

Likewise 4.0 Programmer's Guide

IN THIS DOCUMENT

- Programmatically change Likewise data in Active Directory.
- Create simple scripts with VBScript.
- Write scripts with C# and .NET.
- Use the objects and interfaces that Likewise provides.
- View the methods and properties of Likewise objects.

Abstract

This document describes how scripts can be used to programmatically modify the information maintained by Likewise in Microsoft Active Directory. It identifies the scripting objects defined by Likewise and explains their use.

The information contained in this document represents the current view of Likewise Software on the issues discussed as of the date of publication. Because Likewise Software must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Likewise, and Likewise Software cannot guarantee the accuracy of any information presented after the date of publication.

These documents are for informational purposes only. LIKewise SOFTWARE MAKES NO WARRANTIES, EXPRESS OR IMPLIED.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in, or introduced into a retrieval system, or transmitted in any form, by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Likewise Software.

Likewise may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Likewise, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2007 Likewise Software. All rights reserved.

Likewise and the Likewise logo are either registered trademarks or trademarks of Likewise Software in the United States and/or other countries. All other trademarks are property of their respective owners.

Likewise Software
15395 SE 30th Place, Suite #140
Bellevue, WA 98007
USA

Table of Contents

INTRODUCTION	4
TURNING ON SCRIPTING.....	4
SIMPLE SCRIPTING WITH VBSCRIPT.....	4
SCRIPTING WITH C# OR VB.NET	8
ADVANCED TOPICS	9
PROGRAMMER'S REFERENCE.....	10
ILikewiseIdentity	10
ILikewiseCell	11
ILikewiseUser	13
ILikewiseGroup	14

Introduction

Likewise joins Unix, Linux, and Mac OS X computers to Active Directory so that you can centrally manage all your computers from one source, authenticate users with the highly secure Kerberos 5 protocol, control access to resources, and apply group policies to non-Windows computers. The result: Vastly simplified account management, improved security, reduced infrastructure costs, streamlined operations, and easier regulatory compliance.

Likewise provides graphical tools to manage Linux and Unix information in Active Directory. However, it can be useful to access and modify the information programmatically. For this purpose, Likewise provides scripting objects that can be used by any programming language that supports the Microsoft Common Object Model, or COM. The scripting objects provide dual interfaces that can be used by languages that use COM early binding, such as C++ and C#, and by languages that use *Idispatch*, such as VBScript and Jscript.

This document describes the Likewise scripting objects and explains how to use them.

Turning On Scripting

Before you can use Likewise scripting objects, you must first register the Likewise DLL that provides this functionality. This DLL, *LikewiseIdentity.dll*, is placed in the Likewise installation directory, typically, *c:\program files\Centeris\LikewiseIdentity*. Register the DLL by using the Microsoft .NET *regasm* utility:

```
RegAsm "c:\program files\Centeris\LikewiseIdentity\LikewiseIdentity.dll"
```

(Note: if your system can not find *regasm*, you may need to use it from *%windir%\Microsoft.NET\Framework\v2.0.50727* or to add this directory to your *path*).

Simple Scripting with VBScript

To access the Likewise scripting objects from VBScript (using either *cscript.exe* or *wscript.exe*), you must first use *CreateObject* to create an instance of the main Likewise object:

```
Dim oLWIMain  
Set oLWIMain = CreateObject("Likewise.Identity")
```

Once you have this object, your first step is to *connect* to the Active Directory domain that you want to access. There are two ways of doing this:

```
Dim oLWI
```

```
Set oLWI = oLWIMain.Initialize
oLWI.Connect "LDAP://server/dc=yourdomain,dc=com"
' Alternatively
' oLWI.ConnectEx "LDAP://server/dc=yourdomain,dc=com", "username", "password"
```

The first form of *Connect* uses your current login credentials whereas the second form allows you to pass in an alternate set of credentials. If you need to make changes to Active Directory, it's essential that you have sufficient privileges to do so.

