Intégration CUAC avec l'AD de Microsoft

Contenu

Introduction

Ce document décrit la manière dans laquelle le Protocole LDAP (Lightweight Directory Access Protocol) fonctionne entre la console de réception de Cisco Unified (CUAC) et la Microsoft Active Directory (AD) et les procédures qui sont utilisées afin d'intégrer les deux systèmes.

Conditions préalables

Conditions requises

Cisco vous recommande de prendre connaissance des rubriques suivantes :

- CUCM
- CUAC
- LDAP
- AD

Composants utilisés

Les informations dans ce document sont basées sur la version 10.x CUAC.

Les informations contenues dans ce document ont été créées à partir des périphériques d'un environnement de laboratoire spécifique. Tous les périphériques utilisés dans ce document ont démarré avec une configuration effacée (par défaut). Si votre réseau est opérationnel, assurez-vous que vous comprenez l'effet potentiel de toute commande.

Informations générales

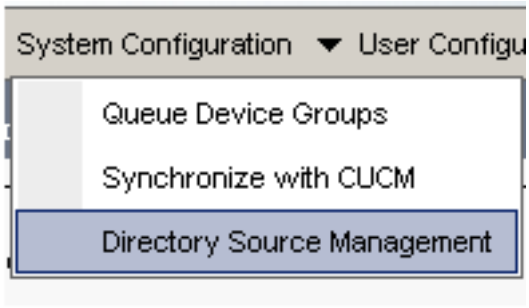
Dans des versions plus tôt CUAC, le serveur obtient des utilisateurs directement de Cisco Unified Communications Manager (CUCM) par l'intermédiaire des requêtes et des filtres de prédéfinis. Avec l'édition premium CUAC (CUACPE), on permet à des des administrateurs pour intégrer et importer des utilisateurs directement de l'AD. Ceci accorde la flexibilité aux administrateurs pour l'implémentation des attributs et des filtres de leur propres choix et conditions requises.

> Remarque: Le CUACPE a été maintenant remplacé par l'édition avancée par CUAC pour des versions 10 et ultérieures.

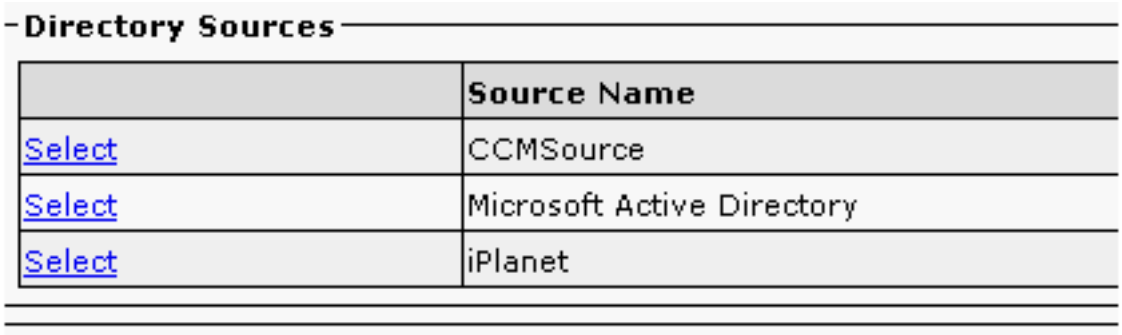
Intégrez l'AD avec CUAC et importez les utilisateurs de l'AD

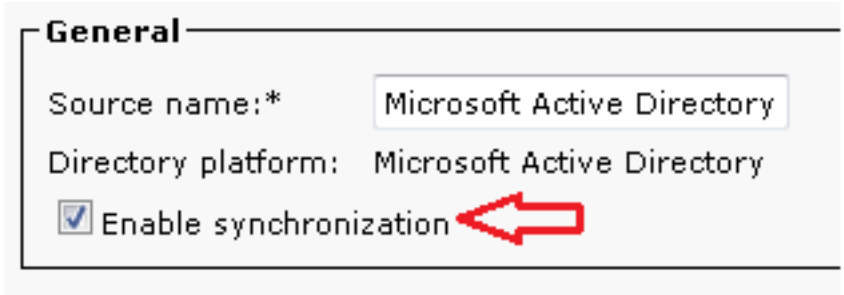
Terminez-vous ces étapes afin d'intégrer le CUAC avec l'AD et importer des utilisateurs de l'AD :

