KAFKA ADMIN CLIENT AND SECURING KAFKA

Đơn vị: Công ty CP Giáo dục và Công nghệ QNET

QNET JOINT STOCK COMPANY

Address: 14th Floor, VTC Online Tower
18 Tam Trinh Street. Hoang Mai District
Hanoi, Vietnam

Quality Network for Education and Technology

ADMIN CLIENT AND SECURING KAFKA

Learning Objective

- AdminClient
- Kafka Security
- Kafka Security Components
- Configure SSL in Kafka
- Secure using ACLs

ADMIN CLIENT AND SECURING KAFKA

AdminClient

AdminClient

To manage and inspect Kafka object: topics, brokers, ACLs

The minimum broker version required is 0.11

Why

Automation: Useful in automating creation, ACLs,

Using Client we can avoid manual creation of any

ADMIN CLIENT AND SECURING KAFKA

Admin Client APIs

Usage of AdminClient APIs

ADMIN CLIENT AND SECURING KAFKA

Admin Client Lifecycle - Creation, Configuring and Closing

AdminClient accepts client config but minimum requirement is to pass bootstrap servers in configuration.
One config is created we use AdminClient create method to get AdminClient

```
Properties props = new Properties();
props.put(AdminClientConfig.BOOTSTRAP_SERVERS_CONFIG, "localhost:9092");
AdminClient admin = AdminClient.create(props);
// TODO: Do something useful with AdminClient
admin.close(Duration.ofSeconds(30));
```

Close but still be in process?

Close method will not stop process immediately
One *close* methods is called, you can't call any other methods and send any more requests

ADMIN CLIENT AND SECURING KAFKA

Admin Client Lifecycle - Two important configurations

client.dns.lookup

Be default, Kafka validates, resolves, and create connections base on the hostname provided in the bootstrap configuration.
It works most of the time but fails to cover two important use case:

- the use of DNS alias:
  - using all-brokers.hostname.com *i*nstead of broker1.hostname.com, broker2.hostname.com.
  - The problem is when you use SASL for authentication, SASL will refuse your request because of wrong dns resolution => If use *client.dns.lookup=resolve_canonial_bootstrap_servers_only* ,

request.timeout.

This configuration limits the time that your application can spend waiting for AdminClient to respond. This includes the time spent on retrying if the client receives a retriable error

ADMIN CLIENT AND SECURING KAFKA

Demo: Perform Various Admin Tasks using AdminClient

KAFKA SECURITY

SECURITY

Security is always a concern in data pipelines.

1. Can we make sure the data going through the pipe is encrypted ?
2. Who are allowed to make modifications to the pipelines ?
3. Is data encrypted when transferring on network?

KAFKA SECURITY

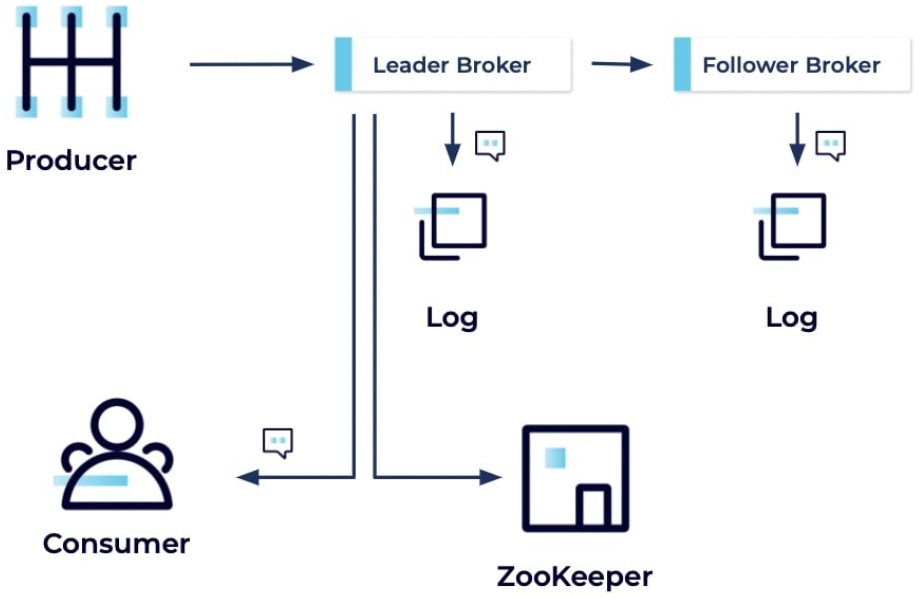
Kafka use a range of security procedures to establish and maintain security