Regardless of which form you use, at this point, you now have an object (*oLWI*) that is connected to Likewise and is ready to access information. This object implements the *ILikewiseIdentity* interface. This interface allows you to access and manipulate Likewise *cells*.

You can get a list of all the cells in a domain or a particular cell in the domain as follows:

```
Dim listCells
Dim oCell
Set listCells = oLWI.GetCells()
For each oCell in listCells
    ' Do something with oCell
Next

Set oCell = oLWI.GetCell("OU=Engineering")
```

Note that when accessing a specific cell, you have to provide the Active Directory *canonical name* (cn) of the cell. Other Likewise scripting objects will follow this pattern. You will specify canonical names that Likewise will interpret relative to the *distinguished name* (dn) that you specified when connecting to your domain.

To create or delete a cell, do this:

```
Set oCell = oLWI.CreateCell("OU=Accounting")

oLWI.DeleteCell aCell
```

Note that to delete a cell, you provide a cell object rather than its canonical name.

Cell objects implement the *ILikewiseCell* interface. This interface provides several properties that let you get/set information about the cell and several methods to let you access and manipulate the users and groups that have been enabled.

```
' Get various cell properties
sDisplayName = oCell.DisplayName
bIsDefaultCell = oCell.IsDefaultCell
nNextUID = oCell.NextUID
```

```
nNextGID = oCell.NextGID
sDefaultHomeDir = oCell.DefaultHomeDirectory
sDefaultShell = oCell.DefaultLoginShell

Dim listUsers
Set listUsers = oCell.GetUsers()
Dim listGroups
Set listGroups = oCell.GetGroups()
```

You can *set* these properties to make changes to a cell. Note that the Likewise objects, in a style similar to Microsoft ADSI (Active Directory Scripting Interface) objects, require that you call a method to explicitly *commit* your changes:

```
' Change the default login shell to use bash and the default home dir
oCell.DefaultLoginShell = "/bin/bash"
oCell.DefaultHomeDir = "/local/%d/%u"
oCell.CommitChanges
```

Property changes are not written to Active Directory until you call *CommitChanges*. This pattern improves performance by delaying Active Directory LDAP write operations until requested.

As with the main object, cell objects let you request *all* the users and groups enabled in the cell or let you access specific ones. Additionally, methods are provided to let you enable and disable users and groups in a cell:

```
' Fetch a particular user
Dim oUser
Set oUser = oCell.GetUser("cn=Bob Smith,cn=Users")

' Enable a new user, specifying "Domain Users" as her primary group
sDU = "cn=Domain Users,cn=Users"
Set oUser = oCell.EnableUser("cn=Jane Done,cn=Users", sDU)

' Enable a new user, with an explicit UID
Set oUser = oCell.EnableUserEx("cn=Clark Kent,cn=Users", sDU, 127)

' Disable a user
oCell.DisableUser oUser

' Fetch a particular group
Dim oGroup
Set oGroup = oCell.GetGroup(sDU)

' Enable a group letting Likewise assign a new GID
Set oGroup = oCell.EnableGroup("cn=Enterprise Admins,cn=Users")

' This time, specify an explicit GID
```

```
Set oGroup = oCell.EnableGroupEx("cn=Printer Operators,cn=Users", 444)

' Disable a group
oCell.DisableGroup oGroup
```

User and Group objects implement the *ILikewiseUser* and *ILikewiseGroup* interfaces (respectively). These follow similar patterns:

```
' Fetch some user properties
sDisplayName = oUser.DisplayName
sSID = oUser.SID
nUID = oUser.UID
nGID = oUser.GID
sLoginName = oUser.LoginName ' Actually, the *alias* name
sHomeDir = oUser.HomeDirectory
sLoginShell = oUser.LoginShell
sGECOS = oUser.GECOS

' Change the user comment field
oUser.GECOS = "A fine user of Likewise"
oUser.CommitChanges

' Fetch some group properties
sDisplayName = oGroup.DisplayName
sSID = oGroup.SID
nGID = oGroup.GID
sGECOS = oGroup.GECOS
sGroupAlias = oGroup.GroupAlias

' Change the GID for the group
oGroup.GID = 723
oGroup.CommitChanges
```

Scripting with C# or VB.NET

To programmatically access Likewise with a .NET language, you don't need to go through the *IDispatch* mechanism described above. Rather than using the *LikewiseIdentity.DLL* library, you can reference the Likewise *DSUtil.DLL* library directly. This library is found in the same Likewise installation directory.

