Benutzerkonten mit PowerShell

Anlegen von Benutzerkonten im Active Directory und anschließender Ausgabe von Benutzername und Passwort

Matthias Hütthaler

Problembeschreibung

Lösungen zum Anlegen mehrerer hundert Accounts im Active Directory gibt es viele. Die möglichen Varianten reichen vom einfachen Powershellscript, bis hin zu kostenpflichtigen, sehr umfangreichen Programmen.

Ich stand vor dem Problem, dass über 400 Gastaccounts für das WLAN-Netz angelegt werden sollten, damit das kabellose Netzwerk auch genutzt werden kann, wenn man keinen gültigen Account im Active Directory besitzt. Die Schwierigkeit bestand darin, dass man sich mit diesem Gastaccount über das Captive Portal gegen einen Windows-Radiusserver authentifiziert:

Bei so gut wie allen Tools zum Anlegen von Benutzeraccounts, die ich gefunden habe, wird ein Standardkennwort festgelegt und dieses müssen die User bei der ersten Anmeldung ändern. Natürlich kann man das Passwort bei der Anmeldung im WLAN-Netz bzw. am Captive Portal nicht selbständig verändern. Somit war dieser Weg für meine Anforderung nicht möglich. Ich suchte nach einer Lösung, mit der ein eigenes Passwort pro Userkonto automatisch generiert wird. Außerdem sollten die neuangelegten Benutzernamen und Passwörter grafisch ausgegeben werden, um in weiterer Folge eine Liste mit Benutzernamen und den dazugehörigen Passwörtern zu erhalten.

Eine weitere Anforderung an das Programm war, die neuerstellten Userkonten automatisch in der richtigen Organisationseinheit einzuordnen. Das heißt, Gast1-10 sollten beispielsweise in die OU „Ausbildung“ und Gast 11-40 in die OU „Fortbildung“ eingeordnet werden.

Außerdem sollte es mir möglich sein, die Passwortkomplexität selbst festlegen zu können. Erfahrungsgemäß gibt es bei vorgegebenen Passwörtern immer die Schwierigkeit, beispielsweise zwischen „0“ (Null), „o“ (kleines o) und „O“ (großes O) zu unterscheiden.

Das Powershellscript

```
Import-Module ActiveDirectory

$ou_mapping = @{
    #Anzahl der User und OU
    'OU=2a,OU=schueler,dc=schule,dc=intern' = 3

}

Function generate-password($length){
    $objRND = new-object System.Random
    #Zeichen für Passwort
    $chars = "abcdefghijkmnpqrstuvwxyzABCDEFGHJKLMNPQRSTUVWXYZ123456789".ToCharArray()
    $pass = ""
    1..($length-1) | %{$pass += $chars[$objRND.Next($chars.Length)]}
    return $pass
}

$userlog = @()
$cnt = 0 #Startnummer
$ou_mapping.GetEnumerator() | %{
    $OU = $_.Name
    1..$_.Value | %{
        $username = "gast$($cnt + $_)" #Name des Accounts
        $password = generate-password 8 #Passwortlänge
        $u = New-ADUser -Path $OU -SamAccountName $username -Name $username -PassThru
        if ($u){
            $u | Set-ADAccountPassword
                -Reset
                -NewPassword (ConvertTo-SecureString $password -AsPlainText -Force)
            $u | Set-Aduser -Enabled $true
            $userlog += New-Object PSObject
                -Property @{Username=$username;OU=$OU;Password=$password}
        }
    }
    $cnt += $_.Value
}
$userlog | ft Username,OU,Password -AutoSize
```

Zusammengefasst sollte also das Tool folgende Funktionalitäten erfüllen:

- Eine Vielzahl an Benutzerkonten soll automatisch im Active Directory anlegt werden.
- Jedes Konto soll automatisch ein eigenes Kennwort erhalten.
- Benutzername und Kennwort sollen grafisch ausgegeben werden.
- Benutzerkonten sollen automatisch in die richtige Organisationseinheit eingeordnet werden.
- Die Passwortkomplexität kann selbst festgelegt werden.

Nach langer Recherche habe ich ein Script auf www.administrator.de gefunden, das alle meine Ansprüche erfüllt bzw. auf dessen Basis ich die Verwaltung der neuen Accounts durchführen konnte.

- `#Anzahl der User und OU`

  Diese Zeile ist eigentlich die wichtigste des ganzen Scripts. Wenn Änderungen vorgenommen werden, dann hier. Denn damit wird festgelegt, wie viele Accounts angelegt und in welcher Organisationseinheit diese eingeordnet werden sollen. In meinem Beispiel werden 3 User in die Organisationseinheit „2a“ neu angelegt. Die OU „2a“ ist Teil der OU „schueler“. Sollte die OU „schueler“ beispielsweise Teil der OU „außenstelle“ sein, müsste diese Zeile so aussehen:

  ```
  'OU=2a,OU=schueler,OU=aussenstelle,dc=schule,dc=intern' =3
  ```

  Die Struktur der Organisationseinheiten wird also von rechts nach links angegeben.