1. Authentication establishes your identity and determines who you are
2. Authorization determines what you allowed to do
3. Encryption protects your data
4. Auditing tracks what you have done or have attempted to do
5. Quotas control how much resources you can utilize

KAFKA SECURITY

SECURITY

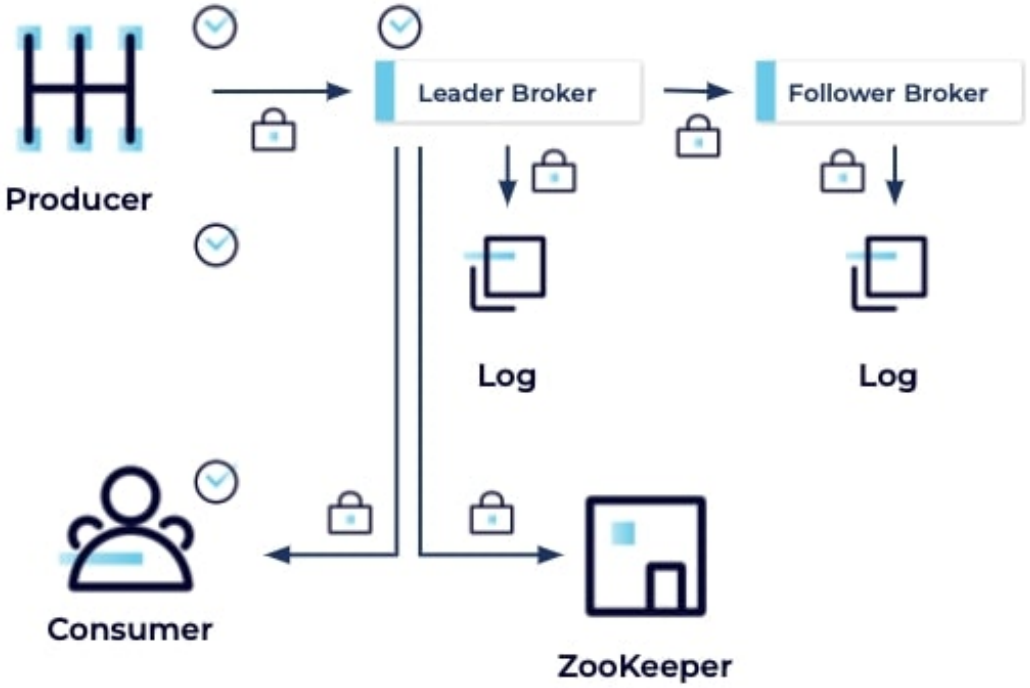
Data flow in Kafka Cluster



KAFKA SECURITY

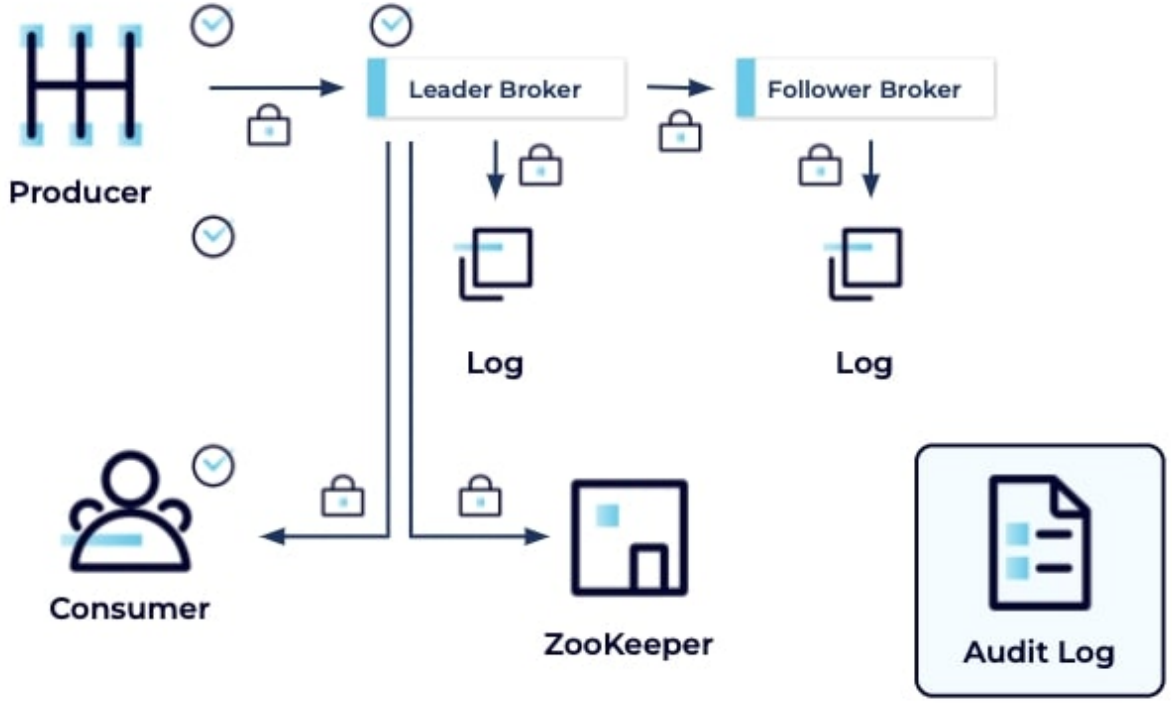
SECURITY

How Kafka Security Works



KAFKA SECURITY

Audit log



KAFKA SECURITY

Security Protocol

PLAINTEXT
: PLAINTEXT transport layer with no authentication. Is suitable only for use within private networks for processing data that is not sensitive since no authentication or encryption is used.