1. Synchronisation de répertoire d'enable pour l'AD sur le CUAC.

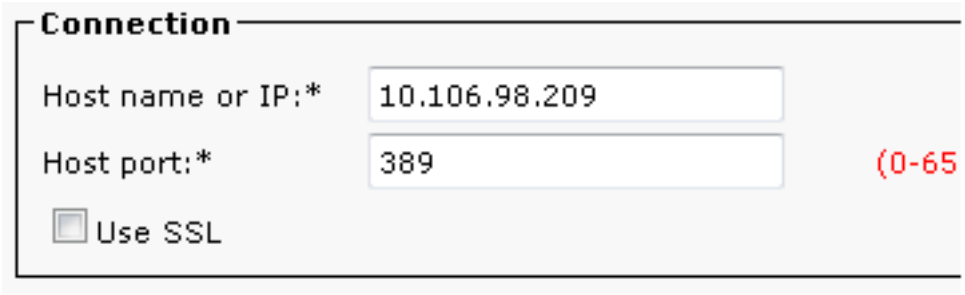


2. **La Microsoft Active Directory** choisie et cochent la case de **synchronisation d'enable** :

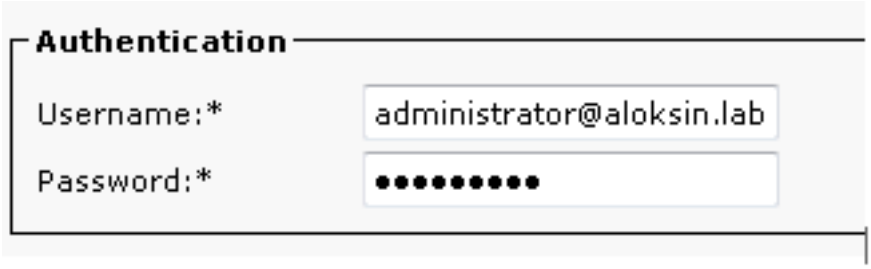




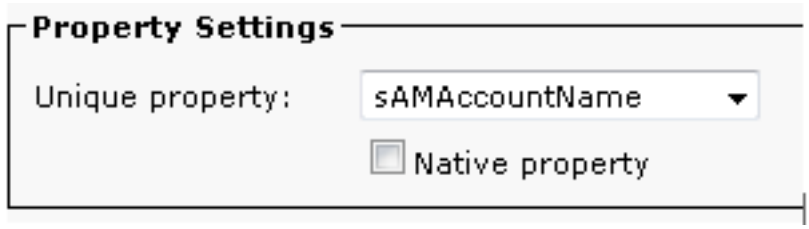
3. Entrez les détails de configuration pour le serveur de Répertoire actif :



Pour cet exemple, **administrator@aloksin.lab** est utilisé pour l'authentification :



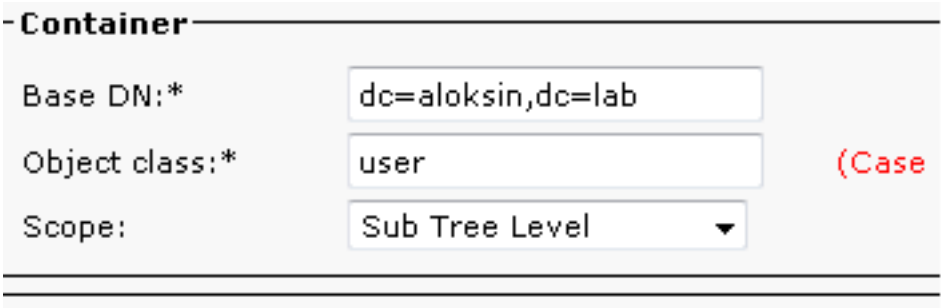
4. Dans la propriété les configurations sectionnent, écrivent les détails de configuration pour la seule propriété, qui apparaît une fois vous écrit les autres détails et clique sur la **sauvegarde**.



