FAQ-O-MATIC.NET

Whitepapers

Erstellen von Suchanfragen mit LDAP-Filtern für Active Directory

Autor: Florian Frommherz
Durchsicht: Nils Kaczenski, Yusuf Dikmenoglu

September 2009

Inhalt

Nicht noch eine Datenbank!

Active Directory ist weitläufig bekannt als Verzeichnisdienst, der eine Vielzahl von Aufgaben in einer Infrastruktur übernimmt: Neben der Authentifizierung von Benutzern sorgt es unter anderem auch dafür, dass Anfragen zur Namensauflösung per DNS beantwortet werden, Benutzer unter Verwendung von sogenannten „Tickets“ Zugriff auf Ressourcen erhalten und Mitarbeiter sowie ihre Clientcomputer von zentraler Stelle aus verwaltet werden können.

Im Mittelpunkt der Pflege des Verzeichnisses steht die Verwaltungskonsole „Active Directory Benutzer und Computer“, die das AD in einer baumförmigen Struktur anzeigt. Mit ihrer Hilfe lässt sich nicht nur die logische Struktur des Verzeichnisses in Organisationseinheiten realisieren, sondern auch einzelne Benutzer-, Gruppen- oder gar Computerobjekte verwalten. Über die „Eigenschaften“ kann der Administrator erweiterte Aufgaben am Objekt durchführen, um Angaben und zusätzliche Informationen zu Objekten einzupflegen. Für Benutzerobjekte können Systemverwalter neben administrativen Kontenoptionen nähere Angaben zu Tätigkeiten und Kontaktinformationen hinterlegen.

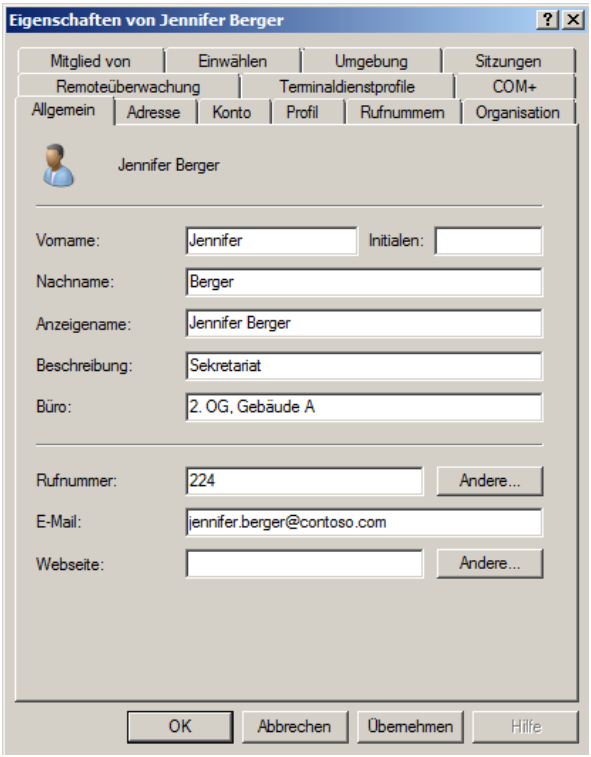


Abbildung 1: Über die Verwaltungskonsole "Active Directory Benutzer und Computer" lassen sich erweiterte Informationen zu Benutzern, Computern und Gruppen speichern.

Dass dies nur die Spitze des Eisberges ist, scheint nur wenigen klar zu sein: Active Directory verwaltet eine Vielzahl weiterer Informationen zu den im Verzeichnis angelegten Objekten, die in der grafischen Oberfläche weder angezeigt noch editiert werden können. Hinter den Daten, die das AD verwalten kann, steht ein mächtiges, erweiterbares Schema, das eine Definition für alle Objekte vorhält. Anhand dieser Definition, die eine eindeutige Objektkennung, den Namen sowie benötigte und optionale Attribute für die Objekterstellung führt, erzeugt Active Directory neue Objekte, die es im Verzeichnis erstellt. Bekannte Objektklassen umfassen unter anderem *user*, *group* oder *organizationalUnit* – wobei das Schema beliebig erweiterbar ist und das Einfügen eigener Objekte und Attribute oder das Modifizieren von bestehenden Objekten erlaubt. Die Speicherung der Objekte und ihrer Attribute erfolgt über eine *Extensible Storage Engine* (ESE), einer Datenbank, die auch in anderen Produkten wie etwa Microsoft Exchange oder die Windows Desktop Search für das Speichern und Abrufen von Informationen eingesetzt wird.

Wie man es von anderen Datenbanken kennt, lässt sich Active Directory demzufolge auch nutzen um benutzer- oder computerkontenbezogene Daten zu speichern und sie im Rahmen von hauseigenen Applikationen abzufragen und weiterzuverwenden. Vorteilhaft ist hierbei, dass eigene Anwendungen sehr einfach an Active Directory angebunden werden können. Auf Grund der Replikation zwischen Domänencontrollern stehen eingepflegte Daten auf allen Domänencontrollern zur Verfügung.

Handwerkszeug

Wer einen Blick hinter die Kulissen und somit in alle verfügbaren Attribute und Objekte des Verzeichnisses blicken möchte, muss sich lediglich mit einer Reihe nutzvoller Werkzeuge befassen die im Internet verfügbar sind.

Um das Verzeichnis zu durchforsten und erweiterte Attribute anzeigen zu können die nicht in der grafischen Oberfläche dargestellt werden, eignen sich zwei Tools die mit den „Support Tools“ für Windows Server 2003 heruntergeladen werden können[1] oder bei einer Installation ab Windows Server 2008 (nicht Server Core) mitgeliefert werden: LDP.exe und ADSIEdit.

LDP.exe ist ein Client für das LDAP, einem offenen Protokollstandard für Verzeichnisdienste. Es kann verwendet werden, eine Verbindung mit beliebigen LDAP-Servern aufzubauen und diese zu durchforsten, Daten hinzuzufügen, zu verändern oder zu löschen. Administratoren können LDP.exe verwenden, weil Active Directory aus einem X.500 kompatiblen LDAP-Server besteht, über den Lese- und Schreibzugriffe an Objekten und Attributen realisiert wird. Auf diese Weise wird die interne Struktur der Datenbank vor dem Benutzer verborgen. Tabellen und Spalten werden nicht angezeigt - stattdessen zeigt sich das AD als Verzeichnisdienst in Form eines Baumes, in dem Container und Objekte gespeichert werden.

Eine Verbindung mit LDAP über LDP erfolgt in drei Schritten: dem Verbinden selbst (connect), dem Anmelden (bind bzw. binding) und der Auswahl der sogenannten Basis-DN, also des „Startzweiges“ im Verzeichnis. „DN“ steht hierbei für „Distinguished name“ und erwartet den Start-Pfad der Ansicht von LDP in der LDAP-Schreibweise. Eine Domäne namens *contoso.com* wird in der Verzeichnisdienstschreibweise beispielsweise mit *DC=contoso,DC=com* angegeben. Eine Subdomäne *sub1.contoso.com* entsprechend *DC=sub1,DC=contoso,DC=com*. „DC“ steht hierbei für „Domain Component“ und trennt zusammen mit einem Komma die jeweiligen Level des vollqualifizierenden Domänennamens. Die Trennung in der LDAP-Schreibweise erfolgt somit nicht wie gewohnt mit einem Punkt „.“, sondern mit der Notation „DC=“. Wird die Organisationseinheit (OU) „Benutzer“ direkt unter dem Domänenobjekt als Basis-DN gewünscht, kann sie wie folgt angegeben werden: *OU=Benutzer,DC=contoso,DC=com*.

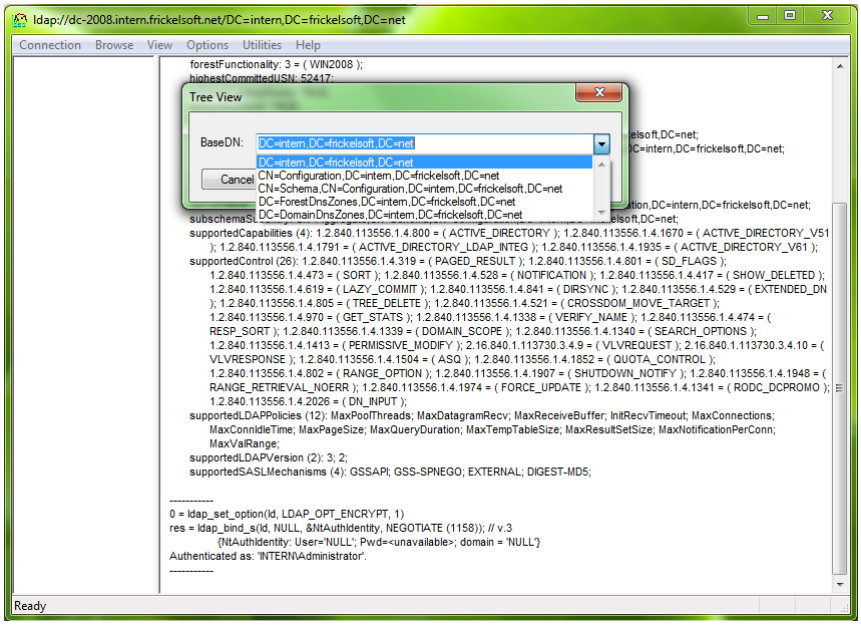


Abbildung 2: LDP möchte wissen, welchen Basiszweig es für die Wurzel des Baumes anzeigen soll

Sobald die Verbindung erfolgreich mit dem Server hergestellt wurde und der Administrator einen gültigen Zweig des Verzeichnisses als Basis-DN angegeben hat, kann mit einem Doppelklick auf den jeweiligen Knoten oder durch auswählen des Pluszeichens vor dem Knoten im Verzeichnis navigiert werden. Die Ansicht von LDP ist dabei zweigeteilt: die linke Spalte zeigt den Verzeichnisbaum mit der Basis-DN als Wurzel und allen Objekten im Baum darunter, die

rechte Spalte zeigt alle Attribute des gerade ausgewählten Objektes, die einen Wert besitzen und nicht *leer* sind. Die linke Spalte sollte, sofern das Domänenobjekt als Basis-DN ausgewählt wurde, der Ansicht aus „Active Directory Benutzer und Computer“ ähneln.