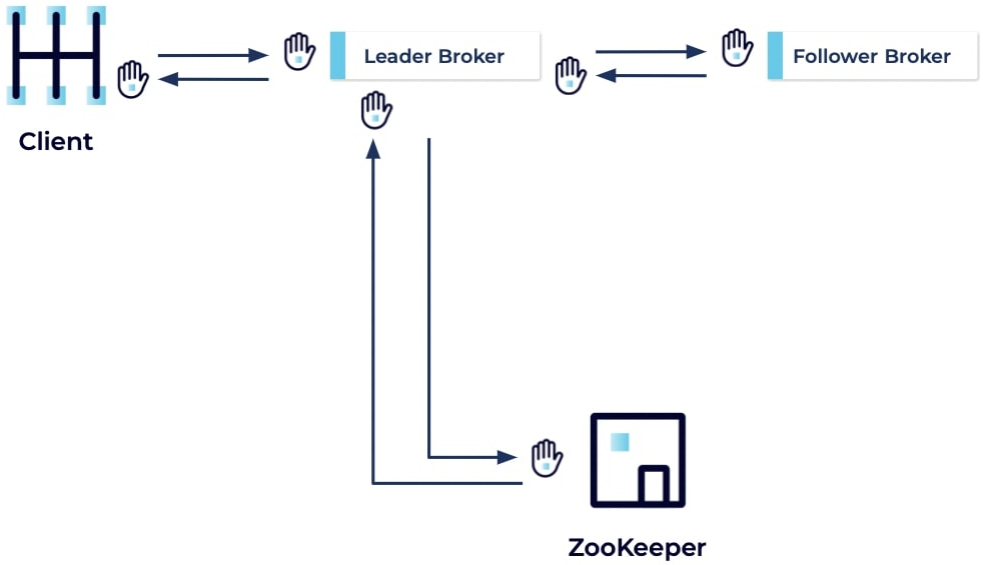
SSL
: SSL transport layer with optional SSL client authentication. Is suitable for use in insecure networks since client and server authentication as well as encryption are supported.

SASL_PLAINTEXT
: PLAINTEXT transport layer with SASL client authentication. Some SASL mechanisms also support server authentication. Does not support encryption and hence is suitable only for use within private networks.

SASL_SSL
: SSL transport layer with SASL authentication. Is suitable for use in insecure networks since client and server authentication as well as encryption are supported.

