P3.4-AUTOMATIZACIÓN EN LA CREACIÓN DE CUENTAS DE USUARIO (W-L)

Cristóbal Suárez Abad

ADMINISTRACIÓN DE SISTEMAS OPERATIVOS - 2º ASIR

Índice

Introducción:

Automatización de administración de cuentas

El departamento de sistemas ha detectado:

- Cuentas activas de empleados que ya no trabajan en la empresa.
- Altas realizadas manualmente con errores de permisos.
- Falta de trazabilidad y auditoría.
- Ausencia de automatización en el ciclo de vida de las cuentas.

Tienes que diseñar e implementar un **sistema automatizado, seguro y auditable** de gestión de cuentas.

Datos para el csv:

 # username,nombre,apellido,departamento,rol,fecha_fin
jgarcia,Juan,Garcia,IT,admin,2025-06-30
mmartin,Maria,Martin,RRHH,user,2025-03-15
aperez,Ana,Perez,Marketing,user,2025-02-01

Linux:

Crear usuario de servicio:

sudo useradd -r -s /usr/sbin/nologin user_mgmt

-r (system): Crea un usuario de sistema. Estos usuarios no tienen fecha de expiración y suelen tener un ID bajo (menor a 1000).

-s /usr/sbin/nologin: Define la "shell" del usuario. Al poner nologin, te aseguras de que nadie pueda entrar físicamente al servidor usando esta cuenta. Es una medida de seguridad crítica.

user_mgmt: Es el nombre del usuario que estás creando.

sudo mkdir /opt/user_mgmt

Crea una carpeta llamada user_mgmt dentro de /opt/. El directorio /opt/ es el lugar estándar en Linux para instalar paquetes de software que no forman parte del sistema operativo base.

sudo chown user_mgmt:user_mgmt /opt/user_mgmt

Cambia el dueño de la carpeta. Ahora, el usuario user_mgmt y el grupo user_mgmt son los propietarios legales de ese directorio. Esto permite que el servicio que ejecutes con ese usuario pueda escribir o leer ahí dentro.

sudo chmod 750 /opt/user_mgmt

Este comando define quién puede hacer qué con la carpeta usando un esquema numérico:

- **7 (Propietario - user_mgmt):** Tiene control total (Lectura, Escritura y Ejecución).
- **5 (Grupo - user_mgmt):** Puede entrar a la carpeta y leer archivos, pero **no** puede crear ni borrar nada.
- **0 (Otros):** El resto de los usuarios del sistema no pueden ni siquiera ver qué hay dentro.

Logs del usuario de servicio:

sudo mkdir /var/log/user_mgmt

sudo chown user_mgmt:user_mgmt /var/log/user_mgmt

Le entrega la propiedad de esa carpeta al usuario de sistema que creaste antes. Esto es fundamental para que el servicio tenga permiso de escribir sus logs ahí.

sudo chmod 750 /var/log/user_mgmt

sudo chmod 750 /var/log/user_mgmt: Establece los permisos de seguridad:

- **7 (Dueño):** El servicio puede leer, escribir y borrar sus propios logs.
- **5 (Grupo):** Los miembros del grupo pueden leer los logs (útil para auditoría).
- **0 (Otros):** Nadie más en el sistema puede ver los logs.

Archivo CSV

Lo colocamos en "/opt/user_mgmt/users.csv"

```
# username,nombre,apellido,departamento,rol,fecha_fin
jgarcia,Juan,Garcia,IT,admin,2025-06-30
mmartin,Maria,Martin,RRHH,user,2025-03-15
aperez,Ana,Perez,Marketing,user,2025-02-01
```

Crear un script que:

- Añada usuarios desde un archivo CSV.
- Compara la fecha_fin con la fecha actual para desabilitar los necesarios
- Asigne grupos y expiración de contraseña.
- Genere un log con la fecha y los usuarios creados y los desabilitados.
- Programe su ejecución automática con cron.
- Programa una revisión diaria de los usuarios a desabilitar con systemd timer.

