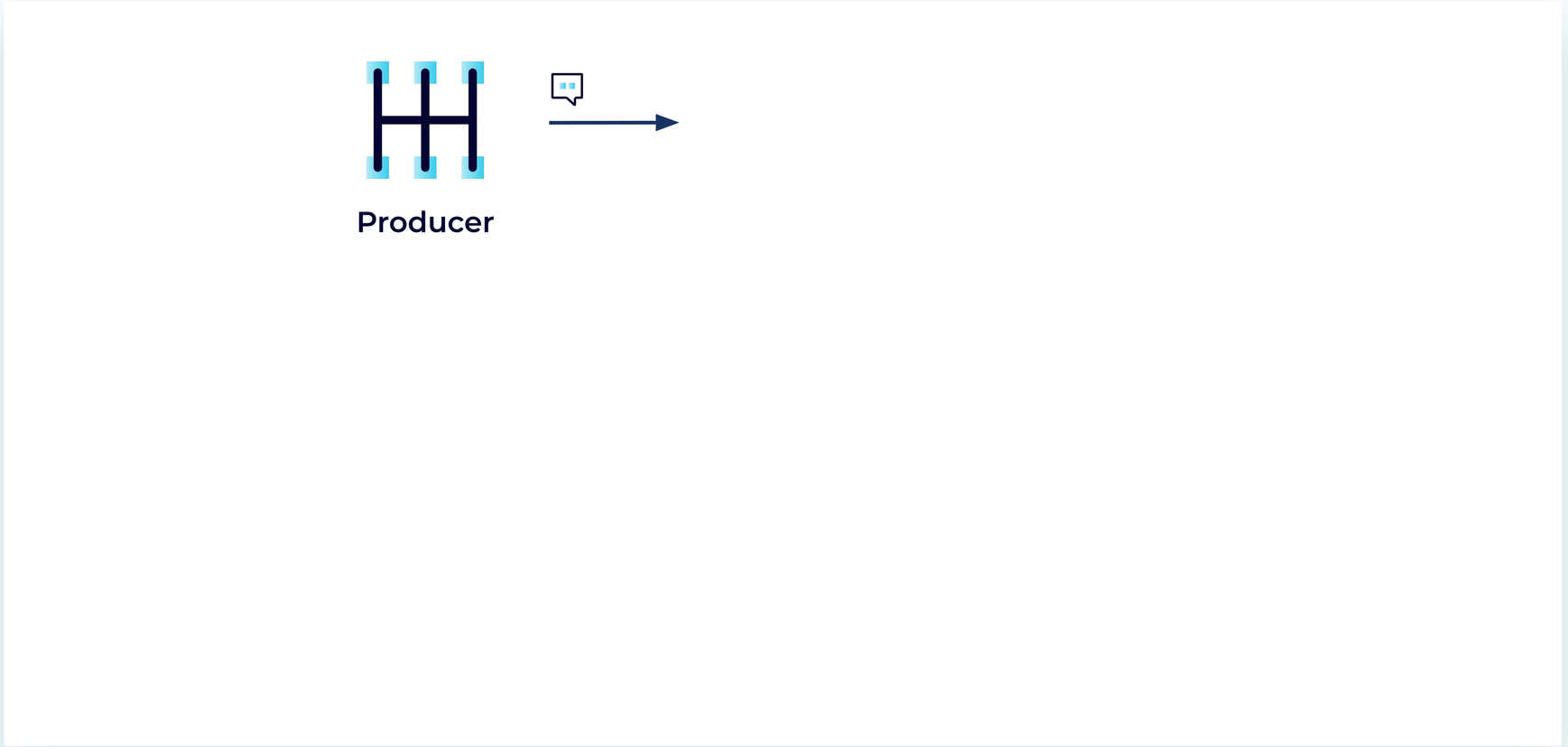
CONFLUENT
Developer

Introduction to Kafka Security

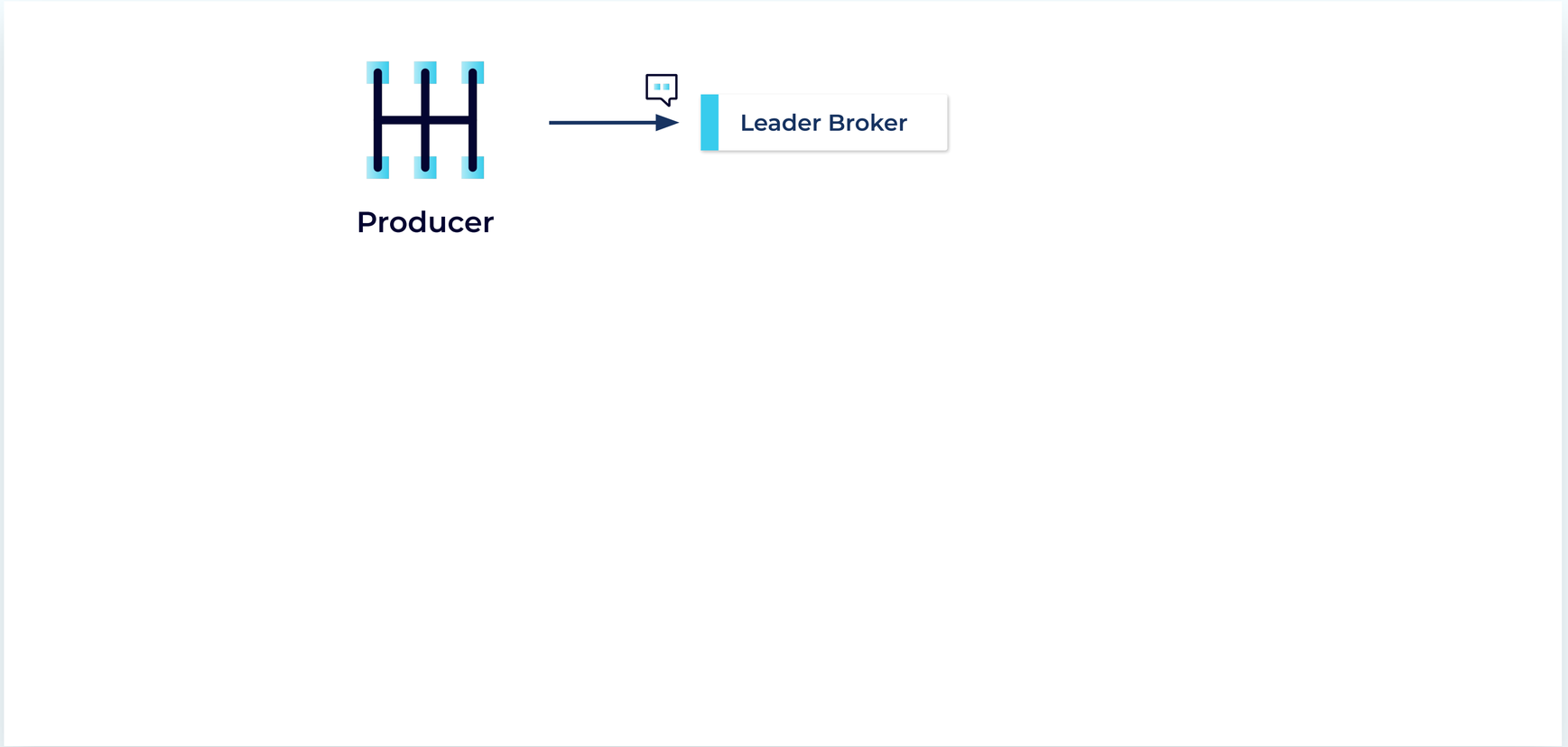
Data Flow in Kafka

CONFLUENT Developer



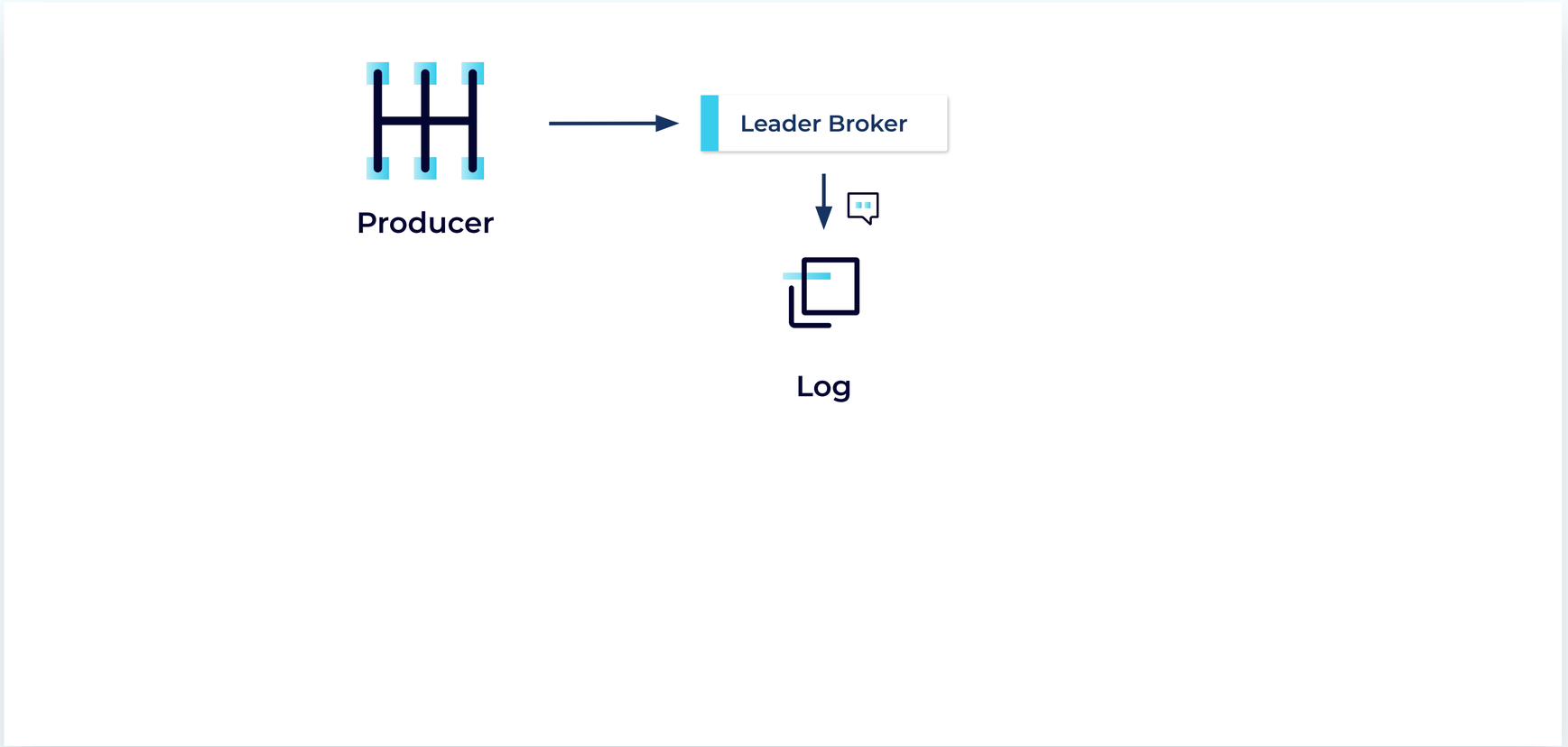
Data Flow in Kafka

CONFLUENT Developer



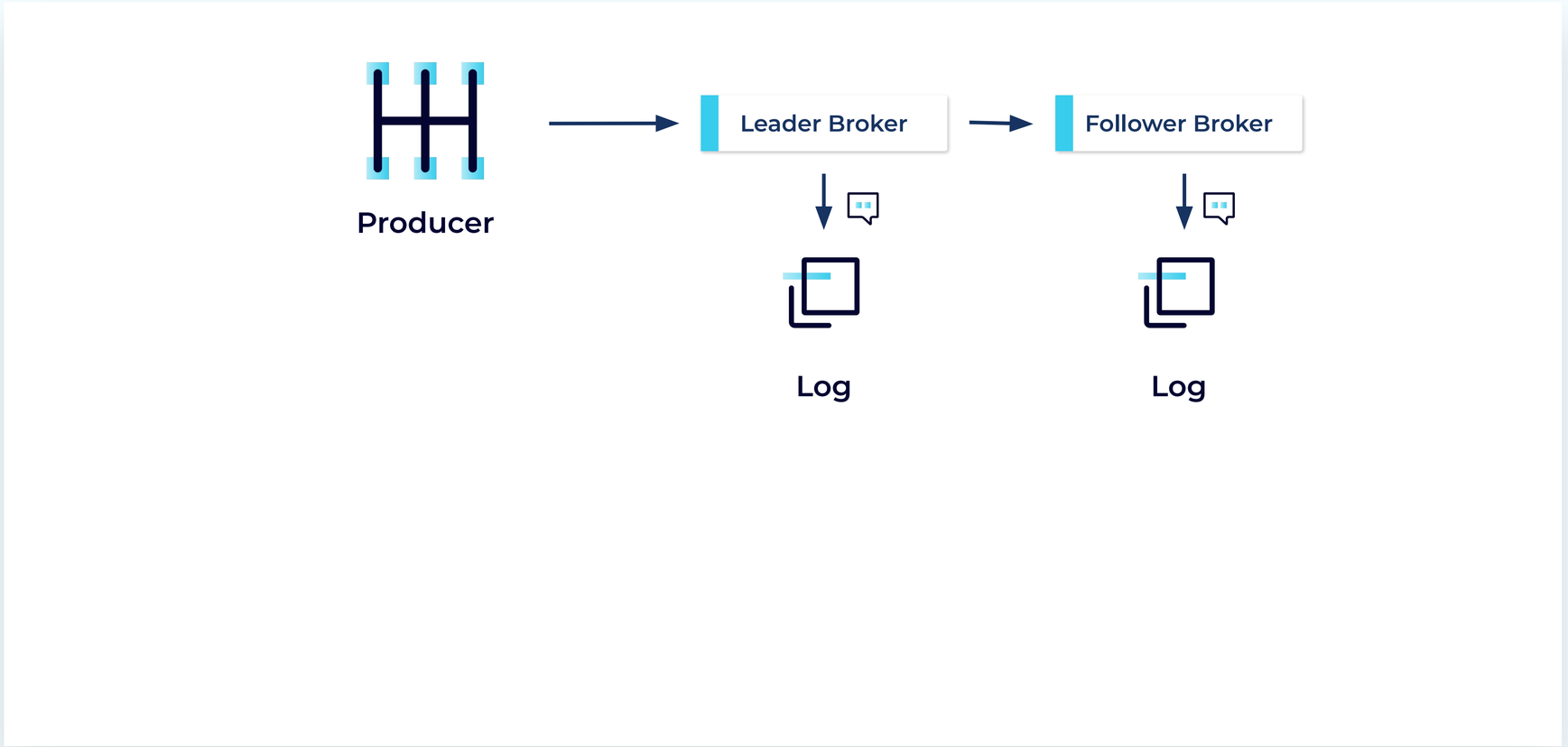
Data Flow in Kafka

CONFLUENT Developer



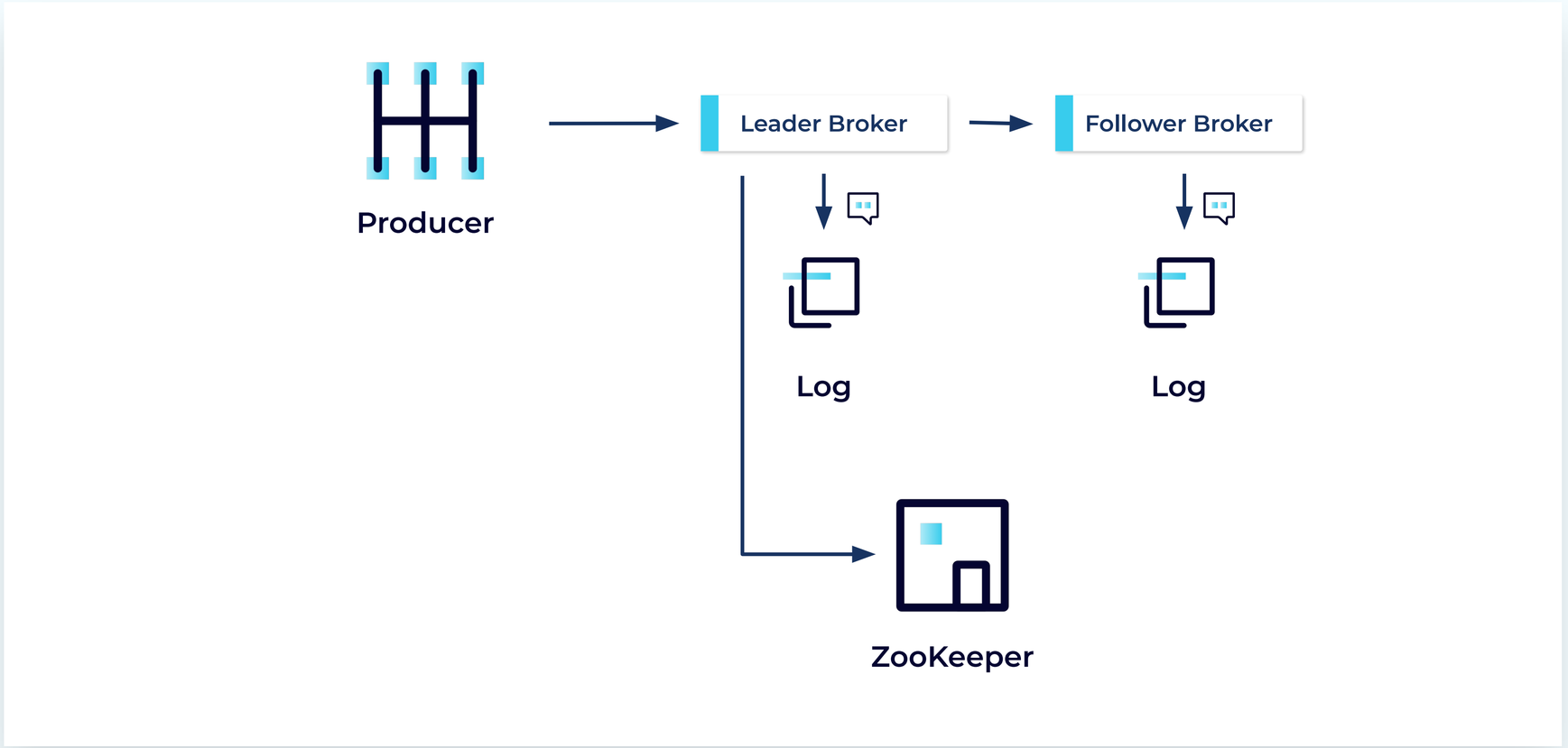
