

*Solutions for Administrators & Developers*

**2nd Edition**



# Active Directory Cookbook™

**O'REILLY®**

*Robbie Allen &  
Laura E. Hunter*

## **Active Directory Cookbook™, Second Edition**

by Robbie Allen and Laura E. Hunter

Copyright © 2006, 2003 O'Reilly Media, Inc. All rights reserved.  
Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*safari.oreilly.com*). For more information, contact our corporate/institutional sales department: (800) 998-9938 or *corporate@oreilly.com*.

**Editor:** Jeff Pepper

**Production Editor:** Laurel R.T. Ruma

**Copyeditor:** Derek Di Matteo

**Proofreaders:** Laurel R.T. Ruma  
and Sanders Kleinfeld

**Indexer:** Ellen Troutman

**Cover Designer:** Karen Montgomery

**Interior Designer:** David Futato

**Illustrators:** Robert Romano and Jessamyn Read

### **Printing History:**

September 2003: First Edition.

June 2006: Second Edition.

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly Media, Inc. *Active Directory Cookbook*, the image of a bluefin tuna, and related trade dress are trademarks of O'Reilly Media, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc. was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher and authors assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

ISBN: 0-596-10202-X

[M]

This excerpt is protected by copyright law. It is your responsibility to obtain permissions necessary for any proposed use of this material. Please direct your inquiries to [permissions@oreilly.com](mailto:permissions@oreilly.com).

# Computers

## 8.0 Introduction

As far as Active Directory is concerned, computers are very similar to users. In fact, computer objects inherit directly from the user object class, which is used to represent user accounts. That means that computer objects possess all of the attributes of user objects and then some. Computers need to be represented in Active Directory for many of the same reasons users do, including the need to access resources securely, utilize GPOs, and have permissions assigned to them.

To participate in a domain, computers need a *secure channel* to a domain controller. A secure channel is an authenticated connection that can transmit encrypted data. To set up the secure channel, a computer must present a password to a domain controller. Similar to the way in which it authenticates a user account, Active Directory will use Kerberos authentication to verify the identity of a computer account. Without the computer object and, by association, the password stored with it that is changed behind the scenes on a regular basis by the operating system, there would be no way for the domain controller to verify a computer is what it claims to be.

### The Anatomy of a Computer

The default location for computer objects in a domain is the `cn=Computers` container located directly off the domain root. You can, however, create computer objects anywhere in a domain. And in Windows Server 2003, you can modify the default location for computer objects as described in Recipe 8.12. Table 8-1 contains a list of some of the interesting attributes that are available on computer objects.

Table 8-1. Attributes of computer objects

Attribute	Description
<code>cn</code>	Relative distinguished name of computer objects.
<code>dnsHostName</code>	Fully qualified DNS name of the computer.

Table 8-1. Attributes of computer objects (continued)

Attribute	Description
lastLogonTimestamp	The approximate timestamp of the last time the computer logged in to the domain. This is a new attribute in Windows Server 2003.
managedBy	The distinguished name (DN) of user or group that manages the computer.
memberOf	List of DNs of the groups the computer is a member of.
operatingSystem	Textual description of the operating system running on the computer. See Recipe 8.13 for more information.
operatingSystemHotFix	Currently not being used, but will hopefully be populated at some point.
operatingSystemServicePack	Service pack version installed on the computer. See Recipe 8.13 for more information.
operatingSystemVersion	Numeric version of the operating system installed on the computer. See Recipe 8.13 for more information.
pwdLastSet	Large integer that can be translated into the last time the computer's password was set. See Recipe 8.9 for more information.
sAMAccountName	NetBIOS-style name of the computer. This is typically the name of the computer with \$ at the end.
userAccountControl	Account flag that defines various account properties. In the case of a computer object, this specifies whether the computer is a member computer or a domain controller.

## 8.1 Creating a Computer

### Problem

You want to create a computer account.

### Solution

#### Using a graphical user interface

1. Open the ADUC snap-in.
2. If you need to change domains, right-click on Active Directory Users and Computers in the left pane, select “Connect to Domain,” enter the domain name, and click OK.
3. In the left pane, browse to the parent container for the computer, right-click on it, and select New → Computer.
4. Enter the name of the computer. If necessary, place a checkmark next to “Assign this computer as a pre-Windows 2000 computer” or “Assign this computer as a backup domain controller.” Click Next to continue.
5. If you will be using this computer account as part of an RIS deployment, place a checkmark next to “This is a managed computer” and enter the GUID that it should use, and then click Next. Otherwise, just click Next to continue.
6. Click Finish.

## Using a command-line interface

You can create a computer object using either the built-in DSAdd utility or AdMod. To create an account using DSAdd, use the following syntax:

```
> dsadd computer "<ComputerDN>" -desc "<Description>"
```

To create a computer account using AdMod, enter the following:

```
> admod -b "<ComputerDN>" objectclass::computer
  sAMAccountName::<ComputerName>$ userAccountControl::4096
  description::"<Description>" -add
```

## Using VBScript

```
' This code creates a computer object.
' ----- SCRIPT CONFIGURATION -----
strBase = "<ParentComputerDN>" ' e.g. cn=Computers,dc=rallencorp,dc=com
strComp = "<ComputerName>"     ' e.g. joe-xp
strDescr = "<Description>"     ' e.g. Joe's Windows XP workstation
' ----- END CONFIGURATION -----

' ADS_USER_FLAG_ENUM
Const ADS_UF_WORKSTATION_TRUST_ACCOUNT = &h1000 ' 4096

set objCont = GetObject("LDAP://" & strBase)
set objComp = objCont.Create("computer", "cn=" & strComp)
objComp.Put "sAMAccountName", strComp & "$"
objComp.Put "description", strDescr
objComp.Put "userAccountControl", ADS_UF_WORKSTATION_TRUST_ACCOUNT
objComp.SetInfo
Wscript.Echo "Computer account for " & strComp & " created"
```

## Discussion

Creating a computer object in Active Directory is not much different from creating a user object. We set the description attribute in the CLI and API solutions, but it is not a mandatory attribute. The only mandatory attribute is sAMAccountName, which should be set to the name of the computer with \$ appended. Also note that these solutions simply create a computer object. This does not mean any user can join a computer to the domain with that computer account. For more information on creating a computer object and allowing a specific user or group to join the computer to the domain, see Recipe 8.2.

## See Also

Recipe 8.2 for creating a computer for a user, MS KB 222525 (Automating the Creation of Computer Accounts), MS KB 283771 (How to Prestage Windows 2000 Computers in Active Directory), MS KB 315273 (Automating the Creation of Computer Accounts), MS KB 320187 (How to Manage Computer Accounts in Active Directory in Windows 2000), and MSDN: ADS\_USER\_FLAG\_ENUM

## 8.2 Creating a Computer for a Specific User or Group

### Problem

You want to create a computer account for a specific user or group to join to the domain. This requires setting permissions on the computer account so that the user or group can modify certain attributes.

### Solution

#### Using a graphical user interface

1. Open the ADUC snap-in.
2. If you need to change domains, right-click on Active Directory Users and Computers in the left pane, select “Connect to Domain,” enter the domain name, and click OK.
3. In the left pane, browse to the parent container for the computer, right-click on it, and select New → Computer.
4. Enter the name of the computer.
5. Under “The following user or group can join this computer to a domain,” click the Change button.
6. Use the Object Picker to select a user or group to join the computer to the domain.
7. Click OK.

#### Using a command-line interface

In the following solution, replace *<ComputerDN>* with the distinguished name of the computer object and *<UserOrGroup>* with the user principal name or NT-style name of a user or group you want to manage the computer:

```
> dsadd computer <ComputerDN>
> dscls <ComputerDN> /G <UserOrGroup>:CALCGRSDDTRC;;
> dscls <ComputerDN> /G <UserOrGroup>:WP;description;
> dscls <ComputerDN> /G <UserOrGroup>:WP;sAMAccountName;
> dscls <ComputerDN> /G <UserOrGroup>:WP;displayName;
> dscls <ComputerDN> /G <UserOrGroup>:WP;"userAccountControl;
> dscls <ComputerDN> /G <UserOrGroup>:WS;"Validated write to service principal\
name";
> dscls <ComputerDN> /G <UserOrGroup>:WS;"Validated write to DNS host name";
```



You can replace the first line of this code with the AdMod code from Recipe 8.1 if you choose.