KAFKA SECURITY

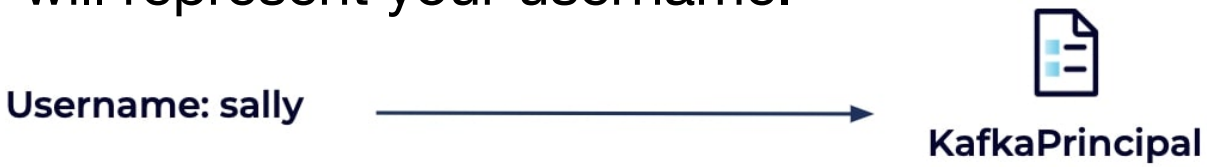
Kafka Authentication Basics



KAFKA SECURITY

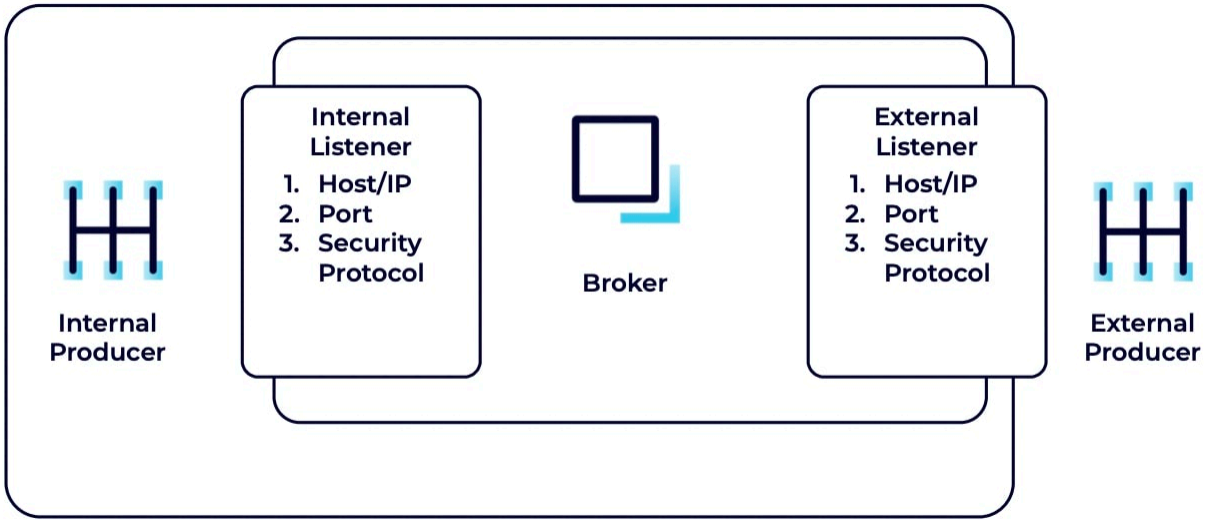
KafkaPrincipal

Internally in Kafka, a client's identity is represented using a KafkaPrincipal object, or principal. So, for example, if you connect to Kafka and authenticate with a username and password, the principal associated with the connection will represent your username.



KAFKA SECURITY

Configuring Authentication: Listeners and Security Protocols



KAFKA SECURITY

Configuring Authentication: Listeners and Security Protocols

The following broker configuration snippet specifies three listeners for a broker: an external network listener, an internal listener, and an inter-broker listener (the inter-broker and internal listeners are configured to use SSL, the external listener SASL_SSL):

```
listeners=EXTERNAL://:9092,INTERNAL://10.0.0.2:9093,BROKER://10.0.0.2:9094
advertised.listeners=EXTERNAL://broker1.example.com:9092,INTERNAL:// broker1.local:9093,BROKER://broker1.local:9094
listener.security.protocol.map=EXTERNAL:SASL_SSL,INTERNAL:SSL,BROKER:SSL
inter.broker.listener.name=BROKER
```

Relatedly, here is a config snippet for a client, specifying that SASL_SSL should be used to communicate with the listed boots servers:

```
security.protocol=SASL_SSL
bootstrap.servers=broker1.example.com:9092,broker2.example.com:9092
```

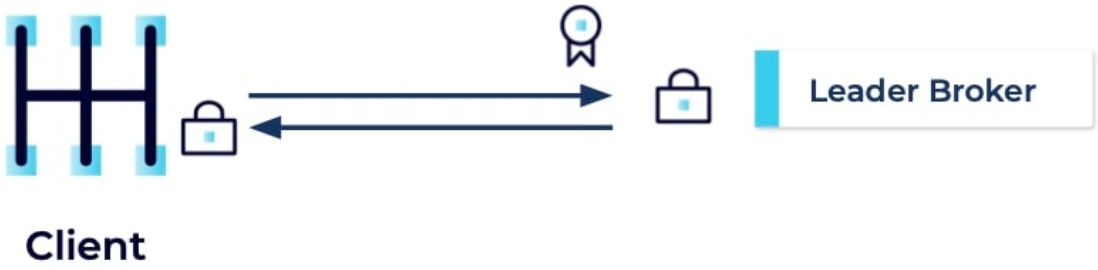
KAFKA SECURITY

Kafka Authentication with SSL

When SSL is enabled for a Kafka listener, all traffic for that channel will be encrypted with TLS, which employs digital cert
for identity verification

Client-

```
ssl.client.auth=required
```



KAFKA SECURITY

Kafka Authentication with SSL

When SSL is enabled for a Kafka listener, all traffic for that channel will be encrypted with TLS, which employs digital certificates for identity verification

inter.broker.listener.name

security.inter.broker.protocol

KAFKA SECURITY

Kafka Authentication with SSL

Once we have the key and trust stores, we can configure TLS for brokers. Brokers require a trust store only if TLS is used for inter-broker communication or if client authentication is enabled:

```
ssl.keystore.location=/path/to/server.ks.p12
ssl.keystore.password=server-ks-password
ssl.key.password=server-ks-password
ssl.keystore.type=PKCS12
ssl.truststore.location=/path/to/server.ts.p12
ssl.truststore.password=server-ts-password
ssl.truststore.type=PKCS12
ssl.client.auth=required
```