Data Flow in Kafka

CONFLUENT Developer



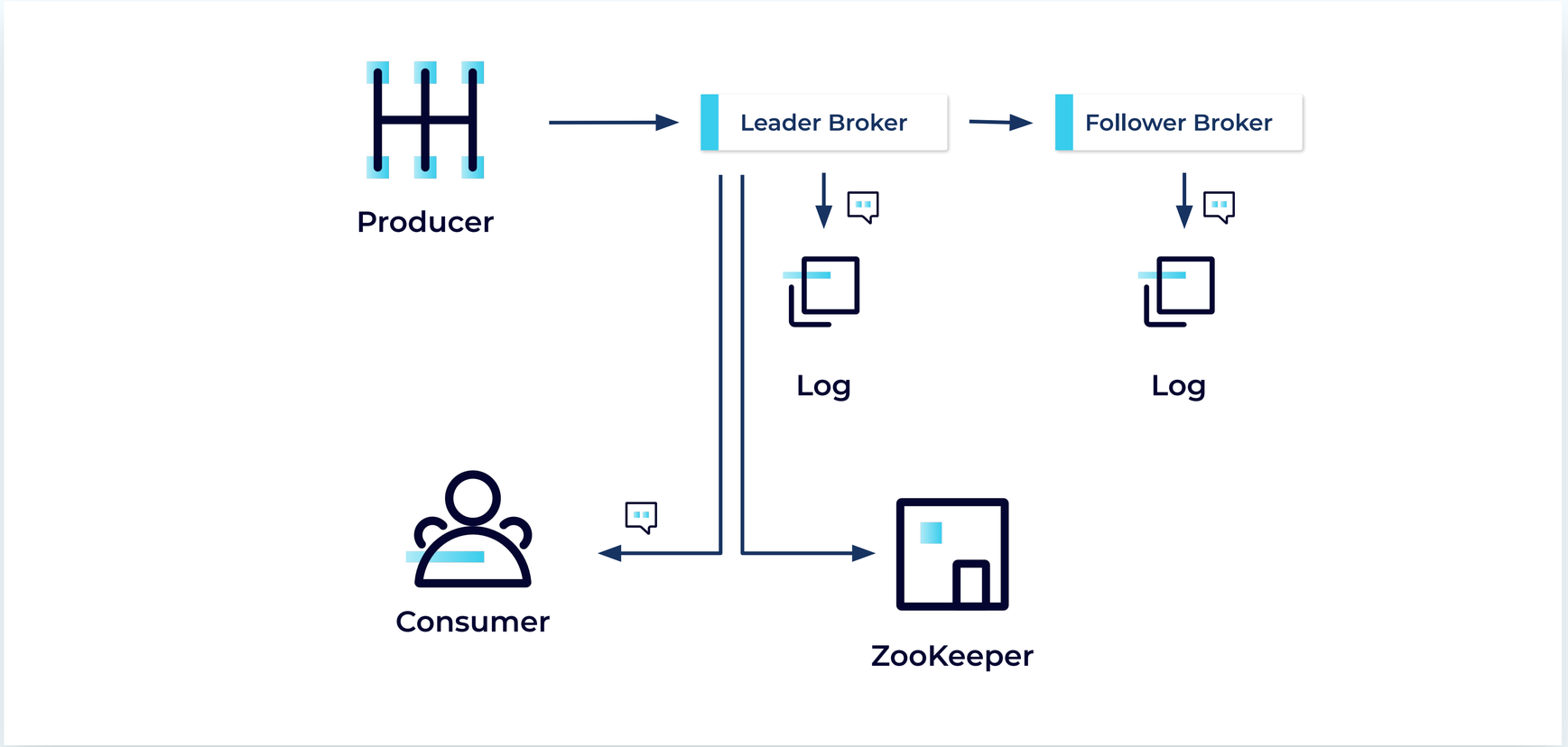
Data Flow in Kafka

CONFLUENT Developer



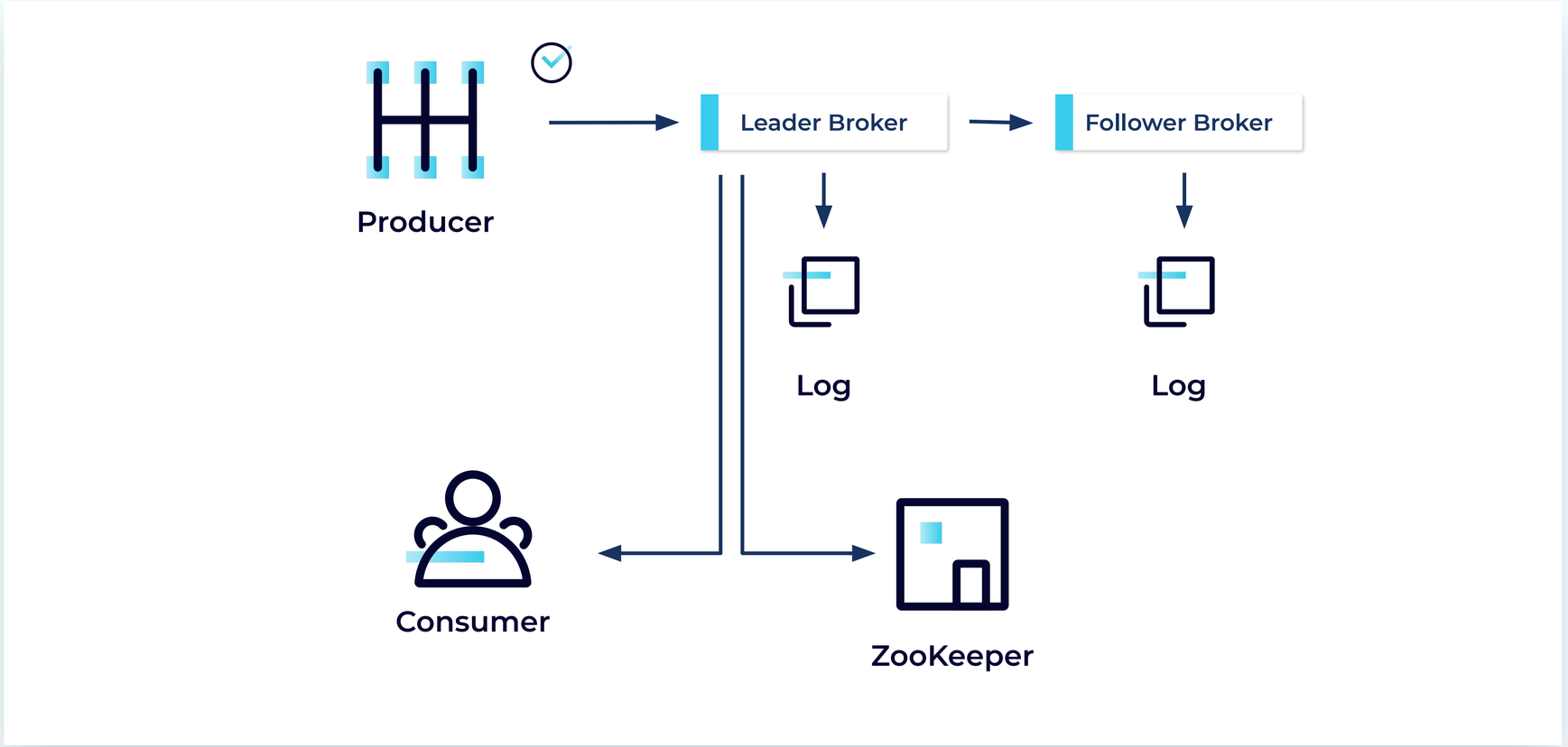
Data Flow in Kafka

CONFLUENT Developer



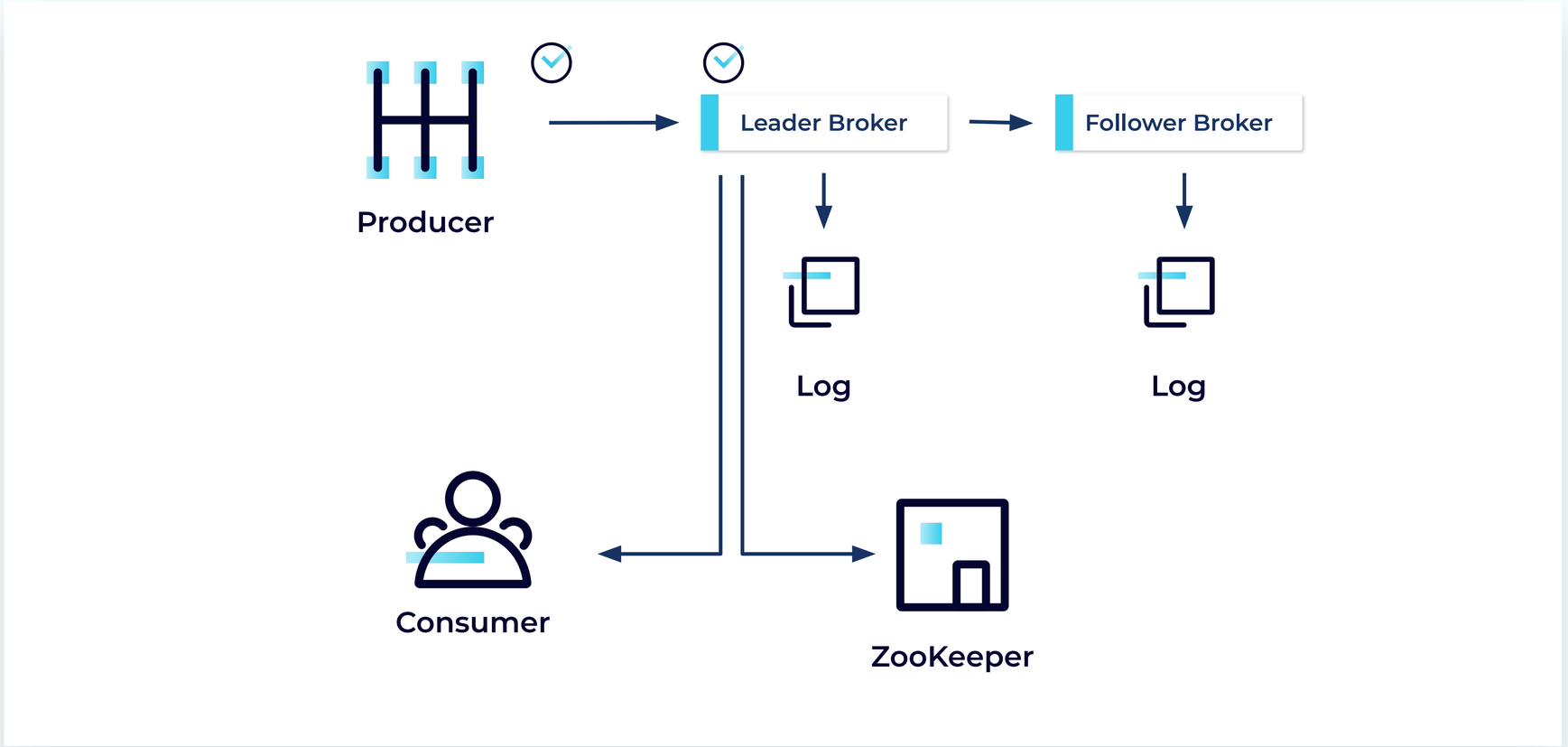
Data Flow in Kafka

CONFLUENT Developer



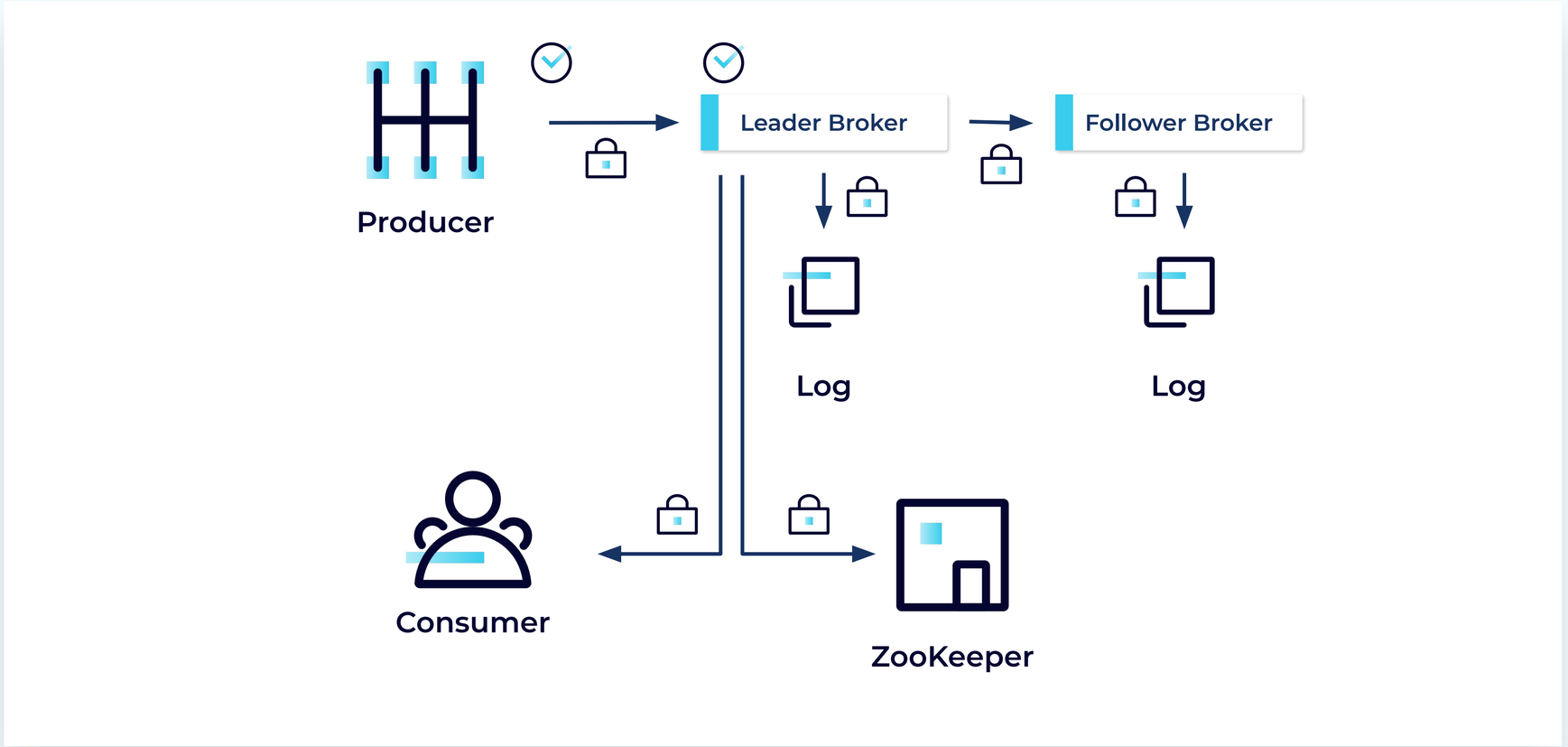
Data Flow in Kafka

