Gestión de permisos en Windows.

Contents

1 Gestión de permisos desde el entorno gráfico

1.1 Los SID de usuarios

Cada vez que abre sesión, la información de identificación utilizada por el usuario (nombre de usuario y contraseña) se transfiere a un monitor de seguridad local que accede al Administrador de seguridad (SAM de Security Account Manager). Este último, asignará un token de acceso que determinará los derechos de acceso que posee ese usuario para todos los objetos "asegurables" (claves del Registro, archivos, carpetas, servicios, procesos, etc.) Este descriptor de seguridad revisa dos informaciones:

- El SID del usuario.
- La lista DACL del objeto al que intenta acceder el usuario.

A continuación, explicaremos estas dos nociones.

Un SID (Security IDentifier) es una manera única de identificar a un usuario o grupo de usuarios. Podemos encontrar estos identificadores en los token de acceso, en las ACL (Access Control List) y en las bases de seguridad de cuentas. Diríjase al apartado siguiente para ver una descripción completa sobre el mecanismo de las ACL.

Los SID son datos de longitud variable que forman una representación jerárquica del actor designado. La sintaxis es la siguiente: S-R-I-XXX-XXX-XXX.

- S: la letra S (para recordar que se trata de un SID).
- R: el número del formato binario de SID.
- I: número entero que identifica la autoridad que ha emitido el SID.
- XXX-XXX-XXX: serie de longitud variable, formada de identificadores de subautoridad o identificadores relativos (Relative IDentifier o RID).

Puede visualizar los SID de esta manera, en la línea de comandos, teclee:

```
whoami /all.
```

Se muestra la información siguiente:

- El SID correspondiente al grupo Administradores es S-1-5-32-544.
- La autoridad que ha emitido este SID tiene como identificador el número 5.
- La subautoridad tiene como identificador el número 32.
- 544 es el RID del grupo Administradores.

Puede comprobar los resultados mostrados con los siguientes comandos:

- whoami
- whoami /user /priv
- whoami /groups

Se mostrarán los privilegios del usuario que está conectado en ese momento. Puede obtener algunos SID de usuarios o entidades de seguridad abriendo este árbol de Registro: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\Current Version\ProfileList.

Finalmente, los SID de algunas entidades se muestran en este otro árbol: HKEY_USERS.

1.2 Las listas de control de acceso

Una lista de control de acceso discrecional (DACL o Discretionary Access Control Lists, comúnmente llamadas ACL) es un mecanismo que permite proteger recursos como los archivos y las claves del Registro. Las DACL contienen las entradas de control de acceso (ACE o Access Control Entry) que funcionan como registros para cada usuario o grupo de usuarios que su SID señala. Estas entradas asocian una entidad de seguridad (una cuenta de usuario, un grupo de cuentas, una entidad de sistema) con una regla que define el uso del recurso. Las DACL y las ACE permiten aceptar o rechazar los privilegios de acceso a los recursos según los permisos que usted quiera darle a las cuentas de usuario. También puede crear una ACE y aplicarla a la DACL de un archivo para impedir que nadie, salvo un administrador, pueda modificar el archivo.

Una lista de control de acceso de sistema (SACL o "ACE de auditoría") es un mecanismo que controla los mensajes de auditoría asociados a un recurso. Las SACL contienen ACE que definen las reglas de auditoría para un recurso determinado.

Así pues, podrá utilizar las DACL para asegurarse de que solo un administrador puede modificar un archivo y las SACL para asegurarse de que se guarden todos los intentos conseguidos de apertura del archivo. Es posible distinguir las ACE positivas y ACE negativas:

En el Explorador de Windows, abra su directorio de usuario.

- Cree una nueva carpeta llamada Prueba.
- Haga clic con el botón secundario del ratón en el submenú **Propiedades**.
- Haga clic en la pestaña **Seguridad**.

Tenga en cuenta que las ACE o autorizaciones visibles aparecen en gris, ya que heredan de la carpeta padre.