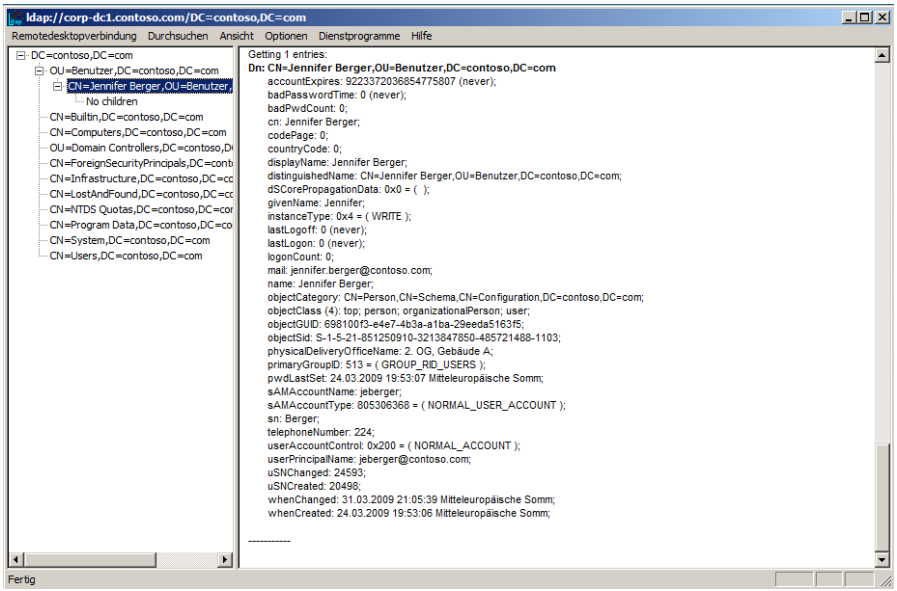


Abbildung 3: Ist die Verbindung erstmal erstmal erstellt und die Basis-DN, hier: DC=contoso,DC=com, ausgewählt, lässt LDP.exe Administratoren im Verzeichnis browsen

Ein weiteres Werkzeug aus den Supporttools für Windows 2000 und Windows Server 2003 ist ADSIEdit. ADSIEdit ist ein Lowlevel –LDAP-Editor, mit dem nicht-GUI-sichtbare Objekte und ihre Attribute editiert werden können. Nach dem Start von ADSIEdit.msc gilt es einen sogenannten „Namenskontext“ auszuwählen. Ein Namenskontext kann hierbei mit einer Festplatte verglichen werden, die in mehrere Partitionen aufgeteilt wurde. Ähnlich wie C:, D: wird das Verzeichnis in Partitionen aufgeteilt, die jeweils unterschiedliche Daten und Aufgaben ausführen. Die Bedeutung der Begriffe „Namenskontext“ und „Partition“ sind dabei identisch einsetzbar. In einem Active Directory-Forest mit einer Domäne existieren anfangs drei Namenskontexte: der Domänennamenskontext, der die Nutzdaten von Active Directory-Domäne beinhaltet und die in „Active Directory Benutzer und Computer“ angezeigt werden (in unserem Beispiel *contoso.com*), einen Schemanamenskontext, indem die Schemadefinitionen für Objekte und Attribute abgelegt sind und einen Konfigurationsnamenskontext, indem Active Directory Dienst- und Konfigurationseigenschaften speichert. Wird der DNS-Dienst als „Active Directory-integriert“ ausgeführt, werden zwei weitere Namenskontexte ab Windows Server 2003 angelegt: „DomainDNSZones“ und „ForestDNSZones“.

Per Rechtsklick auf „ADSI Editor“ und Auswahl „Verbindung herstellen…“ kann der Namenskontext gewählt werden. Von hier aus verhält sich ADSIEdit ähnlich wie LDP – mit dem kleinen Unterschied, dass Attribute des gerade ausgewählten Objekts nicht im rechten Teil des Programms angezeigt werden. Diese müssen durch den Administrator explizit über einen Rechtsklick und der Auswahl „Eigenschaften“ im Kontextmenü angezeigt werden. Im öffnenden Fenster werden sämtliche Attribute eines Objekts angezeigt – auch jene, die bisher keinen Wert zugewiesen bekamen. Eine Änderung der Attributwerte kann über einen Doppelklick auf das Attribut geschehen. ADSIEdit öffnet daraufhin eine Eingabemaske, in der die neuen Werte eingetragen werden können.

Geht es darum im Verzeichnis nach Daten zu suchen, liefert Microsoft seit Windows Server eine Reihe von Kommandozeilenprogrammen mit, die Systemverwaltern die Administration ihrer Domänen vereinfachen sollen. Die „DS-Tools“, wie sie auf Grund ihres gemeinsamen Namens genannt werden, lassen Suchanfragen an den Verzeichnisdienst starten, um sie dann entweder auf der Kommandozeile auszugeben oder mit Hilfe des Pipe-Operators in ein anderes DS-Tool weiterzuleiten, das dann weitere Operationen vornimmt. Bild 3 zeigt die Kommandozeile mit den beiden Tools „DSQuery“ und „DSGet“. DSQuery wird verwendet, um Objekte aus dem Verzeichnis abzufragen. Das

Suchergebnis, im Beispiel eine Liste von Benutzern aus der OU „Customer Support“, kann dann zu DSGet weitergeleitet werden, das detaillierte Informationen zu Objekten liefern kann. Im Beispiel wird die abgefragte Benutzerliste per Pipe-Symbol nach DSGet geleitet, das anschließend für alle Benutzer den *sAMAccountName* sowie die Emailadresse ausgibt. Wollten wir Modifikationen an der von DSQuery zurückgegebenen Benutzerliste durchführen, könnten wir das Ergebnis nach DSMod schicken, das für uns Änderungen an allen Benutzern vornimmt. Bild 4 zeigt das veränderte Kommando das dafür sorgt, dass alle Benutzer der DSQuery-Benutzerliste ihre Passworte bei der nächsten Anmeldung ändern müssen.

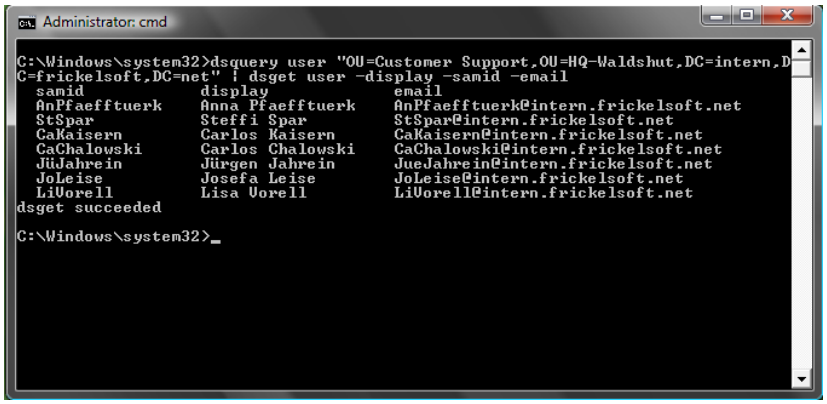


Abbildung 4: Die DS*-Tools können mit Hilfe der Pipe miteinander verbunden werden.

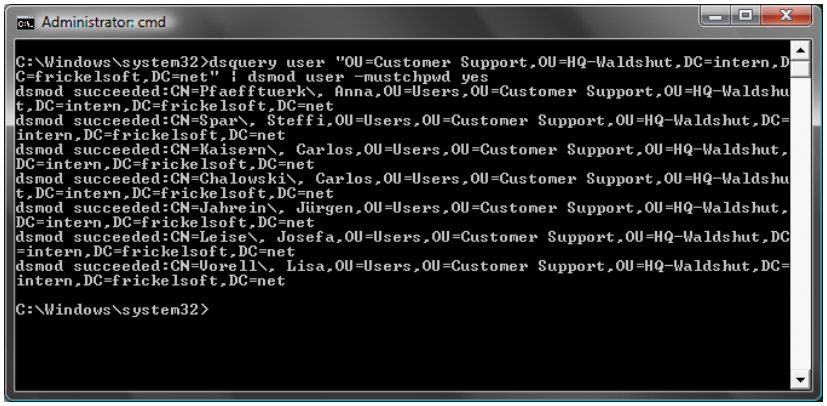


Abbildung 5: Mit Hilfe der Pipe kann aus einer Ergebnisliste eine Änderung an allen gefundenen Objekten vorgenommen werden.

Ein Blick in die Hilfe der DS-Tools verrät, dass die Programme eine Vielzahl von Parametern mitbringen, die häufig durchgeführte Aktionen schnell und bequem durchführen lassen. Was aber, wenn die vorgefertigte Funktionalität nicht mehr ausreicht? In vielen Fällen bietet es sich an, ein wenig tiefer in die Technik hinter dem Verzeichnisdienst und den Abfragen zu blicken und selbst Abfragen und Modifikationen zu entwerfen. Hierfür eignen sich zwei kostenfreie Kommandozeilenprogramme von Joe Richards: ADFind und ADMod [2]. ADFind ist ein mächtiges Werkzeug für Anfragen an Verzeichnisdienste, ADMod führt Modifikationen an Objekten und Attributen durch. Auf den ersten Blick ähnelt die Funktionsbeschreiben der beiden „AD*“-Tools der der DS-Tools. Die AD*-Tools von Joe Richards sind aber weitaus flexibler und erlauben durchaus umfangreichere Abfragen. Ein Blick in die Hilfe, mit dem Schalter /? verrät, wie viele Optionen hier zur Verfügung stehen.

Die Anatomie eines Filters

Abfragen erstellen zu können ist zwar schön und gut, jedoch sollte man sich, bevor man das Erstellen von Abfragen angeht, notieren, was die geforderte Abfrage leisten soll. Sind die Anforderungen klar definiert, gilt es zu prüfen, ob die gewünschten Daten überhaupt im Verzeichnis vorhanden sind und falls ja, in welchem Format sie vorliegen. Active Directory kennt eine Reihe von Datentypen und verschiede Speicherarten für Daten. Wer erweiterte Abfragen erstellen will oder mit einer selbst erstellten Applikation Abfragen erstellen möchte, muss sich über die Speicherung der Daten im Klaren sein.

Wie bereits erwähnt, besteht das Verzeichnis aus einer Vielzahl von Objekten. Die Objekte sind durch ihre Objektklasse definiert, durch die sie eine Reihe von Pflicht-, Optional- und Systemattributen erhalten. Es ist nicht möglich, die letzten zehn Computer anzeigen zu lassen, an dem sich ein Benutzer angemeldet hat – oder die größten fünf Dokumente, die an einem bestimmten Drucker ausgedruckt wurden. Für diese Daten existiert in Active Directory kein Attribut – weder am Benutzerobjekt, noch am Druckerobjekt. Es können nur Daten abgefragt werden, die im Verzeichnis gespeichert sind.

Da für eine erfolgreiche Abfrage die Attribute im Verzeichnis von Nöten sind, gilt es herauszufinden, wie die Attributnamen lauten, nach denen gesucht und gefiltert werden soll. Die bereits vorgestellten Werkzeuge ADSIEdit und LDP lassen durch Objekte im Verzeichnis browsen und ihre Attribute durchleuchten: LDP zeigt beim Doppelklick auf einen Benutzer alle Attribute an, die mit einem Wert gefüllt sind – so lässt sich bereits ein Überblick über die vorhandenen Attribute eines Objektes gewinnen. Wie findet man aber heraus, welches Attribut das gerade gesuchte ist?