CONFLUENT Developer



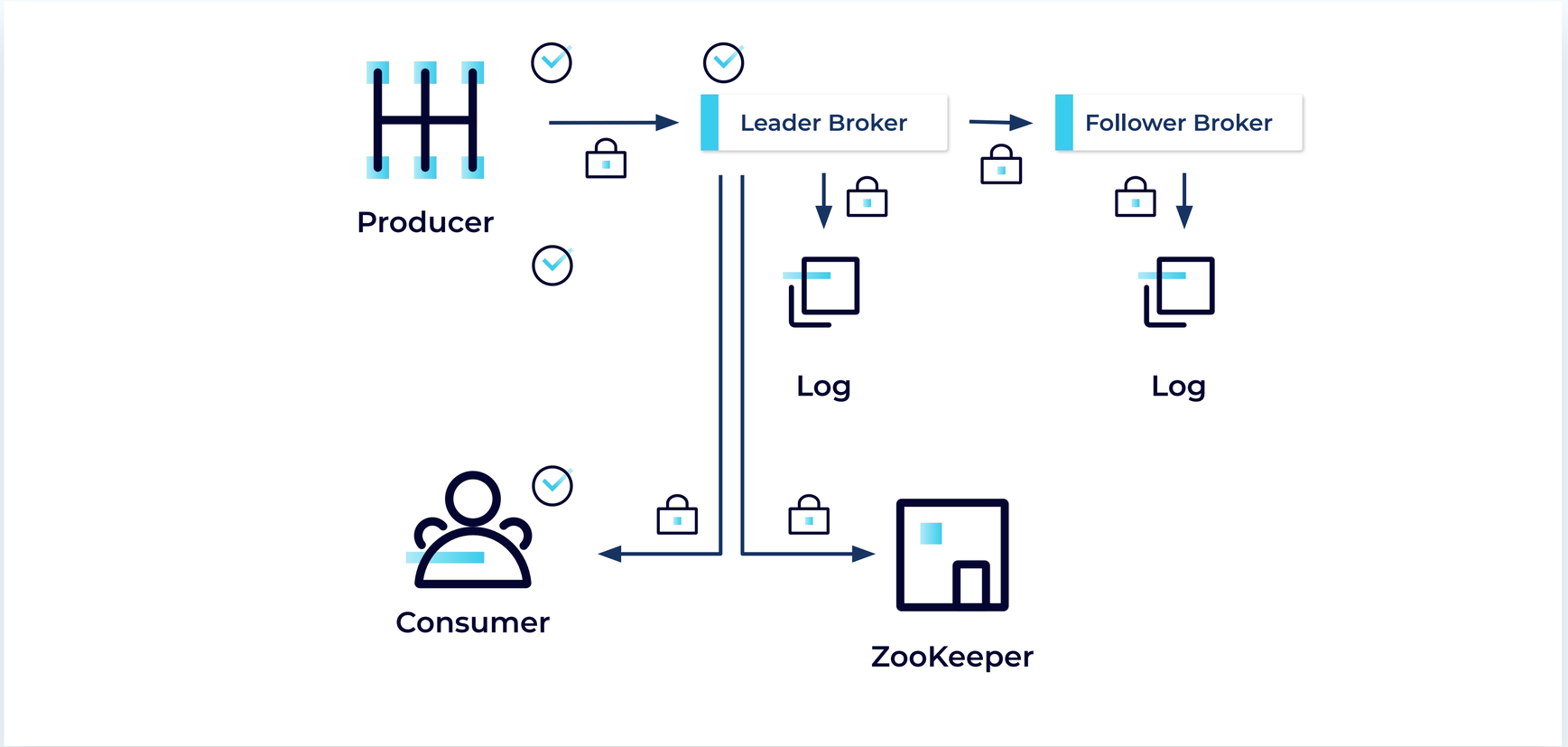
Data Flow in Kafka

CONFLUENT Developer



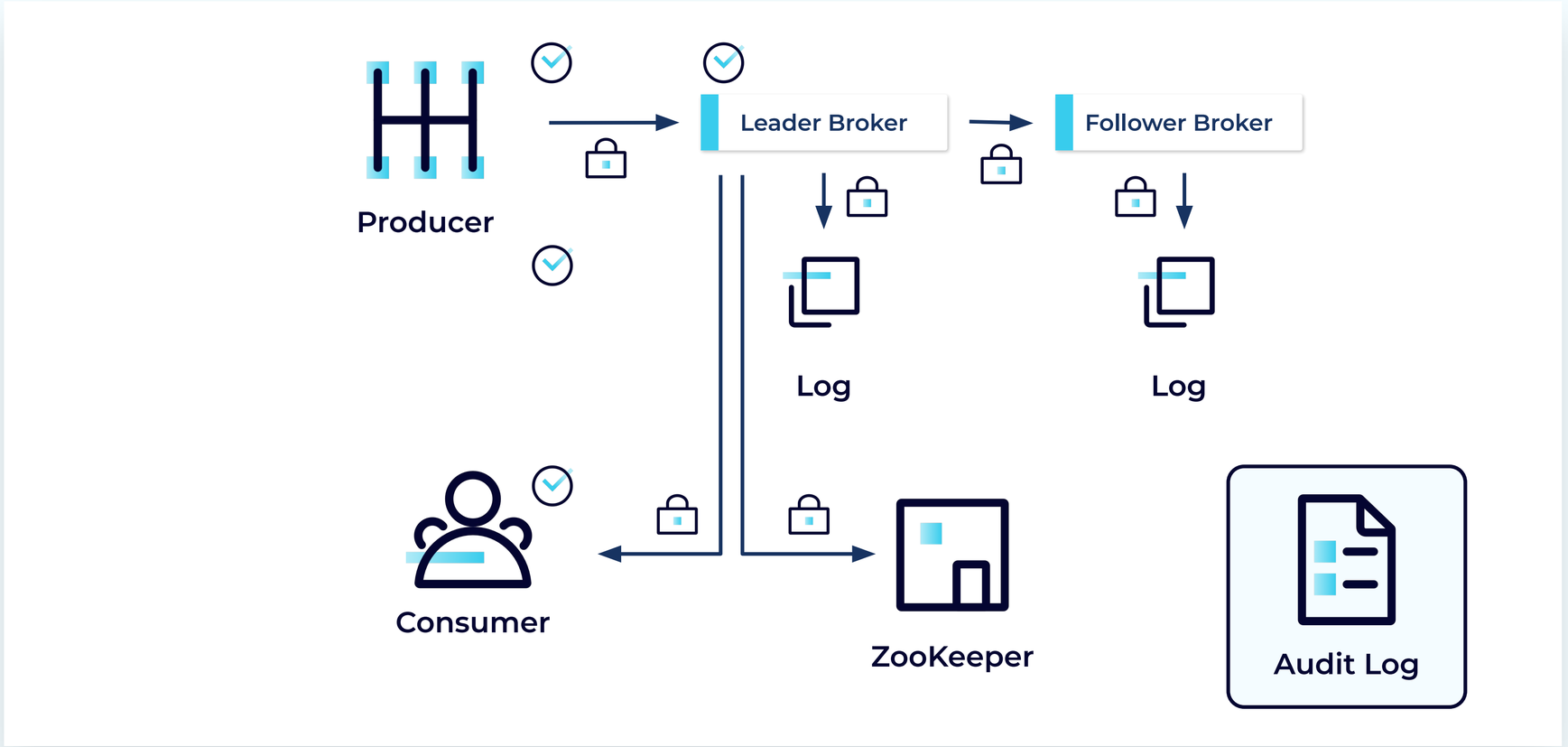
Data Flow in Kafka

CONFLUENT Developer



Data Flow in Kafka

CONFLUENT Developer

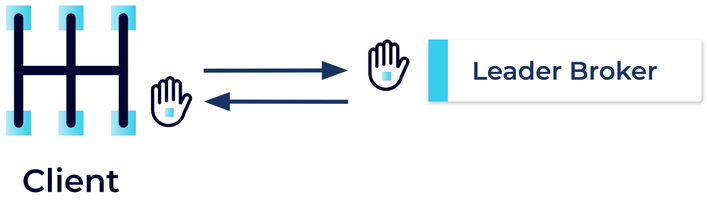


CONFLUENT
Developer

Kafka Authentication Basics

Data Flow in Kafka

CONFLUENT Developer