De hecho, la carpeta que acaba de crear ha heredado los permisos en vigor de la carpeta principal. Este mecanismo de encadenamiento se conoce como "herencia" y lo primero que haremos será desactivarlo:

- Haga clic en el botón **Opciones avanzadas**.
- Haga clic en el botón **Deshabilitar herencia**.
- Haga clic en **Convertir los permisos heredados en permisos explícitos en este objeto**.
- Haga clic en **Aceptar**.
- A continuación, haga clic en **Aceptar**.
- Haga clic en el botón **Editar**.
- Seleccione su nombre de usuario.

Ahora puede seleccionar las casillas **Denegar** para configurar una ACE negativa.

Cuando el sistema inicia una comprobación de acceso, empezará de manera sistemática, por las ACE negativas. Así pues, los permisos "Denegar" siempre tienen prioridad sobre los permisos "Permitir".

Existe una particularidad en los sistemas operativos NT: cuando un usuario crea un archivo, él es el propietario (owner). El SID del propietario se coloca en el descriptor de seguridad que el sistema de archivos NTFS tiene para el objeto correspondiente. El propietario tiene permisos para leer el descriptor de seguridad y así, por ejemplo, modificar la ACL de un archivo.

Para conocer el propietario de la carpeta que acaba de crear, haga clic en la pestaña **Seguridad**, después en el botón **Opciones avanzadas**. Se visualiza directamente el propietario de la carpeta. Puede hacer clic en el enlace **Modificar** para cambiar de propietario.

El propietario de un objeto siempre tiene permisos para leer y modificar la DACL de los objetos que él ha creado, motivo por el que el control de acceso se califica de discrecional (está a discreción del propietario).

1.3 Tomar posesión de un objeto

Para tomar posesión de un objeto, acceda a la pestaña **Seguridad** y después Haga clic en el botón **Opciones avanzadas**. Se visualiza directamente el propietario de la carpeta. Por defecto, será su cuenta de usuario la que aparezca como el propietario del recurso. Esto se puede cambiar rápidamente de la siguiente manera:

- Haga clic en el enlace **Cambiar** (debe buscar o directamente introducir la cuenta o grupo de usuarios que desea definir como propietarios. Puede añadir otros grupos de usuarios haciendo clic en el botón correspondiente).
- Seleccione el grupo de administradores y haga clic en **Aplicar** (si desea que esta operación se aplique a todos los objetos secundarios, seleccione la casilla **Remplazar el propietario en subcontenedores y objetos**).

Un cuadro de diálogo le avisará de que tendrá que cerrar las propiedades del objeto para que el cambio de propietario sea efectivo.

1.4 Utilizar los permisos NTFS

Pongamos ahora el ejemplo de un administrador llamado Juan que desea compartir una carpeta con permiso de escritura con un usuario llamado Marcos y con permiso de solo lectura con otra usuaria llamada Ana.

Primero cree una carpeta llamada **Prueba**, dentro de la carpeta Users\Public\Documents. Dentro de esta, cree el archivo que deberá ser visible, lo puede llamar Fichero.txt.

Cualquier usuario tendrá acceso a la carpeta y podrá modificar el documento, ya que la entidad de sistema **INTERACTIVE** posee las autorizaciones especiales para el contenido de esta carpeta. La entidad reúne a todos los usuarios que han abierto una sesión interactiva en Windows.

Empiece por desactivar el mecanismo de herencia, copiar los permisos y eliminar el grupo **INTERACTIVE**.

La carpeta ya no será accesible para los usuarios Marcos y Ana.

Hay que señalar que debido a que usted forma parte del grupo de administradores, no tendrá ningún problema de acceso a la carpeta.

Una vez que ha realizado este primer paso, añada el usuario llamado Ana. Ana podrá visualizar el contenido del archivo, pero no podrá eliminarlo, modificarlo ni crear otros documentos. Por defecto, las tres autorizaciones genéricas que se han añadido son las siguientes: **Lectura y ejecución** - **Mostrar el contenido de la carpeta** - **Lectura**.

