```powershell
# Get-ADUserTokenSize.ps1
# Written by: Jeff Brusoe
# Last Updated: September 16, 2021
#
# See formula in this article:
# https://docs.microsoft.com/en-US/troubleshoot/windows-server/windows-security/kerberos-authentication-problems-if-user-belongs-to-groups
#
# According to that article, the formula for calculating the token size is:
# TokenSize = 1200 + 40d + 8s
#
# d is the sum of:
#   - The number of memberships in universal groups that
#     are outside the user's account domain.
#   - The number of SIDs stored in the account's sIDHistory attribute.
#
# s is the sum of:
#   - The number of memberships in universal groups that are
#     inside the user's account domain.
#   - The number of memberships in domain-local groups.
#   - The number of memberships in global groups.

[CmdletBinding()]
param(
    [ValidateNotNullOrEmpty()]
    [string]$HSDomainFQDN = "hs.wvu-ad.wvu.edu",

    [ValidateNotNullOrEmpty()]
    [string]$OutputFilePath = (Get-HSCNetworkLogPath) +
                                "4ADMigrationProject\"
)

try {
    Set-HSCEnvironment -ErrorAction Stop

    $TokenSizeLog = $OutputFilePath + "UserTokenSize\" +
                        (Get-Date -format "yyyy-MM-dd") +
                        "-UserTokenSize.csv"

    New-Item $TokenSizeLog -ItemType File -Force -ErrorAction Stop
}
catch {
    Write-Warning "Unable to configure HSC environment"
    Invoke-HSCExitCommand -ErrorCount $Error.Count
}

try {
    Write-Output "Getting list of AD users"

    $GetADUserParams = @{
        Filter = "*"
        Properties = @(
            "memberOf",
            "SIDHistory",
            "whenChanged",
            "whenCreated"
        )
        ErrorAction = "Stop"
    }
```

```powershell

    #$ADUsers = Get-ADUser "jbrusoe" -Properties memberOf
    $ADUsers = Get-ADUser @GetADUserParams
}
catch {
    Write-Warning "Unable to get list of AD users"
    Invoke-HSCExitCommand -ErrorCount $Error.Count
}

foreach ($ADUser in $ADUsers) {
    $TokenSize = 1200
    $d = $ADUser.SIDHistory.Count
    $s = 0

    Write-Output $("Current User: " + $ADUser.SamAccountName)

    $ADGroups = $ADUser.memberOf
    foreach ($ADGroup in $ADGroups) {
        Write-Output "Examining Group: $ADGroup"

        #Parse DN up to find domain and group name
        $Domain = $ADGroup

        if ($Domain.indexOf("OU=") -ge 0) {
            $Domain = $Domain.subString($Domain.lastIndexOf("OU="))
        }
        else {
            $Domain = $Domain.subString($Domain.lastIndexOf("CN="))
        }

        $Domain = $Domain.substring($Domain.indexOf(",") + 1).Trim()

        $GroupName = $ADGroup
        $GroupName = $GroupName.subString(0,$GroupName.indexOf(","))
        $GroupName = $GroupName -replace "CN=",""

        Write-Output "Domain: $Domain"
        Write-Output "Group Name: $GroupName"

        $FQDN = (Convert-HSCDNToFQDN $Domain).FQDN
        Write-Output "FQDN: $FQDN"

        if ($FQDN -eq $HSDomainFQDN) {
            $CurrentADGroup = Get-ADGroup -Identity $ADGroup

            if (($CurrentADGroup.GroupScope -eq "Universal") -OR
                ($CurrentADGroup.GroupScope -eq "DomainLocal") -OR
                ($CurrentADGroup.GroupScope -eq "Global")) {
                $s++
            }
        }
        else {
            $CurrentADGroup = Get-ADGroup -Identity $ADGroup -Server $FQDN

            if ($CurrentADGroup.GroupScope -eq "Universal") {
                $d++
            }
        }

        $CurrentADGroup
```

```
119
120         Write-Output "-------------------------"
121     }
122
123     Write-Output "d = $d"
124     Write-Output "s = $s"
125
126     $TokenSize = $TokenSize + (40*$d) + (8*$s)
127     Write-Output "Token Size: $TokenSize"
128
129     $TokenSizeObject = [PSCustomObject]@{
130         SamAccountName = $ADUser.SamAccountName
131         UserPrincipalName = $ADUser.UserPrincipalName
132         AccountCreationDate = $ADUser.whenCreated
133         AccountChangeDate = $ADUser.whenChanged
134         TokenSize = $TokenSize
135         d = $d
136         s = $s
137         DistinguishedName = $ADUser.DistinguishedName
138     }
139
140     $TokenSizeObject
141     $TokenSizeObject | Export-Csv $TokenSizeLog -NoTypeInformation -Append
142
143     Write-Output "************************"
144 }
145
146 Invoke-HSCExitCommand -ErrorCount $Error.Count
```