## Using VBScript

```
' This code creates a computer object and grants a user/group rights over it.
' ----- SCRIPT CONFIGURATION -----
strComputer = "<ComputerName>" ' e.g. joe-xp
strUser      = "<UserOrGroup>"   ' e.g. joe@rallencorp.com or RALLENLORP\joe
strDescr    = "<ComputerDescr>" ' e.g. Joe's workstation
strDomain   = "<ComputerDomain>" ' e.g. rallencorp.com
' ----- END CONFIGURATION -----

#####
' Constants
#####

' ADS_USER_FLAG_ENUM
Const ADS_UF_PASSWD_NOTREQD          = &h0020
Const ADS_UF_WORKSTATION_TRUST_ACCOUNT = &h1000

' ADS_ACETYPE_ENUM
Const ADS_ACETYPE_ACCESS_ALLOWED      = &h0
Const ADS_ACETYPE_ACCESS_ALLOWED_OBJECT = &h5

' ADS_FLAGTYPE_ENUM
Const ADS_FLAG_OBJECT_TYPE_PRESENT = &h1

' ADS_RIGHTS_ENUM
Const ADS_RIGHT_DS_SELF              = &h8
Const ADS_RIGHT_DS_WRITE_PROP        = &h20
Const ADS_RIGHT_DS_CONTROL_ACCESS    = &h100
Const ADS_RIGHT_ACTRL_DS_LIST        = &h4
Const ADS_RIGHT_GENERIC_READ         = &h80000000
Const ADS_RIGHT_DELETE               = &h10000
Const ADS_RIGHT_DS_DELETE_TREE       = &h40
Const ADS_RIGHT_READ_CONTROL         = &h20000

' schemaIDGUID values
Const DISPLAY_NAME                   = "{bf967953-0de6-11d0-a285-00aa003049e2}"
Const SAM_ACCOUNT_NAME               = "{3e0abfd0-126a-11d0-a060-00aa006c33ed}"
Const DESCRIPTION                    = "{bf967950-0de6-11d0-a285-00aa003049e2}"

' controlAccessRight rightsGUID values
Const USER_LOGON_INFORMATION         = "{5f202010-79a5-11d0-9020-00c04fc2d4cf}"
Const USER_ACCOUNT_RESTRICTIONS     = "{4C164200-20C0-11D0-A768-00AA006E0529}"
Const VALIDATED_DNS_HOST_NAME       = "{72E39547-7B18-11D1-ADEF-00C04FD8D5CD}"
Const VALIDATED_SPN                  = "{F3A64788-5306-11D1-A9C5-0000F80367C1}"

#####
' Create Computer
#####

set objRootDSE = GetObject("LDAP://" & strDomain & "/RootDSE")
set objContainer = GetObject("LDAP://cn=Computers," & _
                             objRootDSE.Get("defaultNamingContext"))
set objComputer = objContainer.Create("Computer", "cn=" & strComputer)
```

```

objComputer.Put "sAMAccountName", strComputer & "$"
objComputer.Put "userAccountControl", _
                ADS_UF_PASSWD_NOTREQD Or ADS_UF_WORKSTATION_TRUST_ACCOUNT
objComputer.Put "description", strDescr
objComputer.SetInfo

'#####
' Create ACL
'#####

set objSD = objComputer.Get("ntSecurityDescriptor")
set objDAcl = objSD.DiscretionaryAcl

' Special: Control Rights, List Children
'     Generic Read, Delete,
'     Delete Subtree, Read Permission
set objACE1 = CreateObject("AccessControlEntry")
objACE1.Trustee = strUser
objACE1.AccessMask = ADS_RIGHT_DS_CONTROL_ACCESS Or _
                    ADS_RIGHT_ACTRL_DS_LIST Or _
                    ADS_RIGHT_GENERIC_READ Or _
                    ADS_RIGHT_DELETE Or _
                    ADS_RIGHT_DS_DELETE_TREE Or ADS_RIGHT_READ_CONTROL
objACE1.AceFlags = 0
objACE1.AceType = ADS_ACETYPE_ACCESS_ALLOWED

' Write Property: description
set objACE2 = CreateObject("AccessControlEntry")
objACE2.Trustee = strUser
objACE2.AccessMask = ADS_RIGHT_DS_WRITE_PROP
objACE2.AceFlags = 0
objACE2.Flags = ADS_FLAG_OBJECT_TYPE_PRESENT
objACE2.AceType = ADS_ACETYPE_ACCESS_ALLOWED_OBJECT
objACE2.ObjectType = DESCRIPTION

' Write Property: sAMAccountName
set objACE3 = CreateObject("AccessControlEntry")
objACE3.Trustee = strUser
objACE3.AccessMask = ADS_RIGHT_DS_WRITE_PROP
objACE3.AceFlags = 0
objACE3.Flags = ADS_FLAG_OBJECT_TYPE_PRESENT
objACE3.AceType = ADS_ACETYPE_ACCESS_ALLOWED_OBJECT
objACE3.ObjectType = SAM_ACCOUNT_NAME

' Write Property: displayName
set objACE4 = CreateObject("AccessControlEntry")
objACE4.Trustee = strUser
objACE4.AccessMask = ADS_RIGHT_DS_WRITE_PROP
objACE4.AceFlags = 0
objACE4.Flags = ADS_FLAG_OBJECT_TYPE_PRESENT
objACE4.AceType = ADS_ACETYPE_ACCESS_ALLOWED_OBJECT
objACE4.ObjectType = DISPLAY_NAME

' Write Property: Logon Information

```

```

set objACE5 = CreateObject("AccessControlEntry")
objACE5.Trustee = strUser
objACE5.AccessMask = ADS_RIGHT_DS_WRITE_PROP
objACE5.AceFlags = 0
objACE5.AceType = ADS_ACETYPE_ACCESS_ALLOWED_OBJECT
objACE5.Flags = ADS_FLAG_OBJECT_TYPE_PRESENT
objACE5.ObjectType = USER_LOGON_INFORMATION

' Write Property: Account Restrictions
set objACE6 = CreateObject("AccessControlEntry")
objACE6.Trustee = strUser
objACE6.AccessMask = ADS_RIGHT_DS_WRITE_PROP
objACE6.AceFlags = 0
objACE6.AceType = ADS_ACETYPE_ACCESS_ALLOWED_OBJECT
objACE6.Flags = ADS_FLAG_OBJECT_TYPE_PRESENT
objACE6.ObjectType = USER_ACCOUNT_RESTRICTIONS

' Write Self: Validated SPN
set objACE7 = CreateObject("AccessControlEntry")
objACE7.Trustee = strUser
objACE7.AccessMask = ADS_RIGHT_DS_SELF
objACE7.AceFlags = 0
objACE7.AceType = ADS_ACETYPE_ACCESS_ALLOWED_OBJECT
objACE7.Flags = ADS_FLAG_OBJECT_TYPE_PRESENT
objACE7.ObjectType = VALIDATED_SPN

' Write Self: Validated DNS Host Name
set objACE8 = CreateObject("AccessControlEntry")
objACE8.Trustee = strUser
objACE8.AccessMask = ADS_RIGHT_DS_SELF
objACE8.AceFlags = 0
objACE8.AceType = ADS_ACETYPE_ACCESS_ALLOWED_OBJECT
objACE8.Flags = ADS_FLAG_OBJECT_TYPE_PRESENT
objACE8.ObjectType = VALIDATED_DNS_HOST_NAME

objDAcl.AddAce objACE1
objDAcl.AddAce objACE2
objDAcl.AddAce objACE3
objDAcl.AddAce objACE4
objDAcl.AddAce objACE5
objDAcl.AddAce objACE6
objDAcl.AddAce objACE7
objDAcl.AddAce objACE8

' #####
' Set ACL
' #####
objSD.DiscretionaryAcl = objDAcl
objComputer.Put "ntSecurityDescriptor", objSD
objComputer.SetInfo
WScript.Echo "Successfully created " & strComputer & _
" and gave rights to " & strUser

```

## Discussion

By default, members of the Authenticated Users group can join up to 10 computers to an Active Directory domain. If you've modified this default behavior or need to allow a user to add computers to the domain on a regular basis, you need to grant certain permissions so that the user has rights to modify the computer object. When you create a computer via the ADUC snap-in, you have the option to select a user or group to manage the computer object and join a computer to the domain using that object. When you use that method, eight ACEs are added to the ACL of the computer object. They are:

- List Contents, Read All Properties, Delete, Delete Subtree, Read Permissions, All Extended Rights (i.e., Allowed to Authenticate, Change Password, Send As, Receive As, Reset Password)
- Write Property for description
- Write Property for sAMAccountName
- Write Property for displayName
- Write Property for Logon Information
- Write Property for Account Restrictions
- Validate write to DNS hostname
- Validate write for service principal name

### Using a graphical user interface

If you want to modify the default permissions that are applied when you select a user or group through the GUI, double-click on the computer object after you've created it and go to the Security tab. For the Security tab to be visible, you have to select View → Advanced Features.