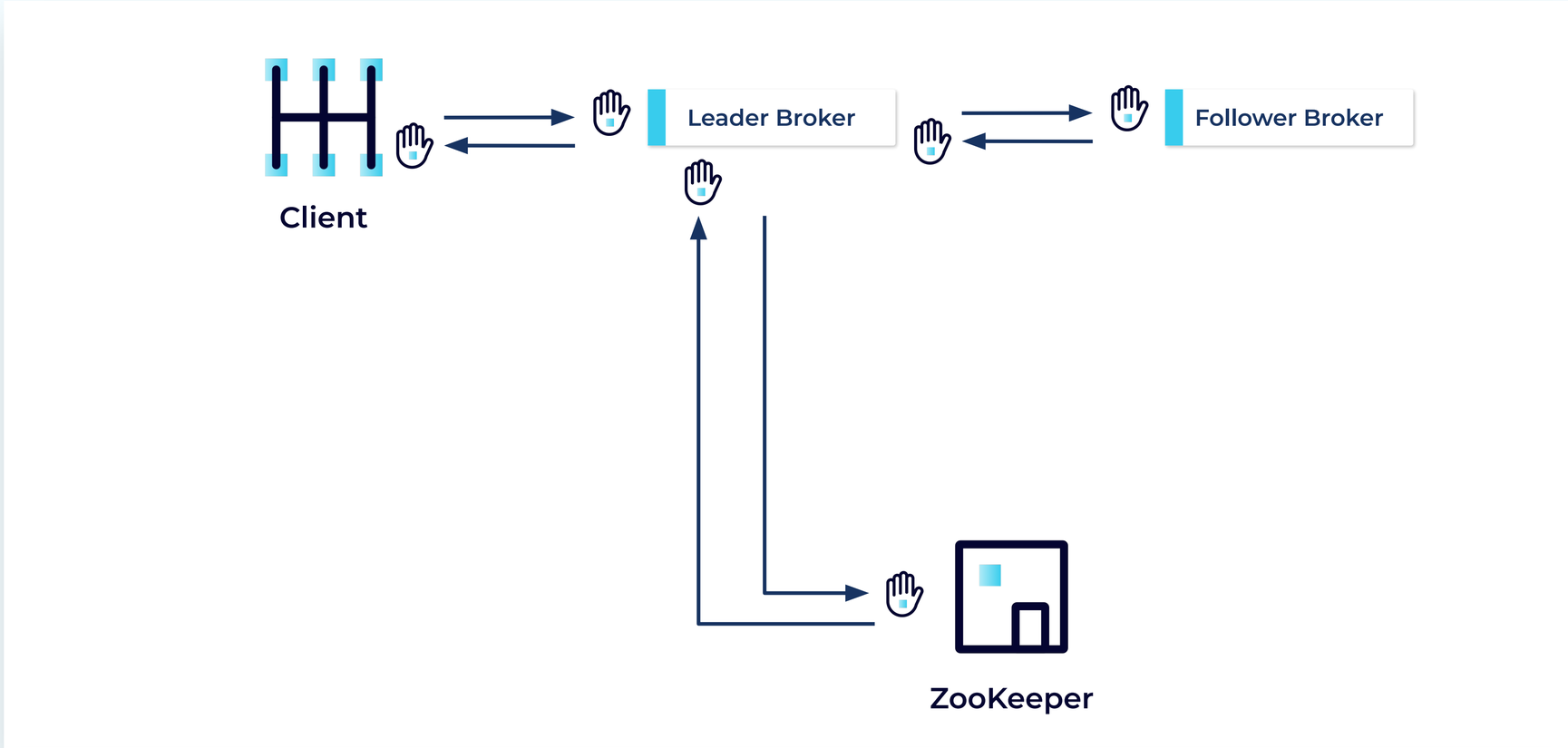


Data Flow in Kafka

CONFLUENT Developer

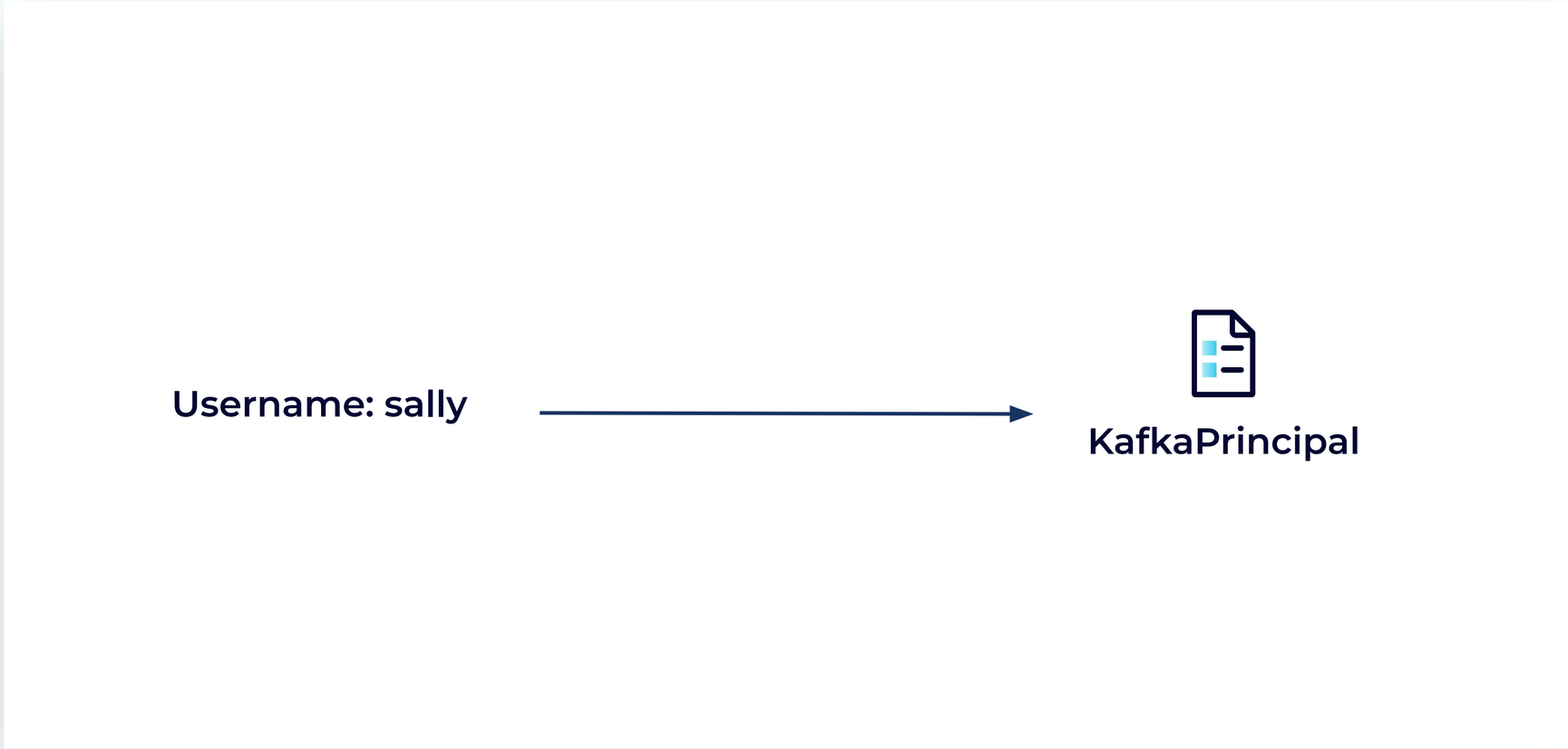
Data Flow in Kafka

CONFLUENT Developer



Data Flow in Kafka

CONFLUENT Developer



Interaction Types

CONFLUENT Developer

Clients

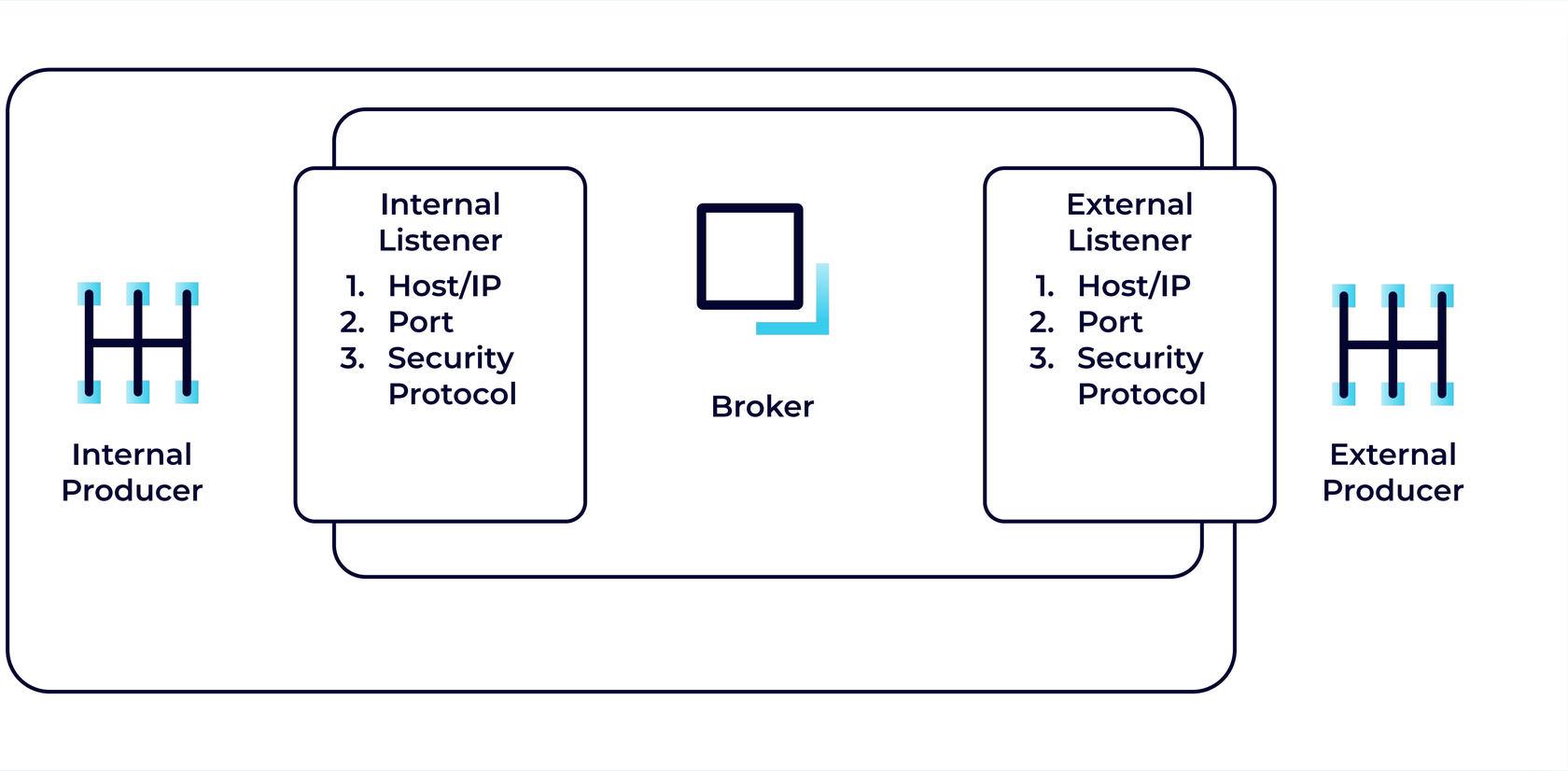
- Users
- Applications
- Services

Brokers

- Broker-to-Broker communication