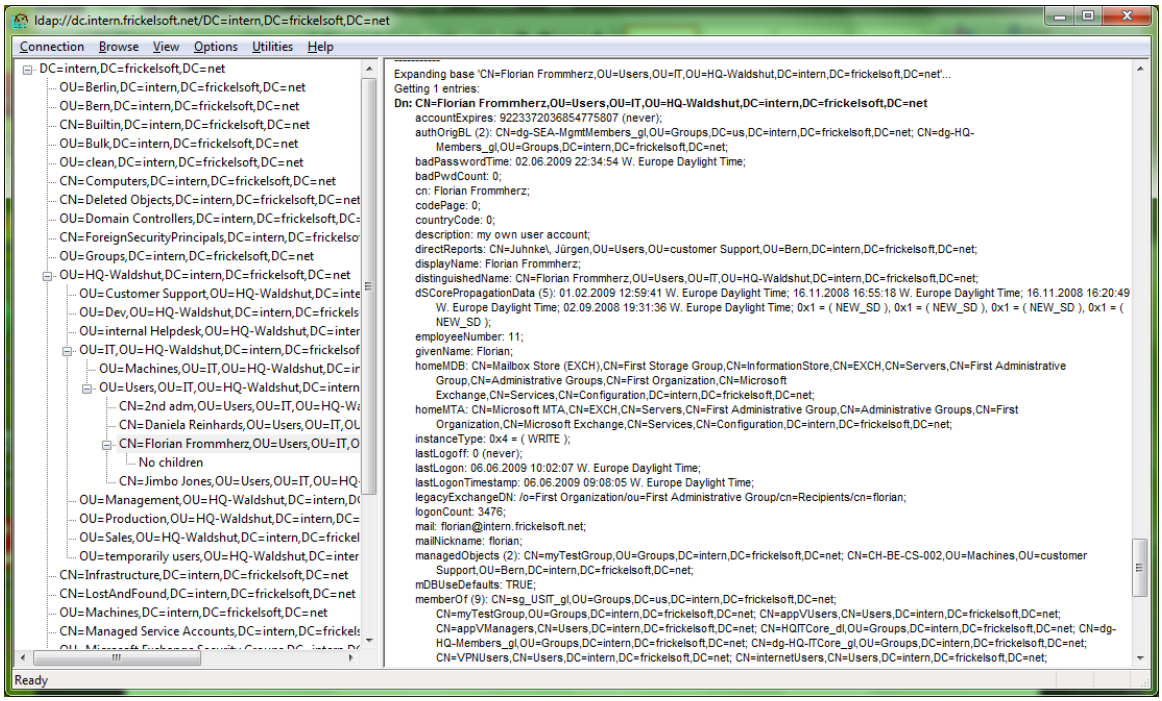


Abbildung 6: LDP zeigt alle gefüllten Attribute des ausgewählten Benutzers an.

Hierfür gibt es drei praktische Ansätze:

(1) Ist die gesuchte Information über die UI sicht- und editierbar (etwa das „Büro“-Feld eines Benutzers in „Active Directory Benutzer und Computer“), kann ein Testbenutzer erstellt werden, dem für das gesuchte Attribut ein Dummywert vergeben wird. Anschließend kann mit LDP oder ADSIEdit in allen Benutzerattributen nach dem Dummywert gesucht werden.

(2) Das MSDN bietet eine Übersicht zu allen Objektklassen [3], etwa auch für das Benutzerobjekt, „User“ [4]. Hinterlegt sind die jeweils verfügbaren Attribute sowie deren Syntax, auf die wir später näher eingehen werden.

(3) Die favorisierte Suchmaschine sollte eine gute Anzahl von Treffern liefern, wenn es um Active Directory und Attribute geht. Die Chance ist groß, dass man, sollte man auf der Suche nach einem Attribut sein, einen Foreneintrag oder ein Blogposting findet, das sich mit einer ähnlichen Suche oder zumindest demselben Attribut bereits befasst hat.

Sind die Attribute und die Attributwerte nach denen man suchen möchte gefunden, muss ein geeignetes Werkzeug für die Suchabfrage gefunden und damit die Suchanfrage erstellt werden. Da Active Directory ein LDAP-konformer Verzeichnisdienst ist, kann für die Abfrage von Daten jede LDAP-Suche verwendet werden. Es ist prinzipiell möglich, jedes Programm zu verwenden, das in der Lage ist, LDAP-Anfragen an einen Verzeichnisserver zu senden. Von den bereits vorgestellten Werkzeugen bieten sich beispielsweise LDP und DSQuery an. Das wohl mächtigste Suchwerkzeug ist wohl das bereits genannte ADFind von joeware, da es neben vielen bereits vorgefertigten Schaltern eine große Anzahl an Suchoptionen beinhaltet. Das Kommandozeilenprogramm wurde speziell für die Suche in Active Directory und ADAM (Active Directory Application Mode, jetzt: „AD LDS“) entwickelt.

Anfragen sind nach einem einfachen Prinzip aufgebaut. Der einfachste Suchanfrage sieht so aus:

```
(attributName <Vergleichsoperator> Wert)
```

Suchen wir etwa alle Objekte, die im Feld „Büro“ die exakte Raumnummer „227“ enthalten, könnte unser Filter wie folgt aussehen:

```
(physicalDeliveryOfficeName=227)
```

Das Attribut, das wir angeben, ist „physicalDeliveryOfficeName“ – das Attribut, das sich hinter dem Feld „Office“ aus „Active Directory Benutzer und Computer“ versteckt. Als Suchwert haben wir „227“ eingegeben.

Für eine Suche mit dem Befehlszeilenprogramm ADFind müsste folgendes Kommando abgesetzt werden:

```
C:\Windows\system32>adfind -default -f "(physicalDeliveryOfficeName=227)" sAMAccountName

AdFind V01.37.00cpp Joe Richards (joe@joeware.net) June 2007

Using server: dc.intern.frickelsoft.net:389
Directory: Windows Server 2003
Base DN: DC=intern,DC=frickelsoft,DC=net

dn:CN=Florian Frommherz,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>sAMAccountName: florian

dn:CN=Jimbo Jones,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>sAMAccountName: jimjones

dn:CN=Daniela Reinhards,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>sAMAccountName: dareinhards


3 Objects returned
```

ADFind wird mit den Parametern „-default“, „-f <filter>“ und „sAMAccountNAme“ aufgerufen. „-Default“ weist ADFind an, die Domänenpartition der Domäne des gerade angemeldeten Benutzers zu verwenden. Nach „-f“ wird der Filter angegeben. Als optionalen Parameter wurde hier im Beispiel ein Attribut angegeben, das ADFind für gefundene

Suchergebnisse anzeigen soll: den „*sAMAccountName*“, der den Anmeldenamen darstellt. Werden keine Anzeigeattribute angegeben, listet ADFind einfach alle Attribute auf, die es zu den Suchergebnissen findet.

Die gleiche Suche kann mit LDP durchgeführt werden. Nach dem Verbinden und Authentifizieren (Connect und Bind) mit dem Domänencontroller kann im LDP-Menü „Browse“ – und anschließend „Search“ ausgewählt werden. Alternativ lässt sich der Suchdialog durch das Tastenkürzel STRG+S öffnen.

Der Suchdialog von LDP erwartet ähnliche Informationen. Im Feld „Base DN“ wird der Startzweig der Suche erwartet – in ADFind konnte der Parameter „-default“ für den Start der Domänenpartition angegeben werden. In LDP muss der komplette Pfad angegeben werden, etwa DC=intern,DC=frickelsoft,DC=net für die Domäne intern.frickelsoft.net. Im Feld „Filter“ wird der LDAP-Filter eingefügt, den der Systemverwalter ausführen möchte.

Die Option „Scope“ wurde bisher noch nicht erwähnt. LDAP bietet die Möglichkeit, die Suchweite einzuschränken. Es werden drei Optionen angeboten: „Base“ schränkt die Suche auf das Basis-Objekt in „Base DN“ ein – es wird effektiv nur das angegebene Objekt durchsucht. „One Level“ weist LDAP an, das Basisobjekt und seine Kindsobjekte und damit die nächste Ebene im Verzeichnisbaum zu betrachten und „Subtree“ wird LDAP zwingen, den gesamten Verzeichnisbaum vom Startzweig an abwärts zu durchsuchen. In ADFind kann die Suchweite mit dem Schalter –s angegeben werden.

Im Feld „Attributes“ wird das Attribut „samaccountname“ eingegeben, das auch im vorherigen Beispiel mit Adfind zum Einsatz kam. Per „Run“ wird die Suche gestartet. Auch mit LDP erhält der Administrator drei Benutzer, die sich offensichtlich das Büro 227 teilen.

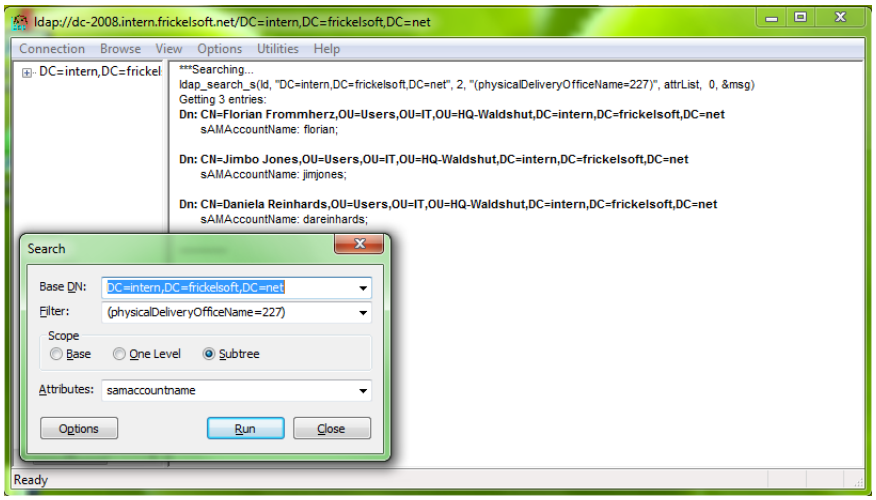


Abbildung 7: Ein Suchbeispiel mit LDP

Die Suche mit anderen bekannten Applikationen geht stets ähnlich zu. In den meisten Fällen wird verlangt, dass der Benutzer einen Startzweig für die Suche angibt und anschließend den Filter formt. Für alle folgenden Beispiele wird das Kommandozeilenprogramm ADFind für die Suchen verwendet; die Beispiele können jedoch ausnahmslos mit LDP durchgeführt werden.