### Using a command-line interface

With the *dsacls* utility, you can specify either a UPN (*user@domain*) or down-level style (*DOMAIN\user*) account name when applying permissions. Also, *dsacls* requires that the *displayName* of the attribute, property set, or extended right you are setting the permission on be used instead of the *LDAPDisplayName*, as one might expect. That is why we had to use “Validated write to service principal name,” which is the *displayName* for the *Validated-SPN controlAccessRight* object, with the ACE for the *SPN-validated write*. *dsacls* is also case sensitive, so be sure to specify the correct case for the words in the *displayName*.

### Using VBScript

After creating the computer object, similar to Recipe 8.1, create an ACE object for each of the eight ACEs previously listed using the *IADsAccessControlEntry* interface.

To apply the ACEs, retrieve the current security descriptor for the computer object, which is stored in the `ntSecurityDescriptor` attribute, and then add the eight ACEs. Finally, call `SetInfo` to commit the change to Active Directory. For more information on setting ACEs and ACLs programmatically, see the `IADsAccessControlEntry` documentation in MSDN.

## See Also

Recipe 8.1 for creating a computer account, MS KB 238793 (Enhanced Security Joining or Resetting Machine Account in Windows 2000 Domain), MS KB 283771 (How to Prestage Windows 2000 Computers in Active Directory), MS KB 320187 (How to Manage Computer Accounts in Active Directory in Windows 2000), MSDN: `IADsAccessControlEntry`, MSDN: `ADS_ACETYPE_ENUM`, and MSDN: `ADS_RIGHTS_ENUM`, MSDN: `ADS_FLAGTYPE_ENUM`

## 8.3 Joining a Computer to a Domain

### Problem

You want to join a computer to a domain after the computer object has already been created in Active Directory.

### Solution

#### Using a graphical user interface

1. Log on to the computer you want to join to the domain and open the Control Panel.
2. Open the System applet.
3. Click the Computer Name tab.
4. Click the Change button.
5. Under “Member of,” select Domain.
6. Enter the domain you want to join and click OK.
7. You may be prompted to enter credentials that grant permission to join the computer.
8. Reboot the computer.

Note that the tabs in the System applet vary between Windows 2000, Windows XP, and Windows Server 2003.

#### Using a command-line interface

```
> netdom join <ComputerName> /Domain <DomainName> /UserD <DomainUserUPN>  
/PasswordD * /User0 <ComputerAdminUser> /Password0 * /Reboot
```

## Using VBScript

```
' This code joins a computer to a domain.
' ----- SCRIPT CONFIGURATION -----
strComputer      = "<ComputerName>"      ' e.g. joe-xp
strDomain         = "<DomainName>"        ' e.g. rallencorp.com
strDomainUser    = "<DomainUserUPN>"     ' e.g. administrator@rallencorp.com
strDomainPasswd  = "<DomainUserPasswd>"
strLocalUser     = "<ComputerAdminUser>" ' e.g. administrator
strLocalPasswd   = "<ComputerUserPasswd>"
' ----- END CONFIGURATION -----

#####
' Constants
#####
Const JOIN_DOMAIN      = 1
Const ACCT_CREATE      = 2
Const ACCT_DELETE      = 4
Const WIN9X_UPGRADE    = 16
Const DOMAIN_JOIN_IF_JOINED = 32
Const JOIN_UNSECURE    = 64
Const MACHINE_PASSWORD_PASSED = 128
Const DEFERRED_SPN_SET = 256
Const INSTALL_INVOCATION = 262144

#####
' Connect to Computer
#####
set objWMILocator = CreateObject("WbemScripting.SWbemLocator")
objWMILocator.Security_.AuthenticationLevel = 6
set objWMIComputer = objWMILocator.ConnectServer(strComputer, _
                                                "root\cimv2", _
                                                strLocalUser, _
                                                strLocalPasswd)

set objWMIComputerSystem = objWMIComputer.Get(_
                            "Win32_ComputerSystem.Name='" & _
                            strComputer & "'")

#####
' Join Computer
#####
rc = objWMIComputerSystem.JoinDomainOrWorkGroup(strDomain, _
                                                strDomainPasswd, _
                                                strDomainUser, _
                                                vbNullString, _
                                                JOIN_DOMAIN)

if rc <> 0 then
    WScript.Echo "Join failed with error: " & rc
else
    WScript.Echo "Successfully joined " & strComputer & " to " & strDomain
end if
```

## Discussion

When trying to add a computer to Active Directory, you can either precreate the computer object as described in Recipes 8.1 and 8.2 before joining it to the domain, or you can perform both operations at the same time.

### Using a graphical user interface

If you have the correct permissions in Active Directory, you can actually create a computer object at the same time as you join it to a domain via the instructions described in the GUI solution. Since the System applet doesn't allow you to specify an OU for the computer object, if it needs to create a computer object, it will do so in the default Computers container. See Recipe 8.15 for more information on the default computers container and how to change it.

### Using a command-line interface

The `netdom` command will attempt to create a computer object for the computer during a join if one does not already exist. An optional `/OU` switch can be added to specify the OU in which to create the computer object. To do so, you'll need to have the necessary permissions to create and manage computer objects in the OU.

There are some restrictions on running the `netdom join` command remotely. If a Windows XP machine has the ForceGuest security policy setting enabled, you cannot join it remotely. Running the `netdom` command directly on the machine works regardless of the ForceGuest setting.

### Using VBScript

In order for the `Win32_ComputerSystem::JoinDomainOrWorkGroup` method to work remotely, you have to use an `AuthenticationLevel` equal to 6 so that the traffic between the two machines (namely the passwords) is encrypted. You can also create computer objects using `JoinDomainOrWorkGroup` by using the `ACCT_CREATE` flag in combination with `JOIN_DOMAIN`.



This function works only with Windows XP and Windows Server 2003 and is not available for Windows 2000 and earlier machines.

Just like with the `netdom` utility, you cannot run this script against a remote computer if that computer has the ForceGuest setting enabled.

## See Also

More information on the ForceGuest setting can be found here: [http://www.microsoft.com/resources/documentation/Windows/XP/all/reskit/en-us/prde\\_ffs\\_ypuh.asp](http://www.microsoft.com/resources/documentation/Windows/XP/all/reskit/en-us/prde_ffs_ypuh.asp), MS KB 238793 (Enhanced Security Joining or Resetting Machine Account in Windows 2000)

Domain), MS KB 251335 (Domain Users Cannot Join Workstation or Server to a Domain), MS KB 290403 (How to Set Security in Windows XP Professional That Is Installed in a Workgroup), MSDN: Win32\_ComputerSystem::JoinDomainOrWorkgroup, and MSDN: NetJoinDomain

## 8.4 Moving a Computer Within the Same Domain

### Problem

You want to move a computer object to a different container or OU within the same domain.

### Solution

#### Using a graphical user interface

1. Open the ADUC snap-in.
2. If you need to change domains, right-click on Active Directory Users and Computers in the left pane, select “Connect to Domain,” enter the domain name, and click OK.
3. In the left pane, right-click on the domain and select Find.
4. Beside Find, select Computers.
5. Type the name of the computer and click Find Now.
6. In the Search Results window, right-click on the computer and select Move.
7. Browse to and select the new parent container or OU.
8. Click OK.



With the Windows Server 2003 version of Active Directory Users and Computers, you can also use the new drag-and-drop functionality to move computers and other objects.

