

CUAC Integration with Microsoft AD



Document ID: 118862

Contributed by Alok Singh, Cisco TAC Engineer.
Mar 30, 2015

Contents

Introduction

Prerequisites

Requirements

Components Used

Background Information

Integrate AD with CUAC and Import Users from AD

LDAP Functionality Between CUAC and AD

LDAP Process Summary

LDAP Process Details

Introduction

This document describes the way in which Lightweight Directory Access Protocol (LDAP) works between the Cisco Unified Attendant Console (CUAC) and the Microsoft Active Directory (AD) and the procedures that are used in order to integrate the two systems.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- CUCM
- CUAC
- LDAP
- AD

Components Used

The information in this document is based on the CUAC Version 10.x.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Background Information

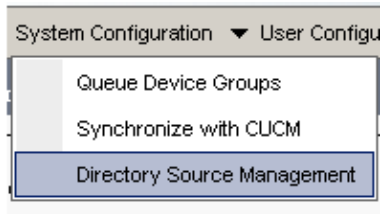
In earlier CUAC versions, the server obtains users directly from the Cisco Unified Communications Manager (CUCM) via predefined queries and filters. With the CUAC Premium Edition (CUACPE), administrators are allowed to integrate and import users directly from the AD. This grants flexibility to administrators for the implementation of attributes and filters of their own choice and requirements.

Note: The CUACPE has now been replaced with the CUAC Advanced Edition for Versions 10 and later.

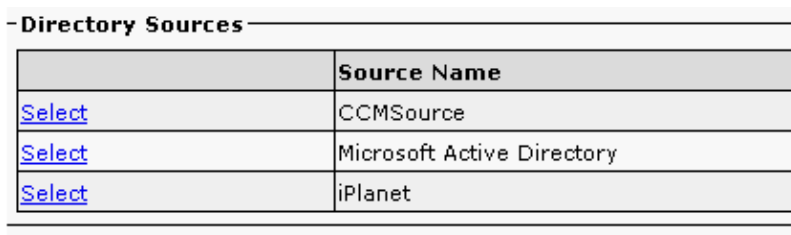
Integrate AD with CUAC and Import Users from AD

Complete these steps in order to integrate the CUAC with the AD and import users from the AD:

1. Enable Directory Synchronization for AD on the CUAC.

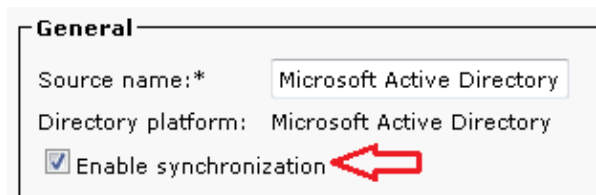


2. Select *Microsoft Active Directory* and check the *Enable synchronization* check box:



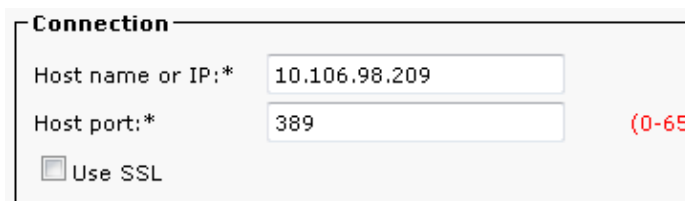
A screenshot of a table titled '- Directory Sources'. The table has two columns: 'Source Name' and an empty column. There are three rows, each with a 'Select' link in the empty column and a source name in the 'Source Name' column.

	Source Name
Select	CCMSource
Select	Microsoft Active Directory
Select	iPlanet



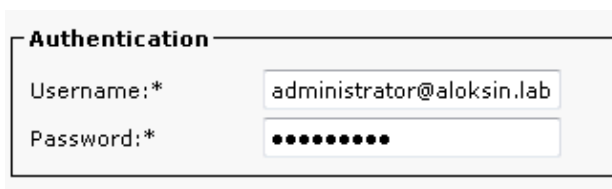
A screenshot of a configuration form titled 'General'. It contains three fields: 'Source name:*' with the value 'Microsoft Active Directory', 'Directory platform: Microsoft Active Directory', and a checked checkbox labeled 'Enable synchronization'. A red arrow points to the 'Enable synchronization' checkbox.