Remarque: C'est une seule valeur pour chaque entrée dans l'AD. S'il y a des valeurs en double, le CUAC tire seulement une entrée.

5. Dans la section de conteneur, écrivez les détails de configuration pour le DN de base, qui est la portée de recherche d'utilisateur dans l'AD.

   Le champ de *classe d'objets* est utilisé par l'AD afin de déterminer la portée demandée de recherche. Par défaut, il est placé *pour entrer en contact*, ainsi il signifie que l'AD recherche des *contacts* (pas utilisateurs) dans la base demandée de recherche. Afin d'importer des *utilisateurs* sur le CUAC, changez la classe d'objets plaçant à l'**utilisateur** :

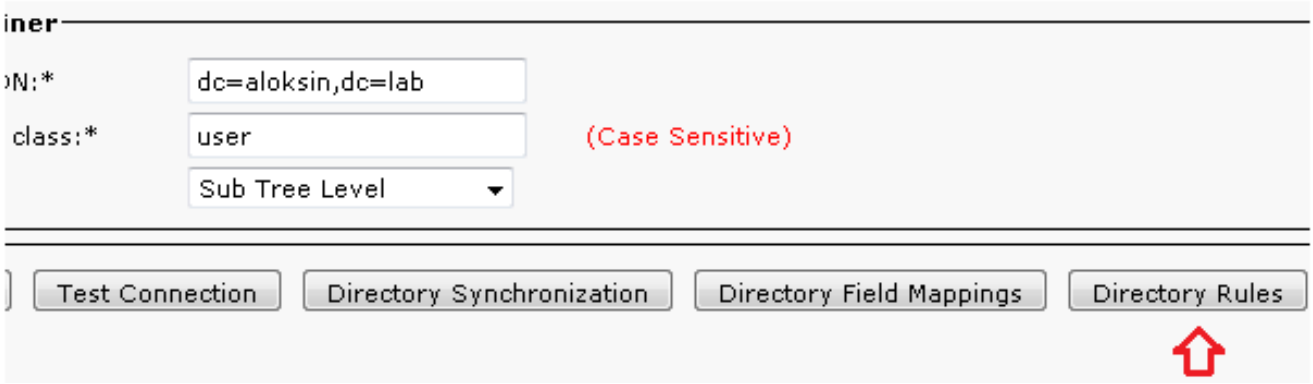


6. Sauvegardez les configurations, cliquez sur les **mappages de champ de répertoire**, et configurez tous les attributs que vous voudriez importer pour n'importe quel utilisateur. Voici

la configuration qui est utilisée dans cet exemple :

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

7. Naviguez vers la page de source de répertoire et cliquez sur les **règles de répertoire** :



8. Cliquez sur Add **nouveau** et créez une règle. Quand vous ajoutez une règle de répertoire, un filtre de règle apparaît par défaut.

Field	Operator	Value
telephoneNumber	=	*

Remarque: Il n'y a aucun besoin de changer le filtre de règle. Il importe tous les utilisateurs qui font configurer un numéro de téléphone.

9. Afin de configurer l'auto-sync avec l'AD, cliquez sur l'onglet de **synchronisation de répertoire**.

10. La configuration est maintenant complète. Naviguez vers la **construction > gestion des services** et redémarrez le module d'extension de LDAP afin de commencer le sync manuellement.