Logische Operatoren

AND / OR-Operatoren

Nachdem der erste erstellte Filter aus einer einfachen Bedingung, nämlich einer Prüfung eines Wertes zu einem bestimmten Attribut bestand, ist es nun Zeit, die nächste Stufe zu erreichen und sich der nächsten Aufgabe zu stellen: mehrere Bedinungen in einem Filter.

Hierbei können mehrere Bedingungen per logischem Und („&“) und dem logischen Oder („|“) verknüpft werden. Die logische Verknüpfung wird dabei vor den Bedingungen angestellt. Soll erneut nach Objekten gesucht werden aus einem bestimmten Büro gesucht werden, die allerdings einen vorgegebenen Vornamen (Attribute *„givenName“*) wie etwa „Daniela“ haben, müsste der Filter so lauten:

```
(&(physicalDeliveryOfficeName=227)(givenName=Daniela))
```

Die beiden Bedingungen, physicalDeliveryOfficeName=227 und givenName=Daniela werden durch das logische Und vor den beiden Bedinungen verknüpft. Die beiden umschließenden Klammern sind hierbei optional und dienen nur der besseren Übersicht. ADFind zeigt uns daraufhin folgende Ergebnisse:

```
C:\Windows\system32>adfind -default -f
"(&(physicalDeliveryOfficeName=227)(givenName=Daniela))" dn sAMAccountName

AdFind V01.37.00cpp Joe Richards (joe@joeware.net) June 2007

Using server: dc.intern.frickelsoft.net:389
Directory: Windows Server 2003
Base DN: DC=intern,DC=frickelsoft,DC=net

dn:CN=Daniela Reinhards,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>sAMAccountName: dareinhards


1 Objects returned
```

Von den bisher drei angezeigten Mitarbeitern wird nur noch eine Mitarbeiterin namens Daniela angezeigt. Um die Suchlogik noch ein wenig zu verschärfen, lässt sich hierzu eine weitere ODER-Bedingung einbauen. Gesucht werden sollen Mitarbeiter im Büro 227, die entweder „Daniela“ oder „Florian“ heißen:

```
C:\Windows\system32>adfind -default -f
"(&(physicalDeliveryOfficeName=227)(|(givenName=Daniela)(givenName=florian)))" dn
sAMAccountName

...

dn:CN=Florian Frommherz,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>sAMAccountName: florian

dn:CN=Daniela Reinhards,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>sAMAccountName: dareinhards


2 Objects returned
```

Der Filter ist zu einer beachtlichen Größe herangewachsen. In einer auseinandergezogenen Schreibweise sieht der Filter nun so aus:

```
(& (physicalDeliveryOfficeName=227) (| (givenName=Daniela)(givenName=florian) ) )
```

Ein für das Verständnis gutes Vorgehen ist die Klammerbeseitigung der Mathematik: innere Klammern vor äußeren Klammern. Nimmt man sich die erste Klammer vor:

```
(| (givenName=Daniela)(givenName=florian) )
```

Sucht dieser Filterterm nach Vornamen, die entweder „Daniela“ oder „Florian“ lauten. Das logische Oder („|“) wird wie das logische Und im vorherigen Beispiel vor die Suchbedingungen gestellt. Objekte, die entweder Operand1 (hier: „givenName=Daniela“) oder Operand2 (hier: „givenName=Florian2) oder beide Merkmale aufweisen können, werden in die Ergebnisliste aufgenommen.

Nun wird der vordere Term betrachtet. Er besteht aus nur einer Bedingung, die per logischem Und an den verschatelten Term verknüpft wird:

```
(& (physicalDeliveryOfficeName=227) ... )
```

Suchergebnisse umfassen alle Objekte, die als Office „227“ eingetragen haben und entweder „Florian“ oder „Daniela“ als Vornamen in „givenName“ tragen.

Warum Computer manchmal auch nur Menschen sind

In den vorherigen Kapiteln wurde Active Directory als Verzeichnisdienst behandelt, der Objekte gemäß seines vorgegebenen Schemas erzeugt. Objekte des Schemas folgen einen bestimmten Vererbungsmuster, das den Objekten untereinander eine Hierarchie in Form eines Stammbaumes beschert. Die erste Objektklasse, von dem alle weiteren Objekte abgeleitet werden, ist „top“. Es enthält einen Mindestsatz an Attributen, die alle von ihm abgeleiteten Objekte erben. Entlang der Hierarchielinie lassen sich alle Objekte des Schemas einordnen. So auch das Objekt „user“, aus dem Benutzerobjekte im Verzeichnis erzeugt werden. Das Attribut „objectClass“ beinhaltet eine Liste aller Objektklassen, denen ein Verzeichnisobjekt angehört. Der Auszug aus LDP.exe für einen Benutzer sieht so aus:

objectClass (4): top; person; organizationalPerson; user;

Zu erkennen ist die Hierarchie, aus der ein Benutzerobjekt im Verzeichnis erstellt wird: Ausgehend von der Objektklasse “top”, gibt es eine Objektklasse „person“, die als direkte Kindsklasse alle Attribute von „top“ erbt. Die Kindsklasse „organizationalPerson“ erbt alle Attribute von „person“. „organizationalPerson“ ist somit im Besitz aller Attribute der Klasse „top“, sowie aller zusätzlich konfigurierten Attribute der Klasse „person“. Zuletzt wird die „user“-Objektklasse aufgeführt, die von „organizationalPerson“ Attribute erbt und deshalb mit allen Vater- und Großvaterklassen verwandt ist.

Interessant wird die Ahnenkunde der Objektklassen, wenn man sich das Attribut „objectClass“ für Computerobjekte ansieht:

objectClass (5): top; person; organizationalPerson; user; computer;

Verglichen mit dem Stammbaum eines Benutzerobjekts lässt sich erkennen, dass die Computerobjekte der selben Hierarchielinie folgen, der Benutzerobjekte im Verzeichnis entspringen – und dass Computerobjekte offensichtlich Kindobjekte der „user“-Objektklasse im AD-Schema sind. Daraus lässt sich ableiten, dass Computer eigentlich Benutzer sind – mit ein paar zusätzlichen Attributen.

Ruft man sich die Funktionsweise der Windows-Sicherheit und der Berechtigungsverwaltung ins Gedächtnis, erhält diese Implementierung plötzlich Sinn: Computern können in Active Directory Rechte zugewiesen werden, ähnlich wie Benutzern. Sie müssen sich auch in Active Directory authentifizieren, um Kerberos-Diensttickets erhalten zu können und in regelmäßigen Abständen ihre Passworte ändern – ähnlich wie Benutzer. Ihnen können Berechtigungen zugewiesen werden und sie sind Mitglied der Gruppe der „Authentifizierten Benutzer“. Im Grunde sind Computer nur eine weitere Art von Benutzern, die in Active Directory verwaltet werden können.

Zum Nachteil wird dieser Umstand wenn man versucht, alle Benutzerobjekte aus dem Verzeichnis abzurufen:

```
C:\Windows\system32>adfind -default -f "(objectClass=user)" -nodn sAMAccountNAme

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc.intern.frickelsoft.net:389
Directory: Windows Server 2003
Base DN: DC=intern,DC=frickelsoft,DC=net

>sAMAccountName: Administrator
>sAMAccountName: Guest
>sAMAccountName: DC$
>sAMAccountName: krbtgt
>sAMAccountName: florian
>sAMAccountName: joderninger
>sAMAccountName: wsus
>sAMAccountName: LoKinski
>sAMAccountName: TiNordmann
…
```

```
>sAMAccountName: BER-SA-002
>sAMAccountName: BER-SA-003
>sAMAccountName: BER-SA-004
>sAMAccountName: CH-BE-CS-000
>sAMAccountName: CH-BE-CS-001
…
```

Denn wird nach objectClass=user gesucht, werden auch die Computerobjekte im Verzeichnis gefunden.

Um nur nach Benutzern suchen zu können, wird ein weiteres Attribut zu Rate gezogen: „objectCategory“. ObjectCategory erlaubt es, innerhalb von Objektklassen feinere Abstufungen zu unternehmen. So werden Objekte weiteren Kategorien zugeordnet. Im Falle von Benutzerobjekten wäre das:

objectCategory: CN=Person,CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net;

oder kurz “Person”. Computer hingegen gehören der Objektkategorie

objectCategory: CN=Computer,CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net;

Die Suche nach Benutzerkonten ohne Computer sieht demnach so aus:

```
C:\Windows\system32>adfind -default -f "&(objectClass=user)(objectCategory=person)" -nodn
sAMAccountName

...

>sAMAccountName: Administrator
>sAMAccountName: Guest
>sAMAccountName: florian
>sAMAccountName: joderninger
>sAMAccountName: wsus
>sAMAccountName: maildelivery
>sAMAccountName: benjamin
>sAMAccountName: recovery
>sAMAccountName: flo
>sAMAccountName: recvrAgnt
>sAMAccountName: jimjones
>sAMAccountName: todarnelli
>sAMAccountName: tiheimgart
>sAMAccountName: SQLservice
>sAMAccountName: LuGottholf
>sAMAccountName: ChMehbauer
>sAMAccountName: LeNaegele
>sAMAccountName: ChHohnzoll
>sAMAccountName: StBeckenbauer
```

Durch das Verbinden der Objektklasse „user“ und der Objektkategorie „person“ werden Computerobjekte von der Suche ausgeschlossen.Wann immer nach Benutzerobjekten gesucht wird, sollte also folgender Filter verwendet werden:

```
(& (objectClass=user)(objectCategory=person))
```

objectCategory=person oder objectCategory=user reichen als Angabe für unseren Filter (anstelle von objectCategory=CN=Person,CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net), da der Filter intern weiterverarbeitet wird und ein „Mapping“ zwischen „person“ und dem eigentlichen DN stattfindet.

