        if ($null -eq $ADUser) {
          Write-Warning "Unable to find AD User"
          $ProxyAdded = $false
        }
        else {
          Write-Output "User found" | Out-Host
          Write-Verbose "Distinguished Name:"
          Write-Verbose $ADUser.DistinguishedName

          $ProxyAdded = $true
        }
      }
      catch {
        Write-Warning "Unable to find AD user based on SamAccountName"
      }
    }

    if ($ProxyAdded)
    {
      #Verify one unique user was found
      try {
        $ADUserCount = ($ADUser | Measure-Object).Count
        Write-Verbose "AD User Count: $ADUserCount"

        if ($ADUserCount -eq 1)
        {
          Write-Verbose "One unique AD User was found"
        }
        else {
          Write-Warning "AD user count doesn't equal 1"
          $ProxyAdded = $false
        }
      }
      catch {
        Write-Warning "Unable to determine AD User Count"
        $ProxyAdded = $false
```

```
      }
    }

    if ($ProxyAdded)
    {
      #At this point, one unique AD user object should be found.
      #Now get proxy addresses to verify that the proxy address isn't already present
      #or that it's not the primary SMTP address
      $ProxyAddresses = $ADUser.proxyAddresses

      foreach ($ProxyAddress in $ProxyAddresses)
      {
        Write-Output "Current Proxy Address from ADUser: $ProxyAddress" | Out-Host

        if (($ProxyAddress.indexOf($NewProxyAddress) -ge 0) -AND ($ProxyAddress -clike "*SMTP*"))
        {
          Write-Output "New Proxy address is the primary SMTP address."
          $ProxyAdded = $false
        }
        elseif ($ProxyAddress.indexOf($NewProxyAddress) -ge 0)
        {
          Write-Output "New proxy address is already a proxy address" | Out-Host
          $ProxyAdded = $false
        }
      }
    }

    if ($ProxyAdded)
    {
      #Now proxy address will be added to AD account
      $NewProxyAddress = $NewProxyAddress.Trim()

      if ($Primary)
      {
        $NewProxyAddress = "SMTP:" + $NewProxyAddress
      }
      else {
        $NewProxyAddress = "smtp:" + $NewProxyAddress
      }

      Write-Output "Adding: $NewProxyAddress"

      try {
        if ($PSCmdlet.ShouldProcess("Adding new proxy address"))
        {
          Write-Verbose "About to set proxy address"
          $ADUser | Set-ADUser -Add @{proxyAddresses=$NewProxyAddress} -ErrorAction Stop
        }

        Write-Output "Successfully added new proxy address" | Out-Host
      }
      catch {
        Write-Warning "Unable to add new proxy address"
        $ProxyAdded = $false
      }
    }
  }
```

```
226
227   end
228   {
229     return $ProxyAdded
230   }
231 }
```