Modifying the Active Directory Schema to Support Mac Systems

Technical White Paper
May 2009

Contents

Overview

One of IT's key roles in today's enterprise environment is client management, which involves centrally defining and controlling how each user's computer functions. This can include restricting access to specific applications or websites, configuring auto-update policies, securing various parts of the file system, and setting display preferences or login scripts. Most Windows administrators are familiar with client management and directory services in the form of Active Directory (AD) Group Policy Objects (GPOs). Group Policies that are well planned and executed can significantly ease setup, security, and support processes for new users and computers. Along with managing Windows PCs through Group Policies, AD can be used to manage Mac systems—an important capability given the growing adoption and popularity of the Mac within enterprises.

The full benefits of directory services can be realized when all desktop, notebook, and server systems are integrated into the same infrastructure. This goal has been difficult to achieve in the past due to the proliferation of incompatible directory services solutions.

Now you can integrate the Mac into an existing AD infrastructure. By joining, or binding, Mac computers to an existing AD domain, you can provide user authentication and policy directly to Mac computers centrally from AD. This white paper is designed to provide system administrators with step-by-step instructions for extending the AD schema to manage Mac systems.

Apple offers additional resources to help with Windows and Mac integration, including:

- Best Practices: Integrating Mac OS X with Active Directory, a high-level white paper designed to help system administrators integrate the Mac into an existing Active Directory infrastructure
- Managing Users and Policies on Mac OS X, a white paper with detailed information on how to manage Mac systems using Workgroup Manager

Background

Apple has developed its own comprehensive client management architecture, commonly called Managed Client for Mac OS X (MCX) or simply Managed Preferences. When Mac OS X clients are integrated into Mac OS X Server, Managed Preferences are stored as records in Open Directory, the native directory services provided in Mac OS X Server. Like Group Policies, Managed Preferences can be used to restrict access to many parts of the Mac OS X interface and to control various user and system settings.

Just as Group Policies are a product of AD, Managed Preferences are a product of Open Directory. AD can be extended with Open Directory objects and attributes so that Mac computers can be integrated into an environment in which user and group records stored in AD can have Managed Preferences associated with them, combining the best of both worlds.

With the introduction of the AD plug-in for Mac OS X v10.4 Tiger, called Mac OS X AD plug-in, Apple made a concerted effort to allow IT administrators to integrate Mac OS X clients and servers easily into existing AD infrastructures. While every AD installation is different, Mac OS X integrates well with the vast majority of them—and with minimal effort.

To support Mac systems using AD, Mac computers must be bound to the existing AD infrastructure. AD can also be used to store managed client preferences directly in AD attributes, which requires modification of the AD schema. Once the schema is modified, Open Directory in Mac OS X can read the managed client preferences and apply them. Then administrators can use Workgroup Manager, a free user management tool from Apple, to populate the managed client settings within AD. This white paper explains how to update the AD schema to support the Mac.

Extending the schema for Apple-specific objects and attributes allows an organization to leverage its current AD infrastructure without additional software for Mac computers. Mac OS X systems can then be fully managed for security and organizational policy using a single directory service.

To integrate Mac computers into AD, you need to start with built-in Windows-based tools to apply the initial schema modifications. After the schema modifications have been applied, you can use Workgroup Manager on a Mac to store policies in AD that will be enforced on any Mac computers bound to AD.

When binding Mac OS X to AD with the built-in Mac OS X AD plug-in, Mac OS X will authenticate passwords against AD. However, Mac OS X does not recognize any AD policies other than passwords and any associated password policies. When Mac OS X is bound to Open Directory on Mac OS X Server, it uses MCX from Open Directory to natively implement additional policies that are stored as XML files within Open Directory. For Mac OS X to store and recognize objects and attributes within AD, the Mac computers must be bound to AD using the AD plug-in, the AD schema must be extended, and the administrator must populate MCX inside these objects and attributes. Once the AD schema is extended, it will replicate any changes to the schema throughout the entire AD forest.

AD Schema Analyzer

To help integrate Mac computers into an existing AD infrastructure, Microsoft provides the AD Schema Analyzer, which connects to a preexisting directory service such as Open Directory and compares that schema to the schema in AD. The AD Schema Analyzer then generates custom LDAP Data Interchange Format (LDIF) files that can be used to modify the schema in AD. By using the AD Schema Analyzer to generate the schema modifications versus using stock LDIF files, system administrators will be made aware of any recent updates to the AD schema from Apple, Microsoft, or third parties. For example, in Windows Server 2003, the RFC 2307 object classes and attributes were added to the AD schema and were no longer required to be added for MCX.