Jokerzeichen

Ist der genaue Wert eines Attributes nicht bekannt oder wird eine Suche nach ähnlich lautenden Attributwerten von Objekten benötigt, kann das Jokerzeichen eingesetzt werden. Das Jokerzeichen für LDAP-Filter ist der Stern („*“):

```
C:\Windows\system32>adfind -default -f "(physicalDeliveryOfficeName=22*)" dn
sAMAccountName physicalDeliveryOfficeName

...

dn:CN=Klum\, Andreas,OU=Users,OU=Sales,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,D
C=net
>physicalDeliveryOfficeName: 220
>sAMAccountName: AnKlum

dn:CN=Feist\, Jusuf,OU=Users,OU=Sales,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC
=net
>physicalDeliveryOfficeName: 220
>sAMAccountName: JuFeist

dn:CN=Klippenspringer\, Christoph,OU=Users,OU=Sales,OU=HQ-Waldshut,DC=intern,DC=
frickelsoft,DC=net
>physicalDeliveryOfficeName: 220
>sAMAccountName: ChKlippenspringer

dn:CN=Fredes\, Lara,OU=Users,OU=Management,OU=HQ-Waldshut,DC=intern,DC=frickelso
ft,DC=net
>physicalDeliveryOfficeName: 224
>sAMAccountName: LaFredes

dn:CN=Suederstedt\, Franka,OU=Users,OU=Management,OU=HQ-Waldshut,DC=intern,DC=fr
ickelsoft,DC=net
>physicalDeliveryOfficeName: 224
>sAMAccountName: FrSuederstedt
...
18 Objects returned
```

Mit dem Filter „(physicalDeliveryOfficeName=22*)“ wird das Verzeichnis angewiesen, alle Objekte des Verzeichnisses aufzulisten, deren Attributwerte in physicalDeliveryOfficeName mit „22“ beginnt. Es spielt dabei keine Rolle, wie viele oder welche weiteren Zeichen nach den beginnenden „22“ folgen. Der Ergenissatz kann demnach aus Objekten bestehen, die das Büro „22“, „228“, „22939393“, „22a“ oder „22 im oberen Stock“ zugewiesen bekamen. Das Jokerzeichen kann auch am Anfang der Suchzeichenkette oder in der Mitte stehen:

```
(physicalDeliveryOfficeName=*22)
```

Größer/Kleiner-Operatoren

Neben dem Vergleichszeichen („=“), das in den bisherigen Filtern eingesetzt wurde, gibt es zwei weitere Vergleichsoperatoren, die für LDAP-Suchen eingesetzt werden können:

<=	Kleiner oder gleich
>=	Größer oder gleich

Anders als bei der Mathematik werden nicht nur Zahlenwerte mit diesen Operatoren verglichen. Das Verzeichnis prüft die Attributwerte lexikographisch - es ist demnach auch gestattet, Buchstaben und andere Muster zu verwenden. Der Filter

```
(givenName >= Ge*)
```

listet alle Objekte, deren Vorname größer als „Ge“ mit beliebig vielen Zeichen dahinter ist. Objekte namens „Gerd“, „Jens“, „Volker“, „Tanja“ oder „Matthias“ würden ausgegeben werden. „Anneliese“, „Berta“, „Christian“ oder „Gabriela“ hingegen nicht.

Ein Zahlenbeispiel mit der Büronummer wäre wie folgt:

```
C:\Windows\system32>adfind -default -f
"(&(physicalDeliveryOfficeName>=220)(givenName=F*))" sAMAccountName
physicalDeliveryOfficeName

...

dn:CN=Florian Frommherz,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,D
C=net
>physicalDeliveryOfficeName: 227
>sAMAccountName: florian

dn:CN=Beinlos\, Florian,OU=Users,OU=Management,OU=HQ-Waldshut,DC=intern,DC=frick
elsoft,DC=net
>physicalDeliveryOfficeName: 222
>sAMAccountName: FlBeinlos

dn:CN=Br³ckenhelfer\, Frank,OU=Users,OU=Management,OU=HQ-Waldshut,DC=intern,DC=f
rickelsoft,DC=net
>physicalDeliveryOfficeName: 222
>sAMAccountName: FrBr³ckenhelfer

dn:CN=Suederstedt\, Franka,OU=Users,OU=Management,OU=HQ-Waldshut,DC=intern,DC=fr
ickelsoft,DC=net
>physicalDeliveryOfficeName: 226
>sAMAccountName: FrSuederstedt

4 Objects returned
```

Das obige Beispiel listet Objekte, deren Vorname mit “F” beginnt und in einem Büro sitzen, dessen Nummer größer oder gleich 220 ist.

Das nächste Beispiel listet Objekte, deren Büronummer zwischen 220 und 225 liegt:

```
C:\Windows\system32>adfind -default -f "(&(physicalDeliveryOfficeName>=220)(phys
```

```
icalDeliveryOfficeName<=225))" sAMAccountName physicalDeliveryOfficeName

...

dn:CN=Klum\, Andreas,OU=Users,OU=Sales,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,D
C=net
>physicalDeliveryOfficeName: 220
>sAMAccountName: AnKlum

dn:CN=Feist\, Jusuf,OU=Users,OU=Sales,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC
=net
>physicalDeliveryOfficeName: 220
>sAMAccountName: JuFeist

dn:CN=Klippenspringer\, Christoph,OU=Users,OU=Sales,OU=HQ-Waldshut,DC=intern,DC=
frickelsoft,DC=net
>physicalDeliveryOfficeName: 220
>sAMAccountName: ChKlippenspringer

dn:CN=Br³ckenhelfer\, Frank,OU=Users,OU=Management,OU=HQ-Waldshut,DC=intern,DC=f
rickelsoft,DC=net
>physicalDeliveryOfficeName: 222
>sAMAccountName: FrBr³ckenhelfer

...

dn:CN=Gottholf\, Rainer,OU=Users,OU=Management,OU=HQ-Waldshut,DC=intern,DC=frick
elsoft,DC=net
>physicalDeliveryOfficeName: 224
>sAMAccountName: RaGottholf

dn:CN=Ulm\, Tina,OU=Users,OU=Management,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,
DC=net
>physicalDeliveryOfficeName: 224
>sAMAccountName: TiUlm


12 Objects returned
```

Offensichtlich gibt es niemanden, der in Büro 225 oder 220 sitzt.

Ein wichtiger Hinweis an dieser Stelle: die Vergleichsoperatoren beinhalten stets das Gleichheitszeichen (>=, <=). In LDAP und damit auch in Active Directory kann nicht mit „größer“ oder „kleiner“ gesucht werden, sondern stets inklusive dem angegebenen Wert. Filter wie „größer als“ oder „kleiner als“ können nicht verwendet werden. Der Filter

```
(physicalDeliveryOfficeName>219)
```

würde in einem Filter-Fehler führen, den der LDAP-Server mit einer Fehlermeldung quittiert und anschließend seinen Dienst verweigert. Sucht man nach Werten, die größer als 100 sind, filtert man nach >=101 (größer oder gleich). Für Werte kleiner 1000, wird nach <=999 (kleiner oder gleich) gesucht.

Nicht-Operator (!)

Die bisher vorgestellten Filter suchen nach bestimmten Kriterien und Attributwerten. Was aber, wenn nach Objekten gesucht werden soll, die einen bestimmten Attributwert nicht haben sollen? Hierfür gibt es den nicht-Operator („!“), der wie folgt eingesetzt werden kann:

```
(!attribut <Operator> Wert)
```

Im Falle des Officebeispiels, wenn alle Benutzer ausser den Büro-227-Mitarbeitern angezeigt werden sollen, wäre das:

```
(!physicalDeliveryOfficeName=227)
```

Der nicht-Operator kann auch in Verbindung mit anderen komplexeren Suchen verwendet werden, beispielsweise mit AND und OR-Suchen:

```
(&(!givenName=F*)(physicalDeliveryOfficeName=227))
```

Der Stern-Operator („enthält Wert“)

Wer den oben genannten Filter

```
(!physicalDeliveryOfficeName=227)
```

ausprobiert hat, wird feststellen, dass eine ganze Reihe mehr Objekte ausgegeben werden als nur die, die nicht das Büro 227 zugewiesen bekamen: nämlich auch alle Objekte, die keinen Wert im Attribut physicalDeliveryOfficeName haben.

```
C:\Windows\system32>adfind -default -f "(!physicalDeliveryOfficeName=227)" -c

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc.intern.frickelsoft.net:389
Directory: Windows Server 2003
Base DN: DC=intern,DC=frickelsoft,DC=net


1364 Objects returned
```

Der Schalter –c in ADFind weist das Tool an, keine Objekte auszugeben, dafür aber die Ergebnisse zu zählen (c für „count“). In diesem Fall werden 1364 Objekte gezählt, die offensichtlich nicht im Büro 227 sind. Ein kurzer Check zeigt:

```
C:\Windows\system32>adfind -default -f "(&(objectCategory=person)(objectClass=user))" -c

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc.intern.frickelsoft.net:389
Directory: Windows Server 2003
Base DN: DC=intern,DC=frickelsoft,DC=net


700 Objects returned
```

Es werden mehr Objekte ohne Büro=227 angezeigt werden, als Benutzer im Verzeichnis hinterlegt sind! Es wird also eine neue Abfrage für die Mitarbeiter benötigt, die nicht im Büro 227 sitzen, dennoch einen Wert in physicalDeliveryOfficeName zugewiesen bekamen.

Für diese Fälle wird der Stern-Operator benutzt. Der Sternoperator ist bekannt aus bereits gezeigten Suchen, in denen er als Jokerzeichen verwendet wird. Um festzustellen, welche Objekte in einem Attribut einen Wert zugewiesen bekamen, kann folgende Syntax verwendet werden:

```
(attribut = *)
```

Objekte mit leeren Attributwerten werden somit ausgeschlossen. Um das vorherige Beispiel zu verbessern, nutzen wir den Filter

```
(&(physicalDeliveryOfficeName=*) (!physicalDeliveryOfficeName=227))
```

Objekte ohne Attributwert können mit einer Kombination aus dem Nicht- und dem Sternoperator gefunden werden:

```
(!attribut = *)
```

Die Ambiguous Name Resolution (ANR) für Suchen verwenden

Eine der wohl häufigsten Anfragen an Active Directory ist die Suche nach Namen. Sowohl Outlook in Verbindung mit Exchange als auch die Personensuche in Windows verwenden eine spezielle Suchanfrage an das Verzeichnis, um möglichst effektiv nach Personeninformationen zu suchen. Woher sollen Outlook oder Windows wissen, ob mit dem Suchwort „San“ ein Vorname wie „Sandra“, ein Nachname wie „Santana“ oder ein Login-Alias oder Mailadresse gemeint ist?

Um externen Anwendungen und Benutzern diese komplexe Abfrage mit einer Vielzahl von Attributen zu ersparen, kennt Active Directory eine besondere Suchfunktion: die Ambiguous Name Resolution (ANR). Mit dem ANR als Suchfilter kann eine ganze Reihe von Attributen auf einen Schlag durchsucht werden. Aus dem Filter

```
(anr = <suchbegriff>)
```

formt Active Directory einen komplizierteren, internen Suchfilter:

```
(|
        (displayName=<suchbegriff>*)
        (givenName=<suchbegriff>*)
        (legacyExchangeDN=<suchbegriff>*)
        (msDS-AdditionalSamaccountName=<suchbegriff>*)
        (msDS-PhoneticCompanyName=<suchbegriff>*)
        (msDS-PhoneticDepartment=<suchbegriff>*)
        (msDS-PhoneticDisplayName=<suchbegriff>*)
        (msDS-PhoneticFirstName=<suchbegriff>*)
        (msDS-PhoneticLastName=<suchbegriff>*)
        (physicalDeliveryOfficeName=<suchbegriff>*)
        (proxyAddresses=<suchbegriff>*)
        (name=<suchbegriff>*)
        (sAMAccountName=<suchbegriff>*)
        (sn=<suchbegriff>*)
)
```

Dem Suchbegriff wird der Stern als Jokerzeichen angefügt, um eine möglichst breite Masse von Ergebnissen zu erlangen. In Windows Server 2008 wird in 14 unterschiedlichen Attributen per logischer ODER-Abfrage nach dem Suchbegriff geforscht. Je nach Betriebssystem oder weiterer installierter Komponenten (zum Beispiel Microsoft Exchange) kann die Anzahl der Suchattribute variieren. Neue Attribute können dem ANR-Set hinzugefügt werden, indem das Schema entsprechend modifiziert wird. Hierzu kann über das Schema-MMC-Snapin die entsprechende Checkbox „Ambiguous Name Resolution (ANR)“ für das Attribut ausgewählt werden. Alternativ kann Bit#2 des Attributes „searchFlags“ für das entsprechende Schema-Attribut gesetzt werden.