Listeners and Security Protocols

CONFLUENT Developer



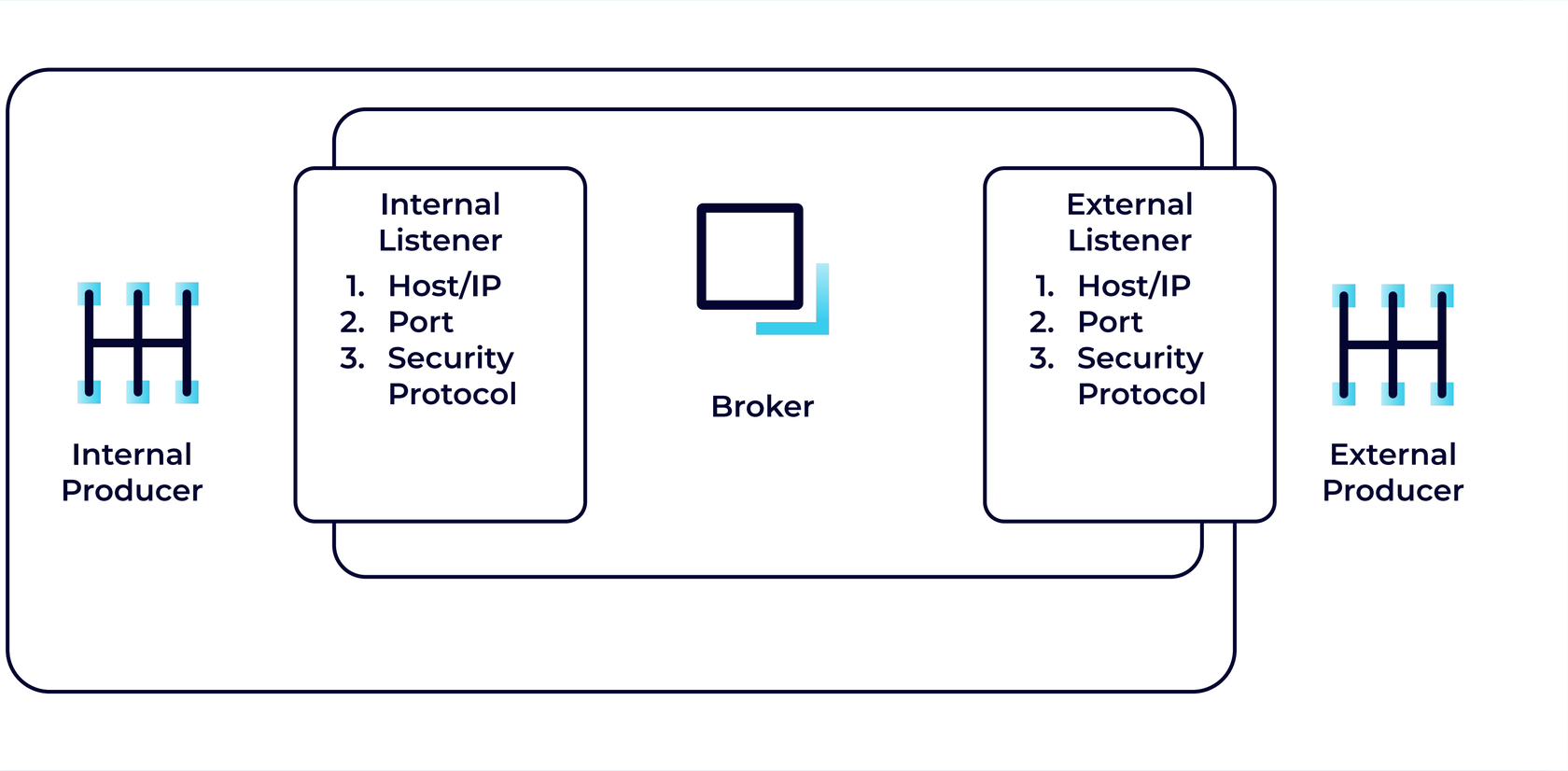
Securing Kafka

Security Protocols

- PLAINTEXT - not secure
- SASL_PLAINTEXT - not secure
- SSL - secure
- SASL_SSL - secure

Listeners and Security Protocols

CONFLUENT Developer



Configuring listeners on the broker

CONFLUENT Developer

Broker configuration

