Active Directory Integration with Confluent Kafka - Hands-On Exercise

Exercise Overview

This comprehensive hands-on exercise demonstrates how to integrate Microsoft Active Directory (AD) with Confluent Kafka for centralized authentication and authorization. You will configure SASL/PLAIN authentication with LDAP callback handlers, set up user mappings, and implement role-based access control (RBAC).

Duration: 4-5 hours
Difficulty: Advanced
Prerequisites: Confluent Platform 7.x installed, Active Directory or OpenLDAP instance, Linux/Ubuntu environment

---

Learning Objectives

By completing this exercise, you will:

1. Understand AD/LDAP integration architecture with Kafka
2. Configure SASL/PLAIN authentication with LDAP callback handlers
3. Set up user search filters and password authentication mechanisms
4. Implement RBAC for fine-grained access control
5. Test AD authentication across producers and consumers
6. Troubleshoot LDAP connectivity and authentication issues
7. Monitor and audit AD-based access patterns

Part 1: Environment Setup and LDAP Configuration

1.1 Verify LDAP/Active Directory Availability

Objective: Establish connectivity to your AD/LDAP server

Steps:

1. **Test LDAP connectivity from the Kafka broker:**

   ```
   ldapsearch -LLL -x -H ldap://your-ldap-host:389 -s "base" -b ""
   supportedSASLMechanisms
   ```

   - For LDAPS (SSL): Replace `ldap://` with `ldaps://`
   - For self-signed certs: Export and use CA certificate
     ```
     LDAPTLS_CACERT=/path/to/CA.cert ldapsearch -LLL -x -H ldaps://your-ldap-host:636
     -s "base" -b "" supportedSASLMechanisms
     ```

2. **Verify AD service account credentials:**

   ```
   ldapsearch -LLL -x -H ldap://your-ldap-host:389 -s "base" -b ""
   -D "CN=kafka_admin,CN=Users,DC=yourcompany,DC=com"
   -w 'your-service-account-password' supportedSASLMechanisms
   ```

3. **Expected output:** Should return `supportedSASLMechanisms` or authentication success

Troubleshooting:

- If `Can't contact LDAP server`: Check firewall, LDAP port (389 for LDAP, 636 for LDAPS)
- If authentication fails: Verify DN and password format (use escape characters if needed)
- Use `ldap://your-hostname:389` rather than IP for proper DNS resolution

1.2 Map Active Directory Users and Groups

Objective: Identify AD users and organizational structure for Kafka authentication

Steps:

1. **List all users in a specific OU (Organizational Unit):**

   ```
   ldapsearch -LLL -x -H ldap://your-ldap-host:389
   -D "CN=kafka_admin,CN=Users,DC=yourcompany,DC=com"
   -w 'service-account-password'
   -b "OU=DataTeam,DC=yourcompany,DC=com"
   "(objectClass=user)" uid mail
   ```

2. **Query for group membership:**

   ```
   ldapsearch -LLL -x -H ldap://your-ldap-host:389
   -D "CN=kafka_admin,CN=Users,DC=yourcompany,DC=com"
   -w 'service-account-password'
   -b "CN=kafka-producers,CN=Users,DC=yourcompany,DC=com"
   "(objectClass=group)" member
   ```

3. **Create a mapping document with:**
   - User DN: `CN=john.doe,OU=DataTeam,DC=yourcompany,DC=com`
   - User ID attribute: `uid=john.doe`
   - Group DN: `CN=kafka-producers,CN=Users,DC=yourcompany,DC=com`
   - Expected roles: producer, consumer, admin

Exercise Task:

Document at least 3 AD users and their organizational mapping for use in subsequent exercises.

Part 2: Configure Kafka Broker for SASL/PLAIN with LDAP

2.1 Install LDAP Callback Handler

Objective: Set up the Confluent LDAP authentication plugin

Steps:

1. **Verify Confluent Platform includes LDAP handler:**

   find $CONFLUENT_HOME -name "*ldap*" -type f

   Should show: `io.confluent.security.auth.provider.ldap.LdapAuthenticateCallbackHandler`

2. **Confirm JAR is in broker classpath:**

   ls -la $CONFLUENT_HOME/share/java/kafka/
   Look for: `confluent-security-*.jar`

3. **If not available, install via Confluent Hub:**

   confluent-hub install confluentinc/kafka-connect-ldap:latest

2.2 Configure broker server.properties for SASL/PLAIN + LDAP

Objective: Enable SASL/PLAIN authentication with LDAP backend

Configuration Steps:

1. **Open the broker configuration file:**

   ```
   sudo nano $CONFLUENT_HOME/etc/kafka/server.properties
   ```

2. **Add/modify SASL/PLAIN listener configuration:**

   Enable SASL on a plaintext listener

   ```
   listeners=PLAINTEXT://0.0.0.0:9092,SASL_PLAINTEXT://0.0.0.0:9093
   advertised.listeners=PLAINTEXT://kafka-
   broker1.yourcompany.com:9092,SASL_PLAINTEXT://kafka-
   broker1.yourcompany.com:9093
   ```

   Configure SASL mechanisms

   ```
   listener.security.protocol.map=PLAINTEXT:PLAINTEXT,SASL_PLAINTEXT:SASL_
   PLAINTEXT
   inter.broker.listener.name=SASL_PLAINTEXT
   ```

   Enable PLAIN mechanism on SASL_PLAINTEXT listener

   ```
   listener.name.sasl_plaintext.sasl.enabled.mechanisms=PLAIN
   listener.name.sasl_plaintext.plain.sasl.jaas.config=org.apache.kafka.common.securit
   y.plain.PlainLoginModule required
   username="kafka_broker"
   password="broker-password";
   ```

   Set the callback handler for LDAP authentication

   ```
   listener.name.sasl_plaintext.plain.sasl.server.callback.handler.class=io.confluent.sec
   urity.auth.provider.ldap.LdapAuthenticateCallbackHandler
   ```

3. **Add LDAP configuration parameters:**

   LDAP Server Connection

   ```
   ldap.java.naming.provider.url=ldap://your-ldap-host:389
   ```

   LDAP Bind Credentials (service account with query permissions)

ldap.java.naming.security.principal=CN=kafka_admin,CN=Users,DC=yourcompany,DC=com
ldap.java.naming.security.credentials=your-service-account-password
ldap.java.naming.security.authentication=simple

User Search Configuration

ldap.user.search.base=OU=DataTeam,DC=yourcompany,DC=com
ldap.user.name.attribute=uid
ldap.user.object.class=user

Password Verification Method

ldap.user.password.attribute=userPassword
ldap.authentication.type=simple

4. **Add Authorization settings (for RBAC later):**

authorizer.class.name=io.confluent.kafka.security.authorizer.ConfluentServerAuthorizer
super.users=User:kafka_broker;User:kafka_admin

5. **Save the configuration**

Configuration Reference Table:

Parameter	Purpose	Example Value
`ldap.java.naming.provider.url`	LDAP server URL	`ldap://ad.company.com:389`
`ldap.java.naming.security.principal`	Bind DN (service account)	CN=kafka_admin,CN=Users,DC=company,DC=com
`ldap.user.search.base`	Base DN for user searches	`OU=DataTeam,DC=company,DC=com`
`ldap.user.name.attribute`	Attribute mapped to username	`uid` or `sAMAccountName`
`ldap.user.object.class`	LDAP object class for users	`user` or `inetOrgPerson`
`ldap.user.password.attribute`	Password storage attribute	`userPassword` or `unicodePwd`

2.3 Restart Kafka Broker

Objective: Apply SASL/LDAP configuration

Steps:

1. **Stop Kafka broker:**

   ```
   confluent local services kafka stop
   ```

   OR

   ```
   $CONFLUENT_HOME/bin/kafka-server-stop.sh
   ```

2. **Verify broker stopped (wait 5-10 seconds):**

   ```
   lsof -i :9093
   ```

   Should return no results

3. **Start broker with new configuration:**

   ```
   confluent local services kafka start
   ```

   OR

   ```
   $CONFLUENT_HOME/bin/kafka-server-start.sh
   $CONFLUENT_HOME/etc/kafka/server.properties
   ```

4. **Monitor broker startup logs:**

   ```
   tail -f $CONFLUENT_HOME/logs/kafka.log | grep -i "ldap|sasl|bind"
   ```
   Look for: `Started NetworkReceiver listening on 0.0.0.0:9093` (SASL port)

5. **Verify listener is active:**

```
netstat -tlnp | grep 9093
```

Troubleshooting:

- If broker fails to start, check: `$CONFLUENT_HOME/logs/kafka.log` for LDAP/SASL errors
- If LDAP connection fails: Verify firewall rules, LDAP host/port, service account credentials
- If bind fails: Check DN format matches your AD structure

Part 3: Create and Test Client Authentication

3.1 Create JAAS Configuration for Client

Objective: Configure a Kafka client to authenticate via AD

Steps:

1. **Create a JAAS configuration file for the client:**

```
cat > /tmp/kafka_client_jaas.conf << 'EOF'
KafkaClient {
org.apache.kafka.common.security.plain.PlainLoginModule required
username="john.doe"
password="john-doe-ad-password";
};
EOF
```

2. **Restrict file permissions (important for security):**

```
chmod 600 /tmp/kafka_client_jaas.conf
```

3. **Verify file contents:**

```
cat /tmp/kafka_client_jaas.conf
```

3.2 Create Client Configuration Properties

Objective: Set up producer/consumer client properties

Steps:

1. **Create client configuration file:**

   ```
   cat > /tmp/kafka_client.properties << 'EOF'
   ```

   Broker Connection

   ```
   bootstrap.servers=your-kafka-broker:9093
   ```

   SASL/PLAIN Configuration

   ```
   security.protocol=SASL_PLAINTEXT
   sasl.mechanism=PLAIN
   sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required
   username="john.doe"
   password="john-doe-ad-password";
   ```

   Client Configuration

   ```
   client.id=ad-auth-producer
   acks=all
   retries=3
   EOF
   ```

2. **For production with TLS/SSL:**

   ```
   cat > /tmp/kafka_client_tls.properties << 'EOF'
   bootstrap.servers=your-kafka-broker:9093
   security.protocol=SASL_SSL
   sasl.mechanism=PLAIN
   sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required
   username="john.doe"
   password="john-doe-ad-password";
   ssl.truststore.location=/path/to/truststore.jks
   ssl.truststore.password=truststore-password
   ssl.truststore.type=JKS
   EOF
   ```

3.3 Test Producer Authentication

Objective: Verify AD user can authenticate and produce messages

Steps:

1. **Create a test topic (as super user):**

   ```
   kafka-topics --bootstrap-server localhost:9092
   --create
   --topic ad-test-topic
   --partitions 3
   --replication-factor 1
   ```

2. **Test producer with AD credentials:**

   ```
   kafka-console-producer --broker-list your-kafka-broker:9093
   --topic ad-test-topic
   --producer-property file=/tmp/kafka_client.properties
   ```

3. **Type test messages:**

   ```
   Hello from AD authenticated producer
   This is message 2 from AD user
   Message 3 with timestamp
   Press Ctrl+D to exit
   ```

4. **Check for authentication errors in broker logs:**

   ```
   grep -i "authentication|ldap|sasl" $CONFLUENT_HOME/logs/kafka.log | tail -20
   ```

Expected Success Indicators:

- Messages accepted without authentication errors
- Broker logs show: `SASL authentication succeeded for user john.doe`
- Topic receives all messages

Common Issues:

- `Authentication failed`: Username/password doesn't match AD
- `User not found`: Check `ldap.user.name.attribute` and user DN structure
- `LDAP bind failed`: Service account credentials wrong or LDAP unreachable

3.4 Test Consumer Authentication

Objective: Verify consumer can authenticate and read messages

Steps:

1. **Test consumer with AD credentials:**

```
kafka-console-consumer --bootstrap-server your-kafka-broker:9093
--topic ad-test-topic
--from-beginning
--consumer-property file=/tmp/kafka_client.properties
```

2. **Verify messages appear:**

```
Hello from AD authenticated producer
This is message 2 from AD user
Message 3 with timestamp
```

3. **Check broker authentication logs:**

```
grep -i "john.doe" $CONFLUENT_HOME/logs/kafka.log
```

Part 4: Implement Role-Based Access Control (RBAC)

4.1 Create AD Groups for Kafka Roles

Objective: Structure AD groups to map to Kafka RBAC roles

Steps:

1. **Verify existing AD groups:**

```
ldapsearch -LLL -x -H ldap://your-ldap-host:389
-D "CN=kafka_admin,CN=Users,DC=yourcompany,DC=com"
-w 'service-account-password'
-b "CN=Users,DC=yourcompany,DC=com"
"(cn=kafka*)" cn member
```