Suchen mit dem ANR sind nicht kostspieliger als andere Suchen. Da nur Attribute in den ANR aufgenommen werden die auch einen Index im Verzeichnis besitzen, kann sich der Abfrageprozessor bei der nach Ergebnissen auf einem effizienten Suchpfad bewegen.

Suche nach Objekten an bestimmten Verzeichnisorten

Suchen nach Verzeichnisdienstobjekten an bestimmten Orten, etwa gewünschten OUs gestalten sich als unmöglich bzw. nicht machbar, da es in Active Directory keine Möglichkeit gibt nach Objektstandorten zu filtern. Der distinguishedName (DN), der in vorherigen Beispielen als Startzweig für Suchen genutzt wurde, ist kein eigenständiges Attribut im Verzeichnis – Active Directory hat keinen Speicherort dafür reserviert. Stattdessen ist distinguishedName ein „constructed" Attribut, das beim Browsen des Verzeichnisses „on demand" anhand der Vater- und Großvaterknoten im Verzeichnisbaum ausgerechnet wird. Folglich lassen sich hiermit auch keine Joker-Suchen ausführen:

```
C:\Windows\system32>adfind -default -f
"(distinguishedName=*OU=newOU,DC=intern,DC=frickelsoft,DC=net)"

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008
Base DN: DC=intern,DC=frickelsoft,DC=net


0 Objects returned
```

Die Suche ergibt keine Treffer, wobei sich mehrere Objekte in der OU befinden. Um nach Objekten in OUs suchen zu können, muss der Administrator die OU als Basis-DN angeben, um die entsprechenden Objekte zu finden:

```
C:\Windows\system32>adfind -b "OU=newOU,DC=intern,DC=frickelsoft,DC=net" dn

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008

dn:OU=newOU,DC=intern,DC=frickelsoft,DC=net

dn:CN=Jake Tomson,OU=newOU,DC=intern,DC=frickelsoft,DC=net

dn:CN=Tamara Albers,OU=newOU,DC=intern,DC=frickelsoft,DC=net


3 Objects returned
```

Eine weitere gängige Praxis ist das Füllen eines nicht verwendeten Objektattributes mit dem Pfad des Objektes. Dabei wird ein Skript erstellt, das sich alle gewünschten Objekte des Verzeichnisses ansieht, sich ihren Pfad merkt und diesen Pfad in ein lesbares, leeres Attribut des Objektes schreibt. Anschließend kann in einem LDAP-Filter nach dem umfunktionierten Attribut gesucht werden.

Active Directory und Zeitstempel

Zeitstempel werden in Active Directory in einem etwas gewöhnungsbedürftigen Format gespeichert. Es gibt zwei Arten von Zeitstempeln in Active Directory. Im ersten Fall handelt es sich um die Speicherung des Zeitstempels im UTC-Format:

```
C:\Windows\system32>adfind -default -f "samaccountname=florian" whenCreated

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008
Base DN: DC=intern,DC=frickelsoft,DC=net

dn:CN=Florian Frommherz,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>whenCreated: 20070324170248.0Z


1 Objects returned
```

Das Attribut ist im Format YYYYMMDDHHmmss.0Z gespeichert. Verwirrender ist dabei ein Zeitstempel anderer Attribute, wie pwdLastSet, das die Zeit der letzten Passwortänderung speichert:

```
C:\Windows\system32>adfind -default -f "samaccountname=florian" pwdLastSet

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008
Base DN: DC=intern,DC=frickelsoft,DC=net

dn:CN=Florian Frommherz,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>pwdLastSet: 128928529479850640


1 Objects returned
```

128928529479850640 ist zugegebenermaßen ein Zeitstempel, der nicht ohne Weiteres zu entschlüsseln ist. In der Tat hat die Zahl eine Bedeutung: Sie beschreibt die Anzahl der seit dem 1. Januar 1601 vergangenen 100 Nanosekunden-Blöcke. Das soll bedeuten: Zwischen dem 1. Januar 1601 und dem Zeitpunkt, an dem florians Passwort das letzte Mal gesetzt wurde sind 128928529479850640 mal 100 Nanosekunden vergangen.

Benutzer von LDP und möglicherweise anderen LDAP-Browsern haben Glück, denn LDP rechnet das Datum bereits korrekt in ein brauchbares Zeitformat um:

pwdLastSet: 23.07.2009 22:02:27 W. Europe Daylight Time;

Für ADFind gibt es einen Schalter, der Zeitstempel automatisch umrechnet: -tdc:

```
C:\Windows\system32>adfind -default -tdc -f "samaccountname=florian" pwdLastSet

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008
Base DN: DC=intern,DC=frickelsoft,DC=net
```

```
dn:CN=Florian Frommherz,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>pwdLastSet: 07/23/2009-22:02:27 W. Europe Daylight Time


1 Objects returned
```

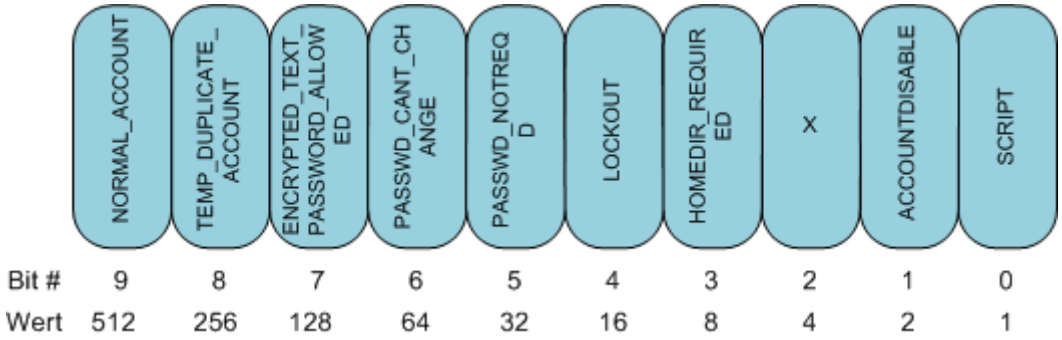
Da alle Zeitstempel von Administrationswerkzeugen mit grafischer Oberfläche benutzerfreundlich angezeigt werden, mögen sie erst zum Zeitpunkt der Erstellung eines Suchfilters interessant werden. Für die Nutzung von Zeiten in Filtern werden tatsächlich die oben gezeigten Zeitstempel verwendet. Das Format von whenCreated lässt sich relativ einfach nutzen:

```
C:\Windows\system32>adfind -default -f "whenCreated>=20090827000000.0Z" samaccountname

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008
Base DN: DC=intern,DC=frickelsoft,DC=net

dn:CN=WIN7--001,OU=Windows 7,OU=Machines,OU=IT,OU=HQ-
Waldshut,DC=intern,DC=frickelsoft,DC=net
>sAMAccountName: WIN7--001$

. . .


8 Objects returned
```

Mit Hilfe des >=-Operators lassen sich Objekte finden, die nach einem bestimmten Datum erstellt wurden (vorausgesetzt es wird whenCreated verwendet), <= lässt Objekte vor einem angegebenen Datum anzeigen. Das Beispiel wirft acht Objekte für das Verzeichnis aus, die nach dem 27.08.2009 00:00:00 erstellt wurden.

Bitmuster vergleichen (userAccountControl)

Active Directory speichert eine Reihe von Informationen in sogenannten Bitmasken. Bitmasken lassen sich als eine Reihe von Einstellungen darstellen, die entweder „aktiv“ oder „inaktiv“ bzw. „wahr“ oder „falsch“ sind. Für ein Beispiel im Umfeld der Benutzerkonten lässt sich das Attribut „userAccountControl“ aufführen. Das Attribut userAccountControl [5] besteht aus einer Reihe von Einstellungen, die alle gemeinsam hinter dem Attribut abrufbar sind. Dabei sieht das Attribut (ein Auszug) so aus:



Deaktiviert ein Administrator in „Active Directory Benutzer und Computer“ ein Benutzerkonto über das Kontextmenü, wird das Attribut „userAccountControl“ geändert. Zuständig für die Kontodeaktivierung ist das Bit #1, das im Bild den Namen „ACCOUNTDISABLE“ trägt. Das Bit wird auf „wahr“ gesetzt, was den Wert des gesamten Attributes um 2 erhöht. Kreuzt der Administrator zusätzlich noch an, dass der Benutzer sein Passwort nicht verändern darf, wird Bit #6, „PASSWD_CANT_CHANGE“ aktiviert, was den Wert des Attributes um 64 erhöht. Das Beispiel zeigt, dass das Benutzerkonto von Florian ein „normales“ Benutzerkonto ist:

```
C:\Windows\system32>adfind -default -f samaccountname=florian userAccountControl

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008
Base DN: DC=intern,DC=frickelsoft,DC=net

dn:CN=Florian Frommherz,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>userAccountControl: 512


1 Objects returned
```

Wäre das Benutzerkonto deaktiviert, also das Bit #1 aktiv, müsste der Wert um „2“ erhöht werden:

```
C:\Windows\system32>adfind -default -f samaccountname=florian userAccountControl

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008
Base DN: DC=intern,DC=frickelsoft,DC=net

dn:CN=Florian Frommherz,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net
>userAccountControl: 514


1 Objects returned
```

Die Schwierigkeit bei der Suche mit LDAP liegt nun darin, den richtigen Wert in der Suche abzufragen. Da es eine Vielzahl von Kombinationen von Bitmustern geben kann, ist es nicht einfach, die richtige „Zahl“, etwa „514“, zu suchen, wenn alle deaktivierten Konten gefunden werden sollen. Der Benutzer „Jimbo“ hat eventuell zusätzlich zu Bit #9 und Bit #1, wie Florian, zusätzlich Bit #0 aktiviert, sodass sein in AD Benutzer und Computer hinterlegtes Logonskript nicht ausgeführt wird. Der Wert seines userAccountControl-Attributes ist 515. Würde die Suche nun nach

```
(userAccountControl=514)
```

filtern, („normaler User“ + „Account deaktiviert“) um alle deaktivierten Konten zu finden, wäre Jimbo nicht in der Resultatsliste. Es muss folglich einen besseren Weg geben, danach zu filtern.