“/opt/user_mgmt/manage_users.sh”

```
#!/bin/bash

CSV="/opt/user_mgmt/users.csv"
LOG="/var/log/user_mgmt/user_mgmt.log"
DATE_NOW=$(date +%F)
ERROR=0

exec >> "$LOG" 2>&1

echo "[$(date)] Inicio ejecución"

# Seguridad básica
if [[ $EUID -ne 0 ]]; then
  echo "ERROR: Debe ejecutarse como root"
  exit 1
fi

# Validar CSV
EXPECTED_FIELDS=6
cut -d',' -f1 "$CSV" | tail -n +2 | sort | uniq -d | grep . && {
  echo "ERROR: Usuarios duplicados en CSV"
  exit 1
}

while IFS=',' read -r username nombre apellido dept rol fecha_fin; do
   [[ "$username" == "username" ]] && continue

   # Validar campos
   [[ -z "$username" || -z "$rol" || -z "$fecha_fin" ]] && ERROR=1
```

```
  # Bloqueo de admins
  if [[ "$rol" == "admin" && "$dept" != "IT" ]]; then
    echo "ALERTA: Intento de admin no autorizado ($username)"
    exit 1
  fi
done < "$CSV"

[[ $ERROR -eq 1 ]] && {
 echo "ERROR: CSV mal formado"
 exit 1
}

# Procesar usuarios
while IFS=',' read -r username nombre apellido dept rol fecha_fin; do
  [[ "$username" == "username" ]] && continue

  if ! id "$username" &>/dev/null; then
    useradd -m -c "$nombre $apellido" "$username"
    passwd -e "$username"

    if [[ "$rol" == "admin" ]]; then
      usermod -aG sudo "$username"
    else
      usermod -aG users "$username"
    fi

    chage -E "$fecha_fin" "$username"
    echo "Usuario creado: $username"
  fi

  # Deshabilitar si procede
  if [[ "$fecha_fin" < "$DATE_NOW" ]]; then
    usermod -L "$username"
    echo "Usuario deshabilitado: $username"
  fi

done < "$CSV"

echo "[$(date)] Fin ejecución"
```

Permisos del Script:

chmod 750 manage_users.sh
chown user_mgmt:user_mgmt manage_users.sh

Cron (altas programadas)

sudo crontab -u user_mgmt -e

Introduce:

0 2 * * * /opt/user_mgmt/manage_users.sh

Systemd Timer (revisión diaria)

Servicio

nano /etc/systemd/system/user-expire.service

[Unit]

Description=Revisión de usuarios expirados

[Service]

Type=oneshot

ExecStart=/opt/user_mgmt/manage_users.sh

Es un **archivo de unidad de Systemd**, que le indica a Linux cómo y cuándo ejecutar el script que mencionamos antes.

Timer

nano /etc/systemd/system/user-expire.timer

[Unit]
Description=Ejecución diaria revisión usuarios

[Timer]
OnCalendar=daily
Persistent=true

[Install]
WantedBy=timers.target

Este es el **activador automático** (el "reloj despertador"). Un archivo .timer de systemd se utiliza para ejecutar un .service de forma programada, sustituyendo en muchos casos al antiguo cron.

systemctl daemon-reexec

Reinicia el proceso gestor de systemd (el PID 1) sin reiniciar el sistema operativo.

systemctl enable --now user-expire.timer

enable**:** Configura el timer para que se inicie automáticamente cada vez que el servidor se encienda (crea un enlace simbólico en timers.target).

--now**:** Activa el timer inmediatamente en este preciso momento. Si no usaras esto, el timer estaría "listo" pero no empezaría a contar hasta el próximo reinicio.

Con estos comandos finales, has puesto en marcha oficialmente todo el sistema.

¿Cómo verificar que todo está funcionando?