The DSUtil library exposes the *LikewiseIdentity* class. This is the actual implementer of the *ILikewiseIdentity* interface described earlier.

In C#:

```
using System;
using Centeris.Likewise.Auth;

namespace ScriptTest
{
    class Program
    {
        static void Main(string[] args)
        {
            // create an instance of LikewiseIdentity
            ILikewiseIdentity lwi = new LikewiseIdentity();

            // connect to AD
            lwi.Connect("LDAP://dc=yourdomain,dc=com");

            // Proceed as with VBScript examples
            ...
        }
    }
}
```

Advanced Topics

It is important to keep in mind that the Likewise objects are associated with other objects in Active Directory. An Identity cell object is associated with an AD organizational unit. Likewise user and group objects are associated with Active Directory objects of the same name. Note that the Likewise objects are **not the same** as the AD objects with which they are associated. Identity objects exist in parallel with the actual AD objects.

Occasionally, it might be necessary to access the AD objects associated with Identity objects. Each Identity object provides a property that lets you do this:

```
' Access the AD user object associated with the Identity user object
Dim oADSIUser
Set oADSIUser = oUser.User

' Access the AD group object associated with the Identity group object
Dim oADSIGroup
Set oADSIGroup = oGroup.Group

' Access the AD organizational unit object associated with the Identity cell
object
Dim oADSIOU
Set oADSIOU = oCell.OU
```

With access to the associated AD objects, you can now use ADSI properties and methods to manipulate the objects. The specifics of these operations are not described here. See the Microsoft documentation and such books as the O'Reilly series on Active Directory:

- Richard, Joe, Robbie Allen, and Alistair G. Lowe-Norris. *Active Directory*. Sebastopol, CA: O'Reilly, 2006.
- Allen, Robbie. *Active Directory Cookbook*. Sebastopol, CA: O'Reilly, 2003.

Programmer's Reference

This appendix describes the objects and interfaces that are provided by Likewise.

ILikewiseIdentity

Properties: (none)

Methods:

```
void Connect(string sLDAPPath)
void ConnectEx(string sLDAPPath, string sUsername, string sPassword)
```

These methods connect the Likewise object to a particular Active Directory domain. The first form uses the current login credentials whereas the second form specifies explicit credentials. In both forms, *sLDAPPath* is a full LDAP distinguished name (dn) identifying the directory to which the object should connect (for example, "LDAP://mydc/dc=mydomain,dc=com").

```
IEnumerable GetCells()
```

This method retrieves a collection of all the cells defined in a domain. The collection can be traversed by using an iterator (e.g. *For Each* in VBScript). Each returned cell object implements the *ILikewiseCell* interface.

```
ILikewiseCell GetCell(string sOUPath)
```

Retrieves an *ILikewiseCell* object for a particular cell identified by *sOUPath*. *sOUPath* is the canonical name of the organizational unit with which the cell is associated (for example, "ou=Accounting,ou=Western Region").

```
ILikewiseCell CreateCell(string sOUPath);
ILikewiseCell CreateCellEx(    string sOUPath,
                                uint nNextUID,
                                uint nNextGID,
                                string sDefaultShell,
                                string sDefaultHomeDirectory)
```

These methods create a new cell in a domain. In both forms, the *sOUPath* is the canonical name of the organizational unit with which the cell is to be associated. In the first form, Likewise will automatically set the *NextUID*, *NetGID*, *DefaultShell* and *DefaultHomeDirectory* properties of the new cell as per the defaults. In the second form, these properties are set explicitly.

```
bool DeleteCell(ILikewiseCell lwc)
```

Deletes a cell from a domain. The cell to be deleted is identified by the *lwc* argument. Note that this is not a canonical name – a cell must first be "gotten" with *GetCell* before it can be deleted.