#### Using a command-line interface

You can move a computer object to a new container using the built-in DSMove utility or AdMod. To use DSMove, enter the following syntax:

```
> dsmove "<ComputerDN>" -newparent "<NewParentDN>"
```

To move a computer object using AdMod, use the following:

```
> admod -b "<ComputerDN>" -move "<NewParentDN>"
```

## Using VBScript

```
' This code moves a computer to the specified container/OU.  
' ----- SCRIPT CONFIGURATION -----  
strCompDN = "<ComputerDN>" ' e.g. cn=joe-xp,cn=Users,dc=rallencorp,dc=com  
strOUDN = "<NewParentDN>" ' e.g. ou=workstations,dc=rallencorp,dc=com  
' ----- END CONFIGURATION -----  
  
set objComp = GetObject("LDAP://" & strCompDN)  
set objOU = GetObject("LDAP://" & strOUDN)  
objOU.MoveHere objComp.ADsPath, objComp.Name
```

## Discussion

You can move computer objects around a domain without much impact on the computer itself. You just need to be cautious of the security settings on the new parent OU, which may impact a user's ability to manage the computer object in Active Directory. Also, if GPOs are used differently on the new parent, it could impact booting and logon times, and how the computer's operating system behaves after a user has logged on.

## See Also

Recipe 4.20 for moving an object to a different OU, and Recipe 8.5 for moving a computer to a different domain

# 8.5 Moving a Computer to a New Domain

## Problem

You want to move a computer object to a different domain.

## Solution

### Using a graphical user interface

To migrate a computer object between domains in the same forest, use the following steps:

1. Open the ADMT MMC snap-in.
2. Right-click on the Active Directory Migration Tool folder and select the Computer Account Migration Wizard.
3. On the Domain Selection page, enter the DNS or NetBIOS name of the source and target domains. Click Next.
4. On the Translate Objects screen, specify which objects should have new ACLs applied in the new domain. Select any, none, or all of the following, and then click Next to continue:

- Files and folders
  - Local groups
  - Printers
  - Registry
  - Shares
  - User profiles
  - User rights
5. On the Security Migration Options screen, select the following options to determine how local user accounts will be migrated into the new domain. Select one of the following and click Next to continue:

*Replace*

This option will replace any references to objects from the source domain with references to objects in the target domain.

*Add*

This option adds references to objects in the target domain while leaving the source domain objects intact.

*Remove*

This option removes all references to source domain objects.

6. On the Naming Conflicts page, configure how the wizard should handle naming conflicts during the migration process. Select one of the following and click Next to continue:
- Ignore conflicting accounts and don't migrate.
  - Replace conflicting accounts.
  - Rename conflicting accounts by adding a designated prefix or suffix.
7. On the Options screen, select the amount of time the wizard should wait before rebooting the target computer into the new domain.
8. Click Next to review your choices and begin the migration process.

### Using a command-line interface

The following command will migrate a computer object from the *rallencorp.com* domain to the *emea.rallencorp.com* domain. It will place the migrated object in the Finance OU and will wait two minutes before rebooting the target computer:

```
ADMT COMPUTER /N "FIN101-A" "FIN101-A" /SD:"emea.rallencorp.com"  
/TD:"emea.rallencorp.com" /TO:"Finance" /RDL:2
```

## Using VBScript

```
set objObject = GetObject("LDAP://TargetDC/TargetParentDN")
objObject.MoveHere "LDAP://SourceDC/SourceDN", vbNullString
```

## Discussion

You can move objects between domains assuming you follow a few guidelines:

- The user requesting the move must have permission to modify objects in the parent container of both domains.
- You need to explicitly specify the target DC (serverless binds usually do not work). This is necessary because the Cross Domain Move LDAP control is being used behind the scenes. (For more information on controls, see Recipe 4.4.)
- The move operation must be performed against the RID master for both domains. This is done to ensure that two objects that are being moved simultaneously don't somehow get assigned the same RID.
- Both domains must be in native mode.

## See Also

Recipe 4.4 for more on LDAP controls, MSDN: IADsContainer::MoveHere, and MS KB 326480 (How to Use Active Directory Migration Tool version 2 to migrate from Windows 2000 to Windows Server 2003)

# 8.6 Renaming a Computer

## Problem

You want to rename a computer.

## Solution

### Using a graphical user interface

1. Log on to the computer either directly or with a remote console application such as Terminal Services.
2. Open the Control Panel and double-click on the System applet.
3. Select the Computer Name tab and click the Change button.
4. Under Computer Name, type the new name of the computer and click OK until you are out of the System applet.
5. Reboot the machine.

## Using a command-line interface

You can rename a computer object by using the built-in *netdom* utility with the following syntax:

```
> netdom renamecomputer <ComputerName> /NewName <NewComputerName> /UserD
<DomainUserUPN> /PasswordD * /UserO <ComputerAdminUser> /PasswordO * /Reboot
```

## Using VBScript

```
' This code renames a computer in AD and on the host itself.
' ----- SCRIPT CONFIGURATION -----
strComputer      = "<ComputerName>"      ' e.g. joe-xp
strNewComputer   = "<NewComputerName>"   ' e.g. joe-pc
strDomainUser    = "<DomainUserUPN>"     ' e.g. administrator@rallencorp.com
strDomainPasswd  = "<DomainUserPasswd>"
strLocalUser     = "<ComputerAdminUser>" ' e.g. joe-xp\administrator
strLocalPasswd   = "<ComputerAdminPasswd>"
' ----- END CONFIGURATION -----

#####
' Connect to Computer
#####
set objWMILocator = CreateObject("WbemScripting.SWbemLocator")
objWMILocator.Security_.AuthenticationLevel = 6
set objWMIComputer = objWMILocator.ConnectServer(strComputer, _
                                                    "root\cimv2", _
                                                    strLocalUser, _
                                                    strLocalPasswd)

set objWMIComputerSystem = objWMIComputer.Get( _
                                                    "Win32_ComputerSystem.Name='" & _
                                                    strComputer & "'")

#####
' Rename Computer
#####
rc = objWMIComputerSystem.Rename(strNewComputer, _
                                  strDomainPasswd, _
                                  strDomainUser)

if rc <> 0 then
    WScript.Echo "Rename failed with error: " & rc
else
    WScript.Echo "Successfully renamed " & strComputer & " to " & _
                strNewComputer
end if

WScript.Echo "Rebooting . . ."
set objWShell = WScript.CreateObject("WScript.Shell")
objWShell.Run "rundll32 shell32.dll,SHExitWindowsEx 2"
```

## Discussion

Renaming a computer consists of two operations: renaming the computer object in Active Directory and renaming the hostname on the machine itself. To do it in one step, which each of the three solutions offers, you must have permission in Active Directory to rename the account and administrator permissions on the target machine. For the rename operation to be complete, you must reboot the computer.



In some cases, renaming a computer can adversely affect services running on the computer. For example, you cannot rename a machine that is a domain controller, Exchange Server, or a Windows Certificate Authority without taking additional steps and precautions.

### Using a graphical user interface

After you rename the computer, you will be prompted to reboot the machine. You can cancel if necessary, but you'll need to reboot at some point to complete the rename operation.

### Using a command-line interface

The `renamecomputer` option in `netdom` is new to Windows Server 2003. It can run remotely and includes a `/Reboot` switch that allows you to automatically reboot the computer after the rename is complete.

### Using VBScript

The `Win32_ComputerSystem::Rename` method must be run on the local machine unless the computer is a member of a domain. Unlike the GUI and CLI solutions, you cannot specify alternate credentials for the connection to the computer other than domain credentials. For this reason, the user and password you use with the `Rename` method must have administrative privileges on the target machine (i.e., part of the Administrators group) and on the computer object in Active Directory.



The `Rename` method is new in Windows XP and Windows Server 2003, and is not available on Windows 2000 and earlier machines.

## See Also

Recipe 4.23 for renaming objects, MS KB 228544 (Changing Computer Name in Windows 2000 Requires Restart), MS KB 238793 (Enhanced Security Joining or Resetting Machine Account in Windows 2000 Domain), MS KB 260575 (How to Use `Netdom.exe` to Reset Machine Account Passwords of a Windows 2000 Domain Controller), MS KB 325354 (How to Use the `Netdom.exe` Utility to Rename a Computer in Windows Server 2003), and MSDN: `Win32_ComputerSystem::Rename`

## 8.7 Add or Remove a Computer Account from a Group

### Problem

You want to add or remove a computer account from an Active Directory security group.

### Solution

#### Using a graphical user interface

1. Open the ADUC snap-in.
2. If you need to change domains, right-click on “Active Directory Users and Computers” in the left pane, select “Connect to Domain,” enter the domain name, and click OK.
3. In the left pane, browse to the parent container of the objects you want to modify.
4. In the right pane, highlight each object you want to modify, right-click, and select Properties.
5. On the Member of tab, click Add.
6. Click the group to which you want to add the computer, and then click Add. To add the computer to more than one group, press Ctrl while selecting the groups you want to add the computer to, and then click Add.
7. To remove a group, select the group object and click Remove.
8. Click OK to finish.