2. **Document AD groups for RBAC mapping:**

AD Group	Kafka Role	Permissions
`CN=kafka-admins,CN=Users,DC=yourcompany,DC=com`	Admin	All operations on all resources
`CN=kafka-producers,CN=Users,DC=yourcompany,DC=com`	Producer	Write to topics, read schemas
`CN=kafka-consumers,CN=Users,DC=yourcompany,DC=com`	Consumer	Read from topics, manage consumer groups
`CN=kafka-connectors,CN=Users,DC=yourcompany,DC=com`	Connector	Deploy and manage connectors

3. **Create mapping file for RBAC policy:**

```
cat > /tmp/rbac_mapping.txt << 'EOF'
```

Subject (User/Group) -> Role Mapping

```
User:john.doe -> kafka-producers
User:jane.smith -> kafka-admins
User:dev-team -> kafka-consumers
User:etl-service -> kafka-connectors
EOF
```

4.2 Configure MDS for RBAC

Objective: Enable Confluent Metadata Service for centralized RBAC

Steps:

1. **Update broker configuration for MDS:**

```
cat >> $CONFLUENT_HOME/etc/kafka/server.properties << 'EOF'
```

Metadata Service (MDS) Configuration

```
confluent.metadata.server.advertised.urls=http://localhost:8090
confluent.metadata.server.listeners=http://0.0.0.0:8090
confluent.metadata.server.authentication.method=BEARER
confluent.metadata.server.user.store=FileUserStore
confluent.metadata.server.user.store.path=/tmp/mds_users.properties
confluent.metadata.server.token.key.path=/tmp/mds_tokenkey
confluent.metadata.server.token.max.lifetime.ms=3600000
```

Authorizer Configuration

```
authorizer.class.name=io.confluent.kafka.security.authorizer.ConfluentServerAuthorizer
confluent.authorizer.access.rule.create=ALLOW
confluent.authorizer.access.rule.create.principal.type=User,Group
confluent.authorizer.access.rule.create.principal.name=*
confluent.authorizer.access.rule.create.operation=All
confluent.authorizer.access.rule.create.resource=*
```

Super Users for Bootstrap

```
super.users=User:kafka_broker;User:kafka_admin;User:mds_admin
EOF
```

2. **Create MDS user store:**

```
cat > /tmp/mds_users.properties << 'EOF'
mds_admin:mds-admin-password
kafka_admin:kafka-admin-password
EOF

chmod 600 /tmp/mds_users.properties
```

3. **Generate MDS token signing key:**

```
openssl genrsa -out /tmp/mds_tokenkey 4096
chmod 600 /tmp/mds_tokenkey
```

4. **Restart broker:**

```
confluent local services kafka stop
confluent local services kafka start
```

5. **Verify MDS is running:**

```
curl -X GET http://localhost:8090/api/v1/metadata/version
```

Should return version information

4.3 Assign RBAC Roles via CLI

Objective: Map AD users to Kafka RBAC roles

Steps:

1. **Install confluent CLI (if not present):**

```
confluent update
```

2. **Create RBAC role binding for producer user:**

```
confluent iam rolebinding create
--principal User:john.doe
--role ResourceOwner
--resource Topic:ad-test-topic
```

3. **Create RBAC role for consumer group:**

```
confluent iam rolebinding create
--principal User:jane.smith
--role ConsumerGroupAdmin
--resource Group:ad-consumer-group-1
```

4. **List RBAC assignments:**

```
confluent iam rolebinding list --principal User:john.doe
```

5. **Create topic-level producer role:**

```
confluent iam rolebinding create
--principal User:john.doe
--role DeveloperWrite
--resource Topic:ad-test-topic
--resource-pattern-type PREFIXED
```

Part 5: Advanced Testing and Troubleshooting

5.1 Test Negative Authentication Scenarios

Objective: Verify proper rejection of invalid credentials

Steps:

1. **Test with wrong password:**

```
kafka-console-producer --broker-list your-kafka-broker:9093
--topic ad-test-topic
--producer-property bootstrap.servers=your-kafka-broker:9093
--producer-property security.protocol=SASL_PLAINTEXT
--producer-property sasl.mechanism=PLAIN
--producer-property
sasl.jaas.config='org.apache.kafka.common.security.plain.PlainLoginModule required
username="john.doe" password="wrong-password";'
```

   Expected: Authentication failure error

2. **Test with non-existent user:**

```
kafka-console-producer --broker-list your-kafka-broker:9093
--topic ad-test-topic
--producer-property bootstrap.servers=your-kafka-broker:9093
--producer-property security.protocol=SASL_PLAINTEXT
--producer-property sasl.mechanism=PLAIN
--producer-property
sasl.jaas.config='org.apache.kafka.common.security.plain.PlainLoginModule required
username="nonexistent.user" password="any-password";'
```

   Expected: User not found error

3. **Test authorization denial:**

   Create a restricted user without write permissions and verify access denied.

