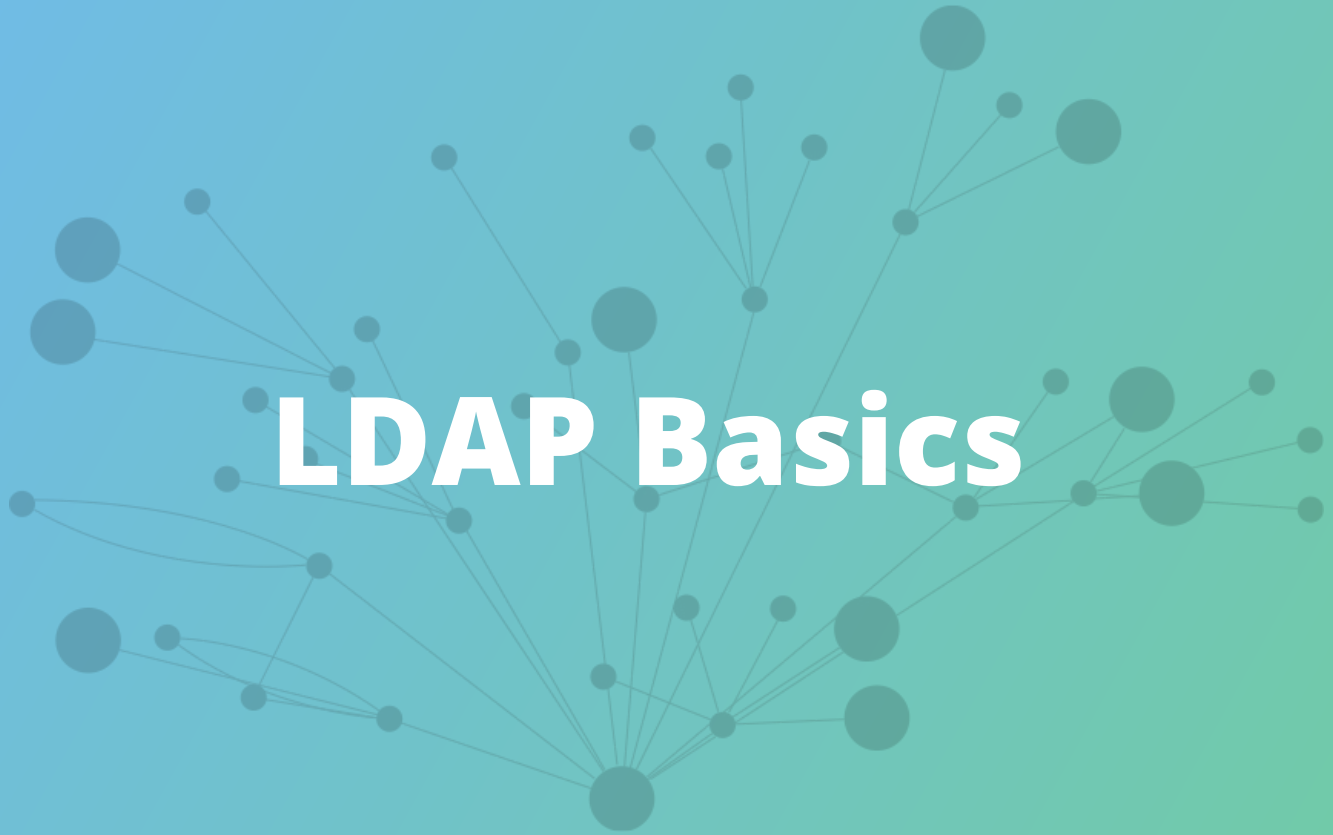
Neo4j LDAP Integration

Technical Discussion for Neo4j Enterprise Customers

Dave Shiposh
Lead Customer Success Architect
david.shiposh@neo4j.com





LDAP Basics



Introduction

LDAP stands for Lightweight Directory Access Protocol. As the name suggests, it is a lightweight client-server protocol for accessing directory services, specifically X.500-based directory services. **LDAP** runs over TCP/IP or other connection oriented transfer services.