#### Using a command-line interface

To add a computer object to a group, use the following syntax:

```
> admod -b "<GroupDN>" member:+:"<ComputerDN>"
```



To remove an object, replace `+:` with `:-` in the previous syntax.

#### Using VBScript

```
' This code adds and removes a computer object from a group.
' ----- SCRIPT CONFIGURATION -----
strGroupDN = "<GroupDN>" ' e.g. cn=SalesGroup,ou=Groups,dc=rallencorp,dc=com
strComputerDN = "<ComputerDN>" ' e.g. cn=Fin101,cn=Computers,dc=rallencorp,dc=com
' ----- END CONFIGURATION -----
```

```
set objGroup = GetObject("LDAP://" & strGroupDN)
' Add a member
objGroup.Add("LDAP://" & strComputerDN)

' Remove a member
objGroup.Remove("LDAP://" & strComputerDN)
```

## Discussion

In Active Directory, both user and computer objects are security principals that can be assigned rights and permissions within a domain. As such, computer objects can be added to or removed from group objects to make for simpler resource administration. You can make this change through ADUC or ADSI Edit, or by manually editing the member attribute of the appropriate group object.

## See Also

MSDN: NT-Group-Members attribute [AD Schema] and MSDN: Member Attribute [AD Schema]

# 8.8 Testing the Secure Channel for a Computer

## Problem

You want to test the secure channel of a computer.

## Solution

### Using a command-line interface

```
> nltest /server:<ComputerName> /sc_query:<DomainName>
```

## Discussion

Every member computer in an Active Directory domain establishes a secure channel with a domain controller. The computer's password is stored locally in the form of an LSA secret and in Active Directory. This password is used by the NetLogon service to establish the secure channel with a domain controller. If for some reason the LSA secret and computer password become out of sync, the computer will no longer be able to authenticate in the domain. The `nltest /sc_query` command can query a computer to verify its secure channel is working. Here is sample output from the command when things are working:

```
Flags: 30 HAS_IP HAS_TIMESERV
Trusted DC Name \\dc1.rallencorp.com
Trusted DC Connection Status Status = 0 0x0 NERR_Success
The command completed successfully
```

If a secure channel is failing, you'll need to reset the computer as described in Recipe 8.9. Here is sample output when things are not working:

```
Flags: 0
Trusted DC Name
Trusted DC Connection Status Status = 1311 0x51f ERROR_NO_LOGON_SERVERS
The command completed successfully
```

## See Also

Recipe 8.9 for resetting a computer and MS KB 216393 (Resetting Computer Accounts in Windows 2000 and Windows XP)

# 8.9 Resetting a Computer Account

## Problem

You want to reset a computer because its secure channel is failing.

## Solution

### Using a graphical user interface

1. Open the ADUC snap-in.
2. If you need to change domains, right-click on Active Directory Users and Computers in the left pane, select "Connect to Domain," enter the domain name, and click OK.
3. In the left pane, right-click on the domain and select Find.
4. Beside Find, select Computers.
5. Type the name of the computer and click Find Now.
6. In the Search Results, right-click on the computer and select Reset Account.
7. Click Yes to verify.
8. Click OK.
9. Rejoin the computer to the domain.

### Using a command-line interface

You can use the DSMod utility to reset a computer's password. You will need to rejoin the computer to the domain after doing this.

```
> dsmod computer "<ComputerDN>" -reset
```

Another option is to use the netdom command, which can reset the secure channel between the computer and the domain controller without affecting the computer's password, so that you do not need to rejoin it to the domain:

```
> netdom reset <ComputerName> /Domain <DomainName> /User0 <UserUPN> /Password0 *
```

You can also use the `nltest` command to reset a secure channel using the following syntax:

```
> nltest /sc_reset:<DomainName>\<DCName>
```

### Using VBScript

```
' This resets an existing computer object's password to initial default.  
' You'll need to rejoin the computer after doing this.  
set objComputer = GetObject("LDAP://<ComputerDN>")  
objComputer.SetPassword "<ComputerName>"
```

## Discussion

When you've identified that a computer's secure channel has failed, you'll need to reset the computer object, which consists of setting the computer object password to the name of the computer. This is the default initial password for new computers. Every 30 days, Windows 2000 and newer systems automatically change their passwords in the domain. After you've set the password, you'll need to rejoin the computer to the domain since it will no longer be able to communicate with a domain controller due to unsynchronized passwords. However, the `netdom reset` command will try to reset the password on both the computer and in Active Directory, which will not necessitate rejoining it to the domain if successful.

From a practical standpoint, you should first attempt to reset the secure channel between the computer and the domain using the `netdom` or `nltest` syntaxes, since doing so will not require you to unjoin and rejoin the computer to the domain; in particular, this will save you from performing the associated reboots involved with rejoining the domain. If resetting the secure channel does not correct the issue you're facing, you can then resort to resetting the computer's password.

## See Also

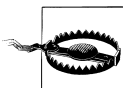
Recipe 8.3 for joining a computer to a domain, Recipe 8.8 for testing a secure channel, MS KB 216393 (Resetting Computer Accounts in Windows 2000 and Windows XP), and MS KB 325850 (How to Use Netdom.exe to Reset Machine Account Passwords of a Windows Server 2003 Domain Controller)

## 8.10 Finding Inactive or Unused Computers

### Problem

You want to find inactive computer accounts in a domain.

## Solution



These solutions might only apply to Windows-based machines. Other types of machines—e.g., Unix, Mac, Network Attached Storage (NAS)—that have accounts in Active Directory might not update their login timestamps or passwords, which are used to determine inactivity.

### Using a command-line interface

The following query will locate all inactive computers in the current forest:

```
> dsquery computer forestroot -inactive <NumWeeks>
```

You can also use `domainroot` in combination with the `-d` option to query a specific domain:

```
> dsquery computer domainroot -d <DomainName> -inactive <NumWeeks>
```

or you can target your query at a specific container:

```
> dsquery computer ou=MyComputers,dc=rallencorp,dc=com -inactive <NumWeeks>
```



These commands can only be run against a Windows Server 2003 domain functional level or higher domain.

You can also use the `OldCmp` joeware utility to create a report of all computer accounts whose passwords are older than a certain number of days (90 by default) by using the following syntax:

```
> oldcmp -report
```



To specify an alternate password age with `oldcmp`, use the `-age x` switch. You can also use the `-llts` switch to use the `lastLogonTimeStamp` attribute to perform the age calculations. (Without this switch, `oldcmp` will use `pwdLastSet` by default.)

### Using Perl

```
#!/perl

#-----
# Script Configuration
#-----
# Domain and container/OU to check for inactive computer accounts
my $domain      = 'amer.rallencorp.com';

# set to empty string to query entire domain
my $computer_cont = 'cn=Computers,';
```

```

# Number of weeks used to find inactive computers
my $weeks_ago = 30;
#-----
# End Configuration
#-----

use strict;
use Win32::OLE;
    $Win32::OLE::Warn = 3;
use Math::BigInt;

# Must convert the number of seconds since $weeks_ago
# to a large integer for comparison against lastLogonTimestamp
my $sixmonth_secs = time - 60*60*24*7*$weeks_ago;
my $intObj = Math::BigInt->new($sixmonth_secs);
    $intObj = Math::BigInt->new($intObj->bmul('10 000 000'));
my $sixmonth_int = Math::BigInt->new(
    $intObj->badd('116 444 736 000 000 000'));
    $sixmonth_int =~ s/^[+]?//;

# Set up the ADO connections
my $connObj = Win32::OLE->new('ADODB.Connection');
$connObj->{Provider} = "AdsDSOObject";
$connObj->Open;
my $commObj = Win32::OLE->new('ADODB.Command');
$commObj->{ActiveConnection} = $connObj;
$commObj->Properties->{'Page Size'} = 1000;

# Grab the default root domain name
my $rootDSE = Win32::OLE->GetObject("LDAP://$domain/RootDSE");
my $rootNC = $rootDSE->Get("defaultNamingContext");

# Run ADO query and print results
my $query = "<LDAP://$domain/$computer_cont$rootNC>";
$query .= "(&(objectclass=computer)";
$query .= "(objectcategory=computer)";
$query .= "(lastlogontimestamp<=$sixmonth_int)";
$query .= "cn,distinguishedName";
$query .= "subtree";
$commObj->{CommandText} = $query;
my $resObj = $commObj->Execute($query);
die "Could not query $domain: ", $Win32::OLE::LastError, "\n"
    unless ref $resObj;

print "\nComputers that have been inactive for $weeks_ago weeks or more:\n";
my $total = 0;
while (!$resObj->EOF) {
    my $cn = $resObj->Fields(0)->value;
    print "\t", $resObj->Fields("distinguishedName")->value, "\n";
    $total++;
    $resObj->MoveNext;
}
print "Total: $total\n";