5.2 Monitor LDAP Authentication in Logs

Objective: Debug and track authentication operations

Steps:

1. **Enable debug logging for LDAP:**

```
export KAFKA_DEBUG=true
export DEBUG_LOGGING_ENABLED=true
```

2. **Check for authentication events:**

```
grep -i "authentication|ldap|user.*success|user.*failed"
$CONFLUENT_HOME/logs/kafka.log
```

3. **Monitor real-time authentication:**

```
tail -f $CONFLUENT_HOME/logs/kafka.log | grep -i "SASL|LDAP|ldap"
```

4. **Capture authentication metrics:**

```
jconsole
```

Navigate to: MBean > kafka.server > BrokerTopicMetrics > MessagesInPerSec

5.3 Test LDAP Connectivity Issues

Objective: Diagnose and resolve LDAP connection problems

Test Matrix:

Scenario	Command	Expected Result
LDAP port unreachable	`telnet your-ldap-host 389`	Connection refused
LDAP server down	`nc -zv your-ldap-host 389`	Port closed
Firewall blocked	Try from Kafka broker	Connection timeout
DNS resolution	`nslookup your-ldap-host`	IP address returned
Service account invalid	`ldapsearch -D` with wrong DN	`Invalid credentials`

Debugging Commands:

Test full LDAP bind and user search

```
ldapsearch -LLL -x -H ldap://your-ldap-host:389
-D "CN=kafka_admin,CN=Users,DC=yourcompany,DC=com"
-w 'password'
-b "OU=DataTeam,DC=yourcompany,DC=com"
"(uid=john.doe)" cn uid mail
```

Check LDAP schema

```
ldapsearch -LLL -x -H ldap://your-ldap-host:389
-s "base" -b "" objectClass
```

Verify password attribute

```
ldapsearch -LLL -x -H ldap://your-ldap-host:389
-D "CN=kafka_admin,CN=Users,DC=yourcompany,DC=com"
```

```
-w 'password'
-b "OU=DataTeam,DC=yourcompany,DC=com"
"(uid=john.doe)" userPassword
```

Part 6: Production Implementation Checklist

6.1 Security Hardening

- Use SASL_SSL instead of SASL_PLAINTEXT in production
- Implement certificate pinning for LDAP connections
- Rotate service account passwords regularly (quarterly minimum)
- Use secrets management (HashiCorp Vault, Kubernetes Secrets)
- Encrypt JAAS configuration files (chmod 600)
- Enable MDS token expiration and rotation
- Implement network segmentation for LDAP traffic
- Enable audit logging for all authentication attempts

6.2 Monitoring and Alerting

- Monitor LDAP connection failures
- Alert on repeated authentication failures (potential brute force)
- Track LDAP query latency (> 500ms is concerning)
- Monitor broker CPU during LDAP authentication peaks
- Set up Prometheus metrics for auth success/failure ratios
- Log all RBAC role changes for compliance

6.3 Failover and High Availability

- Configure LDAP replication/clustering
- Implement LDAP failover with multiple directory servers
- Test authentication during LDAP failover scenarios
- Document RTO/RPO for AD integration
- Implement local cache for frequently accessed users (optional)
- Document fallback authentication mechanisms

Exercise Validation Checklist

Success Criteria

- LDAP server connectivity verified from Kafka broker
- AD users and groups successfully queried
- Broker SASL/PLAIN listener active on port 9093
- Client authentication succeeds with valid AD credentials
- Client authentication fails with invalid credentials
- Producer can write messages with AD authentication
- Consumer can read messages with AD authentication
- RBAC role bindings enforced correctly
- Authentication events visible in broker logs
- LDAP connectivity issues diagnosed and resolved

Expected Outcomes

Upon successful completion:

1. **Authentication:** AD users authenticate to Kafka via SASL/PLAIN with LDAP backend
2. **Authorization:** RBAC controls topic access based on AD group membership
3. **Auditability:** All authentication/authorization events logged for compliance
4. **Scalability:** Supports organization-wide identity management via AD
5. **Troubleshooting:** Can diagnose and resolve common AD integration issues

References and Documentation

Resource	URL	Purpose
Confluent LDAP Auth	https://docs.confluent.io/platform/current/security/authentication/ldap/client-authentication-ldap.html	Official LDAP configuration guide
RBAC Overview	https://docs.confluent.io/platform/current/security/authorization/rbac/overview.html	Role-based access control documentation
SASL/PLAIN	https://docs.confluent.io/kafka/latest/authentication/authentication_sasl/authentication_sasl_plain.html	SASL/PLAIN mechanism reference
Security Best Practices	https://docs.confluent.io/platform/current/security/index.html	Complete security configuration
Active Directory LDAP	https://learn.microsoft.com/en-us/windows/server/identity/ad-ds/manage/component-updates/ldap-query-basics	AD LDAP query reference

Appendix: Sample Configuration Files

A1: Complete broker server.properties (AD Integration)

Cluster Setup

```
broker.id=1
zookeeper.connect=localhost:2181
```

Listeners and Security Protocol

```
listeners=PLAINTEXT://0.0.0.0:9092,SASL_PLAINTEXT://0.0.0.0:9093
advertised.listeners=PLAINTEXT://localhost:9092,SASL_PLAINTEXT://localhost:9093
listener.security.protocol.map=PLAINTEXT:PLAINTEXT,SASL_PLAINTEXT:SASL_PLAIN
TEXT
inter.broker.listener.name=SASL_PLAINTEXT
```

SASL/PLAIN Configuration

```
listener.name.sasl_plaintext.sasl.enabled.mechanisms=PLAIN
listener.name.sasl_plaintext.plain.sasl.jaas.config=org.apache.kafka.common.security.plain.
PlainLoginModule required
username="kafka_broker"
password="broker-secret";
listener.name.sasl_plaintext.plain.sasl.server.callback.handler.class=io.confluent.security.aut
h.provider.ldap.LdapAuthenticateCallbackHandler
```

LDAP Configuration

```
ldap.java.naming.provider.url=ldap://ad.yourcompany.com:389
ldap.java.naming.security.principal=CN=kafka_admin,CN=Users,DC=yourcompany,DC=co
m
ldap.java.naming.security.credentials=SecurePassword123
ldap.java.naming.security.authentication=simple
ldap.user.search.base=OU=DataTeam,DC=yourcompany,DC=com
ldap.user.name.attribute=uid
ldap.user.object.class=user
ldap.user.password.attribute=userPassword
```

Authorization

```
authorizer.class.name=io.confluent.kafka.security.authorizer.ConfluentServerAuthorizer
super.users=User:kafka_broker;User:kafka_admin
```

Log Configuration

```
log.dirs=/var/kafka-logs
num.partitions=3
default.replication.factor=1
```

A2: Sample AD User LDAP Entry

```
dn: CN=john.doe,OU=DataTeam,DC=yourcompany,DC=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
cn: john.doe
uid: john.doe
sAMAccountName: john.doe
givenName: John
sn: Doe
userPassword: HashedPassword123
accountStatus: active
memberOf: CN=kafka-producers,CN=Users,DC=yourcompany,DC=com
```

A3: Troubleshooting Commands Quick Reference

Verify LDAP server responsiveness

```
ldapsearch -x -H ldap://ad.company.com:389 -s base -b "" supportedSASLMechanisms
```

Search for specific user

```
ldapsearch -x -H ldap://ad.company.com:389 -b "DC=company,DC=com" "(uid=john.doe)"
```

List group members

```
ldapsearch -x -H ldap://ad.company.com:389 -b "CN=kafka-
producers,CN=Users,DC=company,DC=com" member
```

Test Kafka authentication

```
kafka-console-producer --broker-list kafka:9093 --topic test
--producer-property bootstrap.servers=kafka:9093
--producer-property security.protocol=SASL_PLAINTEXT
--producer-property sasl.mechanism=PLAIN
--producer-property sasl.jaas.config='...'
```

Monitor authentication in real-time

```
tail -f $CONFLUENT_HOME/logs/kafka.log | grep -i "authentication|ldap"
```

Check active connections

```
netstat -tlnp | grep -E "9092|9093"
```