3. Input the configuration details for the Active Directory Server:



A screenshot of a configuration form titled 'Connection'. It contains three fields: 'Host name or IP:*' with the value '10.106.98.209', 'Host port:*' with the value '389' and a red '(0-65)' next to it, and a checkbox labeled 'Use SSL' which is unchecked.

For this example, *administrator@aloksin.lab* is used for authentication:



A screenshot of a configuration form titled 'Authentication'. It contains two fields: 'Username:*' with the value 'administrator@aloksin.lab' and 'Password:*' with a masked password represented by ten black dots.

4. In the Property Settings section, enter the configuration details for the Unique property, which appears once you enter the other details and click *Save*.

Property Settings

Unique property: ▼

Native property

Note: This is a unique value for each entry in the AD. If there are duplicate values, the CUAC pulls only one entry.

- In the Container section, enter the configuration details for the Base DN, which is the user search scope in the AD.

The *Object class* field is used by the AD in order to determine the requested search scope. By default, it is set to *contact*, which means that the AD looks for *contacts* (not users) in the requested search base. In order to import *users* on the CUAC, change the Object class setting to *user*:

- Container

Base DN:*

Object class:* (Case Sensitive)

Scope: ▼

- Save the settings, click **Directory Field mappings**, and configure all of the attributes that you would like to import for any user. Here is the configuration that is used in this example:

Source Fields	Destination Fields	Default
telephoneNumber	Extension	
mail	Email	
givenName	First Name	
sn	Last Name	


- Navigate to the Directory source page and click **Directory Rules**:

inner

Base DN:*

Object class:* (Case Sensitive)

Scope: ▼

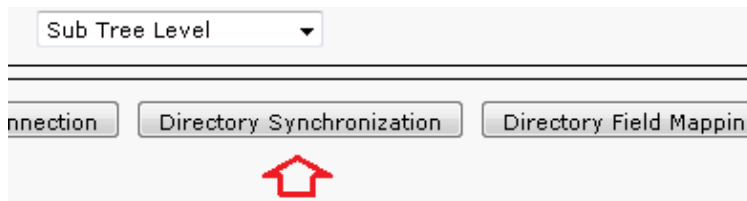


- Click **Add New** and create a rule. When you add a directory rule, a rule filter appears by default.

Field	Operator	Value
telephoneNumber	=	*

Note: There is no need to change the rule filter. It imports all of the users that have a telephone number configured.

- In order to configure auto-sync with the AD, click the **Directory Synchronization** tab.



10. The configuration is now complete. Navigate to **Engineering > Service Management** and restart the LDAP plugin in order to start the sync manually.

LDAP Functionality Between CUAC and AD

LDAP Process Summary

Here is a summary of the LDAP process between the CUAC and the AD:

1. A TCP session is established between the two servers (CUAC and AD).
2. The CUAC sends a BIND request to the AD and authenticates via the user that is configured in the Authentication settings.
3. Once the AD successfully authenticates the user, it sends a BIND Success notification to the CUACPE.
4. The CUAC sends a SEARCH request to the AD, which has the search scope information, filters for the search, and attributes for any filtered user.
5. The AD scans for the requested object (configured in the Object Class settings) in the search base. It filters out objects that match the criteria (filter) detailed in the SEARCH request message.
6. The AD responds to the CUAC with the search results.

Here is a sniffer capture that illustrates these steps:

3.208	10.106.98.209	TCP	49992 > ldap [SYN] Seq=0 win=8192 Len=0 MSS=1460 WS=
3.209	10.106.98.208	TCP	ldap > 49992 [SYN, ACK] Seq=0 Ack=1 win=8192 Len=0 M
3.208	10.106.98.209	TCP	49992 > ldap [ACK] Seq=1 Ack=1 win=65536 Len=0
3.208	10.106.98.209	LDAP	bindRequest(3) "administrator@aloksin.lab" simple
3.209	10.106.98.208	LDAP	bindResponse(3) success
3.208	10.106.98.209	LDAP	searchRequest(4) "dc=aloksin,dc=lab" wholeSubtree
3.209	10.106.98.208	LDAP	searchResEntry(4) "CN=Suhail Angi,CN=Users,DC=aloksif

LDAP Process Details

Once the configuration on the CUAC is completed and the LDAP plugin is restarted, the CUAC server sets up a TCP session with the AD.

The CUAC then sends a BIND request in order to authenticate with the AD server. If the authentication is successful, the AD sends a BIND Success response to the CUAC. With this, both servers attempt to set up a session on port 389 in order to sync users and their information.

