Makrosprachen für die Computerverwaltung

Annahme:

Die Benutzer befinden sich in einer Windows-Domäne mit einem Domänencontroller.

Benutzer- und Gruppeninfos können aus dem Active Directory geholt werden.

Die Automatisierung erfolgt mittels Script Sprachen, wodurch man sich nicht durch die „Oberfläche“ klicken muss, wenn sich eine Aufgabe oft wiederholt. Ist der Windows Server ohne GUI, wird alles über Shells erledigt.

# Active Directory

Der Active Directory Service ist ein Verzeichnisdienst welcher von Microsoft entwickelt wurde. ADS basiert auf dem Lightweight Directory Access Protocol und arbeitet mit dem Domain Name System für die Adressenspeicherung sowie Kerberos für die Authentifizierung. In der DNS-Datenbank werden die Informationen über Netzwerkdienste und die Zuordnung des kompletten Domainnamens zu den IP-Adressen gespeichert. Im Active Directory sind alle relevanten Informationen von Benutzern, Rechnern, Ressourcen, Netzwerken usw. abgelegt. Die Domäne bildet die Basiseinheit des Active Directory. In ihr werden die Ressourcen und die Benutzer in Gruppen zusammengefasst.

## Organisationseinheiten

Organisationseinheiten (auch Organizational Unit genannt), sind Containerobjekte, welche zum Gruppieren von Objekten im Active Directory dienen. Eine Organisationseinheit kann neben Objekten auch andere Organisationseinheiten enthalten. Dadurch, dass die Hierarchie dieser Container frei definierbar ist, wird die Administration vom Active Directory erheblich vereinfacht. Die Organisationseinheiten sind die unterste Ebene vom Active Directory, in der administrative Rechte aufgeteilt werden können.

Beispiel für den Zugriff aus vbs:

Set oTargetOU = GetObject("LDAP://ou=5BHIFS,ou=EDVO,ou=Schueler,ou=Benutzer,dc=htl-vil,dc=local")

oTargetOU.Filter = Array("user")

For each usr in oTargetOU

if instr(usr.SamAccountName, "$") = 0 then ….

# Automatisierung

Für die Automatisierung bieten sich folgende Möglichkeiten an:

## DOS-Shell bzw. cmd-Shell

Vorteile

* Es gibt Kontrollstrukturen wie Schleifen und Verzweigungen, Eingabe und Ausgabe
* Es gibt Variablen, Kommandozeilenparameter
* Umleitung der Ausgabe in Dateien mit > möglich
* Einige Befehle sind sehr gut und mächtig zB NET, cacls

Nachteile

* Wenig Gestaltungsmöglichkeiten
* Probleme mit Zeichensätzen wie Umlaute
* Schlechtes Error-Handling
* Keine Arrays, Variablenhandling „gewöhnungsbedürftig“

## Windows Script Host

Verarbeitet vbs und java script  
ist Interpreter

drei wichtige, mächtige Objekte:

* Networkobject (alle Infos vom ActiveDirectory – alle Benutzer und Ous)
* Filesystemobject (Zugriff auf alle lokalen Ordner und Dateien und auf die Freigaben)
* ShellObject alle Befehle, die es auch in der cmd gibt

Vorteile VBS

* Viele Beispiele im Netz vorhanden, einfache Syntax. .net objekte vewendbar
* Einfache Kontrollstrukturen
* Brauchbare Objekte
* Zusammenarbeit mit cmd über den Befehl Wscript.Run(“dosbefehl“)
* Errorhandling einfach aber immerhin vorhanden (on error resume, err.number, err.clear …)

Nachteile

* Keine vernünftigen IDEs
* Unvollständige Methoden zB für das Einbinden von Netzwerkdruckern

## Powershell

Vorteile PWS

* Enormer Sprach- und Befehlsumfang, man kann damit fast alles automatisieren
* Gute Layoutmöglichkeiten
* funktioniert auch ohne GUI
* ist objektorientiert
* Beispiele im Internet wachsen zusehend
* Powershell\_Ise nette IDE

Nachteile

* Eigentlich keine. Befehle sind mächtig aber komplex und nicht leicht zu merken

## Vbs und DOS

Es besteht die Möglichkeit mit vbs automatisch DOS bat-Dateien zu erstellen und diese anschließend auszuführen. Die Vorteile beider Methoden werden hierbei optimal kombiniert.