```

## Discussion

### Using a command-line interface

The `dsquery computer` command is very handy for finding inactive computers that have not logged in to the domain for a number of weeks or months. You can pipe the results of the query to the `dsrcm` command-line utility if you want to remove the inactive computer objects from Active Directory in a single command. Here is an example that would delete all computers in the current domain that have been inactive for 12 weeks or longer:

```
> for /F "usebackq" %i in (`dsquery computer domainroot -inactive 12`) do dsrcm %i
```

You can also use `OldCmp` to disable inactive accounts, and then either delete them or move them to an alternate OU. `OldCmp` has a number of safeties built into the utility to prevent you from deleting a large number of computer accounts without meaning to. For example, `OldCmp` will not delete an account unless it has first been disabled, it will not modify more than 10 objects at a time unless you manually specify a higher limit, and it simply will not do anything at all to a domain controller computer account under any circumstances. Unless you have a requirement for quickly removing unused computer objects, we'd recommend allowing them to remain inactive for at least three months before removing them. If you don't really care when the objects get removed, use a year (i.e., 52 weeks) to be on the safe side.

### Using Perl

With Windows 2000 Active Directory, the only way you can determine if a computer is inactive is to query either the `pwdLastSet` or `lastLogon` attributes. The `pwdLastSet` attribute is a 64-bit integer that translates into the date and time the computer last updated its password. Since computers are supposed to change their password every 30 days, you could run a query that finds the computers that have not changed their password in several months. This is difficult with VBScript because it does not handle 64-bit integer manipulation very well. There are third-party add-ons that provide 64-bit functions, but none of the built-in VBScript functions can do it and it is nontrivial to implement without an add-on.

The `lastLogon` attribute can also be used to find inactive computers because that attribute contains a 64-bit integer representing the last time the computer logged into the domain. The problem with the `lastLogon` attribute is that it is *not replicated*. Since it is not replicated, you have to query every domain controller in the domain to find the most recent `lastLogon` value. As you can imagine, this is less than ideal, especially if you have a lot of domain controllers.

Fortunately, in Windows Server 2003, Microsoft added a new attribute called `lastLogonTimestamp` to user and computer objects. This attribute contains the approximate last logon timestamp (again in a 64-bit, large-integer format) for the user or computer and is replicated to all domain controllers. It is the approximate last logon

because the domain controllers will update the value only if it hasn't been updated for a certain period of time (such as a week). This prevents the attribute from being updated constantly and causing a lot of unnecessary replication traffic.

Since VBScript was out of the question, we turned to our first love—Perl. It is very rare to find a problem that you can't solve with Perl and this is no exception. The biggest issue is manipulating a number to a 64-bit integer, which we can do with the `Math::BigInt` module.

First, we determine the time in seconds from 1970 for the date that we want to query computer inactivity against. That is, take the current time and subtract the number of weeks we want to go back. Then we have to convert that number to a big integer. The last step is simply to perform an ADO query for all computers that have a `lastLogonTimestamp` less than or equal to the value we just calculated.

## See Also

Recipe 6.31 for finding users whose accounts are about to expire

# 8.11 Changing the Maximum Number of Computers a User Can Join to the Domain

## Problem

You want to grant users the ability to join more or fewer than 10 computers to a domain. This limit is called the *machine account quota*.

## Solution

### Using a graphical user interface

1. Open the ADSI Edit MMC snap-in and connect to the Domain Naming Context.
2. Right-click on the `domainDNS` object for the domain you want to change and select Properties.
3. Edit the `ms-DS-MachineAccountQuota` attribute and enter the new quota value.
4. Click OK twice.

### Using a command-line interface

In the following LDIF code replace `<DomainDN>` with the distinguished name of the domain you want to change and replace `<Quota>` with the new machine account quota:

```
dn: <DomainDN>
changetype: modify
replace: ms-DS-MachineAccountQuota
ms-DS-MachineAccountQuota: <Quota>
-
```

If the LDIF file was named *change\_computer\_quota.ldf*, you would then run the following command:

```
> ldifde -v -i -f change_computer_quota.ldf
```

You can also make this change using AdMod, as follows:

```
> admod -b <DomainDN> ms-DS-MachineAccountQuota::<Quota>
```

## Using VBScript

```
' This code sets the machine account quota for a domain.
' ----- SCRIPT CONFIGURATION -----
intQuota = <Quota>
strDomain = "<DomainDNSName>" ' e.g. emea.rallencorp.com
' ----- END CONFIGURATION -----

set objRootDSE = GetObject("LDAP://" & strDomain & "/RootDSE")
set objDomain = GetObject("LDAP://" & objRootDSE.Get("defaultNamingContext"))
objDomain.Put "ms-DS-MachineAccountQuota", intQuota
objDomain.SetInfo
WScript.Echo "Updated user quota to " & intQuota
```

## Discussion

In a default Active Directory installation, members of the Authenticated Users group can add and join up to 10 computer accounts in the default Computers container. The number of computer accounts that can be created is defined in the ms-DS-MachineAccountQuota attribute on the domainDNS object for a domain. The default setting is artificially set to 10, but you can easily change that to whatever number you want, including 0, via the methods described in the Solution section. If you set it to 0, users have to be granted explicit permissions in Active Directory to join computers; refer to Recipe 8.3 for instructions on granting these permissions.

Another method for granting users the right to add computer objects, although not recommended, is via Group Policy. If you grant the “Add workstation to domain” right via Computer Configuration → Windows Settings → Security Settings → Local Policies → User Rights Assignment on a GPO that’s been linked to the Domain Controllers OU, then users will be able to create computer accounts even if they do not have create child permissions on the default Computers container. This is a holdover from Windows NT to maintain backward compatibility and should not be used unless absolutely necessary. In fact, a good security best practice would be to *remove* this user right from any user or group objects that do not require it.

## See Also

Recipe 8.3 for permissions needed to join computers to a domain, MS KB 251335 (Domain Users Cannot Join Workstation or Server to a Domain), and MS KB 314462 (“You Have Exceeded the Maximum Number of Computer Accounts” Error Message When You Try to Join a Windows XP Computer to a Windows 2000 Domain)

## 8.12 Modifying the Attributes of a Computer Object

### Problem

You want to modify one or more attributes of a computer object.

### Solution

#### Using a graphical user interface

1. Open ADSI Edit.
2. If an entry for the naming context you want to browse is not already displayed, do the following:
  - a. Right-click on ADSI Edit in the right pane and click “Connect to...”
  - b. Fill in the information for the naming context, container, or OU you want to add an object to. Click on the Advanced button if you need to enter alternate credentials.
  - c. In the left pane, browse to the container or OU that contains the computer object you want to modify. Once you’ve found the object, right-click on it and select Properties.
3. Right-click the attribute you want to modify and select Edit.
4. Enter the new value and click OK.
5. Click Apply, followed by OK.

#### Using a command-line interface

Create an LDIF file called *modify\_object.ldf* with the following contents:

```
dn: <ComputerDN>
changetype: modify
add: <AttributeName>
<AttributeName>: <AttributeValue>
-
```

Then run the following command:

```
> ldifde -v -i -f modify_object.ldf
```

To modify an object using AdMod, you’ll use the following general syntax:

```
> admod -b <ComputerDN> <attribute>:<operation>:<value>
```

For example, you can add a location to a computer object using the following syntax:

```
> admod -b cn="Fin101,cn=Computers,dc=rallencorp,dc=com"
location: "Berlin, Germany"
```

#### Using VBScript

```
' The following code will modify the location attribute
' of a computer object.
```

```
Set objComputer = GetObject ("LDAP://<ComputerDN>")  
  
objComputer.Put "Location" , "<NewLocationValue>"  
objComputer.SetInfo
```

## Discussion

Like all objects within Active Directory, computer objects have various attributes that can be queried, modified, and deleted during the day-to-day management of your domain. Because computer objects inherit from the user class, they include similar informational attributes to the user objects, as well as attributes that are specific to computer objects, including:

- Location
- Description
- operatingSystemVersion
- operatingSystemServicePack
- sAMAccountName
- pwdLastSet
- primaryGroupID

## See Also

Recipe 8.10 for finding inactive or unused computers, Recipe 8.13 for finding computers with a particular OS, and MSDN: Computer System Hardware Classes [WMI]

## 8.13 Finding Computers with a Particular OS

### Problem

You want to find computers that have a certain OS version, release, or service pack in a domain.