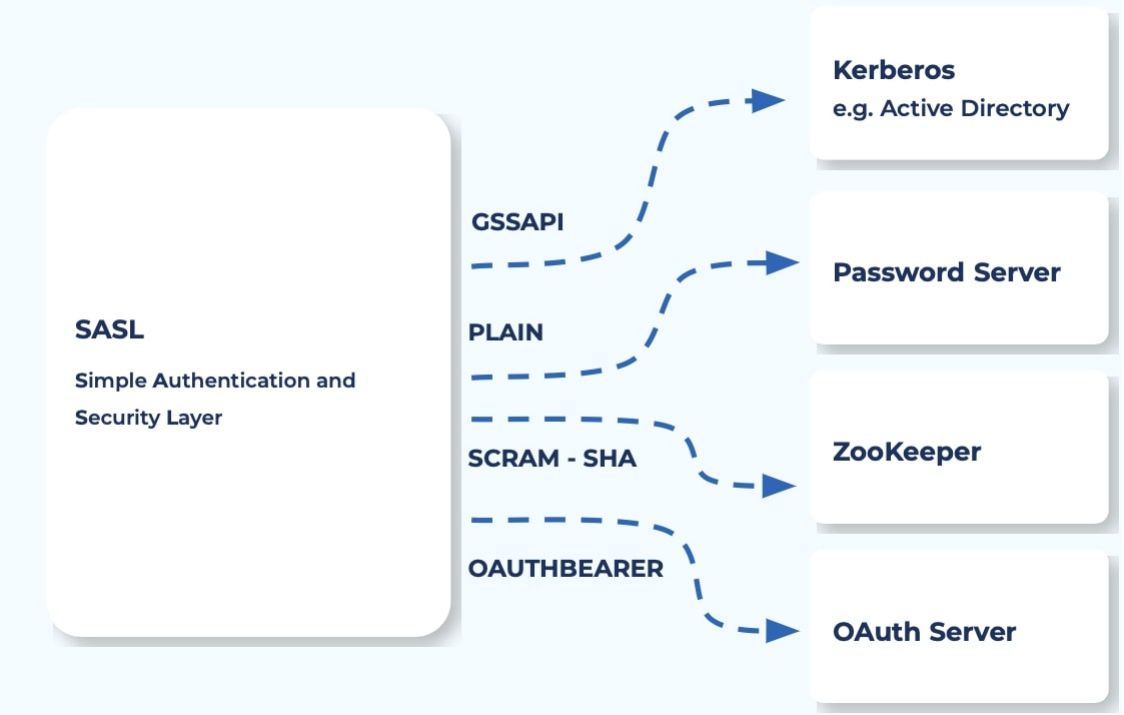
Clients are configured with the generated trust store. The key store should be configured for clients if client authentication is required.

```
ssl.truststore.location=/path/to/client.ts.p12
ssl.truststore.password=client-ts-password
ssl.truststore.type=PKCS12
ssl.keystore.location=/path/to/client.ks.p12
ssl.keystore.password=client-ks-password
ssl.key.password=client-ks-password
ssl.keystore.type=PKCS12
```