ILikewiseCell

Properties:

```
string DisplayName (Read only)
```

Contains the displayable name of the cell. This is actually the *displayName* attribute of the organizational unit with which the cell is associated.

```
object OU (Read only)
```

Returns ADSI object of the organizational unit with which the cell is associated.

```
string GuidOU (Read only)
```

Returns the GUID of the organizational unit with which the cell is associated.

```
uint NextUID (Read/write)  
uint NextGID (Read/write)
```

These properties contain the next user ID and group ID that will be automatically allocated to users and groups (respectively) enabled in the cell.

```
bool IsDefaultCell (Read only)
```

This read-only property is set to *true* if the cell is *default cell* for the domain. The default cell is associated with the top-most domain object rather than with any particular organizational unit.

```
string DefaultHomeDirectory (Read/write)
```

Contains the default home directory that will be applied to any new users enabled in the cell. The value of this property should use "%D" and "%U" as placeholders for the user's domain and user name. For example, setting this property to "/home/%d/%u" (its default value) means that the user joe@mydomain.com will have his home directory set to "/home/mydomain/joe".

```
string DefaultLoginShell (Read/write)
```

Contains the default shell program that will be applied to any new users enabled in the cell. By default, this is set to "/bin/bash".

List<string> **LinkedCells** (Read)

Returns the list of cells to which this one is linked. The Likewise Agent searches linked cells for UID, GID and other information *after* searching the current cell. The cells are searched in order with a match stopping the search operation. This implies that linked cells that appear earlier in the list take precedence over ones later in the list.

The list contains the *GUIDs* for the linked cells. While the property is read-only, the returned list can be added to/removed from/modified.

Methods:

bool **CommitChanges** ()

Writes any modified property values out to Active Directory. If you've changed any property settings, you *must* call this method to make those changes permanent.

IEnumerable **GetUsers** ()

Returns a collection of all the users enabled in the cell. Each of the objects in the enumeration implements the *ILikewiseUser* interface.

ILikewiseUser **GetUser**(string sCNUser)

Returns an *ILikewiseUser* object for the object identified by the *sCNUser* argument. This argument should be the AD *canonical name* of the user, for example, "cn=Joe Smith,cn=Users".

ILikewiseUser **EnableUser**(string sCNUser, string sCNPrimaryGroup)
ILikewiseUser **EnableUserEx**(string sCNUser, string sCNPrimaryGroup, uint nUID)

These methods enable a user to access computers in the cell. The user is identified by the AD canonical name in the *sCNUser* argument. The *sCNPrimaryGroup* argument identifies an AD *group* that has been previously enabled in the cell. The user's Linux/UNIX *primary group id* will be set to the GID of this enabled group. The first form of this method lets Likewise automatically assign a UID in the cell to the user. The second form specifies an explicit UID.

bool **DisableUser**(ILikewiseUser lwu)

This method disables a user in a cell, disallowing access to Linux/UNIX computers in the organizational unit with which the cell is associated. The *lwu* argument specifies an *ILikewiseUser* object previously obtained with *GetUsers* or *GetUser*.

ArrayList **GetGroups** ()

Returns a collection of all the groups enabled in the cell. Each of the objects in the enumeration implements the *ILikewiseGroup* interface.

