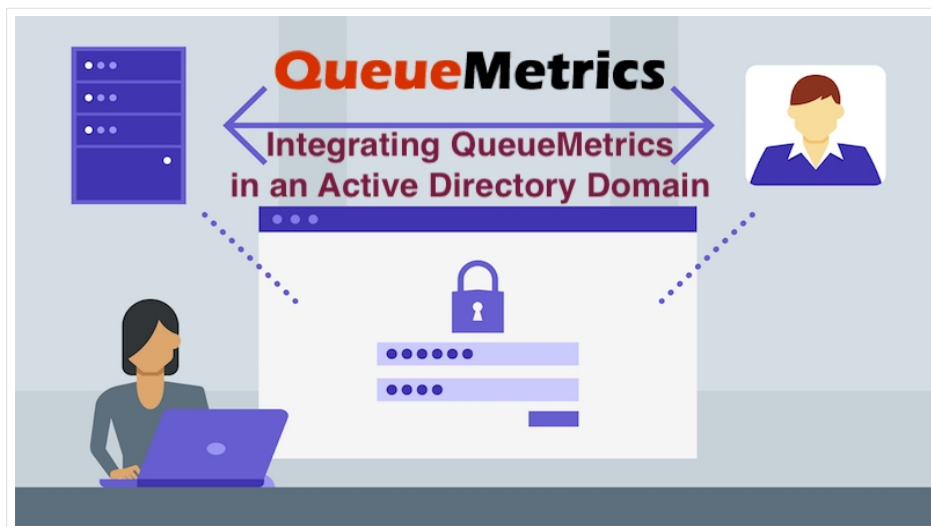


Integrating QueueMetrics in an Active Directory Domain



Active Directory (AD) is a directory service developed by Microsoft for Windows domain networks. It is based on the LDAP protocol and is widely used to provide centralized management of users in large organizations, from centralized password storage to provisioning of all company services.

By using Active Directory or similar solutions based on LDAP (e.g. OpenLDAP, Apache Directory Services and many others) your company can manage all users across all services and all computers in a relatively straightforward way - you do not need to manage credentials and access for each user in every application you run.

QueueMetrics can integrate with Active Directory and any other LDAP server by leveraging its LDAP APIs to verify user credentials.

Getting started: credentials and terminology

The first thing you have to do is to make sure that you have a valid set of credentials on your hand. This is something you have to get from your Active Directory system administrator.

You need:

- A LDAP URI to connect to your Domain Controller, e.g. `ldap:ad.example.com:389`
- A LDAP Distinguished Name and its password. A DN is your “login” and usually has either a long form like `CN=Peter Parker,CN=Users,DC=example,DC=com` or (in Active Directory systems) can be the user’s e-mail address. In LDAP terminology, logging on to the remote LDAP server is called binding.

- An LDAP “base”, that is the point in the LDAP tree we should start searching. In ActiveDirectory systems it looks something like `CN=Users,DC=example,DC=com`

At this point you should install a small tool called `ldapsearch`; on Linux you can get it by issuing:

```
yum install -y openldap-clients
```

When done, you can test that everything is correct by running the following command (replace URI, DN, password and base as appropriate):

```
ldapsearch -H ldap://ad.example.com:389 -x -w 'Spiderman' -D "pparker@example.com" -b "CN=Users,DC=example,DC=com"
```

If you get a (possibly long) list of objects printed out, your credentials are valid and you can see users in your company’s directory.

```
# extended LDIF
#
# LDAPv3
# base <CN=Users,DC=example,DC=com> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#
...many elements skipped....

# Guest, Users, example.com
dn: CN=Guest,CN=Users,DC=example,DC=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
cn: Guest
description: Built-in account for guest access to the computer/domain
```