KAFKA SECURITY

Enable SASL_SSL for Kafka

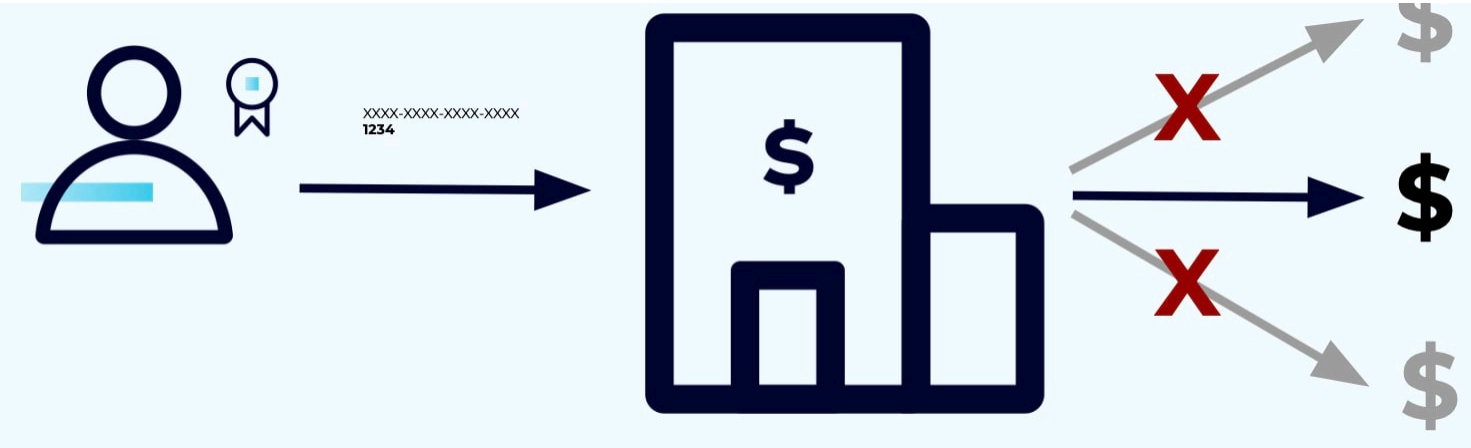
SASL-SSL (Simple Authentication and Security Layer) uses TLS encryption like SSL but differs in its authentication process. To use the protocol, you must specify one of the four authentication methods supported by Apache Kafka: GSSAPI, Plain, SCRAM-SHA-256/512, or OAUTHBEARER. One of the main reasons you might choose SASL-SSL over SSL is because you'd like to integrate Kafka, for example, with an existing Kerberos server in your organization, such as Active Directory or LDAP.



KAFKA SECURITY

Authorization

Authorization detemines what an entity can do on a system once it has been authenticated.
Consider an ATM, once you successfully authenticate with your card and PIN, you are able to access your account only, not all



KAFKA SECURITY

ACLs

KAFKA SECURITY

ACLs

ACL	Kafka requests	Notes
`Cluster:ClusterAction`	Inter-broker requests, including controller requests and follower fetch requests for replication	Should only be granted to brokers.
`Cluster:Create`	`CreateTopics` and auto-topic creation	Use `Topic:Create` for fine-grained access control to create specific topics.
`Cluster:Alter`	`CreateAcls`, `DeleteAcls`, `AlterReplicaLogDirs`, `ElectReplicaLeader`, `AlterPartitionReassignments`	
`Cluster:AlterConfigs`	`AlterConfigs` and `IncrementalAlterConfigs` for broker and broker logger, `AlterClientQuotas`	
`Cluster:Describe`	`DescribeAcls`, `DescribeLogDirs`, `ListGroups`, `ListPartitionReassignments`, describing authorized operations for cluster in Metadata request	Use `Group:Describe` for fine-grained access control for ListGroups.
`Cluster:DescribeConfigs`	`DescribeConfigs` for broker and broker logger, `DescribeClientQuotas`	
`Cluster:IdempotentWrite`	Idempotent `InitProducerId` and `Produce` requests	Only required for nontransactional idempotent producers.

KAFKA SECURITY

ACLs

`Topic:Create`	`CreateTopics` and auto-topic creation
`Topic:Delete`	`DeleteTopics`, `DeleteRecords`
`Topic:Alter`	`CreatePartitions`
`Topic:AlterConfigs`	`AlterConfigs` and `IncrementalAlterConfigs` for topics
`Topic:Describe`	Metadata request for topic, `OffsetForLeaderEpoch`, `ListOffset`, `OffsetFetch`
`Topic:DescribeConfigs`	`DescribeConfigs` for topics, for returning configs in `CreateTopics` response

KAFKA SECURITY

ACLs

ACL	Kafka requests	Notes
`Topic:Read`	`Consumer Fetch`, `OffsetCommit`, `TxnOffsetCommit`, `OffsetDelete`	Should be granted to consumers.
`Topic:Write`	`Produce`, `AddPartitionToTxn`	Should be granted to producers.
`Group:Read`	`JoinGroup`, `SyncGroup`, `LeaveGroup`, `Heartbeat`, `OffsetCommit`, `AddOffsetsToTxn`, `TxnOffsetCommit`	Required for consumers using consumer group management or Kafka-based offset management. Also required for transactional producers to commit offsets within a transaction.
`Group:Describe`	`FindCoordinator`, `DescribeGroup`, `ListGroups`, `OffsetFetch`	
`Group:Delete`	`DeleteGroups`, `OffsetDelete`	
`TransactionalId:Write`	`Produce` and `InitProducerId` with transactions, `AddPartitionToTxn`, `AddOffsetsToTxn`, `TxnOffsetCommit`, `EndTxn`	Required for transactional producers.
`TransactionalId:Describe`	`FindCoordinator` for transaction coordinator	
`DelegationToken:Describe`	`DescribeTokens`	