Ahora que el motor está encendido, puedes usar estos comandos de control para dormir tranquilo:

Comprobar el cronómetro:

systemctl list-timers user-expire.timer

(Verás cuándo fue la última vez que corrió y cuánto falta para la próxima).

```
[root@server2asir user_mgmt]$systemctl list-timers user-expire.timer
NEXT                         LEFT     LAST PASSED UNIT              ACTIVATES
Tue 2025-12-30 00:00:00 UTC 14h left n/a  n/a    user-expire.timer user-expire.service

1 timers listed.
Pass --all to see loaded but inactive timers, too.
 Cristobal CON ROOT  lunes 29 diciembre 2026 09:44
```

Ver si el servicio se activó correctamente:

systemctl status user-expire.service

```
[root@server2asir user_mgmt]$systemctl status user-expire.service
○ user-expire.service - Revisión de usuarios expirados
     Loaded: loaded (/etc/systemd/system/user-expire.service; static)
     Active: inactive (dead)
TriggeredBy: ● user-expire.timer
 Cristobal CON ROOT  lunes 29 diciembre 2026 09:45
```

(Aunque el timer esté activo, el servicio solo aparecerá como "inactive (dead)" si ya terminó su tarea, lo cual es normal para un oneshot).

Ver los logs (la prueba de fuego): Como configuraste la carpeta de logs antes, podrías revisar si el script escribió algo:

ls -l /var/log/user_mgmt

cat /var/log/user_mgmt/user_mgmt.log

Solución problemas ejecución script:

- Arreglar formato del archivo

Convierte a formato Unix:

apt install dos2unix

dos2unix manage_users.sh

- O Problemas de permisos:

chmod +x manage_users.sh

./manage_users.sh

Windows

Un script PowerShell que:

- Cree usuarios desde un CSV. Son usuarios de AD
- Asigne OU, grupos o políticas básicas segun su rol.
- Se ejecute con el Programador de tareas.

Archivo CSV:

Mismo CSV, ubicado por ejemplo en: **C:\scripts\users.csv**

```
# username,nombre,apellido,departamento,rol,fecha_fin
jgarcia,Juan,Garcia,IT,admin,2025-06-30
mmartin,Maria,Martin,RRHH,user,2025-03-15
aperez,Ana,Perez,Marketing,user,2025-02-01
```