Requirements

The following list outlines the components required to use the AD Schema Analyzer to generate LDIF files and modify the schema in AD:

- Windows XP with Service Pack 2 with .NET Framework 2.0 or later installed
- A test AD domain controller with the specified organization's current schema with Windows Server 2003 R2 schema or greater
- Mac OS X Server v10.5 or later promoted to an Open Directory Master
- Mac OS X v10.5 or later with Workgroup Manager installed to test schema modifications

Creating the LDIF Schema Modifications

Using the AD Schema Analyzer to generate the required schema modifications is the first step in connecting Mac computers to both AD and Open Directory. The following example shows how to accomplish this task.

1. On the Windows XP computer, download and install Microsoft's Active Directory Application Mode (ADAM) directory services toolset: www.microsoft.com/windowsserver2003/adam/default.mspx. While ADAM provides much greater functionality than the AD Schema Analyzer tool, it is the only tool required for generating the schema modifications. Note that ADAM does not currently run in Windows Vista, though it is available in Windows Server 2008 as part of the Lightweight Directory Services (LDS) role.
2. Choose Start > All Programs > ADAM > ADAM Tools Command Prompt.
3. Select the AD Schema Analyzer tool by entering "ADSchemaAnalyzer" and pressing Return.
4. Choose File > Load Target Schema and enter the DNS name of the Open Directory Master.
5. Select Simple bind type and leave the other fields blank. The server type should be left as Automatic.
6. Choose File > Load Base Schema. Enter the DNS name of the AD domain controller, the AD user name and password (an administrator's credentials are not required), and the domain where the AD user's account resides. Select Secure bind type to enable the domain text box.
7. Choose Schema > Hide Present Elements. The object classes and attributes contained in Open Directory, but absent from AD, will be shown.

   Expand the Classes folder and then select the following classes and attributes. Note that you should not select all of the attributes because the other attributes may already be included within AD or are not needed. For example, a user in AD is composed of the following objectClasses: User, Person, and OrganizationalPerson. The User objectClass already contains the userCertificate and jpegPhoto attributes, so you do not need to include them in the apple-user objectClass. By adding apple-user to the objectClasses of the User objectClass in AD, Apple-specific attributes will be added to a User. The object classes and attributes that are added are nearly all "apple-" specific. The only exception is ttl (time to live), which is an attribute used to determine how long a value specified will last. The "Mac OS X Directory Data" appendix in the Mac OS X Server Open Directory Administration guide (www.apple.com/server/macosx/resources) describes the object classes and attributes selected in the following list.

[+] apple-computer-list

[+] subclassOf: top

[X] possSuperior: top

[+] rdnAttId: cn

[+] mayContain: apple-computer-list-groups

[+] mayContain: apple-computers

[X] mayContain: apple-computer-list-groups

[X] mayContain: apple-generateduid

[+] mayContain: apple-keyword

[+] mayContain: apple-mcxflags

[+] mayContain: apple-mcxsettings

[X] mustContain: cn

Make sure that any other nonspecific attributes are deselected (have an "*x*" in the checkbox). The only checkboxes that should be selected within the classes are shown above. When selecting schema classes and attributes, be sure to enable the desired class and then click to disable undesired attributes within a class.

For example, for the apple-computer-list, select apple-computer-list and then deselect mayContain: apple-generateduid, possSuperior: top, mayContain: apple-generateduid, and mustContain: cn.

The following is a list of objectClasses and attributes to select. Make sure all of the objectClasses and attributes are selected, and deselect any attributes that are not contained in the list.

