```
set objWMI = GetObject("winmgmts:\\.\root\cimv2")
set fso = CreateObject("Scripting.FileSystemObject")
```

Lecture 7
Working with ADSI

Print page

Welcome to Lecture 7.

Domain, group and user maintenance is probably one of the top administrative tasks that we want to automate.

Administering a directory service often involves numerous repetitive tasks such as creating, deleting, and modifying users, groups, organizational units, computers, and other directory resources. Performing these steps manually by using graphical user interface (GUI) tools is time-consuming, tedious, and error prone. Active Directory Service Interfaces (ADSI) is the technology that allows you to create custom scripts to automate such repetitive tasks. In this lecture we will learn how to manipulate domains, groups and users via ADSI.

Unfortunately if you do not have a network or system administrative role on the computer or network you are using, you will be unable to perform all the tasks we discuss in this lecture. However, all the programs provided in this lecture can be run with an Administrator user account, therefore you should be able to run them at home.

Key words

Directory Service, ADSI (Active Directory Service Interfaces), Network Resources, Distributed Computing, Domain, Group, User Manipulation, Domain Name, Domain Object, Domain Component, Group Name, Group Object, User Name, User Object, Organization Unit, Common Name, Administrative Role

Reference to textbook chapters

This lecture covers Chapters 14, 15 & 16. (Don Jones, VBScript, WMI and ADSI unleashed : using VBSscript, WMI, and ADSI to automate Windows administration eBook: Chapter 14. Working with ADSI Providers; Chapter 15. Manipulating Domains; Chapter 16. Manipulating Users and Groups).

Active Directory Services Interface (ADSI): basic concepts

Active Directory (AD) is a directory service used to store information about the network resources across a domain and user information on the local computer. Microsoft provides many tools to enumerate, diagnose, and manage AD, and more and more commercial tools are also available. ADSI is one of the most useful tools.

Similar to WMI, Active Directory Service Interfaces (ADSI) is an object library. It can be used in a distributed computing environment for managing network resources from different network providers. Administrators and developers can use ADSI services to enumerate and manage the resources in a directory service, no matter which network environment contains the resource.

The tasks that ADSI can perform:

• creating, deleting, and modifying users, groups, organizational units, computers;

• managing printers, and
• locating resources in a distributed computing environment.

Like WMI, you can write ADSI client applications in many languages, such as Visual Basic, C++, JScript, VBScript.

Note that Active Directory runs on Windows Server 2008, Microsoft Windows Server 2003, and Windows 2000 Server domain controllers. Only client applications using ADSI may run on Windows Vista, Windows XP, Windows 2000, Windows NT 4.0.

ADSI providers

ADSI provides a well-defined set of objects, methods, and properties to interact with supported directories. Abstracted access to an independent directory resource is via an ADSI provider. Surprisingly, we can use several ADSI service providers to access same information. The commonly used ADSI providers are WinNT, LDAP, NDS, and NWCOMPAT. In this lecture, we mainly discuss the use of the WinNT provider.

The WinNT provider can connect to any NT-compatible domain, therefore can be used in any Windows NT compatible operating systems, such as Windows 2000 or above. The following code is a typical ADSI program using WinNT provider.

Remember that we ever use WinNT ADSI in lecture three:

```
Set oNetwork = CreateObject("WScript.Network")
sDomain = oNetwork.UserDomain
sUser = oNetwork.UserName

Set oUser = GetObject("WinNT://" & sDomain & "/" & sUser & ",user")
For Each oGroup In oUser.Groups
MsgBox oGroup.Name
Next
```

The fourth line of the code gets an ADSI object from WinNT with specified a WinNT ADsPath. This object contains the information of the user who's currently logging in.

In general, in order to get a WinNT provided ADSI object, you have to specify a WinNT ADsParh. The syntax of WinNT ADsPath is the following:

```
WinNT:
WinNT://<domain name>
WinNT://<domain name>/<server>
WinNT://<domain name>/<object path>
WinNT://<domain name>/<object name>
WinNT://<domain name>/<object name>,<object class>
WinNT://<server>
WinNT://<server>/<object name>
WinNT://<server>/<object name>,<object class>
```

The domain name can be either a NETBIOS name or a DNS name, which can be retrieved through WScript.Network object. The server is the name of a specific server within the domain. The object path is the path of an object, such as "printserver1/printer2". The object name is the name of a specific object. The object class is the class name of the named object. The information about an object can also be retrieved through WScript.Network object. With my computer, for the user object of my account, the ADsPath will be "WinNT://jamie-win/jamie,user", where "jamie-win" is the domain name and "jamie" is the object name and "user" is the specified class.

Another more commonly used but more complicated ADSI provider is LDAP Provider. LDAP stands for Lightweight Directory Access Protocol. The LDAP provider implements a set of ADSI objects that support various ADSI interfaces, therefore it can work with any LDAP compatible directory, not just Active Directory.