Script PowerShell

manage-users.ps1

```
# ================================
# CONFIGURACIÓN
# ================================
$CSV = "C:\scripts\users.csv"
$LOG = "C:\scripts\user_mgmt.log"
$BaseDN = "DC=empresa,DC=local"
$Today = Get-Date

# ================================
# LOGGING
# ================================
function Write-Log {
  param ([string]$Message)
  Add-Content -Path $LOG -Value "[$(Get-Date -Format 'yyyy-MM-dd HH:mm:ss')] $Message"
}

Write-Log "===== INICIO EJECUCIÓN ====="
```

```
# ==============================
# COMPROBACIONES INICIALES
# ==============================
if (-not (Get-Module -ListAvailable -Name ActiveDirectory)) {
  Write-Log "ERROR CRÍTICO: Módulo ActiveDirectory no disponible"
  exit 1
}

Import-Module ActiveDirectory

if (-not (Test-Path $CSV)) {
  Write-Log "ERROR CRÍTICO: CSV no encontrado"
  exit 1
}

# ==============================
# CARGA Y VALIDACIÓN DEL CSV
# ==============================
$users = Import-Csv $CSV

if ($users.Count -eq 0) {
  Write-Log "ERROR CRÍTICO: CSV vacío"
  exit 1
}

# Duplicados
$duplicates = $users | Group-Object username | Where-Object { $_.Count -gt 1 }
if ($duplicates) {
  Write-Log "ERROR CRÍTICO: Usuarios duplicados en CSV"
  exit 1
}

# Validación de campos
foreach ($u in $users) {
  if (-not $u.username -or -not $u.nombre -or -not $u.apellido `
    -or -not $u.departamento -or -not $u.rol -or -not $u.fecha_fin) {
    Write-Log "ERROR CRÍTICO: CSV mal formado"
    exit 1
  }

  if ($u.rol -eq "admin" -and $u.departamento -ne "IT") {
    Write-Log "ALERTA DE SEGURIDAD: Intento de admin no autorizado ($($u.username))"
    exit 1
  }
```

```
}

# ==============================
# CREACIÓN DE OUs AUTOMÁTICA
# ==============================
$departamentos = $users.departamento | Sort-Object -Unique

foreach ($dept in $departamentos) {
  if (-not (Get-ADOrganizationalUnit -Filter "Name -eq '$dept'" -ErrorAction
SilentlyContinue)) {
    New-ADOrganizationalUnit -Name $dept -Path $BaseDN
    Write-Log "OU creada automáticamente: $dept"
  }
}

# ==============================
# CREACIÓN DE GRUPOS (SI NO EXISTEN)
# ==============================
if (-not (Get-ADGroup -Filter "Name -eq 'GRP-Admins'" -ErrorAction SilentlyContinue)) {
  New-ADGroup -Name "GRP-Admins" -GroupScope Global -Path $BaseDN
  Write-Log "Grupo creado: GRP-Admins"
}

if (-not (Get-ADGroup -Filter "Name -eq 'GRP-Usuarios'" -ErrorAction SilentlyContinue)) {
  New-ADGroup -Name "GRP-Usuarios" -GroupScope Global -Path $BaseDN
  Write-Log "Grupo creado: GRP-Usuarios"
}

# ==============================
# PROCESAMIENTO DE USUARIOS
# ==============================
foreach ($u in $users) {

  $ouPath = "OU=$($u.departamento),$BaseDN"
  $fechaFin = [datetime]::ParseExact($u.fecha_fin, 'yyyy-MM-dd', $null)

  $existingUser = Get-ADUser -Filter "SamAccountName -eq '$($u.username)'" -
ErrorAction SilentlyContinue

  if (-not $existingUser) {

    New-ADUser `
      -SamAccountName $u.username `
      -Name "$($u.nombre) $($u.apellido)" `
```

```
        -GivenName $u.nombre `
        -Surname $u.apellido `
        -Enabled $true `
        -AccountPassword (ConvertTo-SecureString "Temporal123!" -AsPlainText -Force) `
        -ChangePasswordAtLogon $true `
        -Path $ouPath

      Write-Log "Usuario creado: $($u.username)"
    }

    # Grupos
    if ($u.rol -eq "admin") {
      Add-ADGroupMember -Identity "GRP-Admins" -Members $u.username -ErrorAction
SilentlyContinue
    }
    else {
      Add-ADGroupMember -Identity "GRP-Usuarios" -Members $u.username -ErrorAction
SilentlyContinue
    }

    # Deshabilitar si está expirado
    if ($fechaFin -lt $Today) {
      Disable-ADAccount -Identity $u.username
      Write-Log "Usuario deshabilitado por fecha_fin: $($u.username)"
    }
}

Write-Log "===== FIN EJECUCIÓN ====="
```