ADFind macht es einfach und lässt uns mit Hilfe der logischen Bit-Operatoren suchen:

```
C:\Windows\system32>adfind -default -bit -f "(&
(objectClass=user)(objectCategory=person)(userAccountControl:AND:=2))" dn

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Transformed Filter: (& (objectClass=user) (objectCategory=person)
(userAccountControl:1.2.840.113556.1.4.803:=2) )
Using server: dc-2008.intern.frickelsoft.net:389
Directory: Windows Server 2008
Base DN: DC=intern,DC=frickelsoft,DC=net

dn:CN=Guest,CN=Users,DC=intern,DC=frickelsoft,DC=net

dn:CN=krbtgt,CN=Users,DC=intern,DC=frickelsoft,DC=net

dn:CN=SUPPORT_388945a0,CN=Users,DC=intern,DC=frickelsoft,DC=net

dn:CN=Recovery,CN=Users,DC=intern,DC=frickelsoft,DC=net

dn:CN=SystemMailbox{93009BBC-F422-47A4-A04E-C77327CCDC62},CN=Microsoft Exchange System
Objects,DC=intern,DC=frickelsoft,
DC=net

dn:CN=SystemMailbox{9C63CC23-1304-4D68-9FB3-8D9E2ACCCBF3},CN=Microsoft Exchange System
Objects,DC=intern,DC=frickelsoft,
DC=net

dn:CN=2nd adm,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net

dn:CN=Jimbo Jones,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net

dn:CN=Damian Weiner,OU=Users,OU=Dev,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net

dn:CN=Jake Tomson,OU=newOU,DC=intern,DC=frickelsoft,DC=net

dn:CN=Tamara Albers,OU=newOU,DC=intern,DC=frickelsoft,DC=net


11 Objects returned
```

Das System findet 11 Benutzer, deren Benutzerkonto momentan deaktiviert ist. Bei näherer Betrachtung des Filters lässt sich feststellen, dass hier mehrere Dinge geschehen sind: zu allererst wird der Parameter „-bit“ verwendet, der ADFind anweist, spezielle Filter zu konvertieren. Im Filter werden drei Bedingungen verwendet, die mit einem logischen Und verknüpft sind – alle müssen erfüllt sein: die ersten beiden Bedingungen sind bereits behandelt worden. Es handelt sich um die Abfrage nach Benutzerkonten. Die dritte Bedingung sieht anders aus als die Bedingungen bisher:

```
(userAccountControl:AND:=2)
```

sie besteht nicht aus der Syntax

```
(attibut <Operator> Wert)
```

wie die bisher verwendeten Filter. Das liegt daran, dass LDAP einige zusätzliche Funktionen, sogenannte „Controls“ enthält, die zur Laufzeit mit ausgeführt und vom LDAP-Server, in diesem Falle Active Directory, verarbeitet werden. Controls haben keinen Namen sondern werden in der OID-Schreibweise dargestellt. Anhand der ADFind-Ausgabe lässt sich erkennen, wie das Control wirklich heißt:

```
(userAccountControl:1.2.840.113556.1.4.803:=2)
```

Nun wird deutlich, was der Schalter “–bit” in ADFind für den Administrator vollbringt: er übersetzt den Namen “AND” in den Controlnamen “1.2.840.113556.1.4.803“.

Intern führt der LDAP-Server eine Berechnung durch. Von allen Benutzerkonten wird das Attribut userAccountControl durchsucht und überprüft, ob das Bit #1 (Wert 2) gesetzt ist. Ist dies der Fall, wird es mit in die Resultatsliste aufgenommen.

Die Verwendung der Controls kann bei allen Bit-Muster-Attributen angewandt werden. Adfind versteht folgende Controls:

AND	`:AND:=`	`:1.2.840.113556.1.4.803:=`
OR	`:OR:=`	`:1.2.840.113556.1.4.804:=`
IN-CHAIN/NESTING	`:INCHAIN:=`	`:1.2.840.113556.1.4.1941:=`

Die Verwendung der Controls OR und IN-CHAIN wird im weiterführenden Whitepaper behandelt.

Bei der Verwendung von LDP oder anderen LDAP-Browsern ist die Handhabung der Controls und letztlich nicht ganz so komfortabel: hier muss der Administrator das zu verwendende Control auswendig kennen:

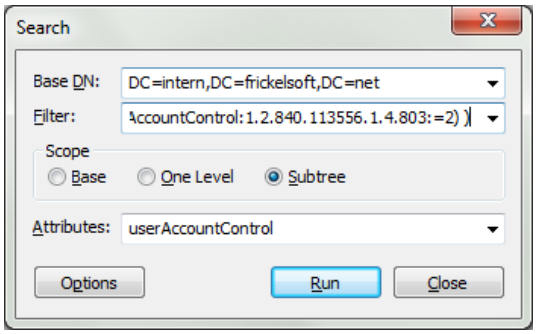


Abbildung 8: LDP kann mit Hilfe des richtigen Controls nach den gewünschten Bitmustern suchen.

Suchergebnisse exportieren

Eine Suche kann noch so gute Ergebnisse liefern, sie wird unnütz, wenn die Ergebnisse nicht weiterverwendet werden können. Um gesuchte Objekte aus dem Verzeichnis weiterverwenden zu können, bieten Suchtools stets Optionen für den Export der Resultate in das CSV-Format. Im CSV-Format angekommen können Ergebnisse weiterverarbeitet werden.

Ein häufig genutztes Tool ist CSVDE. CSVDE ist kommandozeilenbasiert – sein Name steht für Comma Separated Value Data Exchange. Für das Ausgeben und Wegspeichern von AD-Objekten kann CSVDE per Parameter angewiesen werden, einen LDAP-Filter zu benutzen, um ganz spezielle Objekte abzurufen.

Als erstes Beispiel soll ein vorheriger Filter noch einmal verwendet werden: es wird nach allen deaktivierten Benutzerkonten gesucht.

```
C:\Users\florian\Downloads>csvde -d "DC=intern,DC=frickelsoft,DC=net" -r
"(&(objectClass=user)(objectCategory=person)(userAccountcontrol:1.2.840.113556.1.4.803:=2)
)" -f output.csv -l dn,samaccountname,userAccountControl

Connecting to "dc-2008.intern.frickelsoft.net"
Logging in as current user using SSPI
Exporting directory to file output.csv
Searching for entries...
Writing out entries
...........
Export Completed. Post-processing in progress...
11 entries exported

The command has completed successfully
```

Wie die Ausgabe des Tools zeigt, wurden 11 Einträge exportiert. Ähnlich wie bei den bereits bekannten Tools erwartet CSVDE eine Reihe von Parametern, um die richtigen Ergebnisse abrufen zu können. Mit dem Parameter „–d“ wird CSVDE die Basis-DN angegeben, an dem die Suche starten soll. Der Schalter „–r“ gibt den Suchfilter an. Hier ist es wichtig, den Suchfilter analog zu den bisherigen Beispielen in Klammern anzugeben. Der Schalter „-f“ weist das Tool an, die Einträge in eine kommaseparierte Datei mit nachfolgendem Dateinamen zu schreiben, wobei es die mit Komma getrennte (leerzeichenlose!) Liste nach dem Parameter „–l“ berücktsichtigt.

CSVDE kennt eine Reihe weiterer Parameter, die über die Hilfe des Tools angezeigt werden können.

Die exportierte Liste der gefundenen Objekte lässt sich nun weiterverarbeiten. Da sie im CSV-Format vorliegt, genügt ein einfacher Texteditor um die Ergebnisse einsehen zu können.

	A	B	C
1	DN	userAccountControl	sAMAccountName
2	CN=Guest,CN=Users,DC=intern,DC=frickelsoft,DC=net	66082	Guest
3	CN=krbtgt,CN=Users,DC=intern,DC=frickelsoft,DC=net	514	krbtgt
4	CN=SUPPORT_388945a0,CN=Users,DC=intern,DC=frickelsoft,DC=net	66050	SUPPORT_388945a0
5	CN=Recovery,CN=Users,DC=intern,DC=frickelsoft,DC=net	66050	recovery
6	CN=SystemMailbox{93009BBC-F422-47A4-A04E-C77327CCDC62},CN=Microsoft Exchange Syst	514	93009BBC-F422-47A4-A
7	CN=SystemMailbox{9C63CC23-1304-4D68-9FB3-8D9E2ACCCBF3},CN=Microsoft Exchange Syst	514	9C63CC23-1304-4D68-9
8	CN=2nd adm,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net	66050	flo
9	CN=Jimbo Jones,OU=Users,OU=IT,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net	514	jimjones
10	CN=Damian Weiner,OU=Users,OU=Dev,OU=HQ-Waldshut,DC=intern,DC=frickelsoft,DC=net	514	daweiner
11	CN=Jake Tomson,OU=newOU,DC=intern,DC=frickelsoft,DC=net	514	jatomson
12	CN=Tamara Albers,OU=newOU,DC=intern,DC=frickelsoft,DC=net	514	taalbers

Abbildung 9: Mit Excel lassen sich die Ergebnisse und die ausgegebenen Attribute übersichtlich anzeigen.

Suchen ausserhalb der Domänenpartition

Suchen über alle Domänen hinweg (mit dem globalen Katalog)

In den bisher aufgeführten Beispielen wurde stets die Suche in einer einzigen Domäne thematisiert. Zweifelsohne gibt es Situationen, in denen es nicht ausreicht in einer Domäne der Gesamtstruktur nach Ergebnissen zu suchen, sondern das Filtern von Objekten auf alle Domänen der Gesamtstruktur auszuweiten. Für die Werkzeuge, auf die in diesem Whitepaper das Hauptaugenmerk gelegt wurde, ADFind und LDP gibt es zwei unterschiedliche Ansätze, mit denen vorgegangen werden muss.

Für eine domänenübergreifende Suche wird stets der „Globale Katalog“ benötigt. Der Globale Katalog ist eine Domänencontroller-Rolle, die neben der jeweils eigenen Domänenpartition auch einen partiellen Attributsatz aller Objekte aller anderen im Forest befindlichen Domänen beinhaltet. Ein Domänencontroller mit GC-Rolle kennt also einen Subset von Attributen aller Objekte der vollständigen Gesamtstruktur. Das Suchen im Globalen Katalog kann demnach Ergebnisse aus allen Domänen der Gesamtstruktur liefern.