  Natürlich müssen die Werte bei „dc=“ entsprechend dem Domänennamen angepasst werden (Distinguished Name) und mit der Zahl hinter dem „=“ wird die Anzahl der neuen User definiert.

  Möchte man bei einem Durchlaufen des Scripts Accounts für beispielsweise verschiedene Klassen erstellen, ergänzt man es an dieser Stelle einfach um mehrere Zeilen:

  ```
  'OU=2a,OU=schueler, dc=schule,dc=intern' =3

  'OU=2b,OU=schueler, dc=schule,dc=intern' =3

  'OU=2c,OU=schueler, dc=schule,dc=intern' =3
  ```

  In diesem Beispiel werden jeweils 3 Accounts für die 2a, 2b und 2c erstellt. Damit dies auch wirklich funktioniert, müssen die jeweiligen Organisationseinheiten bereits im Active Directory angelegt sein.

- `#Zeichen für Passwort`

  Hier wird die Passwortkomplexität beeinflusst. Die Passwörter der Gastaccounts erhalten in meiner Schule der Einfachheit halber keine Sonderzeichen. Möchte man aus Sicherheitsgründen welche, ergänzt man diese einfach. Außerdem haben die Kennwörter kein „0“ (Null), „O“ (großes O), „o“ (kleines o), „I“ (großes I) und „l“ (kleines l), weil die Verwechslungsgefahr dieser Zeichen sehr groß ist. Dadurch kommt es erfahrungsgemäß bei vorgegebenen Passwörtern zu Problemen. Aus diesem vorgegebenen Zeichenpool wird das Passwort geniert. Die Länge des Kennworts wird an einer anderen Stelle geändert.

  Natürlich muss das Kennwort den Passwortrichtlinien innerhalb der Domäne entsprechen.

- `#Startnummer`

  Soll eingestellt werden, dass die Userkonten eine bestimmte Nummer am Ende des Namens haben, wird das hier eingetragen. In diesem Beispiel werden 3 Accounts mit den Namen „Gast1“, „Gast2“ und „Gast3“ angelegt. Ist jedoch schon „Gast1-3“ im Active Directory vorhanden und soll „Gast4-6“ erstellt werden, passt man den Wert folgendermaßen an: `$cnt = 3`

  Die Anzahl der neuanzulegenden Accounts wird, wie bereits beschrieben, festgelegt.

- `#Name des Accounts`

  Sollen die Benutzerkonten nicht „gast1“ heißen, sondern beispielsweise „schueler1“, bessert man diese Zeile folgendermaßen um:

  ```
  $username = "schueler$($cnt + $_)"
  ```

- `#Passwortlänge`

  Die Länge des Passwortes kann hier geändert werden. In diesem Beispiel ist das Passwort 8 Zeichen lang. Selbstverständlich muss man auch hier wieder die Richtlinien der Domäne berücksichtigen.

Starten des Scripts

Zuerst sollte man sicherstellen, das Script im richtigen Dateiformat angelegt zu haben (Dateiendung: „.ps“).

Um es ausführen zu können, muss zuerst die Powershellkonsole mit Administratorrechten gestartet werden. Standardmäßig ist die Powershell besonders „sicher“ und führt nur direkt eingegebene Befehle aus, aber keine Scripts. Dies muss erst ermöglicht werden, indem man eine execution policy festlegt. Der Befehl dazu lautet:

```
Set-ExecutionPolicy unrestricted
```

Mit dem Wert „unrestricted“ kann man alle Scipts ausführen. Der gegenwärtigen Status der execution policy wird mit folgendem Befehl ausgegeben:

```
Get-ExecutionPolicy
```

Anschließend navigiert man in gewohnter Manier zum Speicherort des Scripts und startet es.

Bei erfolgreicher Ausführung wird jetzt Benutzername und Passwort in folgender Form ausgegeben:

```
Username OU                                    Password
-------- --                                    --------
gast1    OU=2a,OU=lehrer,dc=     ,dc=         UmWVII0
gast2    OU=2a,OU=lehrer,dc=     ,dc=         d3jBOX0
gast3    OU=2a,OU=lehrer,dc=     ,dc=         Nkkr9eP
```

Diese Tabelle kann man sich bequem in ein Textverarbeitungsprogramm kopieren.

Gerne schicke ich das Scripts per Mail, um aufwendiges Abtippen zu vermeiden.