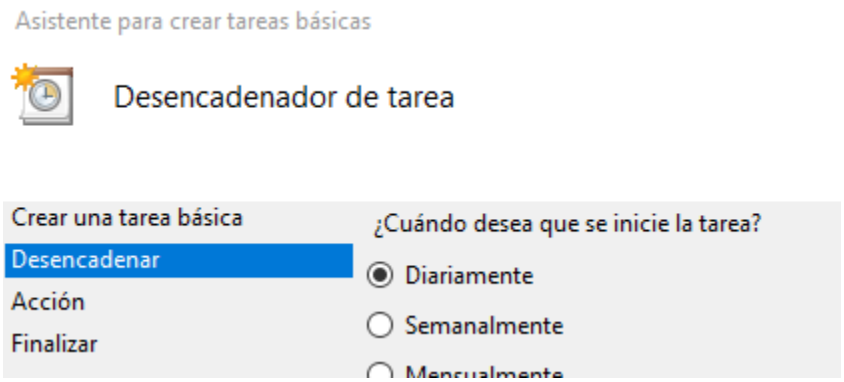
Programador de tareas

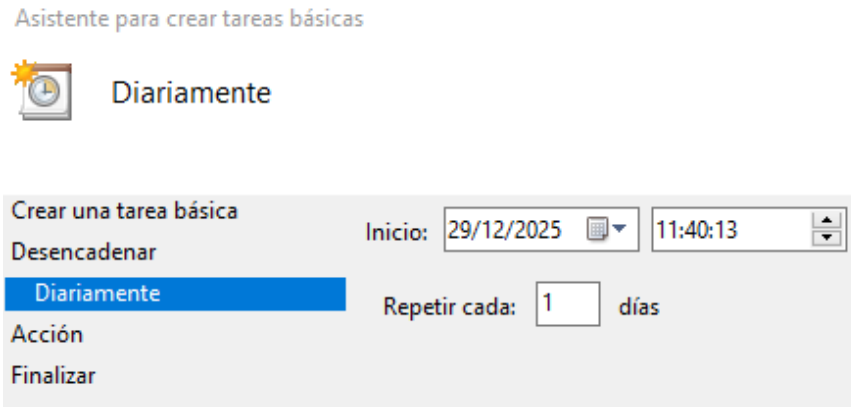
Ejecutar como Administrador de dominio

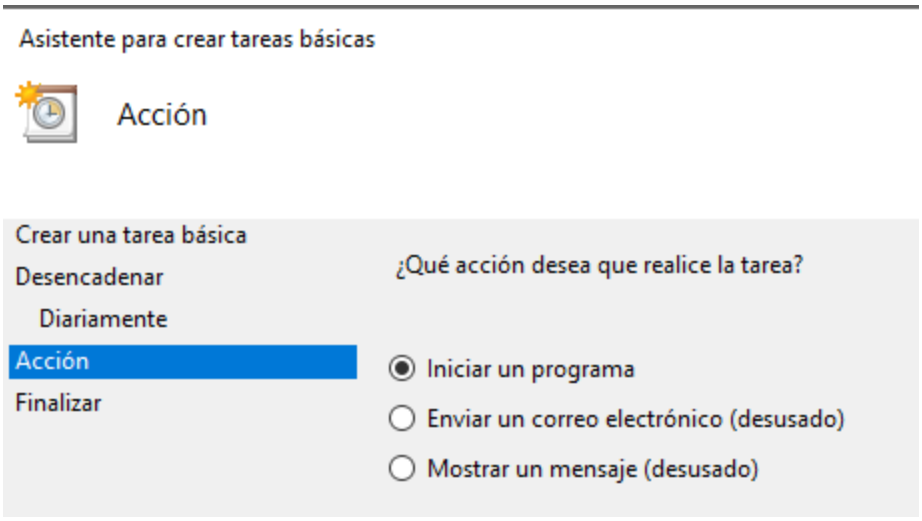
Acción:

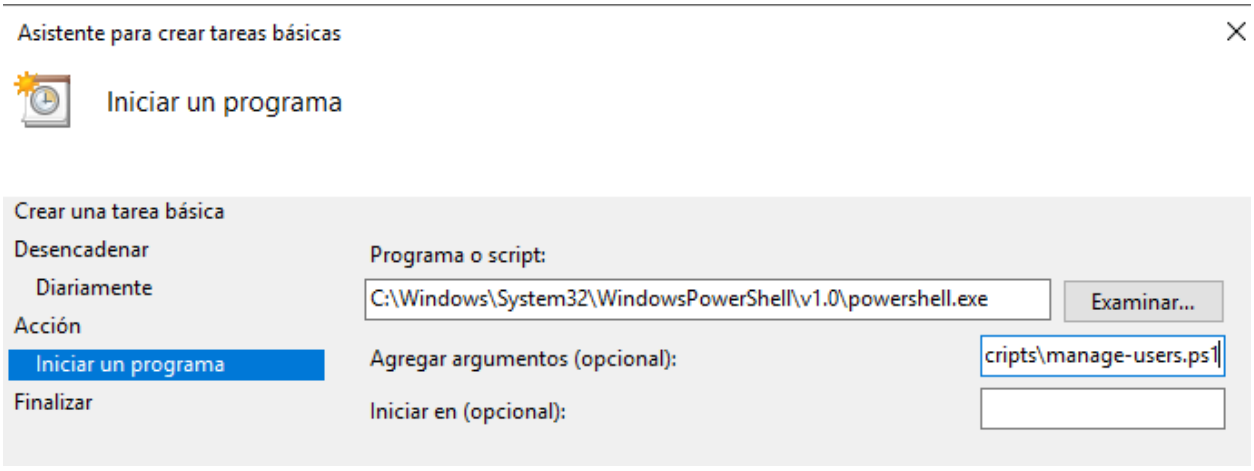
powershell.exe -ExecutionPolicy Bypass -File C:\scripts\manage-users.ps1