Fonctionnalité de LDAP entre CUAC et AD

Résumé de processus de LDAP

Voici un résumé du processus de LDAP entre le CUAC et l'AD :

1. Une session TCP est établie entre les deux serveurs (CUAC et AD).

2. Le CUAC envoie une demande de GRIPPAGE à l'AD et l'authentifie par l'intermédiaire de l'utilisateur qui est configuré dans les configurations d'authentification.

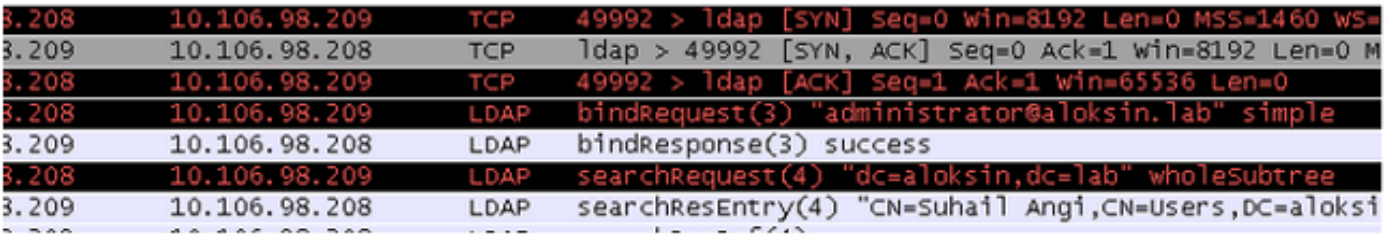
3. Une fois que l'AD authentifie avec succès l'utilisateur, il envoie une notification de succès de GRIPPAGE au CUACPE.

4. Le CUAC envoie une demande de RECHERCHE à l'AD, qui a les informations de portée de recherche, des filtres pour la recherche, et les attribue pour n'importe quel utilisateur filtré.

5. Les balayages d'AD pour l'objet demandé (configuré dans les configurations de classe d'objets) dans la base de recherche. Il filtre les objets qui apparient les critères (filtre) détaillés dans le message de demande de RECHERCHE.

6. L'AD répond au CUAC avec les résultats de la recherche.

Voici une capture de renifleur qui illustre ces étapes :

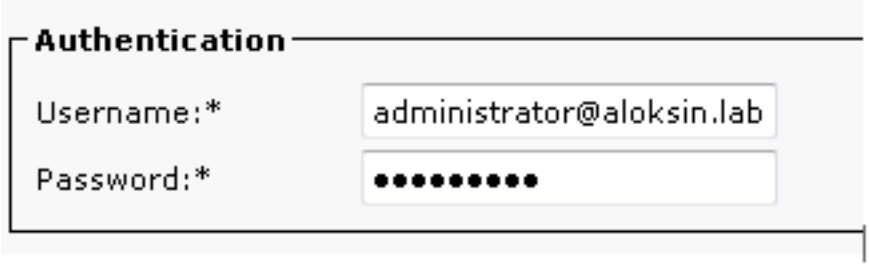


Détails du processus de LDAP

Une fois la configuration sur le CUAC est terminée et le module d'extension de LDAP est redémarré, le serveur CUAC a installé une session TCP avec l'AD.

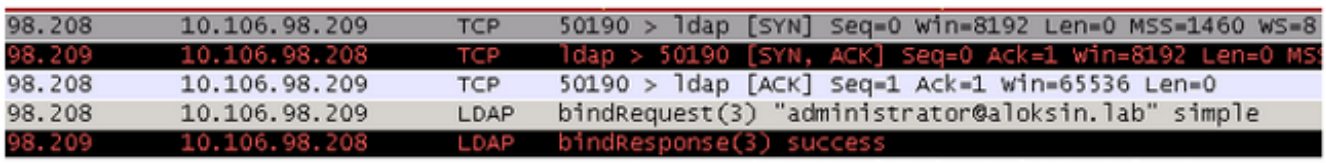
Le CUAC envoie alors une demande de GRIPPAGE afin d'authentifier avec le serveur d'AD. Si l'authentification est réussie, l'AD envoie une réponse de succès de GRIPPAGE au CUAC. Avec ceci, les deux serveurs tentent d'installer une session sur des utilisateurs de sync du port 389 et leurs informations.