apple-computer

subclassOf: top

rdnAttId: cn

mayContain: apple-category

mayContain: apple-computer-list-groups

mayContain: apple-keyword

mayContain: apple-mcxflags

mayContain: apple-mcxsettings

mayContain: apple-networkview

mayContain: apple-service-url

mayContain: apple-xmlplist

mayContain: macAddress

mayContain: ttl

apple-computer-list

subclassOf: top

rdnAttId: cn

mayContain: apple-computer-list-groups

mayContain: apple-computers

mayContain: apple-keyword

mayContain: apple-mcxflags

mayContain: apple-mcxsettings

```
apple-configuration
    subclassOf: top
    rdnAttId: cn
    mayContain: apple-data-stamp
    mayContain: apple-keyword
    mayContain: apple-xmlplist
    mayContain: ttl
apple-group
    subclassOf: top
    rdnAttId: cn
    mayContain: apple-group-homeowner
    mayContain: apple-group-homeurl
    mayContain: apple-keyword
    mayContain: apple-mcxflags
    mayContain: apple-mcxsettings
    mayContain: apple-user-picture
    mayContain: ttl
apple-location
    subclassOf: top
    rdnAttId: cn
    mayContain: apple-dns-domain
    mayContain: apple-dns-nameserver
apple-neighborhood
    subclassOf: top
    rdnAttId: cn
    mayContain: apple-category
    mayContain: apple-computeralias
    mayContain: apple-keyword
    mayContain: apple-neighborhoodalias
    mayContain: apple-nodepathxml
    mayContain: apple-xmlplist
    mayContain: ttl
apple-serverassistant-config
    subclassOf: top
    rdnAttId: cn
    mayContain: apple-xmlplist
```

apple-service

subclassOf: top
rdnAttId: cn
mayContain: apple-dnsname
mayContain: apple-keyword
mayContain: apple-service-location
mayContain: apple-service-port
mayContain: apple-service-url
mayContain: ipHostAddress
mustContain: apple-service-type

apple-user

subclassOf: top
rdnAttId: cn
mayContain: apple-imhandle
mayContain: apple-keyword
mayContain: apple-mcxflags
mayContain: apple-mcxsettings
mayContain: apple-user-authenticationhint
mayContain: apple-user-class
mayContain: apple-user-homequota
mayContain: apple-userhomesoftquota
mayContain: apple-user-mailattribute
mayContain: apple-user-picture
mayContain: apple-user-printattribute
mayContain: apple-webloguri

mount

subclassOf: top
rdnAttId: cn
mayContain: mountDirectory
mayContain: mountDumpFrequency
mayContain: mountOption
mayContain: mountPassNo
mayContain: mountType

Do not select any attributes in the Attributes section because the attributes associated with the apple object classes are automatically selected.

8. Choose File > Create LDIF File and save the LDIF file. This LDIF file contains all the schema modifications required for AD.
9. Verify that you receive the message, "LDIF file created: 36 attributes, 10 classes, 0 property sets, 0 updated present elements." If you did not export 36 attributes and 10 classes, recheck your selections and export again.

Modifying the Resulting LDIF File

Once you export the LDIF file from AD Schema Analyzer, the file must be modified. Some of the object classes need to be changed, and some object classes require additional prefixes. You will also need to specify where in AD the required objects can be created.

Updating objectClassCategory

The LDIF file from the AD Schema Analyzer results in all objectClasses being assigned an attribute objectClassCategory with a value of 1. An objectClassCategory of 1 means that the object is a structural type, and an objectClassCategory of 3 is an auxiliary type. Structural objectClasses can be used to make objects within AD, while Auxiliary objectClasses can only be used to extend a structural object (or other auxiliary objects). User, Group, and Computer objectClasses exist within AD already, so the Apple associated objectClasses that extend Users, Groups, and Computers need to be changed to the objectClassCategory of 3 (Auxiliary).

1. Open the LDIF file in Wordpad.
2. Find the apple-user, apple-group, and apple-computer objectClasses in the Classes section of the LDIF file and change the objectClassCategory to 3, as shown for the apple-user objectClass:

```
# Class: apple-user
dn: cn=cls-apple-user,cn=Schema,cn=Configuration,dc=X
changetype: ntdsschemaadd
objectClass: classSchema
governsID: 1.3.6.1.4.1.63.1000.1.1.2.1
ldapDisplayName: apple-user
adminDescription: apple user account
objectClassCategory: 3
```

3. To effectively use Auxiliary objectClasses, they need to be associated with current objectClasses in AD. At the end of the LDIF file, add the following three sections to extend the User, Computer, and Group objectClasses in AD with the apple-user, apple-computer, and apple-group auxiliary classes:

```
# Add the new class to the user object
dn: CN=User,CN=Schema,CN=Configuration,DC=X
changetype: modify
add: auxiliaryClass
auxiliaryClass: apple-user
-
```

```
# Add the new class to the computer object
dn: CN=Computer,CN=Schema,CN=Configuration,DC=X
changetype: modify
add: auxiliaryClass
auxiliaryClass: apple-computer
-


# Add the new class to the group object
dn: CN=Group,CN=Schema,CN=Configuration,DC=X
changetype: modify
add: auxiliaryClass
auxiliaryClass: apple-group
-
```

Note: After each "-" line above, there *must* be a blank line, or the schema modifications will fail.