KAFKA SECURITY

Creating ACLs

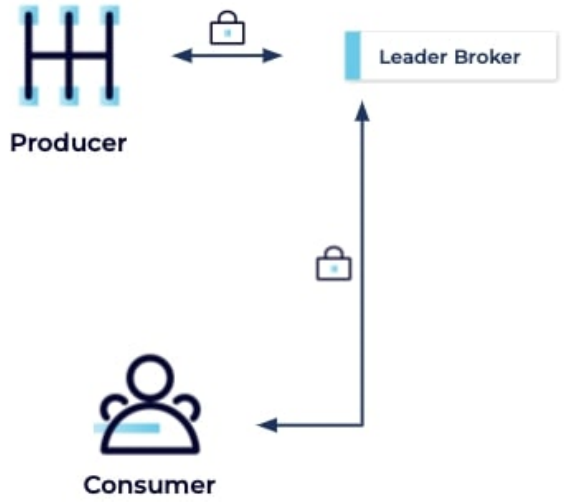
Kafka-acts

```
kafka-acls --bootstrap-server localhost:9092 \
  --command-config adminclient-configs.conf  \
  --add \
  --allow-principal User:alice \
  --allow-principal User:fred \
  --operation read
  --operation write \
  --topic finance`
```

KAFKA SECURITY

Encryption: Client/Cluster

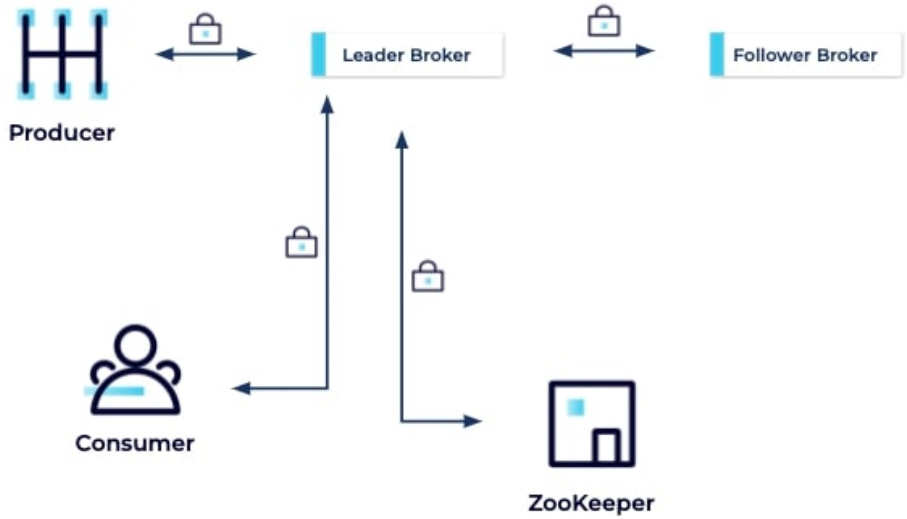
The simplest encryption setup consists of encrypted traffic between clients and the cluster, which is important if clients access the cluster through an unsecured network such as the public internet:



KAFKA SECURITY

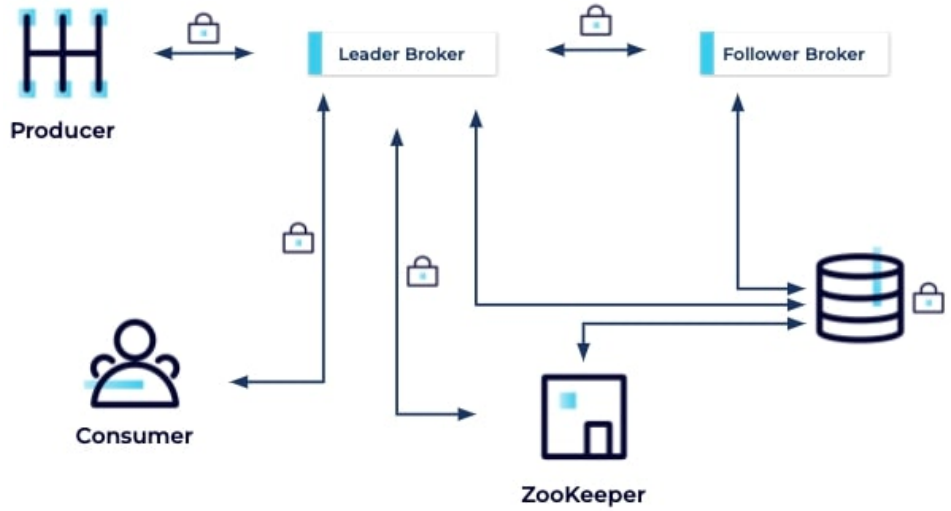
Encryption: Client/Cluster - Inter-Broker - Broker/Zookeeper

The next thing to consider is encrypting traffic between brokers, and between brokers and ZooKeeper. Even private networks can be breached, so you want to be sure the traffic on your private network is resistant to eavesdroppers or anyone who wishes to tamper with it while it is in motion.