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

Configuring the client

CONFLUENT Developer

Client configuration

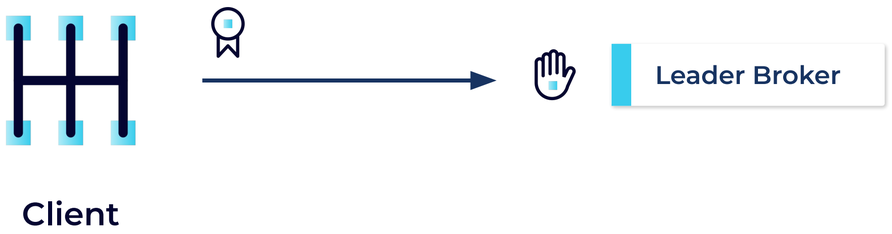
security.protocol=SASL_SSL

bootstrap.servers=broker1.example.com:9092,broker2.example.com:9092

CONFLUENT
Developer

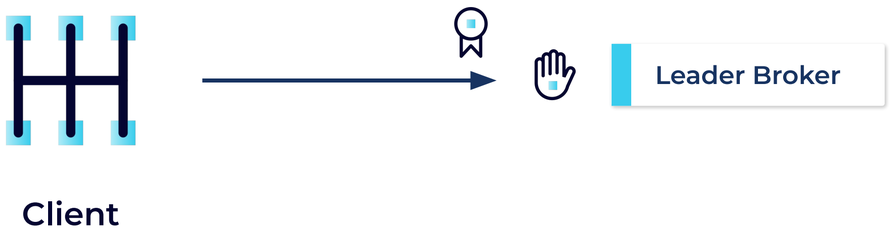
Kafka Authentication with SSL and SASL_SSL