Here is the configuration on the server that defines the Distinguished name, which is used for authentication in the BIND transaction:

Authentication	
Username:*	administrator@aloksin.lab
Password:*	●●●●●●●●

These messages appear in the packet captures:

- Here is the TCP handshake, followed by the BIND request:

98.208	10.106.98.209	TCP	50190 > ldap [SYN] Seq=0 win=8192 Len=0 MSS=1460 WS=8
98.209	10.106.98.208	TCP	ldap > 50190 [SYN, ACK] Seq=0 Ack=1 win=8192 Len=0 MS
98.208	10.106.98.209	TCP	50190 > ldap [ACK] Seq=1 Ack=1 win=65536 Len=0
98.208	10.106.98.209	LDAP	bindRequest(3) "administrator@aloksin.lab" simple
98.209	10.106.98.208	LDAP	bindResponse(3) success

- Here is the expansion of the BIND request:

```

Lightweight Directory Access Protocol
  LDAPMessage bindRequest(3) "administrator@aloksin.lab" simple
    messageID: 3
    protocolOp: bindRequest (0)
      bindRequest
        version: 3
        name: administrator@aloksin.lab
        authentication: simple (0)
          simple: 633173633031323321
          [Response To: 81]
  
```

- Here is the expansion of BIND response, which indicates successful authentication of the user (*administrator* in this example):

```

Lightweight Directory Access Protocol
  LDAPMessage bindResponse(3) success
    messageID: 3
    protocolOp: bindResponse (1)
      bindResponse
        resultCode: success (0)
        matchedDN:
        errorMessage:
        [Response To: 80]
        [Time: 0.002073000 seconds]
  
```

Upon a successful bind, the server sends a SEARCH request to the AD in order to import users. This SEARCH request contains the filter and attributes that are used by the AD. The AD then searches for users within the defined search base (as detailed in the SEARCH request message), which fulfills the criteria in the filter and the attributes verification.

Here is an example of the SEARCH request that is sent by the CUCM:

```

Lightweight Directory Access Protocol
  LDAPMessage searchRequest(2) "dc=aloksin,dc=lab" wholeSubtree
    messageID: 2
    protocolOp: searchRequest (3)
      searchRequest
        baseObject: dc=aloksin,dc=lab
        scope: wholeSubtree (2)
        derefAliases: derefAlways (3)
  
```

```

sizeLimit: 0
timeLimit: 0
typesOnly: False
Filter: (&&(objectclass=user)(!(objectclass=Computer)))
  (!(UserAccountControl:1.2.840.113556.1.4.803:=2))
    filter: and (0)
      and: (&&(objectclass=user)(!(objectclass=Computer)))
        (!(UserAccountControl:1.2.840.113556.1.4.803:=2))
          and: 3 items
            Filter: (objectclass=user)
              and item: equalityMatch (3)
                equalityMatch
                  attributeDesc: objectclass
                  assertionValue: user
            Filter: (!(objectclass=Computer))
              and item: not (2)
                Filter: (objectclass=Computer)
                  not: equalityMatch (3)
                    equalityMatch
                      attributeDesc: objectclass
                      assertionValue: Computer
            Filter: (!(UserAccountControl:1.2.840.113556.1.4.
803:=2))
              and item: not (2)
                Filter: (UserAccountControl:1.2.840.113556
.1.4.803:=2)
                  not: extensibleMatch (9)
                    extensibleMatch UserAccountControl
                      matchingRule: 1.2.840.113556.
1.4.803
                      type: UserAccountControl
                      matchValue: 2
                      dnAttributes: False
attributes: 15 items
  AttributeDescription: objectguid
  AttributeDescription: samaccountname
  AttributeDescription: givenname
  AttributeDescription: middlename
  AttributeDescription: sn
  AttributeDescription: manager
  AttributeDescription: department
  AttributeDescription: telephonenumber
  AttributeDescription: mail
  AttributeDescription: title
  AttributeDescription: homephone
  AttributeDescription: mobile
  AttributeDescription: pager
  AttributeDescription: msrtcslip-primaryuseraddress
  AttributeDescription: msrtcslip-primaryuseraddress

```

[Response In: 103]

controls: 1 item

Control

```

controlType: 1.2.840.113556.1.4.319 (pagedResultsControl)
criticality: True
SearchControlValue
  size: 250
  cookie: <MISSING>

```

When the AD receives this request from the CUCM, it searches for users in the **baseObject: dc=aloksin,dc=lab**, which satisfies the filter. Any user who does not fulfill the requirements that are detailed by the filter is left out. The AD responds to the CUCM with all of the filtered users and sends the values for the requested attributes.