Mit skritp.vbs in eine bat-skript Datei schreiben und dieses dann ausführen.

Trick für die Automatisierung:

Eine Datei namens zB yes erzeugenund dort ein j mit Return einfügen  
Dann kann man ein script ausführen, dass zB eine Bestätigung mit j benötigt in dem man

Script.bat <yes ausführt. Die Eingabe kommt nun nicht von der Tastatur sondern aus dem File, in dem j steht.

## Kix

Kix ist eine eigene Skriptsprache für Windows-Rechner und benötigt einen eigenen Interpreter namens kixstart, der am einfachsten im gleichen Ordner wie die scripts liegt

Vorteile

* Großer Sprachumfang für Zugriff auf AD Objekte, Files,….
* Gute Layoutmöglichkeit, auch mit Farben
* Gute Kontrollstrukturen und Variablen in gewohnter Syntax
* Leichter Zugriff auf Windows Objekte

### KIX Beispiel

Schaut ein bißchen aus wie basic, man kann mit at auf bestimmte Cursorpositionen schreiben  
Variablen in Zeichenketten direkt mit @ einfügbar (wie bei js und php) zB unten die Variabel userid  
= Operator ist auch Vergleichsoperator!!!  
Ordnername mit $ am Ende bedeutet versteckter Ordner, wird im Explorer nicht angezeigt, Linux machts trotzdem  
use h: ist wie der Shell Befehl net use

; \*\*\* Home-Laufwerke für Schüler verbinden

if ingroup("Schüler") OR ingroup("SchülerITNT")

at (5,8) "verbinde Netzlaufwerk H: ... "

if ($abteilung = "EDVO")

use h: "\\gandalf\@userid$"

$home = "gandalf"

endif

if ($abteilung = "IT")

use h: "\\triton\@userid$"

$home = "triton"

endif

endif

Erkläre zB die folgenden Scriptzeilen:

a.)

VBS Script mit Zugriff auf das Active Directory über LDAP  
gültige Benutzernamen im AD user Objekt der HTL haben kein $ Zeichen drinnen

Dim Klasse=array("1AHIF","2AHIF","3AHIF")

for i = 0 to ubound(Klasse)

Set oTargetOU = GetObject("LDAP://ou="& Klasse(i),ou=EDVO,ou=Schueler,ou=Benutzer,dc=htl-vil,dc=local")

oTargetOU.Filter = Array("user")

For each usr in oTargetOU

if instr(usr.SamAccountName, "$") = 0 then

BenutzerName=usr.SamAccountName

....

Jetzt könnte man zB ein File öffnen und Infos reinschreiben

Set fso = CreateObject("Scripting.FileSystemObject")

Set datei=fso.OpenTextFile("Infos",ForWriting)

Fso.WriteLine(BenutzerName)

b.)

Dos Command erzeugt Freigabe mit Freigabe Permissions  
Achtung Freigabe Permissions sind nicht gleich den Zugriffseinstellung im Reiter „Sicherheit“, würden diese aber einschränken,wenn zB in den Sicherheitseinstellung „Full Control“ steht und in der Freigabe nur „Read“, dann würde jemand der übers Netzwerk zugreift nur Lesen können.  
Wenn in der Freigabe „Full“ steht und in den Sicherheitseinstellungen nur „Read“ dann gilt nur das read. Es gilt also immer der strengere Teil.

chcp 1252

NET SHARE apeltm$="u:\schüler\home\APELT Moritz" /GRANT:Everyone,FULL /CACHE:None

chcp codepage damit Umlaute, Sonderzeichen funtkionieren

$ bedeutet, dass Directory versteckt ist, also nicht von Haus aus im Explorer angezeigt wird (man muss den Namen wissen)

....