SSL

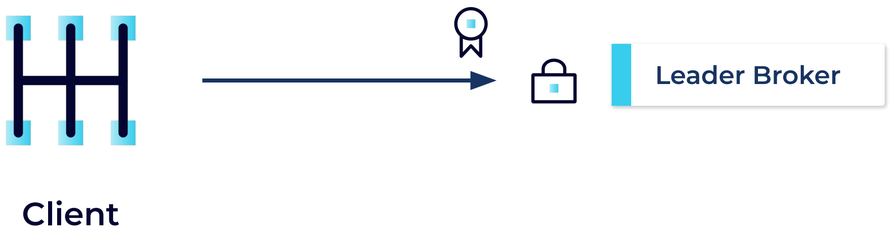


SSL

CONFLUENT
Developer



SSL

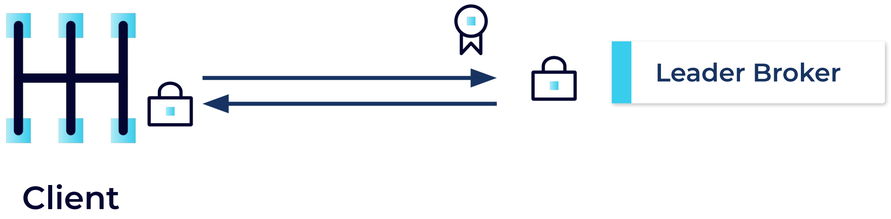


SSL

CONFLUENT Developer

Broker configuration

ssl.client.auth=required



Inter-broker Authentication

CONFLUENT Developer

inter.broker.listener.name

security.inter.broker.protocol

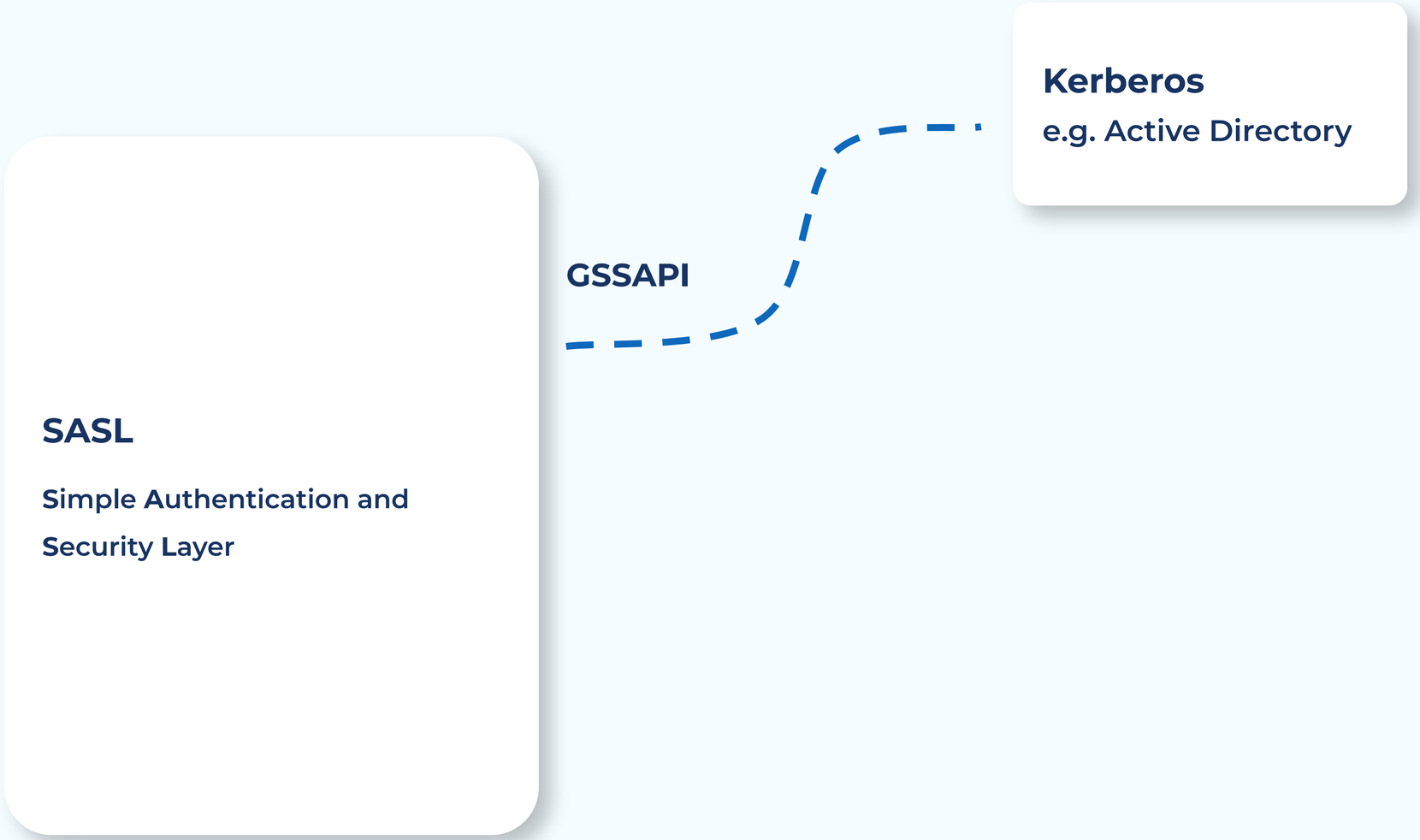
SASL_SSL

SASL

Simple Authentication and Security Layer

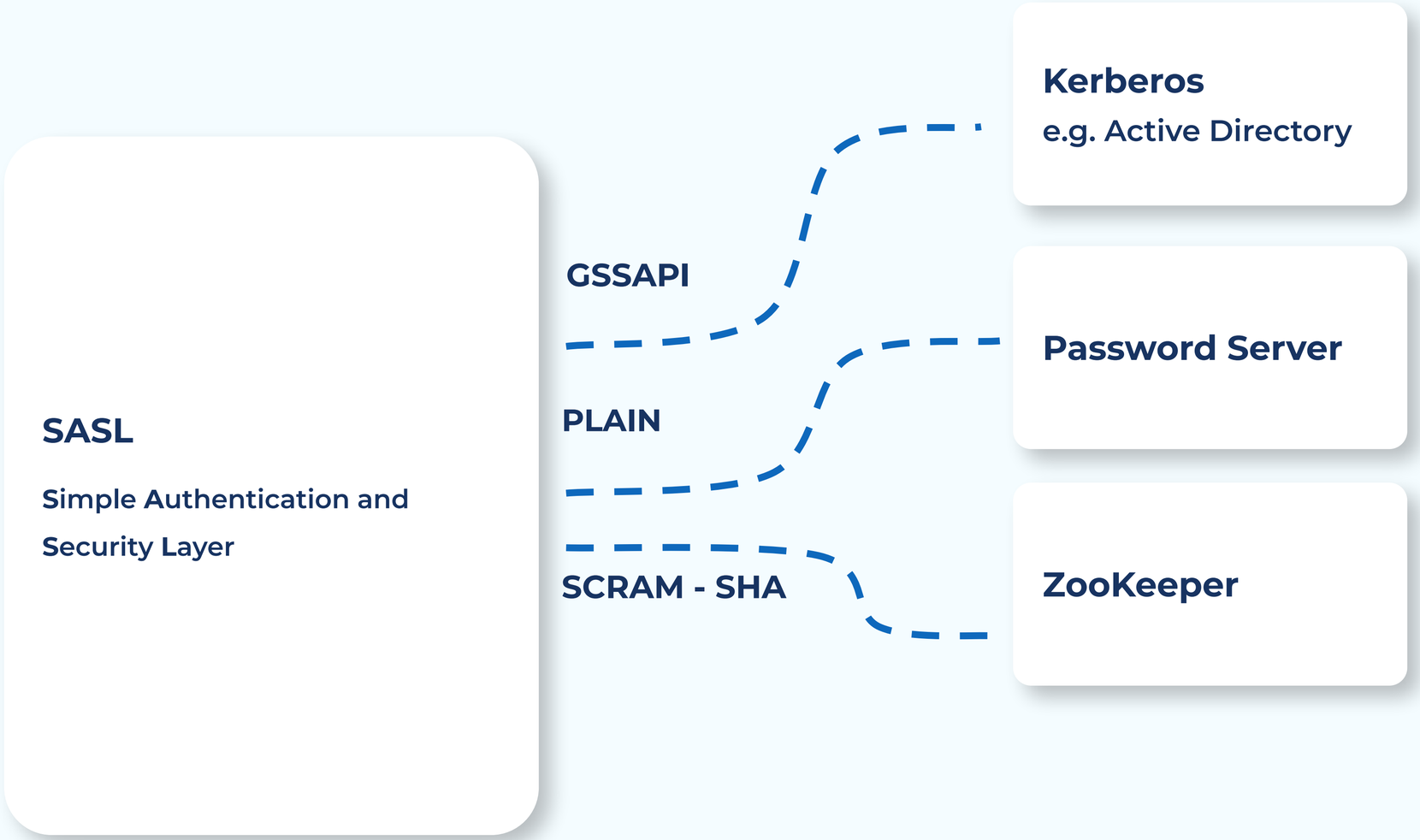
SASL_SSL

CONFLUENT Developer



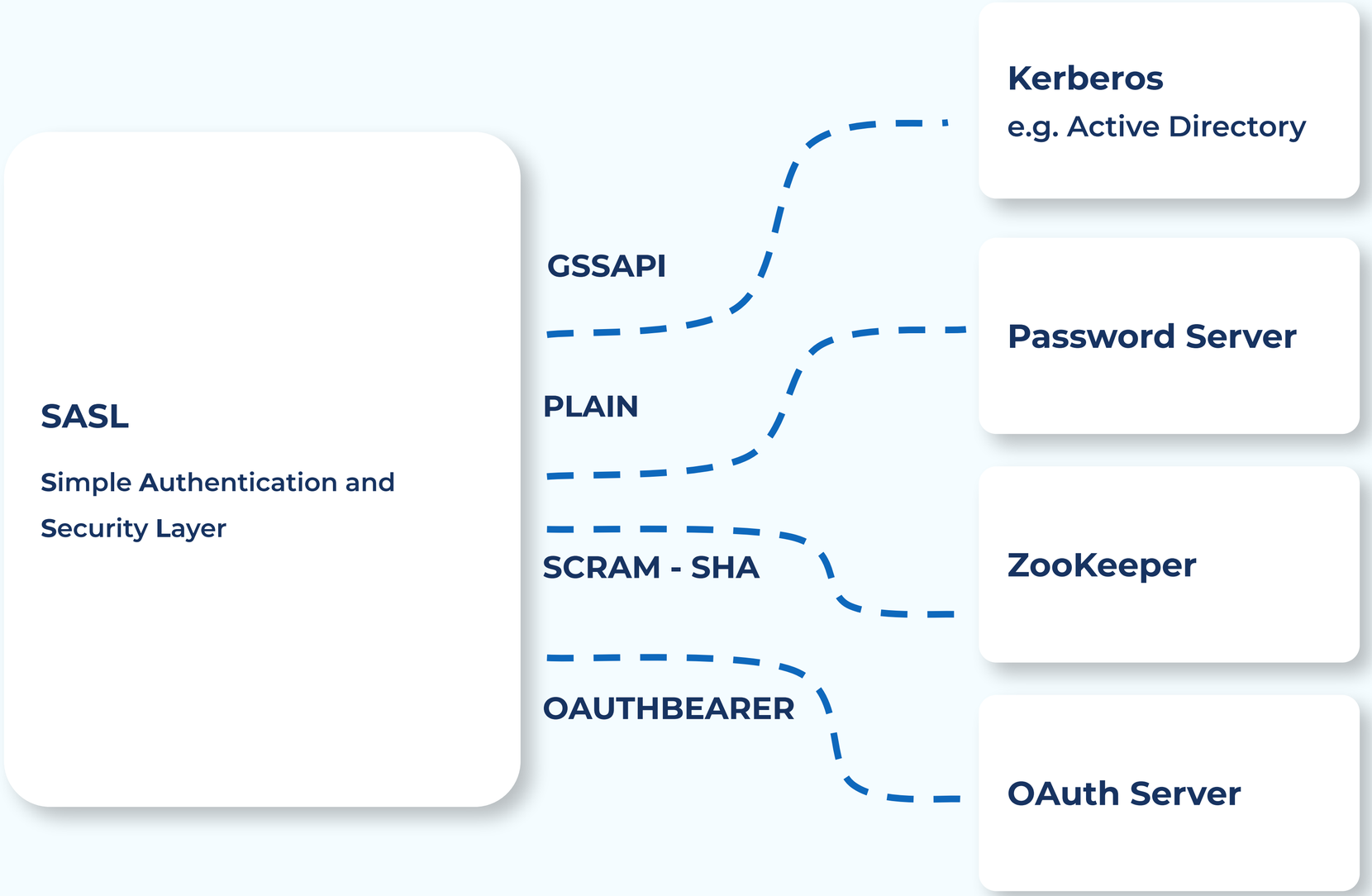
SASL_SSL

CONFLUENT Developer



SASL_SSL

CONFLUENT Developer



Considerations

CONFLUENT Developer

- Use filesystem permissions to restrict access to files containing security information

Considerations

- Use filesystem permissions to restrict access to files containing security information
- Avoid storing passwords in plaintext anywhere on the system

Considerations

- Use filesystem permissions to restrict access to files containing security information
- Avoid storing passwords in plaintext anywhere on the system
- Use disk encryption or a secure credential store