distinguishedName: CN=Guest,CN=Users,DC=example,DC=com
instanceType: 4
whenCreated: 20210812150455.0Z
whenChanged: 20210812150455.0Z
uSNCreated: 8197
memberOf: CN=Guests,CN=Builtin,DC=example,DC=com
name: Guest
objectGUID:: P5yymq0Cwkm4w7/5otuyMQ==
userAccountControl: 66082
badPwdCount: 0
codePage: 0
countryCode: 0
badPasswordTime: 0
lastLogoff: 0
lastLogon: 0
pwdLastSet: 0
primaryGroupID: 514
objectSid:: AQUAAAAAAAAUAAAAeigMKVkbGFLp3j0A9QEAAA==
accountExpires: 9223372036854775807
logonCount: 0
sAMAccountName: Guest
sAMAccountType: 805306368
objectCategory: CN=Person,CN=Schema,CN=Configuration,DC=example,DC=com
isCriticalSystemObject: TRUE
dSCorePropagationData: 20210812150617.0Z
dSCorePropagationData: 16010101000001.0Z

search result

```
search: 2  
result: 0 Success  
  
# numResponses: 29  
# numEntries: 28
```

See how each entry has a lot of attributes; you can use them in queries to retrieve the exact directory entries you are looking for. If you were able to make it so far, this means that your credentials are valid and you can start configuring QueueMetrics to access LDAP.

Using Active Directory from QueueMetrics

Generally speaking, you need a local QueueMetrics account for each user in AD. Therefore, you have to create “normal” accounts, enable them, set them to the correct classes etc, and then set a random password so they cannot be used.

In the simplest case, you just want QueueMetrics to take the user supplied login and password, try and use them to “bind” to the server, and if the server allows a successful binding (that is, if credentials are correct), load the user that is defined in QM with the same login.

If the user fails AD verification, you can tell QM that the login should be rejected outright (and that’s the default) or to attempt a “local” login. This is called a “delegated” authentication because the LDAP server actually delegates QM to do its own job, and it is useful so you don’t have to define each and every user in LDAP for them to work.

To make such a scenario work, you just add the following lines to your configuration.properties:

```
auth.externalSource=ldap  
auth.verboseLog=false  
  
auth.ldapServerUrl=ldap://ad.example.com:389  
auth.ldapBind=cn=${login},dc=example,dc=com  
auth.ldapFailureDelegates=true
```

Now you log in with one of your users, and it should work as expected. Note how values, e.g. the `ldapBind`, are built by replacing placeholders with the values your user entered.

Using different log-ins

Sometimes, you need to use different users from the ones your users physically input to log in. This is quite common, as sadly ActiveDirectory systems are company-wide entities, and often you have no control on the format of log-ins that are created there.

QM is rather picky in terms of how an agent’s login is supposed to look like, so there is a need to tell QM to use a different login. Luckily, in Active Directory, the default user schema has a set of empty Extension Attributes that

may be set freely for such purposes. They will appear in LDAP with a name like `msDS-cloudExtensionAttribute7`. In one of them, you should specify the QM login to use, and this for each user that need to access QM.

So, let's say that you want a user to log in with their company's ID (eg `pparker`). We will first add a domain part so that the user is actually `pparker@example.com`, therefore forming a valid value for binding. Once the bind is successful, we go look for a user that has an account name matching the login entered, and when we find it, we read their Extension Attribute number 7 to determine the actual login that QM will use.

So, if the user appears like this in AD:

```
# Peter Parker, Users, example.com
dn: CN=Peter Parker,CN=Users,DC=example,DC=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
cn: Peter Parker
sn: Parker
givenName: Peter
distinguishedName: CN=Peter Parker,CN=Users,DC=example,DC=com
displayName: Peter Parker
name: Peter Parker
sAMAccountName: pparker
userPrincipalName: pparker@example.com
objectCategory: CN=Person,CN=Schema,CN=Configuration,DC=example,DC=com
msDS-cloudExtensionAttribute7: agent/101
mail: pparker@example.com
```

We can use the following configuration in QM:

```
auth.externalSource=ldap
auth.verboseLog=false
```

```
auth.IdapServerUrl=ldap://ad.example.com:389  
auth.IdapBind=${login}@example.com  
auth.IdapFailureDelegates=true  
  
auth.IdapLoginAttr=msDS-cloudExtensionAttribute7  
auth.IdapBase=CN=Users,dc=example,dc=com  
auth.IdapQuery=(sAMAccountName=${login})
```

At this point, you can log-in with `pparker` and QM will open up the page for `agent/101`.

