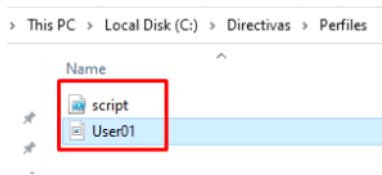


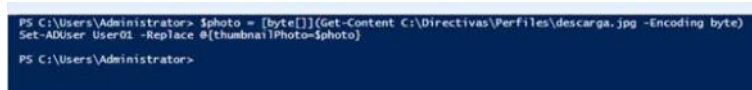
# GPO's AVATARES (Perfil Usuario)

martes, 3 de octubre de 2023 11:18

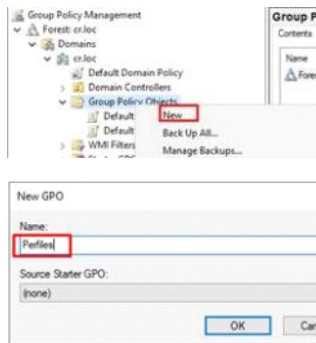
-Nuestro primer paso ha sido, dentro de la carpeta compartida llamada **Directivas**, crear una nueva carpeta nombrada **Perfiles**, en la que añadimos una foto a nuestro gusto (**User01**) y el script que se ejecutará en los equipos de los usuarios (**script**):



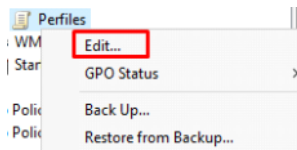
-A continuación, añadimos estos comandos, para asignar dicha foto (descarga) al usuario User01 en este caso, si tuviéramos más usuarios haríamos lo mismo, pero con otros usuarios y otras fotos:



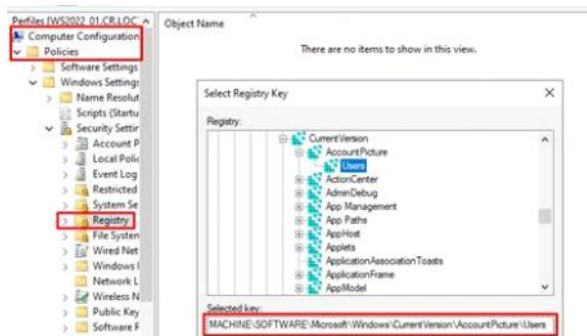
-Creamos una política nueva llamada Perfiles:



-Entramos a editarla dando click derecho sobre ella – **edit**:



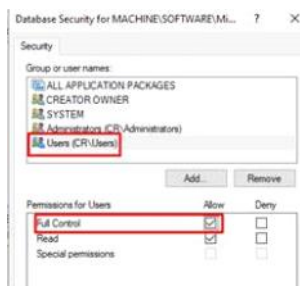
-Damos permisos control total a los usuarios para que puedan cambiar la foto de perfil en sus respectivos equipos:



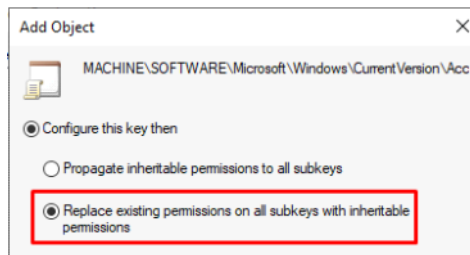
## -Script:

```
Function ResizeImage {
    Param (
        [Parameter(Mandatory = $True, HelpMessage = "image in byte")]
        [ValidateNotNull()]
        $imageSource,
        [Parameter(Mandatory = $True, HelpMessage = "Between 16 and 1000")]
        [ValidateRange(16, 1000)]
        $canvasSize,
        [Parameter(Mandatory = $True, HelpMessage = "Between 1 and 100")]
        [ValidateRange(1, 100)]
        $imgQuality = 100
    )
    [void][System.Reflection.Assembly]::LoadWithPartialName("System.Drawing")
    $imageBytes = [byte[]]$imageSource
    $ms = New-Object IO.MemoryStream($imageBytes, 0, $imageBytes.Length)
    $ms.Write($imageBytes, 0, $imageBytes.Length)
    $bmp = [System.Drawing.Image]::FromStream($ms, $true)
    # Image size after conversion
    $canvasWidth = $canvasSize
    $canvasHeight = $canvasSize
    # Set picture quality
    $smEncoder = [System.Drawing.Imaging.Encoder]::Quality
    $encoderParams = New-Object System.Drawing.Imaging.EncoderParameters(1)
    $encoderParams.Param(0) = New-Object System.Drawing.Imaging.EncoderParameter($smEncoder, $imgQuality)
    # Get image type
    $imgImageCodeInfo = [System.Drawing.Imaging.ImageCodeInfo]::GetImageEncoders() | Where-Object { $_.MimeType -eq "image/jpeg" }
    # Get aspect ratio
    $ratioX = $bmp.Width / $bmp.Height
    $ratioY = $canvasHeight / $canvasWidth
    $ratio = $ratioX / $ratioY
    if ($ratioX -lt $ratioY) {
        $ratio = $ratioX
    }
    # Create an empty picture
    $newWidth = [int]($bmp.Width * $ratio)
    $newHeight = [int]($bmp.Height * $ratio)
    $bmpResized = New-Object System.Drawing.Bitmap($newWidth, $newHeight)
    $graph = [System.Drawing.Graphics]::FromImage($bmpResized)
    $graph.Clear([System.Drawing.Color]::White)
    $graph.DrawImage($bmp, 0, 0, $newWidth, $newHeight)
    # Create an empty stream
    $ms = New-Object IO.MemoryStream
    $bmpResized.Save($ms, $imgImageCodeInfo, $encoderParams)
    # cleanup
    $bmpResized.Dispose()
    $bmp.Dispose()
    return $ms.ToArray()
}

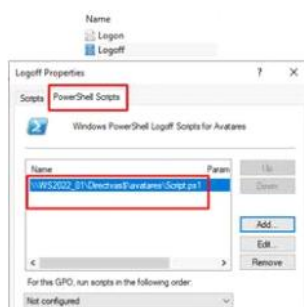
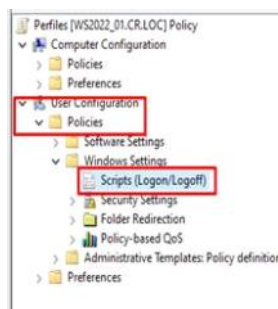
$ADUserInfo = ([ADSISearcher]("&(objectCategory=User)&(SAMAccountName=ServiceName)")).FindOne().Properties
$ADUserInfo_sids = [System.Security.Principal.WindowsIdentity]::GetCurrent().User.Value
If ($ADUserInfo.thumbnailphoto) {
    $img_sizes = @(32, 40, 48, 96, 192, 200, 240, 448)
    $img_base = "C:\Users\Public\AccountPictures\"
    $reg_key = "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\AccountPicture\Users\$ADUserInfo_sids"
    If ((Test-Path -Path $reg_key -eq $false) | New-Item -Path $reg_key | Write-Verbose "Reg key exist [$reg_key]" )
    Try {
        ForEach ($size in $img_sizes) {
            $sid = $img_base + $size + $ADUserInfo_sids
            If ((Test-Path -Path $sid -eq $false) | New-Item -Item -Path $sid -Attributes "Hidden" )
            $file_name = "ImageSize_$size.jpg"
            $spath = $sid + $file_name
            Write-Verbose "Create file: [$file_name]"
            try {
                $imgSource = $ADUserInfo.thumbnailphoto -canvasSize $size -imgQuality 100 -Set-Content -Path $spath -Encoding Byte -Force -ErrorAction Stop
                Write-Verbose "File saved: [$file_name]"
            }
            catch {
                If (Test-Path -Path $spath) {
                    Write-Warning "File exist [$spath]"
                }
                else {
                    Write-Warning "File not exist [$spath]"
                }
            }
            $name = "ImageSize"
            try {
                $null = New-ItemProperty -Path $reg_key -Name $name -Value $spath -Force -ErrorAction Stop
            }
            catch {
                Write-Warning "Reg key edit error [$reg_key] [$name]"
            }
        }
    }
    Catch {
        Write-Error "Check permissions to files or registry."
    }
}
```



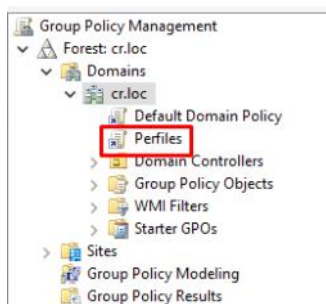
-Reemplazamos los permisos anteriores y añadimos los nuevos:



-Una vez activada la política de equipo, entramos en las políticas de usuario, para especificar que, cuando los usuarios tengan la sesión cerrada, ejecute el equipo automáticamente el script creado:



Añadimos la ruta compartida, si no, los usuarios no reconocen la ruta si añadimos la local.  
-Linkamos la directiva de Perfiles al dominio, ya que es, tanto de tipo equipo como de tipo usuario:



-Y por último forzamos la actualización de las GPO's:

```
C:\Users\Administrator>gpupdate /force
Updating policy...

Computer Policy update has completed successfully.
User Policy update has completed successfully.
```

-Comprobación (reiniciamos el equipo y cerramos sesión una o dos veces, para que se apliquen las configuraciones):