Note: Objects cannot be imported. Only *users* are imported. This is because the filter that is sent in the

SEARCH request message includes *objectclass=user*. Hence, the AD searches only for users, not contacts. The CUACM has all of these mappings and a filter by default.

The CUAC is not configured by default; there are no mapping details configured in order to import attributes for users, so you must input these details manually. In order to create these mappings, navigate to **System Configuration > Directory Source Management > Active Directory > Directory Field Mapping**.

Administrators are allowed to map fields per their own requirements. Here is an example:

Directory Source				
Microsoft Active Directory				
Field Mappings				
		Source Fields	Destination Fields	Default Value
<input type="checkbox"/>	Select	telephoneNumber	Extension	
<input type="checkbox"/>	Select	mail	Email	
<input type="checkbox"/>	Select	givenName	First Name	
<input type="checkbox"/>	Select	sn	Last Name	

The Source Field information is sent to the AD in the SEARCH request message. When the AD sends the SEARCH response message, these values are stored in the Destination Fields on the CUACPE.

Note that the CUAC by default has the Object Class set to *contacts*. If this default setting is used, the filter that is sent to the AD appears as shown here:

```
Filter: (&(&(objectclass=contact)( .....))
```

With this filter, the AD never returns any users to the CUACPE, since it searches for *contacts* in the search base, not *users*. For this reason, you must change Object Class to *user*:

Container

Base DN:*

Object class:* (Case Sensitive)

Scope:

Up to this point, these settings have been configured on the CUAC:

- Connections details
- Authentication (distinguished user for binding)
- Container settings
- Directory mapping

In this example, the Unique property is configured as *sAMAccountName*. If you restart the LDAP plugin on the CUAC and check the SEARCH request message, it contains no attributes or filter except the *ObjectClass=user*:

```
Lightweight Directory Access Protocol
LDAPMessage searchRequest(224) "dc=aloksin,dc=lab" wholeSubtree
messageID: 224
protocolOp: searchRequest (3)
searchRequest
baseObject: dc=aloksin,dc=lab
scope: wholeSubtree (2)
derefAliases: neverDerefAliases (0)
sizeLimit: 1
timeLimit: 0
typesOnly: True
```

```

Filter: (ObjectClass=user)
  filter: equalityMatch (3)
    equalityMatch
      attributeDesc: ObjectClass
      assertionValue: user
  attributes: 0 items
[Response In: 43]

```

Note that the Directory rule is missing here. In order to sync the contacts with the AD, you must create a rule. By default, there is no Directory rule configured. As soon as one is created, a filter is already present. There is no need to change the filter, as you must import all of the users that have a telephone number.

Field	Operator	Value
telephoneNumber	=	*

Restart the LDAP plugin in order to initiate a sync with the AD and import the users. Here is the SEARCH request from the CUAC:

```

Lightweight Directory Access Protocol
  LDAPMessage searchRequest(4) "dc=aloksin,dc=lab" wholeSubtree
    messageID: 4
    protocolOp: searchRequest (3)
      searchRequest
        baseObject: dc=aloksin,dc=lab
        scope: wholeSubtree (2)
        derefAliases: neverDerefAliases (0)
        sizeLimit: 0
        timeLimit: 15
        typesOnly: False
        Filter: (&(&(objectclass=user)(telephoneNumber=*)
          (!(UserAccountControl:1.2.840.113556.1.4.803:=2))))
          filter: and (0)
            and: (&(&(objectclass=user)(telephoneNumber=*))
              (!(UserAccountControl:1.2.840.113556.1.4.803:=2)))
              and: 3 items
                Filter: (objectclass=user)
                  and item: equalityMatch (3)
                    equalityMatch
                      attributeDesc: objectclass
                      assertionValue: user
                Filter: (telephoneNumber=*)
                  and item: present (7)
                    present: telephoneNumber
                Filter: (!(UserAccountControl:1.2.840.113556.
                  1.4.803:=2))
                  and item: not (2)
                    Filter: (UserAccountControl:1.2.840.113556.
                      1.4.803:=2)
                      not: extensibleMatch (9)
                        extensibleMatch UserAccountControl
                          matchingRule: 1.2.840.113556.1.
                            4.803
                          type: UserAccountControl
                          matchValue: 2
                          dnAttributes: False
          attributes: 10 items
            AttributeDescription: TELEPHONENUMBER
            AttributeDescription: MAIL
            AttributeDescription: GIVENNAME
            AttributeDescription: SN
            AttributeDescription: SAMAccountName
            AttributeDescription: ObjectClass
            AttributeDescription: whenCreated
            AttributeDescription: whenChanged