Ahora añada el usuario llamado Marcos:

- Haga clic en el botón **Opciones avanzadas**, y seleccione el usuario **Marcos**.
- Haga clic en el botón **Editar**, y a continuación en el enlace **Mostrar permisos avanzados**.
- Seleccione estas cuatro casillas:
  - **Crear archivos/escribir datos**
  - **Crear carpetas/anexar datos**
  - **Escribir atributos**
  - **Escribir atributos extendidos**

Por lo que respecta al usuario Marcos, este puede editar el contenido del archivo y añadir otros documentos, pero en ningún caso podrá:

- Cambiar el conjunto de permisos NTFS.
- Tomar posesión de la carpeta.
- Eliminar la carpeta o el archivo.

2 Gestión de permisos desde el CMD

2.1 Tomar posesión de un directorio

El comando TakeOwn permite a un administrador (en Windows) recuperar el acceso que se le ha denegado a un archivo al haberse cambiado el propietario del archivo.

La sintaxis es la siguiente:

```
TAKEOWN [/S sistema] [/U usuario [/P contraseña]]] /F~
nombre_de_archivo~ [/A] [/R [/D linea_de_comandos]]
```

Los modificadores son:

- /s: indica el sistema remoto al que conectarse.
- /u: [dominio\]usuario: especifica el contexto de usuario en el que el comando debe ejecutarse. Este modificador no puede utilizarse sin /s.
- /p: [contraseña]: define la contraseña del contexto de un usuario determinado.
- /f : nombre_de_archivo: indica el nombre del archivo o directorio. Puede utilizar el carácter genérico * para englobar varios archivos.
- /a: asigna la posesión al grupo de administradores y no al usuario actual. Este modificador no es específico, la posesión del archivo se asignará al usuario conectado en ese momento.
- /r: trata el comando en modo recursivo. La operación se realizará en un conjunto de directorios y subdirectorios.
- /d: línea_de_comandos: permite definir una respuesta predeterminada que se utilizará aunque el usuario actual no posea el permiso "mostrar lista de carpetas" en un directorio. Esto se produce durante el proceso recursivo (/R) de subdirectorios. Utilice los valores "O" para tomar posesión o "N" para ignorar.

Aquí le mostramos un ejemplo de uso.

Después de una instalación de Windows, algunos directorios situados en otra partición ya no son accesibles, ni siquiera desde una cuenta de usuario con privilegios de administrador. La explicación es sencilla: las ACL se configuran en función del SID que ya no existe en el sistema.

En este caso, puede utilizar estos dos comandos:

- takeown /f Nombre_del_directorio /r /d
- icacls Nombre_del_directorio /setowner usuario [/T] [/C] [/L]

¡El acceso al directorio será entonces posible! Tenga en cuenta que deberá ejecutar el símbolo del sistema como administrador, de lo contrario aparecerá un mensaje que indica que el acceso ha sido denegado.

2.2 Modificar las listas de control de acceso

Mediante el símbolo del sistema, puede modificar las ACL de los archivos utilizando una herramienta llamada icacls.

El comando iCACLS permite mostrar o cambiar una lista de control de acceso (ACL) para los archivos y carpetas del sistema de archivos.

Para mostrar los permisos actuales de NTFS en una carpeta específica (por ejemplo, c:\prueba), abra un símbolo del sistema y ejecute el comando:

```
icacls c:\prueba
```

Este comando devolverá una lista de todos los usuarios y grupos a los que se les han asignado permisos en este directorio. Intentemos entender la sintaxis de los permisos devueltos por el comando iCACLS:

```
c:\prueba Nombre de usuario:(OI)(CI)(M)
          BUILTIN\Administradores:(I)(OI)(CI)(F)
          BUILTIN\Usuarios:(I)(OI)(CI)(RX)
          CREATOR_OWNER:(I)(OI)(CI)(IO)(F)

Procesado con éxito 1 archivo; error al procesar 0 archivos
```

El nivel de acceso a los recursos se especifica después de cada grupo o usuario. Los permisos de acceso se indican mediante las abreviaturas. A continuación se muestra una lista completa de los permisos que se pueden establecer usando la utilidad icacls:

- Configuración de la herencia de iCACLS:
  - (OI) – objeto heredado;
  - (CI) – contenedor heredado;
  - (IO) – sólo heredar;
  - (NP) – no propagar heredar;
  - (I) – permiso heredado del contenedor padre.
- Lista de permisos de acceso básicos:
  - D – borrar el acceso;
  - F – acceso completo;
  - N – sin acceso;
  - M – modificar el acceso;
  - RX – acceso de lectura y ejecución;
  - R – acceso de sólo lectura;
  - W – acceso de sólo escritura.
- Lista de permisos avanzados:
  - DE – borrar;
  - RC – control de lectura;
  - WDAC – escribe DAC;
  - WO – escribe propietario;
  - S – sincronizar;
  - AS – seguridad del sistema de acceso;
  - MA – permisos máximos permitidos;
  - GR – lectura genérica;
  - GW – escritura genérica;
  - GE – genérico ejecutar;
  - GA – genérico todos;
  - RD – directorio de datos/listas de lectura;
  - WD – escribir datos/añadir archivo;
  - AD – añadir datos/subdirectorio AD;
  - REA – leer atributos extendidos;
  - WEA – escribir atributos extendidos;
  - X – ejecutar/travesar;
  - DC – borrar niño;

- RA – atributos de lectura;
- WA – atributos de escritura.

Si necesita encontrar todos los objetos en el directorio especificado y sus subdirectorios en los que se especifica el SID de un usuario y grupo específico, utilice el comando:

```
icacls c:\prueba / findsid [User/Group_SID] /t /c /l /q
```

2.3 Guardar y restaurar los ACL de NTFS usando ICACLS.

Usando el comando icacls, puedes guardar el ACL del objeto actual en un archivo de texto, y luego aplicar la lista de permisos guardados al mismo u otros objetos (una especie de forma de ACL de respaldo).

Para exportar las ACL actuales a la carpeta c:\prueba y guardarlas en el archivo folder_ACLs.txt, ejecute el comando:

```
icacls c:\prueba /save c:\temp\folder_ACLs.txt /t
```

Este comando guarda las ACLs no sólo del propio directorio, sino en todas las subcarpetas y archivos. El archivo de texto resultante se puede abrir con el Bloc de notas de Windows y enumera los SID de usuario y la lista de permisos mediante la sintaxis SDDL.

Para aplicar los ACLs de acceso guardados (restaurar permisos), ejecute el comando:

```
icacls c:\prueba /restore c:\temp\folder_ACLs.txt
```

Por lo tanto, el proceso de transferencia de ACLs de una carpeta a otra se hace mucho más fácil.

2.4 Ejemplos de uso de iCACLS para otorgar/modificar permisos de carpetas o archivos

Con el comando icacls, puedes cambiar las listas de acceso a la carpeta. Por ejemplo, quieres conceder al usuario John los permisos para editar el contenido de la carpeta c:\prueba. Ejecute el comando:

```
icacls c:\prueba /grant John:M
```

Conceder al grupo de NYUsers un permiso de control total y aplicar toda la configuración a las subcarpetas:

```
icacls "c:\prueba" /grant NYUsers:F /Q /C /T
```

Puedes quitar todos los permisos de John usando el comando:

```
icacls c:\prueba /remove John
```

Además, puede impedir que un usuario o grupo de usuarios accedan a un archivo o carpeta utilizando la denegación explícita de la manera siguiente:

```
icacls c:\prueba /deny "NYUsers:(CI)(M)"
```

Tengan en cuenta que las normas de prohibición tienen mayor prioridad que las de autorización.