### Solution

#### Using a graphical user interface

1. Open LDP.
2. From the menu, select Connection → Connect.
3. For Server, enter the name of a domain controller (or leave blank to do a serverless bind).
4. For Port, enter 389.
5. Click OK.
6. From the menu, select Connection → Bind.

7. Enter credentials of a user to perform the search.
8. Click OK.
9. From the Menu, select Browse → Search.
10. For Base DN, enter the base of where you want your search to begin.
11. For Filter, enter a filter that contains the OS attribute you want to search on. For example, a query for all computers that are running Windows XP would be the following:
 

```
(&(objectclass=computer)(objectcategory=computer)(operatingSystem=Windows XP Professional))
```
12. Select the appropriate Scope based on how deep you want to search.
13. Click the Options button if you want to customize the list of attributes returned for each matching object.
14. Click Run, and the results will be displayed in the right pane.

You can also perform this search using the Active Directory Users and Computers MMC snap-in (*dsa.msc*), as follows:

1. Open the ADUC MMC snap-in.
2. Right-click on the domain, OU, or container that you wish to search, and click Find.
3. In the Find drop-down box, select Computers.
4. Click on the Advanced tab. Click on Field and select Operating System.
5. Select the Condition that you want to search on from one of the following:
  - Starts with
  - Ends with
  - Is (exactly)
  - Is not
  - Present
  - Not present
6. In the Value field, enter the value that you want to search for, such as “Windows Server 2003.”
7. Click Find Now.

### Using a command-line interface

You can query for computer objects of a particular operating system using either DSQuery or AdFind. To perform the query with DSQuery, use the following syntax:

```
> dsquery * <DomainDN> -scope subtree -attr "*" -filter "(&(objectclass=computer)(objectcategory=computer)(operatingSystem=Windows Server 2003))"
```

To use AdFind, enter the following:

```
> adfind -b <DomainDN> -s subtree -f
"(&(objectclass=computer)(objectcategory=computer)
(operatingSystem=Windows Server 2003))"
```

## Using VBScript

```
' This code searches for computer objects that have Service Pack 1 installed.
' ----- SCRIPT CONFIGURATION -----
strBase = "<LDAP://& " & "<DomainDN>" & ">";
' ----- END CONFIGURATION -----

strFilter = "(&(objectclass=computer)(objectcategory=computer)" & _
"(operatingSystemServicePack=Service Pack 1));"
strAttrs = "cn,operatingSystem,operatingSystemVersion," & _
"operatingSystemServicePack;"
strScope = "subtree"

set objConn = CreateObject("ADODB.Connection")
objConn.Provider = "AdsDSOobject"
objConn.Open "Active Directory Provider"
Set objRS = objConn.Execute(strBase & strFilter & strAttrs & strScope)
objRS.MoveFirst
while Not objRS.EOF
    Wscript.Echo objRS.Fields(0).Value
    Wscript.Echo objRS.Fields(1).Value
    Wscript.Echo objRS.Fields(2).Value
    Wscript.Echo objRS.Fields(3).Value
    Wscript.Echo objRS.Fields(4).Value
    Wscript.Echo
    objRS.MoveNext
wend
```

## Discussion

When a computer joins an Active Directory domain, the operating system attributes are updated for the computer object. There are four of these attributes, which can be used in queries to find computers that match certain OS-specific criteria, like service pack level.

These attributes include the following:

### operatingSystem

Descriptive name of the installed Operating System—e.g., Windows Server 2003, Windows 2000 Server, and Windows XP Professional

### operatingSystemVersion

Numerical representation of the operating system—e.g., 5.0 (2195) and 5.2 (3757)

operatingSystemServicePack

Current service pack level if one is installed—e.g., Service Pack 2 and Service Pack 3



This recipe typically applies only to Windows-based machines. Other types of machines (e.g., Unix) that have accounts in Active Directory might not automatically update their OS attributes, though some newer Unix- or Linux-based NAS devices have been configured to do. Additionally, the `operatingSystem` attribute does not distinguish between Windows NT 4 server and Windows NT 4 workstation.

## 8.14 Binding to the Default Container for Computers



This recipe requires the Windows Server 2003 domain functional level.

### Problem

You want to bind to the default container that new computer objects are created in.

### Solution

#### Using a graphical user interface

1. Open LDP.
2. From the menu, select Connection → Connect.
3. For Server, enter the name of a domain controller (or leave blank to do a serverless bind).
4. For Port, enter 389.
5. Click OK.
6. From the menu, select Connection → Bind.
7. Enter credentials of a domain user.
8. Click OK.
9. From the menu, select View → Tree.
10. For the DN, enter:  

```
<WKGUID=aa312825768811d1aded00c04fd8d5cd,<DomainDN>>
```

where `<DomainDN>` is the distinguished name of a domain.
11. Click OK.
12. In the lefthand menu, you can now browse the default computers container for the domain.

## Using a command-line interface

With tools like *netdom*, if there is an option to specify only the name of the computer and not its DN or parent container, the computer object will be created in the default Computers container by default. You can use the *redircmp* utility to change this default location, as we will discuss in Recipe 8.15.

## Using VBScript

```
' This code illustrates how to bind to the default computers container.
' ----- SCRIPT CONFIGURATION -----
strDomain = "<DomainDNSName>" ' e.g. apac.rallencorp.com
' ----- END CONFIGURATION -----

' Computer GUID as defined in ntdsapi.h
Const ADS_GUID_COMPUTRS_CONTAINER = "aa312825768811d1aded00c04fd8d5cd"

set objRootDSE = GetObject("LDAP://" & strDomain & "/RootDSE")
set objCompContainer = GetObject("LDAP://<WKGUID=" & _
                                ADS_GUID_COMPUTRS_CONTAINER & "," & _
                                objRootDSE.Get("defaultNamingContext") & ">" )
WScript.Echo objCompContainer.Get("distinguishedName")
```

## Discussion

In much the same way that the TCP/IP protocol defines a list of well-known ports that are commonly used by industry applications (TCP 20 and 21 for FTP, TCP port 80 for HTTP, etc.), Active Directory defines Well-Known GUIDs that map to container objects that are present in every AD installation. The Domain NC defines the following WKGUIDs:

- Users
- Computers
- System
- Domain Controllers
- Infrastructure
- Deleted Objects
- Lost and Found

The Configuration NC also defines its own Deleted Objects WKGUID.

For example, the default computers container has the following WKGUID:

```
aa312825768811d1aded00c04fd8d5cd
```

You can use the GUID to bind to the default computers container in the domain using the following ADsPath:

```
LDAP://<WKGUID=aa312825768811d1aded00c04fd8d5cd,dc=apac,dc=rallencorp,dc=com>
```

The list of well-known objects for a domain is contained in the `wellKnownObjects` attribute of the `domainDNS` object for the domain. The `wellKnownObjects` attribute is multivalued with `DNWithBinary` syntax. The following is an example of what that attribute looks like for the *rallencorp.com* domain:

```
B:32:AA312825768811D1ADED00C04FD8D5CD:CN=Computers,DC=rallencorp,DC=com;
B:32:F4BE92A4C777485E878E9421D53087DB:CN=Microsoft,CN=Program
Data,DC=rallencorp,DC=com;
B:32:09460C08AE1E4A4EA0F64AEE7DAA1E5A:CN=Program Data,DC=rallencorp,DC=com;
B:32:22B70C67D56E4EFB91E9300FCA3DC1AA:
CN=ForeignSecurityPrincipals,DC=rallencorp,DC=com;
B:32:18E2EA80684F11D2B9AA00C04F79F805:CN=Deleted Objects,DC=rallencorp,DC=com;
B:32:2FBAC1870ADE11D297C400C04FD8D5CD:CN=Infrastructure,DC=rallencorp,DC=com;
B:32:AB8153B7768811D1ADED00C04FD8D5CD:CN=LostAndFound,DC=rallencorp,DC=com;
B:32:AB1D30F3768811D1ADED00C04FD8D5CD:CN=System,DC=rallencorp,DC=com;
B:32:A361B2FFFFD211D1AA4B00C04FD7D83A:OU=Domain Controllers,DC=rallencorp,DC=com;
B:32:A9D1CA15768811D1ADED00C04FD8D5CD:CN=Users,DC=rallencorp,DC=com;
```

Each value has the format of:

```
B:NumberOfBytes:GUID:DistinguishedName
```

As you can see, the GUID for the first value is the same as the one we used in the `ADsPath` above to bind to the default computers container.