Voici la configuration sur le serveur qui définit le nom unique, qui est utilisé pour l'authentification dans la transaction de GRIPPAGE :

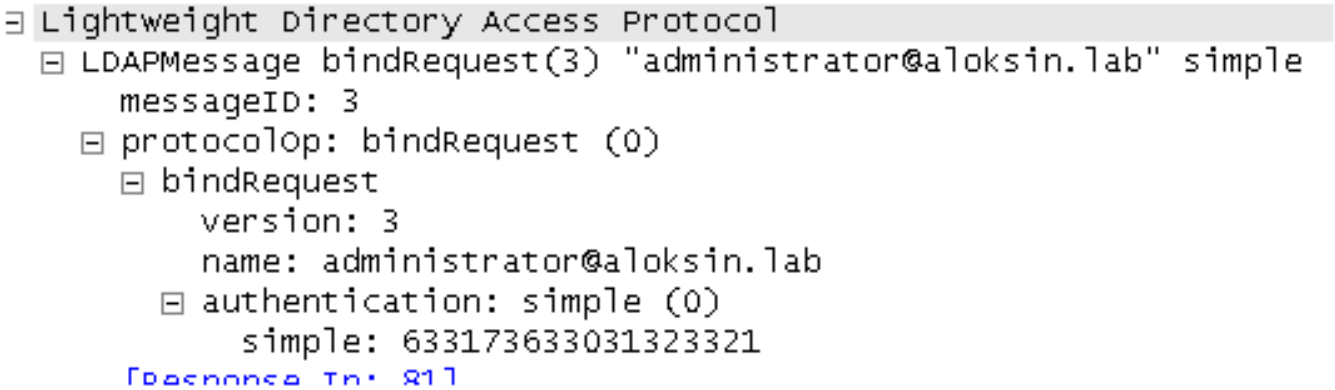


Ces messages apparaissent dans les captures de paquet :

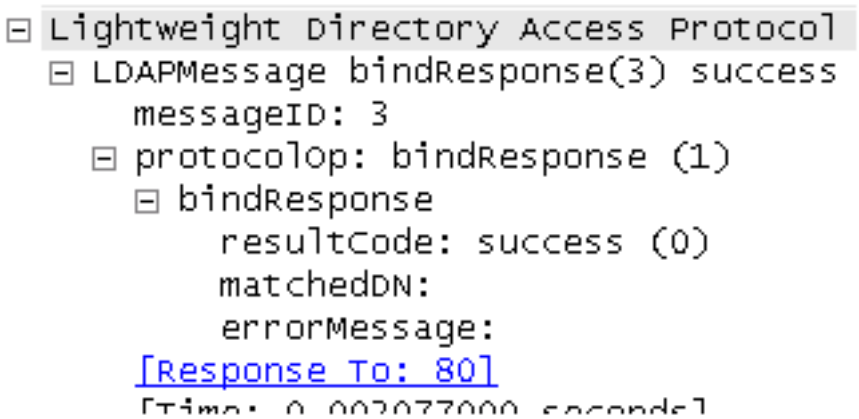
- Voici la prise de contact de TCP, suivie de la demande de GRIPPAGE :



- Voici l'extension de la demande de GRIPPAGE :



- Voici l'extension de la réponse de GRIPPAGE, qui indique l'authentification réussie de l'utilisateur (**administrateur** dans cet exemple) :



Sur un grippage réussi, le serveur envoie une demande de RECHERCHE à l'AD afin d'importer des utilisateurs. Cette demande de RECHERCHE contient le filtre et les attributs qui sont utilisés par l'AD. L'AD recherche alors des utilisateurs dans la base définie de recherche (comme détaillé dans le message de demande de RECHERCHE), qui remplit les critères dans le filtre et la vérification d'attributs.