neo4j

Introduction

Protocol overview

- A client starts an LDAP session by connecting to an LDAP server, called a Directory System Agent (DSA), by default on TCP and UDP port 389, or on port 636 for LDAPS (LDAP over SSL).
- The client then sends an operation request to the server, and the server sends responses in return.

neo4j

Operations

The client may request the following operations:

- **StartTLS*** – use the LDAPv3 Transport Layer Security (TLS) extension for a secure connection
- **Bind*** – authenticate and specify LDAP protocol version
- **Search*** – search for and/or retrieve directory entries
- **Compare** – test if a named entry contains a given attribute value
- **Add** a new entry
- **Delete** an entry
- **Modify** an entry
- **Modify Distinguished Name** (DN) – move or rename an entry
- **Abandon** – abort a previous request
- **Extended** Operation – generic operation used to define other operations
- **Unbind** – close the connection (not the inverse of Bind)

neo4j

Neo4j LDAP Integration



Let’s review the docs!

[10.4. Integration with LDAP - Chapter 10. Authentication and authorization]



Enabling LDAP

```
# Turn on security:
dbms.security.auth_enabled=true

# Choose LDAP connector as security provider for both authentication and authorization:
dbms.security.authentication_providers=ldap
dbms.security.authorization_providers=ldap
# 3.5
dbms.security.auth_provider=ldap
# or
dbms.security.auth_providers=ldap,native

# Configure LDAP to point to the LDAP server:
dbms.security.ldap.host=ldap://myldap.example.com
```


neo4j

Authentication and Authorization Settings

```
dbms.security.ldap.authentication.user_dn_template=cn={0},cn=Users,dc=example,dc=com

dbms.security.ldap.authorization.user_search_base=cn=Users,dc=example,dc=com
dbms.security.ldap.authorization.user_search_filter=(&(objectClass=*)(cn={0}))

dbms.security.ldap.authorization.use_system_account=true
dbms.security.ldap.authorization.system_username=cn=search-account,cn=Users,dc=example,dc=com
dbms.security.ldap.authorization.system_password=secret
dbms.security.ldap.authentication.use_samaccountname=true

dbms.security.ldap.authorization.group_membership_attributes=memberOf
dbms.security.ldap.authorization.group_to_role_mapping=\
  "cn=Neo4j Read Only,cn=Users,dc=neo4j,dc=com" = reader ;\
  "cn=Neo4j Read-Write,cn=Users,dc=neo4j,dc=com" = publisher ;\
  "cn=Neo4j Schema Manager,cn=Users,dc=neo4j,dc=com" = architect ;\
  "cn=Neo4j Administrator,cn=Users,dc=neo4j,dc=com" = admin ;\
  "cn=Neo4j Procedures,cn=Users,dc=neo4j,dc=com" = allowed_role
```


neo4j

Various Authorization Options

`dbms.security.ldap.authorization.use_system_account=false`

- User will search their own group membership

```
# Turn on security:
dbms.security.auth_enabled=true

# Choose LDAP connector as security provider for both authentication and authorization:
dbms.security.authentication_providers=ldap
dbms.security.authorization_providers=ldap

# Configure LDAP to point to the AD server:
dbms.security.ldap.host=ldap://ec2-54-224-47-87.compute-1.amazonaws.com

# Provide details on user structure within the LDAP system:
dbms.security.ldap.authentication.user_dn_template=CN={0},CN=Users,DC=example,DC=com
dbms.security.ldap.authorization.user_search_base=CN=Users,DC=example,DC=com
dbms.security.ldap.authorization.user_search_filter=(&(objectClass=*)(CN={0}))
dbms.security.ldap.authorization.group_membership_attributes=memberOf

# Configure the actual mapping between groups in the LDAP system and roles in Neo4j:
dbms.security.ldap.authorization.group_to_role_mapping=\
  "CN=Neo4j Read Only,CN=Users,DC=example,DC=com"      = reader       ;\
  "CN=Neo4j Read-Write,CN=Users,DC=example,DC=com"     = publisher    ;\
  "CN=Neo4j Schema Manager,CN=Users,DC=example,DC=com" = architect    ;\
  "CN=Neo4j Administrator,CN=Users,DC=example,DC=com"  = admin
```


neo4j

Various Authorization Options

```
dbms.security.ldap.authorization.use_system_account=true
dbms.security.ldap.authentication.use_samaccountname=false
```