```



```

AttributeDescription: uSNCreated
AttributeDescription: uSNChanged
[Response In: 11405]
controls: 1 item
Control
controlType: 1.2.840.113556.1.4.319 (pagedResultsControl)
SearchControlValue
size: 500
cookie: <MISSING>

```

If the AD finds users that match the criteria detailed in the SEARCH request message, then it sends a *SearchResEntry* message that contains the user information.

```

8.208 10.106.98.209 TCP 49992 > 1dap [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=8 SACK_PERM=1
8.209 10.106.98.208 TCP 1dap > 49992 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=8 SACK_PERM=1
8.208 10.106.98.209 TCP 49992 > 1dap [ACK] Seq=1 Ack=1 Win=65536 Len=0
8.208 10.106.98.209 LDAP bindRequest(3) 'administrator@aloksin.lab' simple
8.209 10.106.98.208 LDAP bindResponse(3) success
8.208 10.106.98.209 LDAP searchRequest(4) 'dc=aloksin,dc=lab' wholesubtree
8.209 10.106.98.208 LDAP searchResEntry(4) "CN=Suhail Angi,CN=Users,DC=aloksin,DC=lab" | searchResEntry(4) "CN=Pra
8.209 10.106.98.208 LDAP searchResRef(4)
8.208 10.106.98.209 TCP 49992 > 1dap [ACK] Seq=358 Ack=3555 Win=65536 Len=0

```

Here is the SearchResEntry message:

Lightweight Directory Access Protocol

LDAPMessage searchResEntry(4) "**CN=Suhail Angi,CN=Users,DC=aloksin,DC=lab**" [4 results]

messageID: 4

protocolOp: searchResEntry (4)

searchResEntry

objectName: CN=Suhail Angi,CN=Users,DC=aloksin,DC=lab

attributes: 9 items

PartialAttributeList item objectClass

type: objectClass

vals: 4 items

top

person

organizationalPerson

user

PartialAttributeList item **sn**

type: sn

vals: 1 item

Angi

PartialAttributeList item **telephoneNumber**

type: telephoneNumber

vals: 1 item

1002

PartialAttributeList item **givenName**

type: givenName

vals: 1 item

Suhail

PartialAttributeList item **whenCreated**

type: whenCreated

vals: 1 item

20131222000850.0Z

PartialAttributeList item **whenChanged**

type: whenChanged

vals: 1 item

20131222023413.0Z

PartialAttributeList item **uSNCreated**

type: uSNCreated

vals: 1 item

12802

PartialAttributeList item **uSNChanged**

type: uSNChanged

vals: 1 item

12843

```

PartialAttributeList item sAMAccountName
  type: sAMAccountName
  vals: 1 item
    sangi
[Response To: 11404]
[Time: 0.001565000 seconds]
Lightweight Directory Access Protocol
LDAPMessage searchResEntry(4) "CN=Pragathi NS,CN=Users,DC=aloksin,DC=lab" [5 results]
messageID: 4
protocolOp: searchResEntry (4)
  searchResEntry
    objectName: CN=Pragathi NS,CN=Users,DC=aloksin,DC=lab
    attributes: 9 items
      PartialAttributeList item objectClass
        type: objectClass
        vals: 4 items
          top
          person
          organizationalPerson
          user
      PartialAttributeList item sn
        type: sn
        vals: 1 item
          NS
      PartialAttributeList item telephoneNumber
        type: telephoneNumber
        vals: 1 item
          1000
          .....
          ...{message truncated}.....
          .....

```

Note: There is no MAIL in the response, even though this attribute is requested. This is because the MAIL ID was not configured for users on the AD.

Once these values are received by the CUAC, it stores them in the Structured Query Language (SQL) table. You can then log into the console, and the console fetches the users list from this SQL table on CUACPE server.