## See Also

Recipe 8.15 for changing the default computers container, and MSDN: Binding to Well-Known Objects Using WKGUID

# 8.15 Changing the Default Container for Computers

## Problem

You want to change the container that computers are created in by default.

## Solution

### Using a graphical user interface

1. Open LDP.
2. From the menu, select Connection → Connect.
3. For Server, enter the name of a domain controller (or leave blank to do a serverless bind).
4. For Port, enter 389.
5. Click OK.
6. From the menu, select Connection → Bind.
7. Enter credentials of a domain user.
8. Click OK.

9. From the menu, select Browse → Modify.
10. For DN, enter the distinguished name of the domainDNS object of the domain you want to modify.
11. For Attribute, enter wellKnownObjects.
12. For Values, enter the following:
 

```
B:32:AA312825768811D1AED00C04FD8D5CD:CN=Computers,<DomainDN>
```

 where <DomainDN> is the same as the DN you enter for the DN field.
13. Select Delete for the Operation and click the Enter button.
14. Go back to the Values field and enter the following:
 

```
B:32:AA312825768811D1AED00C04FD8D5CD:<NewComputersParent>,<DomainDN>
```

 where <NewComputersParent> is the new parent container for new computer objects (e.g., ou=RAllenCorp Computers).
15. Select Add for the Operation and click the Enter button.
16. Click the Run button.
 

The result of the operations will be displayed in the right pane of the main LDP window.

## Using a command-line interface

```
> redircmp "<NewParentDN>"
```

## Using VBScript

```
' This code changes the default computers container.
' ----- SCRIPT CONFIGURATION -----
strNewComputersParent = "<NewComputersParent>" ' e.g. OU=RAllenCorp Computers
strDomain              = "<DomainDNSName>"      ' e.g. rallencorp.com
' ----- END CONFIGURATION -----

Const COMPUTER_WKGUID = "B:32:AA312825768811D1AED00C04FD8D5CD:"
' ADS_PROPERTY_OPERATION_ENUM
Const ADS_PROPERTY_APPEND = 3
Const ADS_PROPERTY_DELETE = 4

set objRootDSE = GetObject("LDAP://" & strDomain & "/RootDSE")
set objDomain = GetObject("LDAP://" & objRootDSE.Get("defaultNamingContext"))
set objCompWK = GetObject("LDAP://" & _
    "<WKGUID=AA312825768811D1AED00C04FD8D5CD," & _
    objRootDSE.Get("defaultNamingContext") & ">")

objDomain.PutEx ADS_PROPERTY_DELETE, "wellKnownObjects", _
    Array( COMPUTER_WKGUID & objCompWK.Get("distinguishedName"))
objDomain.PutEx ADS_PROPERTY_APPEND, "wellKnownObjects", _
    Array( COMPUTER_WKGUID & strNewComputersParent & ", " &
    objRootDSE.Get("defaultNamingContext") )

objDomain.SetInfo
WScript.Echo "New default Computers container set to " & _
    strNewComputersParent
```

## Discussion

Most Active Directory administrators do not use the Computers container within the Domain naming context as their primary computer repository. One reason is that since it is a container and not an OU, you cannot apply Group Policy Objects to it. If you have another location where you store computer objects, you might want to consider changing the default container used to bind to the computers container by changing the well-known objects attribute, as shown in this recipe. This can be beneficial if you want to ensure computers cannot sneak into Active Directory without having the appropriate group policies applied to them. While you can also apply GPOs at the site or the domain level, forcing new computers into a particular Organizational Unit ensures that those computers receive the Group Policy settings that you want them to receive through GPOs linked at the OU level. However, this does not protect you from an administrator (whether intentionally or accidentally) explicitly creating a computer object in the incorrect OU; this only protects you from applications or utilities that do not allow or do not require you to specify an OU when creating the computer.



See Recipe 8.14 for more information on how well-known objects are specified in Active Directory.

## See Also

MS KB 324949 (Redirecting the Users and Computers Containers in Windows Server 2003 Domains)

# 8.16 Listing All the Computer Accounts in a Domain

## Problem

You want to obtain a list of all computer accounts in an Active Directory domain.

## Solution

### Using a graphical user interface

1. Open the Active Directory Users and Computers MMC snap-in.
2. Right-click on the domain node and select Find.
3. In the Find drop-down box, select Computers and click Find Now.

All computer objects in the domain will be displayed in the Search Results window.

## Using a command-line interface

```
> adfind -default -f (objectCategory=computer)
```

## Using VBScript

```
' The following script will enumerate all computer accounts  
' within an Active Directory domain.
```

```
Const ADS_SCOPE_SUBTREE = 2  
strDomain = "<DomainDN>"  
  
Set objConnection = CreateObject("ADODB.Connection")  
Set objCommand = CreateObject("ADODB.Command")  
objConnection.Provider = "ADsDSOObject"  
objConnection.Open "Active Directory Provider"  
  
Set objCommand.ActiveConnection = objConnection  
objCommand.CommandText = _  
    "Select Name, Location from 'LDAP://' & strDomain & "' " _  
    & "Where objectCategory='computer'"  
objCommand.Properties("Page Size") = 1000  
objCommand.Properties("Searchscope") = ADS_SCOPE_SUBTREE  
Set objRecordSet = objCommand.Execute  
objRecordSet.MoveFirst  
  
Do Until objRecordSet.EOF  
    Wscript.Echo "Computer Name: " & objRecordSet.Fields("Name").Value  
    Wscript.Echo "Location: " & objRecordSet.Fields("Location").Value  
    objRecordSet.MoveNext  
Loop
```

## Discussion

### Using VBScript

To obtain a list of domain controllers, rather than just computer objects, you should query the Configuration NC rather than the domain NC, and replace "where objectCategory=computer" with "where objectCategory=ntDSDSA".

## See Also

MSDN: Object Class and Object Category [Active Directory] and MSDN: Object-Class Attribute [AD-Schema]

# 8.17 Identifying a Computer Role

## Problem

You want to identify the role that a particular computer serves in an Active Directory domain.

# Solution

## Using a graphical user interface

1. Open the Active Directory Users and Computers MMC snap-in.
2. Right-click on the domain node and select Find.
3. In the Find drop-down box, select Computers and click Find Now.

The role of each computer will be displayed in the Machine Role column in the Search Results window.

## Using a command-line interface

```
> wmic computersystem get domainrole
```

For a domain controller that holds the PDC Emulator FSMO role, this will return the following output:

```
DomainRole  
5
```



For a DC that doesn't hold the PDCe FSMO, this command will return a value of 4.

## Using VBScript

```
' The following code will return the domain role of the  
' local computer.  
strComputer = "."  
Set objWMIService = GetObject("winmgmts:" _  
    & "{impersonationLevel=impersonate}!\\" _  
    & strComputer & "\root\cimv2")  
Set colComputers = objWMIService.ExecQuery _  
    ("Select DomainRole from Win32_ComputerSystem")  
For Each objComputer in colComputers  
    Select Case objComputer.DomainRole  
        Case 0  
            strComputerRole = "Standalone Workstation"  
        Case 1  
            strComputerRole = "Member Workstation"  
        Case 2  
            strComputerRole = "Standalone Server"  
        Case 3  
            strComputerRole = "Member Server"  
        Case 4  
            strComputerRole = "Backup Domain Controller"  
        Case 5  
            strComputerRole = "Primary Domain Controller"  
    End Select  
    Wscript.Echo strComputerRole  
Next
```

## Discussion

### Using a command-line interface

WMIC is the command-line component of the Windows Management Instrumentation that uses aliases to enable you to easily access WMI namespaces from the command line. To run `wmic` against a remote computer, specify the `/node:"<ComputerFQDN>"` switch.

### Using VBScript

Rather than relying on an `if...else` construct to produce output, this script uses `Select Case`. In situations where there are numerous possible outcomes for a conditional statement, `Select Case` can produce far more elegant code than using numerous `if...else` statements.