- System account will login and search for the users group membership
- user_dn_template still required for authentication

```
need to have access rights to Neo4j or any other systems.
dbms.security.ldap.authorization.use_system_account=true
dbms.security.ldap.authorization.system_username=CN=Neo4j Search Account,CN=Users,DC=example,DC=com
dbms.security.ldap.authorization.system_password=P@ssword1
```


neo4j

Various Authorization Options

```
dbms.security.ldap.authorization.use_system_account=true
dbms.security.ldap.authentication.use_samaccountname=true
```

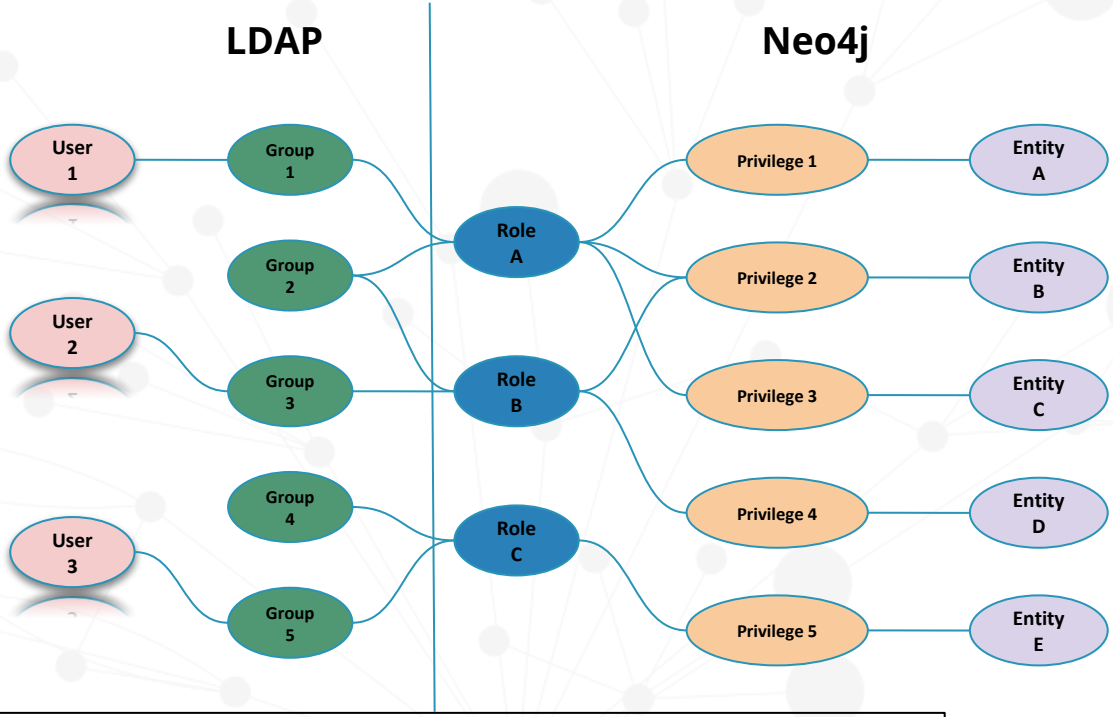
- System account will login and search for the user based on their samaccountname and will do both group membership as well as return the users DN so that the bind (authentication) can be completed for the user.
- user_dn_template not required

```
dbms.security.ldap.authorization.use_system_account=true
dbms.security.ldap.authorization.system_username=CN=Neo4j Search Account,CN=Users,DC=example,DC=com
dbms.security.ldap.authorization.system_password=P@ssword1

# Perform authentication with sAMAccountName instead of DN.
# Using this setting requires dbms.security.ldap.authorization.system_username and
dbms.security.ldap.authorization.system_password to be used, since there is no way to log in through LDAP
directly with the sAMAccountName.
# Instead, the login name will be resolved to a DN that will be used to log in with.
dbms.security.ldap.authentication.use_samaccountname=true
```