Frecuencia: diaria






Asistente para crear tareas básicas
Acción
Crear una tarea básica
Desencadenar
Diariamente
Acción
Finalizar
¿Qué acción desea que realice la tarea?
Iniciar un programa
Enviar un correo electrónico (desusado)
Mostrar un mensaje (desusado)


Asistente para crear tareas básicas
Iniciar un programa
Crear una tarea básica
Desencadenar
Diariamente
Acción
Iniciar un programa
Finalizar
Programa o script:
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Examinar...
Agregar argumentos (opcional):
cripts\manage-users.ps1
Iniciar en (opcional):

Solución para poder ejecutar el script en Windows:

Install-WindowsFeature AD-Domain-Services -IncludeManagementTools

Esto instala:

- Active Directory Domain Services
- Módulo PowerShell ActiveDirectory
- Consolas administrativas

Promocionar a Controlador de Dominio

Install-ADDSForest `

-DomainName "empresa.local" `

-DomainNetbiosName "EMPRESA"

O Simplemente, instala desde el GUI Active Directory o con el script:

```
#
# Script de Windows PowerShell para implementación de AD DS
#

Import-Module ADDSDeployment
Install-ADDSForest `
-CreateDnsDelegation:$false `
-DatabasePath "C:\Windows\NTDS" `
-DomainMode "WinThreshold" `
-DomainName "empresa.local" `
-DomainNetbiosName "EMPRESA" `
-ForestMode "WinThreshold" `
-InstallDns:$true `
-LogPath "C:\Windows\NTDS" `
-NoRebootOnCompletion:$false `
-SysvolPath "C:\Windows\SYSVOL" `
-Force:$true
```

Gestión de errores y ataques simulados:

Un csv mal formado y con usuarios duplicados. Debe registrar el fallo mediante log y no realizar acciones parciales

Linux

```
  GNU nano 6.2
username,nombre,apellido,departamento,rol,fecha_fin
jgarcia,Juan,Garcia,IT,admin,2025-06-30
mmartin,Maria,Martin,RRHH,user,2025-03-15
aperez,Ana,Perez,Marketing,user,2025-02-01
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBBBBBBBBBBBBBBBBBB
CCCCCCCCCCCCCCCCCCCCCCCC
```

```
 Cristobal CON ROOT  lunes 29 diciembre 2026 10:59
[root@server2asir user_mgmt]$cat /var/log/user_mgmt/user_mgmt.log
[lun 29 dic 2025 10:59:34 UTC] Inicio ejecución
ERROR: CSV mal formado
 Cristobal CON ROOT  lunes 29 diciembre 2026 10:59
```

Windows

```
users.csv: Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
username,nombre,apellido,departamento,rol,fecha_fin
jgarcia,Juan,Garcia,IT,admin,2025-06-30
mmartin,Maria,Martin,RRHH,user,2025-03-15
aperez,Ana,Perez,Marketing,user,2025-02-01
AAAAAAAAAAAAAAAAAAAAAAAAA
VVVVVVVVVVVVVV
CCCCCCCCCCCCCCCCCCCC
```

```
[2025-12-29 12:45:16] ===== INICIO EJECUCION =====
[2025-12-29 12:45:17] ERROR CRÍTICO: CSV mal formado
```