ILikewiseGroup **GetGroup**(string sCNGroup)

Returns an *ILikewiseGroup* object for the object identified by the *sCNGGroup* argument. This argument should be the AD *canonical name* of the group, for example, "cn=Domain Admins,cn=Users".

```
ILikewiseGroup EnableGroup(string sCNGGroup)  
ILikewiseGroup EnableGroupEx(string sCNGGroup, uint nGID)
```

These methods enable the use of an AD group within a Likewise cell. Enabling a group assigns it a group ID in that cell. The first form of this method lets Likewise automatically assign a GID. The second form specifies the GID explicitly. In both forms, the *sCNGGroup* argument should be set to the canonical name of the AD group that is to be enabled in the cell (for example, "cn=Enterprise Admins,cn=Users"). Note that enabling a group in a cell does **not** automatically enable the users that are members of the group.

```
bool DisableGroup(ILikewiseGroup lwg)
```

This method disables a group in a cell. The *lwg* argument specifies an *ILikewiseGroup* object previously obtained with *GetGroups* or *GetGroup*.

ILikewiseUser

Properties:

```
string DisplayName (Read only)
```

Contains the displayable name of the user. This is actually the *displayName* attribute of the AD user object with which the *ILikewiseUser* object is associated.

```
object User (Read only)
```

Returns ADSI object of the AD *user* object with which this *ILikewiseUser* object is associated.

```
string SID (Read only)
```

Returns the SID of the AD user object with which this *ILikewiseUser* object is associated.

```
uint UID (Read/write)  
uint GID (Read/write)
```

These properties contain the UID and GID of the user as assigned within the cell from which this *ILikewiseUser* object was obtained.

```
string LoginName (Read/write)
```

This property contains the Linux/UNIX login name of the user as set within the cell from which this *ILikewiseUser* object was obtained. If this is left blank, the login name will be the same as the *sAMAccountName* of the user which this *ILikewiseUser* object is associated. If it is not blank, the *LoginName* is considered an "alias" for the user.

string **HomeDirectory** (*Read/write*)

Contains the home directory for the user as set within the cell from which this *ILikewiseUser* object was obtained. Note that this property must **not** contain any spaces (even if the user's name contains them).

string **LoginShell** (*Read/write*)

Contains the Linux/UNIX shell that will be started initially when the user logs on to a Linux/UNIX computer in the cell.

string **GECOS** (*Read/write*)

Contains the GECOS "comment" field associated with the user within the current cell.

Methods:

bool **CommitChanges** ()

Writes any modified property values out to Active Directory. If you've changed any property settings, you *must* call this method to make those changes permanent.

ILikewiseGroup

Properties:

string **DisplayName** (*Read only*)

Contains the displayable name of the group. This is actually the *displayName* attribute of the AD group object with which the *ILikewiseGroup* object is associated.

object **Group** (*Read only*)

Returns ADSI object of the AD *group* object with which this *ILikewiseGroup* object is associated.

string **SID** (*Read only*)

Returns the SID of the AD group object with which this *ILikewiseGroup* object is associated.

uint **GID** (*Read/write*)

This properties contains the GID of the group as assigned within the cell from which this *ILikewiseGroup* object was obtained.

string **GECOS** (Read/write)

Contains the GECOS "comment" field associated with the group within the current cell.

string **GroupAlias** (Read/write)

Contains the group alias of the group. Group aliases are when mapping an AD group to a different group name on UNIX/Linux.

Methods:

bool **CommitChanges** ()

Writes any modified property values out to Active Directory. If you've changed any property settings, you *must* call this method to make those changes permanent.

ABOUT LIKewise

Likewise® Software solutions improve management and interoperability of Windows, Linux, and UNIX systems with easy to use software for Linux administration and cross-platform identity management.

Likewise provides familiar Windows-based tools for system administrators to seamlessly integrate Linux and UNIX systems with Microsoft Active Directory. This enables companies running mixed networks to utilize existing Windows skills and resources, maximize the value of their Active Directory investment, strengthen the security of their network and lower the total cost of ownership of Linux servers.

Likewise Software is a Bellevue, WA-based software company funded by leading venture capital firms Ignition Partners, Intel Capital, and Trinity Ventures. Likewise has experienced management and engineering teams in place and is led by senior executives from leading technology companies such as Microsoft, F5 Networks, EMC and Mercury.