It's not working - what can I do?

Make sure you set `auth.verboseLog=true` and try logging in, while checking the file `queuemetrics-(date).log`.

If you see entries like this:

```
it.loway.tpf.transaction.user.AuthOverLDAP.authOverRpc LDAP: Binding 'pparker@example.com' on  
'ldap://ad.example.com:389'  
  
it.loway.tpf.transaction.user.AuthOverLDAP.authOverRpc LDAP auth failed for 'pparker'  
  
    javax.naming.AuthenticationException: [LDAP: error code 49 - 80090308: LdapErr: DSID-  
0C090439, comment: AcceptSecurityContext error, data 52e, v4563]
```

This means that you got a wrong password. Check it with `ldapsearch`. Note that when you use delegate authentication, LDAP is always checked first, so you'll get an error before it loads the local user.

When all goes well, you'll see a log like:

```
it.loway.tpf.transaction.user.AuthOverLDAP.authOverRpc LDAP: Binding 'pparker@example.com' on  
'ldap://ad.example.com:389'  
  
it.loway.tpf.transaction.user.AuthOverLDAP.queryLdapAttribute LDAP: Attribute 'msDS-  
cloudExtensionAttribute7' is set to 'agent/101'
```

If you use custom LDAP queries, make sure you test them.

It is very useful to check the audit logs (Home Page -> System Administration -> Audit Logs) so you can see errors and successful logins.

Date	User	Session	Action	Text 1	Text 2	Text 3	Text 4
2021-08-23 23:12:37	Agent/101	AC62F6...97576D	Logon	Agent/101	10.0.2.2	pparker	

Note that if you use login rewriting, the original login supplied by the user appears in column 3, while the actual user that logged in to QM appears in column 1.
On errors, of course, the only login available is the one the user supplied.

Going deeper

What about web services?

In general, all web services (as any other log-in) will be checked against ActiveDirectory. This may or may not be what you want, e.g. because

- some services (e.g. data upload, with the `webqloader` user used to upload data) may trigger a login multiple times per second
- sometimes you just are not allowed by company policies to create service users in LDAP, as they are not really “employees” so they cannot belong there.

To handle such cases, it is possible to specify a list of users that are NOT to be checked on LDAP, but only locally, e.g.

```
auth.ldapIgnoreUsers=webqloader,robot
```

And those will only have their password checked on the local QM user database.

Will cron jobs keep working if I switch to LDAP?

Yes, they will. They do NOT perform a log-on, but only “impersonate” the user that is supposed to run them.

Can I use Secure LDAP?

LDAP is an old protocol, and it is based on clear-text; this exposes it password sniffing and other forms of credential theft.

You can transparently use a LDAPS server, just by replacing `ldap://ad.example.com:389` with `ldaps://ad.example.com:686`, as long as your DC supports it.

Is this available in my QM system?

LDAP support is available since QM 21.04; rewriting is available since QM 21.04.4

Are there other external authentication systems supported by QM?

Yes: an HTTP/S JSON driver (useful to integrate with anything else, supporting rewriting, ignored users and even transparent user creation), and a legacy XML-RPC driver.

QueueMetrics References

QueueMetrics software is available on premise or as a cloud hosted service for FreePBX, Yeastar S PBX, Grandstream, Issabel, FusionPBX and many other Asterisk distros.

For more technical information please refer to the [User Manual](#).

Visit www.queuemetrics.com for a free 15 days full featured trial.

Attend our [Free Webinars](#) for a live demonstration of QueueMetrics.