Voici un exemple de la demande de RECHERCHE qui est envoyée par le 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: msrtcsip-primaryuseraddress
                AttributeDescription: msrtcsip-primaryuseraddress
        [Response In: 103]
        controls: 1 item
            Control
                controlType: 1.2.840.113556.1.4.319 (pagedResultsControl)
                criticality: True
                SearchControlValue
                    size: 250
                    cookie: <MISSING>
```

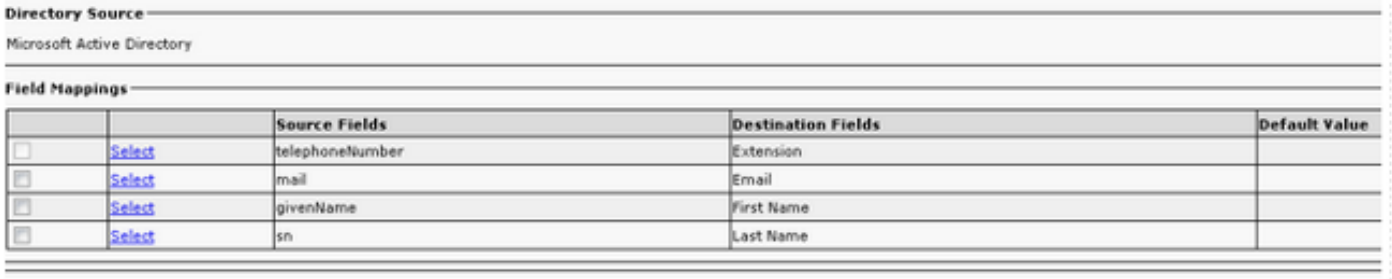
Quand l'AD reçoit cette demande du CUCM, il recherche des utilisateurs dans le **baseObject : dc=aloksin, dc=lab**, qui satisfait le filtre. N'importe quel utilisateur qui ne remplit pas les conditions qui sont détaillées par le filtre est laissé. L'AD répond au CUCM avec tous les utilisateurs filtrés et envoie les valeurs pour les attributs demandés.

> Remarque: Des objets ne peuvent pas être importés. Seulement des *utilisateurs* sont importés. C'est parce que le filtre qui est introduit le message de demande de RECHERCHE

inclut l'**objectclass=user**. Par conséquent, les recherches d'AD seulement des utilisateurs, pas contacts. Le CUCM a tous ces mappages et filtre par défaut.

Le CUAC n'est pas configuré par défaut ; il n'y a aucun mappage détaille configuré afin d'importer des attributs pour des utilisateurs, ainsi vous devez entrer ces détails manuellement. Afin de créer ces mappages, naviguez vers la **Gestion de configuration système > de source de répertoire > le mappage de champ de Répertoire actif > de répertoire**.

On permet à des des administrateurs pour tracer des champs par leurs propres conditions requises. Voici un exemple :

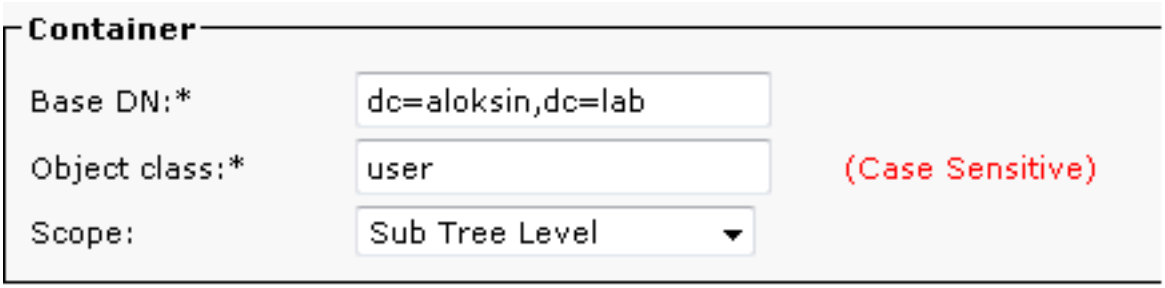


Les informations de champ de source sont envoyées à l'AD dans le message de demande de RECHERCHE. Quand l'AD envoie le message de réponse de RECHERCHE, ces valeurs sont enregistrées dans les champs de destination sur le CUACPE.

Notez que le CUAC par défaut a la classe d'objets réglée aux *contacts*. Si cette valeur par défaut est utilisée, le filtre qui est envoyé à l'AD apparaît comme affiché ici :

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

Avec ce filtre, l'AD ne retourne jamais aucun utilisateur au CUACPE, puisqu'il recherche des *contacts* dans la base de recherche, pas des *utilisateurs*. Pour cette raison, vous devez changer la classe d'objets à l'**utilisateur** :



Jusqu'à ce point, ces configurations ont été configurées sur le CUAC :

- Détails de connexions
- Authentification (utilisateur distingué pour lier)
- Configurations de conteneur
- Mappage de répertoire

Dans cet exemple, la seule propriété est configurée comme **sAMAccountName**. Si vous redémarrez le module d'extension de LDAP sur le CUAC et vérifiez le message de demande de RECHERCHE, il ne contient aucun attribut ou filtre excepté l'**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]
```