Puedes activar o desactivar los permisos de los objetos de la carpeta/archivo usando la opción /inherencia del comando icacls. Para desactivar los permisos de herencia en el objeto del sistema de archivos y copiar el control de acceso actual (permisos explícitos), ejecute la lista de comandos:

```
icacls c:\prueba /inherencia:d
```

Para desactivar la herencia y eliminar todos los permisos heredados, corre:

```
icacls c:\prueba /inherencia:r
```

Para habilitar los permisos heredados en el objeto de archivo o carpeta:

```
icacls c:\prueba /inherencia:e
```

En algunos casos, es posible que reciba el error " **Se niega el acceso** " al tratar de cambiar los permisos de un archivo o carpeta utilizando la herramienta icacls. En este caso, primero asegúrate de que ejecutas la ventana de cmd con derechos elevados (ejecuta como administrador). Dado que la herramienta icacls no es una herramienta de UAC, no verás la solicitud de elevación.

Si el error persiste, enumere los permisos actuales del archivo y asegúrese de que su cuenta tiene los derechos de "Cambiar permisos" en el archivo.

3 Gestión de permisos con PowerShell

3.1 Mostrar ACL para archivos y carpetas

El primer cmdlet de PowerShell utilizado para administrar los permisos de archivos y carpetas es "get-acl"; enumera todos los permisos de objeto. Por ejemplo, obtengamos la lista de todos los permisos para la carpeta con la ruta del objeto " *\fs1\shared\sales"* :

```
get-acl C:\MyFolder | Format-List
```

3.2 Copiar permisos de archivos y carpetas

Para copiar permisos, un usuario debe poseer tanto la carpeta de origen como la de destino. El siguiente comando copiará los permisos de la carpeta "Contabilidad" a la carpeta "Ventas":

```
get-acl C:\MyFolder | Set-Acl C:\NewFolder
```

Como podemos ver en la salida de los comandos "get-acl" antes y después de la copia de permisos, los permisos de la carpeta compartida "Ventas" se han cambiado.

3.3 Establecer permisos de archivos y carpetas

El cmdlet "set-acl" de PowerShell se usa para cambiar el descriptor de seguridad de un elemento específico, como un archivo, una carpeta o una clave de registro; es decir, se utiliza para modificar permisos de archivos o carpetas. La siguiente secuencia de comandos establece el permiso "Control total" para para "username" en la carpeta "MyFolder":

```
$acl = Get-Acl C:\MyFolder

$AccessRule = New-Object System.Security.AccessControl.FileSystemAccessRule(" ↩
    username","FullControl","Allow")

$acl.SetAccessRule($AccessRule)

$acl | Set-Acl C:\MyFolder
```

Si desea establecer otros permisos para usuarios o grupos de seguridad, selecciónelos de la siguiente tabla:

Access Right	Access Right's Name in PowerShell
Full Control	FullControl
Traverse Folder / Execute File	ExecuteFile
List Folder / Read Data	ReadData
Read Attributes	ReadAttributes
Read Extended Attributes	ReadExtendedAttributes
Create Files / Write Data	CreateFiles
Create Folders / Append Data	AppendData
Write Attributes	WriteAttributes
Write Extended Attributes	WriteExtendedAttributes
Delete Subfolders and Files	DeleteSubdirectoriesAndFiles
Delete	Delete
Read Permissions	ReadPermissions

También hay conjuntos de permisos de derechos de acceso básicos que se pueden aplicar:

Access Rights Set	Rights Included in the Set	Name of the Set in PowerShell
Read	List Folder / Read Data	Read
	Read Attributes	
	Read Extended Attributes	
	Read Permissions	
Write	Create Files / Write Data	Write
	Create Folders / Append Data	
	Write Attributes	
	Write Extended Attributes	
Read and Execute	Traverse folder / Execute File	ReadAndExecute
	List Folder / Read Data	
	Read Attributes	
	Read Extended Attributes	
	Read Permissions	
Modify	Traverse folder / Execute File	Modify
	List Folder / Read Data	
	Read Attributes	
	Read Extended Attributes	
	Create Files / Write Data	
	Create Folders / Append Data	
	Write Attributes	
	Write Extended Attributes	
	Delete	
	Read Permissions	

3.4 Eliminar permisos de usuario

Para eliminar un permiso, utilice el parámetro "RemoveAccessRule". Eliminemos el permiso "Control total" para "username" en la carpeta "MyFolder":

```
$acl = Get-Acl C:\MyFolder

$AccessRule = New-Object System.Security.AccessControl.FileSystemAccessRule(" ↩
    username","FullControl","Allow")

$acl.RemoveAccessRule($AccessRule)

set-acl -path C:\MyFolder -AclObject $acl
```