neo4j

LDAP Role Mapping in Neo4j 4.0



Caching

`dbms.security.ldap.authentication.cache_enabled=true`

- Determines whether or not to cache the result of authentication via the LDAP server

`dbms.security.auth_cache_ttl=10m`

- Is the time to live (TTL) for cached authentication and authorization info.

`dbms.security.auth_cache_use_ttl=true`

- Enable time-based eviction of the authentication and authorization info cache for external auth providers (LDAP or plugin).

`dbms.security.auth_cache_max_capacity=10000`

- The maximum capacity for authentication and authorization caches (respectively).

neo4j

Secure LDAP

- Use LDAPS **OR** StartTLS=true

```
dbms.security.ldap.use_starttls=true

dbms.security.ldap.host=ldaps://myldap.example.com
dbms.security.ldap.host=ldaps://myldap.example.com:636
```

- LDAP Server Cert needs to be added to the JVM Keystore on each Neo4j Server:

```
keytool -import -alias _alias_name_ -keystore ..\lib\security\cacerts -file _path_to_cer_file
```


neo4j

ldapsearch (or ldp.exe) is your friend

```
#ldapsearch -v -H ldap://<dbms.security.ldap.host> -x -D <dbms.security.ldap.authentication.user_dn_template : replace {0}> -W -b
<dbms.security.ldap.authorization.user_search_base> "<dbms.security.ldap.authorization.user_search_filter : replace {0}>"
<dbms.security.ldap.authorization.group_membership_attributes>
ldapsearch -v -H ldap://myactivedirectory.example.com:389 -x -D cn=john,cn=Users,dc=example,dc=com -W -b cn=Users,dc=example,dc=com "
(&(objectClass=*)(cn=john))" memberOf
```

- Start test without group_membership_attributes
  - This will return all users details

```
filter: (&(objectClass=*)(cn=Neo4j Admin))
requesting: All userApplication attributes
# extended LDIF
#
# LDAPv3
# base <cn=Users,dc=example,dc=com> with scope subtree
# filter: (&(objectClass=*)(cn=Neo4j Admin))
# requesting: ALL
#

# Neo4j Admin, Users, example.com
dn: CN=Neo4j Admin,CN=Users,DC=example,DC=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
cn: Neo4j Admin
givenName: Neo4j Admin
distinguishedName: CN=Neo4j Admin,CN=Users,DC=example,DC=com
instanceType: 4
```

Troubleshooting / Tricks



Troubleshooting

- Use ldapsearch
- Check security.log
  - Can set: dbms.logs.security.level=DEBUG
  - Narrow down if the issue is related to:
    - Connection/Security
    - Authentication (user doesn't exist/password)

      ldap_bind: Invalid credentials (49) additional info: 80090308: LdapErr: DSID-0C0903A9, comment: AcceptSecurityContext error, data 52e, v1db1
    - Authorization

      Neo.ClientError.Security.Forbidden: Read operations are not allowed for user 'gavin' with no roles.
- Check LDAP Server Logs

neo4j

Tricks

- Multiple Providers (3.5)

  dbms.security.auth_provider**s**=native, ldap

- “Bypass” role mapping

  dbms.security.ldap.authorization.group_membership_attributes=objectClass

  dbms.security.ldap.authorization.group_to_role_mapping=\

  "user" = admin

OR

  dbms.security.ldap.authorization.group_membership_attributes=CN

  dbms.security.ldap.authorization.group_to_role_mapping=\

  "NeoTest" = publisher

neo4j

Thank You!