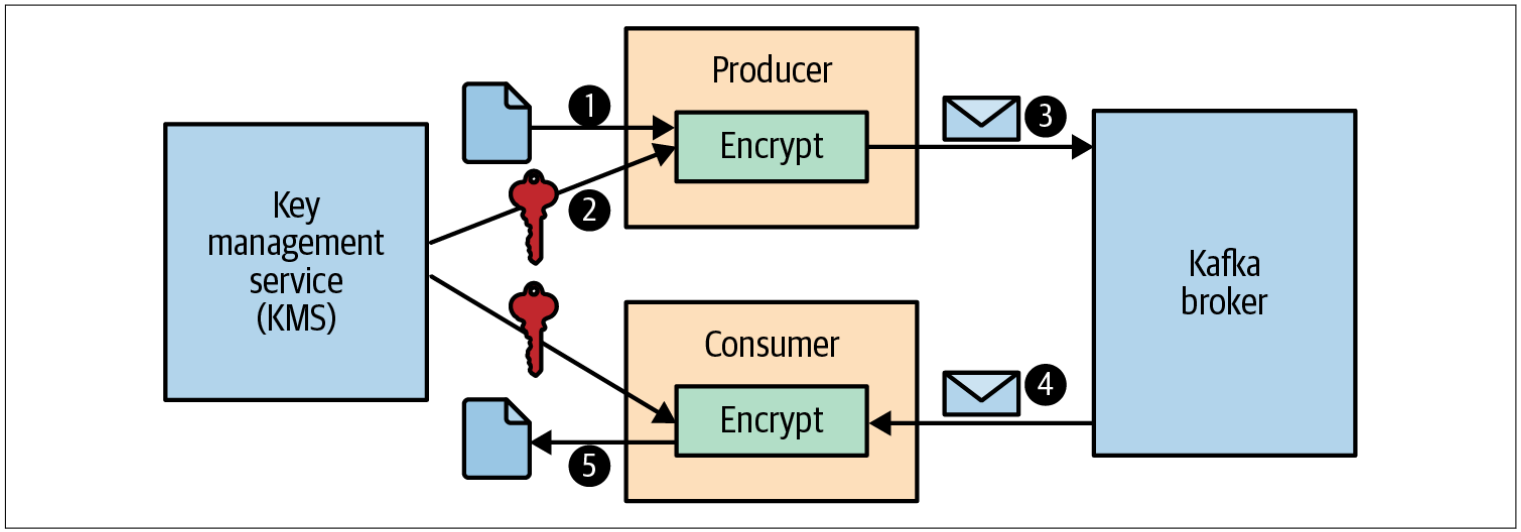
KAFKA SECURITY

Encryption:Client/Cluster – Inter-Broker – Broker/ZooKeeper – Data at Rest



KAFKA SECURITY

End-to-end encryption with Key Management Service



KAFKA SECURITY

Hands On: Setting Up Encryption

KAFKA SECURITY

Hands On: Requiring Encryption for Broker Traffic

KAFKA SECURITY

Securing Zookeeper

SSL client Authentication:

- Each broker and CLI must use same Distinguished Name for Authorization
- Use wildcard certificates OR Subject Alternative Name with list of broker hostnames

SASL:

- Integrate with Kerberos
- Use TLS encryption
  - ssl.clientAuth=none in the Zookeeper configuration
- Configure each broker with same Kerberos principal

KAFKA SECURITY

Securing Zookeeper

SSL + SASL

- Able to use either identity
- No need to use same distinguished name
- Ability to use hostnames in the distinguished name

KAFKA SECURITY

Audit logs

1. Insight - Logging every attempted operation
2. Security - Monitoring and validating operations
3. Impact - Debugging client interactions
4. Compliance - Generating audit reports

KAFKA SECURITY

Security Recommendation

1. Encrypt the file system
2. Secure data in transit
3. Set up a system for administering ACLs
4. Rotate your keys
5. Dynamically update certificates
6. Enable reauthentication
7. Protect Zookeeper
8. Setup and monitor audit logs
9. Play, tinker, break things

CONCLUSION

Quality Network for Education and Technology

XIN CHÂN THÀNH CẢM ƠN!