Notez que la règle de répertoire manque ici. Sync les contacts avec l'AD, vous devez créer une règle. Par défaut, il n'y a aucune règle de répertoire configurée. Dès qu'un sera créé, un filtre est déjà présent. Il n'y a aucun besoin de changer le filtre, comme vous devez importer tous les utilisateurs qui ont un numéro de téléphone.

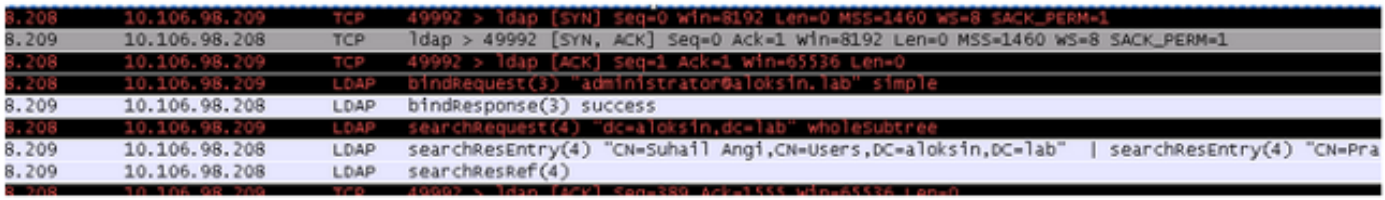
Field	Operator	Val
telephoneNumber	=	*

Redémarrez le module d'extension de LDAP afin d'initier un sync avec l'AD et importer les utilisateurs. Voici la demande de RECHERCHE du 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>
```

Si l'AD le trouve les utilisateurs qui apparient les critères détaillés dans la RECHERCHE demandent le message, alors lui envoie un message de *SearchResEntry* qui contient les informations utilisateur.



Voici le message de SearchResEntry :

```
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}.........
                           .....
```

Remarque: Il n'y a aucune MESSAGERIE dans la réponse, quoique cet attribut soit demandé. C'est parce que l'ID de MESSAGERIE n'a pas été configuré pour des utilisateurs sur l'AD.

Une fois que ces valeurs sont reçues par le CUAC, il les enregistre dans la table du SQL (SQL). Vous pouvez alors se connecter dans la console, et la console cherche la liste utilisateurs de cette table SQL sur le serveur CUACPE.