Similar to WinNT ADSI provider, to access an ADSI LDAP object, we need to specify a LDAP ADsPath. The syntax is slightly different from the one for WinNT ADsPath.

```
' Bind to the root of the LDAP namespace.
LDAP:
' Bind to a specific server.
```

```
LDAP://server01
' Bind to a specific server using the specified port number.

LDAP://server01:390
' Bind to a specific object.
LDAP://CN=Jeff Smith,CN=users,DC=fabrikam,DC=com
' Bind to a specific object through a specific server.
LDAP://server01/CN=Jeff Smith,CN=users,DC=fabrikam,DC=com
```

We have to follow the following conventions when we specify a LDAP ADsPath:

- Use DC when specifying any portion of a domain name. Always list the domain name components in their regular order. For example, a domain named scm.uws.edu.au would have an LDAP path of "dc=scm,dc=uws,dc=edu,dc=au". DC stands for "Domain Component".
- Use OU when specifying an "Organizational Unit".
- Use CN when specifying a "Common Name", such as a user, group, or any of the built-in AD containers.

With the ADSI LDAP provider, you can only create a global user account. Local accounts reside in the SAM database and must be created using the WinNT provider. Since I don't have a network administrative credentials in my office network, I can't create a global user. Therefore I can't run any LDAP query in my computer. If you have a network environment at home, please try the code listed in the textbook Chapter 15.

Manipulating domains

If you have administrative credentials of your network, you can access and change the setting of the network domain.

To query domain information by using the WinNT provider, simply do the following (specifying your domain name; the domain name can be retrieved through WScript.Network object):

```
Dim objDomain
Set objDomain = GetObject("WinNT://MyDomain")
```

Then you can get the information about your domain by using get method. For instance,

```
objDomain.Get("MinPasswordAge")
```

will give you the minimum number of days a user must keep this password.

To set this this property, simply do something like this:

```
objDomain.put("MinPasswordAge", 10)
```

The properties of a domain you can operate by using the WinNT provider are:

- MinPasswordLength
- MinPasswordAge
- MaxPasswordAge
- MaxBadPasswordsAllowed
- PasswordHistoryLength
- AutoUnlockInterval
- LockoutObservationInterval

See WinNT Schema's Mandatory and Optional Properties from Microsoft.

For settings of Organization Units, please refer to the textbook pages 260-263.

Manipulating users and groups

To manipulate users and groups, we first get an domain object as we did above (`objDomain`). Then you can call Create method to create a new user or group:

```
Set oNetwork = WScript.CreateObject("WScript.Network")
sDomain = oNetwork.UserDomain
sUser = oNetwork.UserName
msgbox "User domain: " & sDomain & "   User name:  " & sUser
Set oDomain = GetObject("WinNT://" & sDomain)

Set oUser = oDomain.Create("user", "Daniel")
oUser.Description = "Test user"
```

```
Set oGroup = oDomain.Create("group", "TestGroup")
oGroup.Description = "Test group"

oUser.SetInfo
oGroup.SetInfo

oUser.GetInfo
oGroup.GetInfo

WScript.echo "User name: " & oUser.Name
WScript.echo "User description: " & oUser.Description

WScript.echo "Group name: " & oGroup.Name
WScript.echo "Group description: " & oGroup.Description
```

This code (CreateUserGroup.vbs) can explain itself. It creates a new user "Daniel", a new group "TestGroup" and display the related information. You might be curious about the methods GetInfo and SetInfo. GetInfo method is used to load into the cache of ADSI object properties from the underlying directory store. SetInfo method saves the cached property values of the ADSI objects to the underlying directory store. Note that any property value changes made by Create or Put methods will be lost if GetInfo (or GetInfoEx) is invoked before SetInfo is called.

If you have an administrative role in a Windows system, you can test the program CreateUserGroup.vbs without needing any change.

If a user name or a group name already exist, you can create a user object or a group object from their names by calling GetObject method:

```
Set oNetwork = WScript.CreateObject("WScript.Network")
sDomain = oNetwork.UserDomain
Set oDomain = GetObject("WinNT://" & sDomain)
Set oUser1 = GetObject("WinNT://" & sDomain & "/Daniel,user")
WScript.echo oUser1.Description
Set oGroup1 = GetObject("WinNT://" & sDomain & "/TestGroup,group")
WScript.echo oGroup1.Description
```

The above code (UserGroupMore.vbs) must be run after CreateUserGroup.vbs has been run. The following code (SetPassword.vbs) demonstrates how to change password programmatically:

```
Set oNetwork = WScript.CreateObject("WScript.Network")
sDomain = oNetwork.UserDomain
Set oDomain = GetObject("WinNT://" & sDomain)
Set oUser = GetObject("WinNT://" & sDomain & "/Daniel,user")
oUser.SetPassword "12345"
oUser.SetInfo
WScript.echo "Password changed"
```

For more information for user and group manipulation, read Microsoft references IADsUser Interface, IADsGroup Interface or more generally ADSI Objects of WinNT. Microsoft also provides a few examples for User Account Management. Please read these examples carefully because a few quiz questions might be based on some of the examples.

All in all, Active Directory Service is a complicated and powerful facility. It takes a while to get familiar with its internal structure and usage. Hope you had an easy time.