In ADFind lässt sich der Globale Katalog mit dem Schalter „-gcb“ aktivieren. Das Werkzeug sucht eigenständig nach einem geeigneten Domänencontroller und liefert Suchergebnisse.

```
C:\Users\florian\Downloads>adfind -gcb -f "(sn=Jansen)" dn

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc-2008.intern.frickelsoft.net:3268
Directory: Windows Server 2008

dn:CN=Eric Jansen,OU=Users,OU=IT,OU=Seattle,DC=us,DC=intern,DC=frickelsoft,DC=net

dn:CN=Dave Jansen,OU=Users,OU=Dev,OU=Munich,DC=intern,DC=frickelsoft,DC=net

dn:CN=James Jansen,OU=cool users,DC=fsft-extra,DC=net


3 Objects returned
```

Die Suche nach Objekten mit dem Nachnamen (sn) “Jansen” bringt drei Treffer: ein Benutzerobjekt aus der Hauptdomäne, ein Objekt aus der Childdomäne und ein Objekt aus einer anderen Domäne der Gesamtstruktur.

Bemerkenswert an der Ausgabe von ADFind ist, dass der Server an einem anderen Port angesprochen wurde. Bisherige Suchen wurden an Port 389 des Domänencontrollers aufgegeben, für Suchen am Globalen Katalog muss der Port 3268 verwendet werden. Dies ist auch die Vorgehensweise, die mit LDP angestrebt werden muss. Im Verbindungsdialog muss anstelle der Voreinstellung „389“ im „Port“-Feld der GC-Port „3268“ eingetragen werden. Anschließend wird bei der Suche der globale Katalog befragt.´

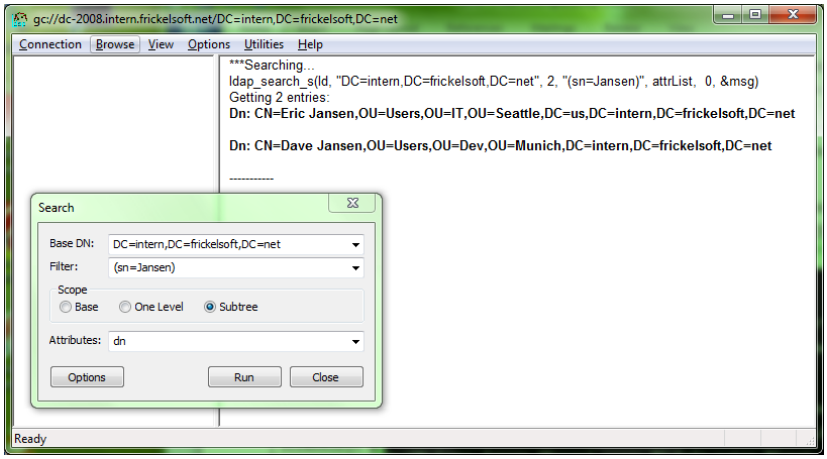


Abbildung 10: Suchergebnisse des Globalen Kataloges mit LDP.

Vergleicht man die Ausgabe von LDP im Bild mit der Ausgabe von ADFind, ist eine Diskrepanz zu erkennen: LDP zeigt nur zwei gefundene Objekte an, ADFind allerdings drei. Zwischen den beiden Aufzeichnungen wurde keines der Objekte gelöscht – die Suche in LDP ist schlichtweg anders eingestellt. Genauer betrachtet ist der Basis-DN schuld am fehlenden, letzten Ergebnis. Die Suche startet bei DC=intern,DC=frickelsoft,DC=net und das in LDP fehlende Objekt ist in DC=fsft-extra,DC=net zu finden. Da die Suche bei DC=intern,DC=frickelsoft,DC=net beginnt und alle Kindsobjekte, somit auch die Subdomäne „us“ durchläuft, werden alle Objekte dieses Domänenbaums gefunden – jedoch nicht die des parallelen Baums fsft-extra.net.

Auch dies ist nur Einstellungssache, denn LDP lässt sich beibringen, dass der Basis-DN anders lauten muss: anstatt DC=intern,DC=frickelsoft,DC=net wird „DC=net“, der größte gemeinsame Teiler der beiden Domänenbäume intern.frickelsoft.net und fsft-extra.net verwendet. Ausgehend von DC=net werden beide Bäume abgearbeitet und alle Ergebnisse gefunden.

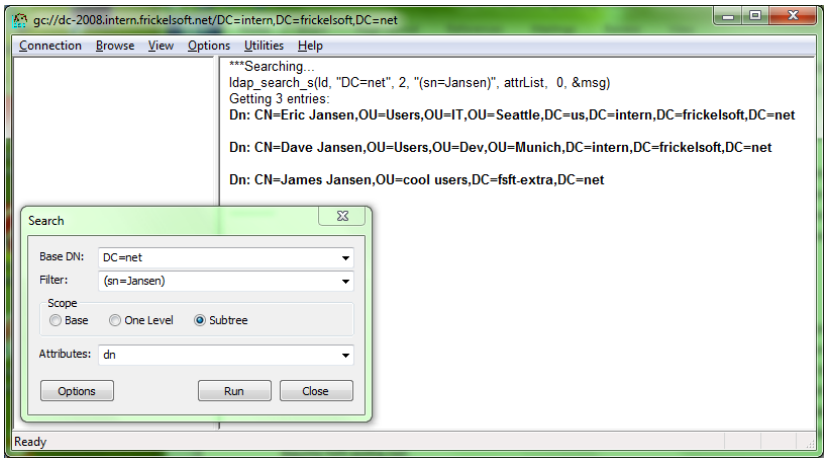


Abbildung 11: Verfügt die Gesamtstruktur über mehrere Domänenbäume, muss der Basis-DN angepasst werden.

Suchen in der Konfigurations- und Schemapartition

Das vorherige Beispiel hat es bewiesen: mit geschickten Händen und einem klugen Köpfchen lässt sich das Basis-DN für die Suche so verändern, dass nicht nur die Domänenpartition sondern auch alle weiteren Partitionen des Domänencontrollers abgefragt werden können – selbstverständlich auch die Partitionen des Schemas und der Konfiguration. ADFind kennt hierfür zwei separate Schalter: „-schema“ für die Schemapartition CN=Schema,CN=Configuration,DC=domain,DC=tld“ und „-config“ für die Konfigurationspartition CN=Configuration,DC=intern,DC=frickelsoft,DC=net.

Ein Beispiel das alle Attribute aus dem Schema ausgibt, die den Anfangsbuchstaben „Q“ aufweisen:

```
C:\Windows\system32>adfind -schema -f "&(objectClass=attributeSchema)(name=q*)" dn

AdFind V01.40.00cpp Joe Richards (joe@joeware.net) February 2009

Using server: dc3.intern.frickelsoft.net:389
Directory: Windows Server 2008 R2
Base DN: CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net

dn:CN=Quality-Of-Service,CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net

dn:CN=Query-Filter,CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net

dn:CN=Query-Policy-BL,CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net

dn:CN=Query-Policy-Object,CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net

dn:CN=QueryPoint,CN=Schema,CN=Configuration,DC=intern,DC=frickelsoft,DC=net


5 Objects returned
```

Man beachte dabei, dass ADFind korrekterweise die Base-DN des Schemas aufführt. In LDP können die Basis-DNs einfach über das Dropdownmenü ausgewählt werden. Der Suchdialog bietet alle schreibbaren Partitionen des verbundenen Domänencontrollers als Auswahlmöglichkeit an.

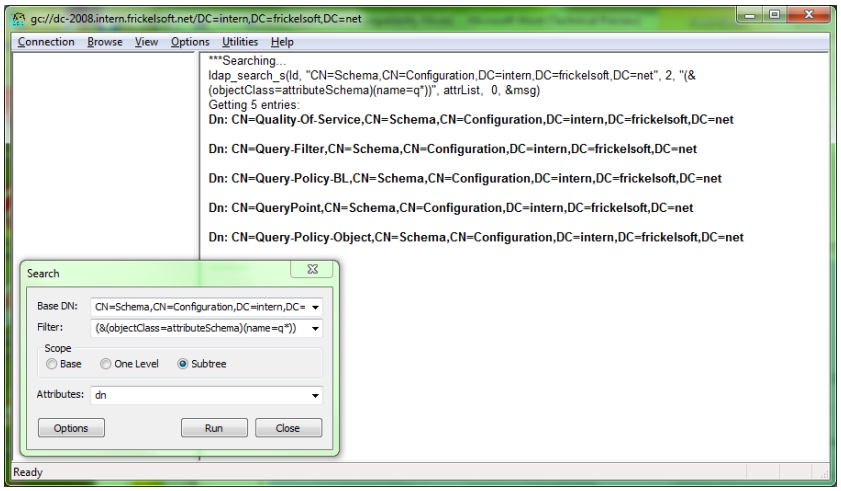


Abbildung 12: Auch LDP lässt den Benutzer in allen Partitionen suchen.

Schlusswort und Ausblick

Nachdem das letzte Kapitel Suchen abschließend die Suche in der Konfigurations- und der Schemapartition beschrieben hat, ist dieses Whitepaper nun zu Ende. Es wurde behandelt, woraus LDAP-Suchen bestehen und wie sie geformt werden können. Ausserdem wurde darauf eingegangen, wie mit verschiedenen Datentypen, etwa Zeitstempeln oder Bitmasken umgegangen werden kann. Die Exportmöglichkeiten von ADFind und CSVDE mit Hilfe der richtigen Parameter rundeten den Überblick ab.

Dieses Whitepaper hat jedoch eine zentrale Frage offen belassen: wie verhalten sich Filter bei Abfragen in großen Objektbereichen? Anwendungsentwickler und Applikationsdesigner defineren ihre Schwerpunkte bei LDAP-Abfragen in anders als Systemadministratoren. Wo Systemadministratoren Abfragen für einen bestimmten Zweck, meist ad hoc abfeuern und das Verzeichnis befragen, erstellen Entwickler Anwendungen, die das Verzeichnis mehrere Male pro Minute, wenn nicht sogar pro Sekunde kontaktieren. In diesen Fällen spielt eine weitere, bisher nicht benannte Komponente eine wichtige Rolle: Effizienz. Die Fragen nach Performance und Abfrageoptimierung und Effizienz sollen in einem weiteren Whitepaper erklärt werden.

Der Autor des Whitepapers ist stets unter <vorname [bei] frickelsoft [Punkt] net> zu erreichen.

Quellen/Referenzen

[1] “Windows Server 2003 Service Pack 1 32-bit Support Tools”, http://www.microsoft.com/downloads/details.aspx?FamilyId=6EC50B78-8BE1-4E81-B3BE-4E7AC4F0912D&displaylang=en

[2] “JoeWare.NET – Free Tools”, http://joeware.net/freetools/index.htm

[3] “All Classes (Windows)”, http://msdn.microsoft.com/en-us/library/ms680938(VS.85).aspx

[4] “User Class (Windows)”, http://msdn.microsoft.com/en-us/library/ms683980(VS.85).aspx

[5] “User-Account-Control Attribute (Windows)”, http://msdn.microsoft.com/en-us/library/ms680832(VS.85).aspx