4. Save the LDIF file, but do not close the document.

Updating Prefixes

The AD Schema Analyzer adds a prefix of "cls" to all object classes and "attr" to all attributes. Because most of the object classes and attributes already have the "apple-" prefix, you need to remove the "attr-" and "cls-" prefix:

1. If you closed it after the previous action, open the LDIF file with Write in Windows.
2. In the Edit menu, select Replace.
3. Search for "dn: cn=cls-" and replace it with "dn: cn=" (do not specify the quotes).
4. Search for "dn: cn=attr-" and replace it with "dn: cn=" (do not specify the quotes).

   Microsoft recommends that all vendor-specific object classes and attributes include a prefix with the vendor's name. The majority of object classes and attributes already include the "apple-" prefix; however, the mount object class does not have a prefix and should be updated with the "apple-" prefix. Each of the following lines lists the addition of the "apple-" prefix to the start of the dn:

Replace:	dn: cn=mountDirectory,cn=Schema,cn=Configuration,dc=X
With:	dn: cn=apple-mountDirectory,cn=Schema,cn=Configuration,dc=X
Replace:	dn: cn=mountDumpFrequency,cn=Schema,cn=Configuration,dc=X
With:	dn: cn=apple-mountDumpFrequency,cn=Schema,cn=Configuration,dc=X
Replace:	dn: cn=mountOption,cn=Schema,cn=Configuration,dc=X
With:	dn: cn=apple-mountOption,cn=Schema,cn=Configuration,dc=X
Replace:	dn: cn=mountPassNo,cn=Schema,cn=Configuration,dc=X
With:	dn: cn=apple-mountPassNo,cn=Schema,cn=Configuration,dc=X
Replace:	dn: cn=mountType,cn=Schema,cn=Configuration,dc=X
With:	dn: cn=apple-mountType,cn=Schema,cn=Configuration,dc=X
Replace:	dn: cn=mount,cn=Schema,cn=Configuration,dc=X
With:	dn: cn=apple-mount,cn=Schema,cn=Configuration,dc=X

5. Save the LDIF file, but do not close the document.

Updating possSuperiors

AD includes an attribute within each object class that specifies what parent an object can have in the directory. This does not apply to auxiliary classes; it applies only to the object classes used to create objects within the directory. You need to specify where the objects can be created. Do so by specifying the possSuperiors attribute. For all apple object classes that can be used to create objects in AD, specify possSuperiors of "organizationUnit" and "container." If you wish to specify where objects can be created, find the following object classes, remove any possSuperiors if they exist, and add "organizationalUnit" and "container" as values for the possSuperiors attribute:

```
apple-computer-list
        possSuperiors: organizationalUnit
        possSuperiors: container
apple-configuration
        possSuperiors: organizationalUnit
        possSuperiors: container
apple-location
        possSuperiors: organizationalUnit
        possSuperiors: container
apple-neighborhood
        possSuperiors: organizationalUnit
        possSuperiors: container
apple-serverassistant-config
        possSuperiors: organizationalUnit
        possSuperiors: container
apple-service
        possSuperiors: organizationalUnit
        possSuperiors: container
apple-mount
        possSuperiors: organizationalUnit
        possSuperiors: container
```

Verifying the LDIF File

To ensure that the LDIF file was correctly created, verify the following information:

- apple-user, apple-group, and apple-computer have an objectClassCategory of 3.
- The Attributes section contains 36 attributes and all attributes, except for ttl, have an attributeID that starts with 1.3.6.1.4.1.63.1000.1.1.1.
- The Classes section contains 10 classes, and all governsIDs start with 1.3.6.1.4.1.63.1000.1.1.2.
- All attributes and classes, except for ttl, have an "apple-" prefix in their dn: and ldapDisplayName.
- After each "-" line, there is a blank line.
- Every objectClass that does not have an objectClassCategory of 3 has organizationalUnit and container as possSuperiors.