Tenga en cuenta que T.Simpson todavía tiene el permiso "Denegar control total". Para eliminarlo, usemos el comando "PurgeAccessRules", que borrará por completo los permisos de "username" en la carpeta "MyFolder":

```
$acl = Get-Acl C:\MyFolder
```

```
$usersid = New-Object System.Security.Principal.Ntaccount ("username")

$acl.PurgeAccessRules($usersid)

$acl | Set-Acl C:\MyFolder
```

Tenga en cuenta que "PurgeAccessRules" no funciona con un nombre de usuario de cadena; solo funciona con SID. Por lo tanto, usamos la clase "Ntaccount" para convertir el nombre de la cuenta de usuario de una cadena en un SID. También tenga en cuenta que "PurgeAccessRules" solo funciona con permisos explícitos; no purga los heredados.

3.5 Deshabilitar o habilitar la herencia de permisos

Para administrar la herencia, usamos el método "SetAccessRuleProtection". Tiene dos parámetros:

- El primer parámetro es responsable de bloquear la herencia de la carpeta principal. Tiene dos estados: “$true” y “$false”.
- El segundo parámetro determina si los permisos heredados actuales se conservan o eliminan. Tiene los mismos dos estados: “$true” y “$false”.

Desactivemos la herencia para la carpeta "MyFolder" y eliminemos también todos los permisos heredados:

```
$acl = Get-Acl C:\MyFolder

$acl.SetAccessRuleProtection($true, $false)

$acl | Set-Acl C:\MyFolder
```

Ahora solo nos queda un permiso de acceso (porque se agregó explícitamente); se eliminaron todos los permisos heredados.

Revirtamos este cambio y habilitemos la herencia para la carpeta "MyFolder" nuevamente:

```
$acl = Get-Acl C:\MyFolder

$acl.SetAccessRuleProtection($false, $true)

$acl | Set-Acl C:\MyFolder
```

3.6 Cambiar la propiedad de archivos y carpetas

Si desea establecer un propietario para una carpeta, debe ejecutar el método "SetOwner". Hagamos que “username” sea el propietario de la carpeta “Ventas”:

```
$acl = Get-Acl C:\MyFolder

$user = New-Object System.Security.Principal.Ntaccount(“username”)

$acl.SetOwner($user)

$acl | Set-Acl C:\MyFolder
```

Tenga en cuenta que nuevamente usamos la clase "Ntaccount" para convertir el nombre de la cuenta de usuario de una cadena en un SID.

3.7 Gestión de permisos con funciones propiedades

A continuación se muestra un ejemplo de cuatro funciones que permiten gestionar los permisos de una manera simplificada:

```
function global:Add-AccessRule($Path, $User, $Grant, $Type)
{
    $acl=Get-Acl $Path
    $AccessRule=New-Object System.Security.AccessControl.FileSystemAccessRule($user, $Grant ↩
        , "ContainerInherit, ObjectInherit", "None", $Type)
    $acl.SetAccessRule($AccessRule)
    Set-Acl $Path -AclObject $acl
}

function global:Get-AccessRule($Path)
{
    (Get-Acl $Path).Access
}

function global:Remove-AccessRule($Path, $User, $Grant, $Type)
{
    $acl = Get-Acl $Path
    $AccessRule = New-Object System.Security.AccessControl.FileSystemAccessRule($user,  ↩
        $Grant, "ContainerInherit, ObjectInherit", "None", $Type)
    $acl.RemoveAccessRule($AccessRule)
    set-acl -path $Path -AclObject $acl
}

function global:Purge-AccessRule($Path, $User)
{
    $acl = Get-Acl $Path
    $usersid = New-Object System.Security.Principal.Ntaccount ($User)
    $acl.PurgeAccessRules($usersid)
    set-acl -path $Path -AclObject $acl
}
```