Updating the Schema on a Domain Controller in the Forest

The LDIF file created in the previous section can now be used to update the schema on a domain controller. The schema changes will be replicated to the rest of the forest during the next replication cycle. A domain controller must be marked to allow write access for schema modifications. The LDIF file can then be imported using the ldifde command.

1. Copy the LDIF file created in the previous section to the domain controller where you plan to implement the AD schema modifications.
2. On the domain controller that has been designated the schema master, import the LDIF file using the ldifde command. The following command assumes that the LDIF file is named "apple-mods.ldf" and the domain name of the domain controller is EXAMPLE.COM:

   ```
   ldifde /j . /k /i /f apple-mods.ldf /v /c "DC=X" "DC=EXAMPLE,DC=COM"
   ```

   Note that /k will ignore errors if objectClasses or attributes already exist in the schema, /i will perform an import, /f specifies the file to import, /v is verbose output, /c will replace "DC=X" with the correctly formatted distinguishing name for the domain, and /j will send a copy of the output to ldif.err and ldif.log in the current directory. Also do not change "DC=X," as it produces the formatting required to use the /c option.

Additional Changes

Indexing the MAC Address

When a Mac searches AD for policy applied to a computer on a computer account or as part of a Computer List, the system searches for the MAC address attribute in computer accounts. The MAC address that is searched for is the MAC address of the primary interface of the system (usually the built-in Ethernet port or wireless port on the MacBook Air). The system does not have to communicate with AD over this network port. The MAC address attribute is populated in a computer account when a Mac computer is bound to AD.

Because a Mac will periodically search for policy applied to its computer account, the MAC address attribute should be indexed within AD to improve search time and reduce overhead on domain controllers. Indexing the MAC address attribute within AD will provide greater responsiveness to searches and reduce CPU overhead on domain controllers.

Instructions on how to index the MAC address attribute in AD are provided in the support document at http://support.apple.com/kb/TS1534.

Verifying the Changes

Once the schema modifications are complete, Workgroup Manager can create policy on Users, Groups, Computers, and Computer Lists. Use the following steps to verify that the schema modifications were successfully applied:

1. In AD Users and Computers, create a security Group and populate it with AD users who log in to Mac computers.
2. On a Mac OS X client with Workgroup Manager (v10.5 or later) installed, bind to AD using the Directory Utility AD plug-in located in the /Applications/Utilities folder. If the client was bound prior to the schema changes being applied, you need to refresh the settings by either rebooting or restarting Directory Services on the client:

   ```
   sudo killall DirectoryService
   ```
3. Launch Workgroup Manager, but do not authenticate. Click Cancel at the authentication window.
4. In the Server menu, select View Directories.
5. In the small globe in the upper right corner of the Workgroup Manager window, select Other. Select AD and All Domains (or your specific domain if that is how you have configured the AD plug-in).

6. Select the Group tab and find the Group you created in step 1.
7. Click the Preferences button and select the System Preferences icon.
8. In the upper right corner, click the lock button and authenticate as an AD Administrator who has write access to attributes of this AD Group.
9. Change the Manage radio button to Always, click Select None, and click Apply.
10. Log out of the Mac and log in as an AD user who is a member of the Group created in step 1. Go to System Preferences and note that all System Preferences panes are dimmed.

Conclusion

By leveraging a current directory infrastructure, IT teams can provide policy and authentication to their existing Mac population and ease the integration of new Mac computers into their organization's network. Extending the AD schema provides an effective way for organizations to comply with security and policy requirements for their Mac computers, while offering increased compatibility to a growing population of Mac users.

Additional Resources

For more information about integrating Mac OS X clients into AD environments—including documentation, training, articles, scripts, and discussions—visit the following sites:

- www.apple.com/business/solutions/it/directory.html
- www.apple.com/server/resources
- www.apple.com/training

You can also reference the following papers for more information:

- Best Practices: Integrating Mac OS X with Active Directory, a high-level white paper designed to help system administrators integrate the Mac into an existing Active Directory infrastructure
- Managing Users and Policies on Mac OS X, a white paper with detailed information on how to manage Mac systems using Workgroup Manager

If you have any questions about the detailed steps in this paper, or any other aspect of integrating Mac OS X systems with Active Directory, please contact your Apple representative or Apple Authorized Reseller for assistance.