Considerations

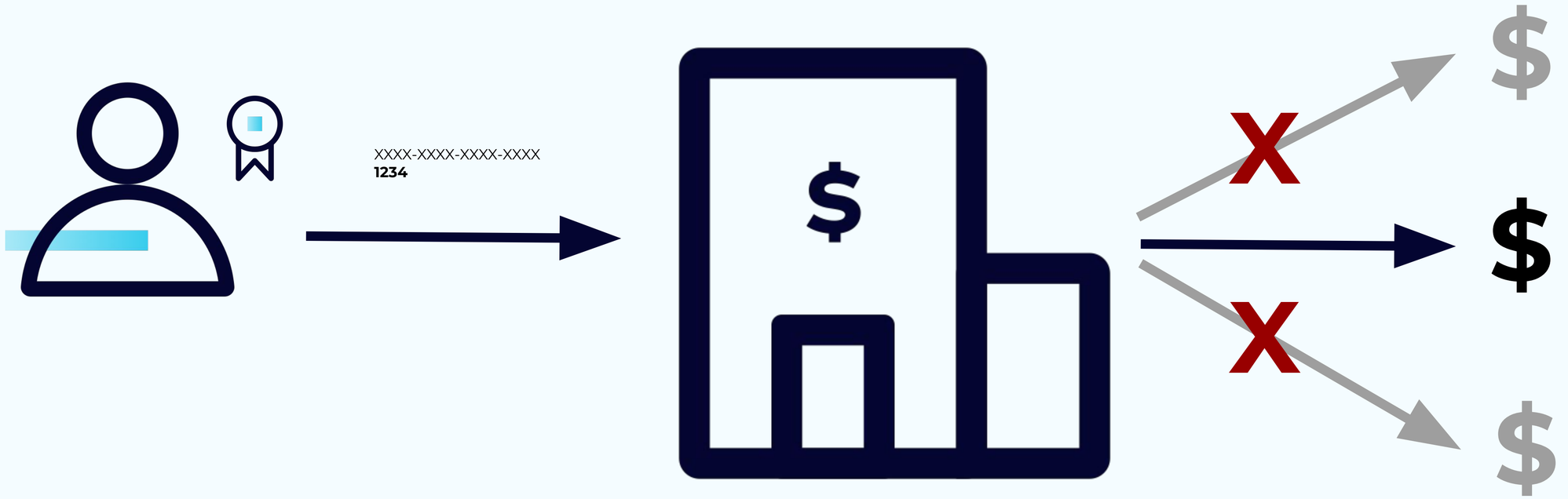
- Use filesystem permissions to restrict access to files containing security information
- Avoid storing passwords in plaintext anywhere on the system
- Use disk encryption or a secure credential store
- Use quotas to limit the impact of malicious client behaviours

Considerations

- Use filesystem permissions to restrict access to files containing security information
- Avoid storing passwords in plaintext anywhere on the system
- Use disk encryption or a secure credential store
- Use quotas to limit the impact of malicious client behaviours
- Apply change control to configuration

CONFLUENT
Developer

Authorization


XXXX-XXXX-XXXX-XXXX
1234
$
X
$
X
$
$

Access control lists (ACLs)

CONFLUENT Developer

Access control lists (ACLs)

CONFLUENT Developer

Pattern type: Literal or Prefixed
Resource name: Supports *
Principal: Supports User:*
Host: Source IP or *

Creating ACLs


```
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
```

Real-world principal names

CONFLUENT Developer

SSL certificate subject name

User:CN=tracking.example.org,OU=TEST,O=Sales,L=PaloAlto,ST=Ca,C=US

Kerberos

User:tracking

SSL subject names to principals

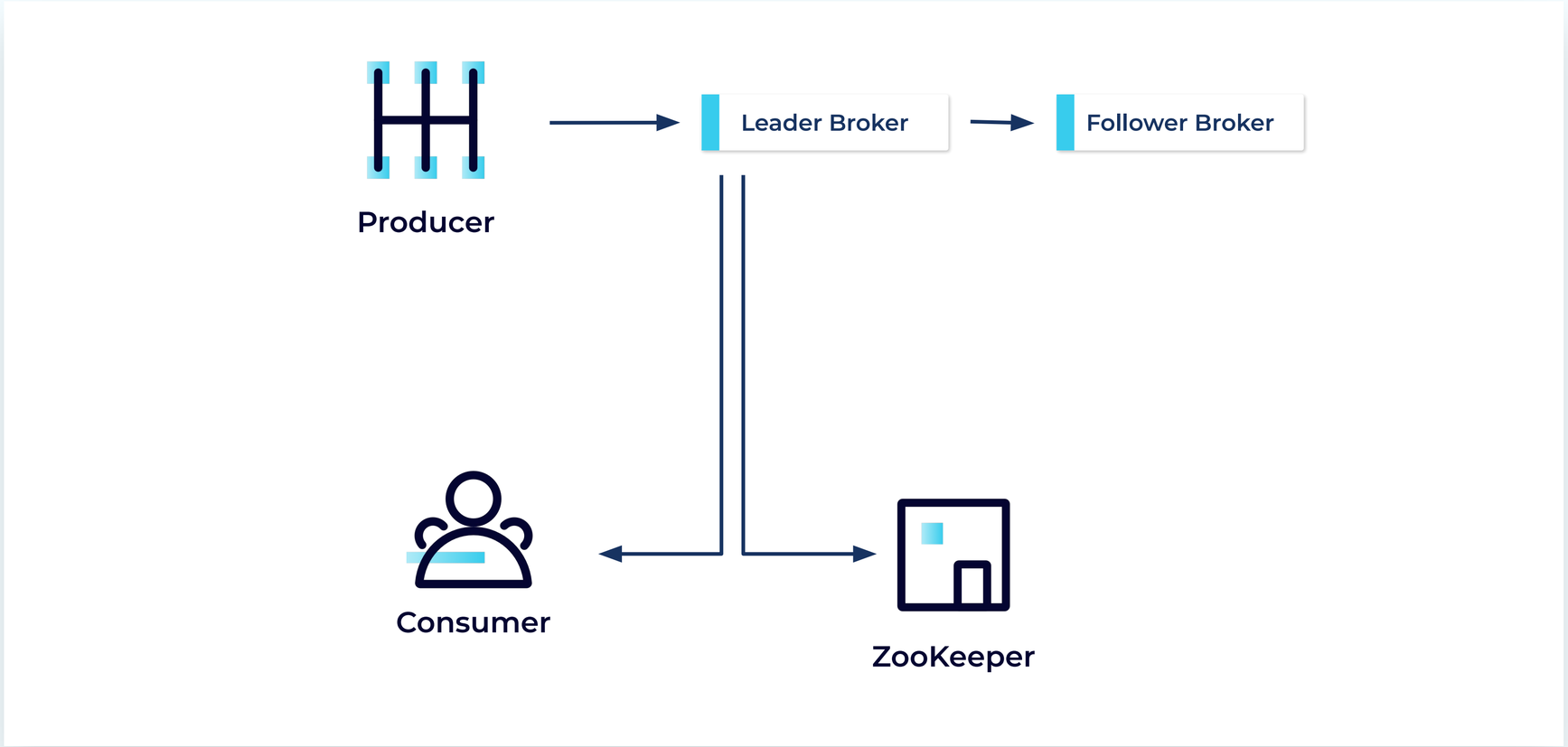
ssl.principal.mapping.rules

Kerberos identities to preferred principal format

sasl.kerberos.principal.to.local.rules

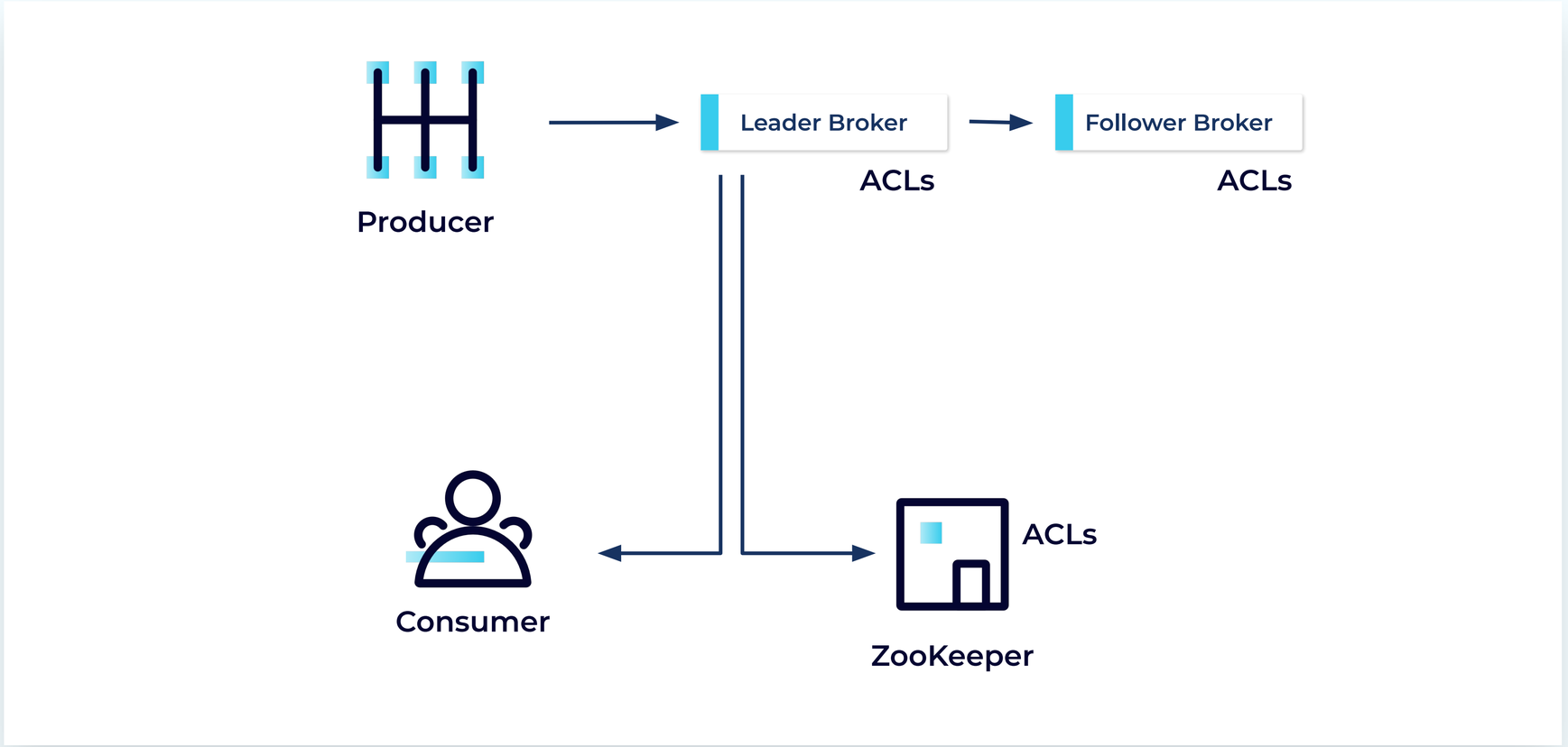
ACLs

CONFLUENT Developer



ACLs

CONFLUENT Developer



How Kafka applies ACLs

CONFLUENT Developer

Broker configuration (with ZooKeeper)

authorizer.class.name=kafka.security.authorizer.AclAuthorizer

How Kafka applies ACLs

CONFLUENT Developer

Broker configuration (with ZooKeeper)

authorizer.class.name=kafka.security.authorizer.AclAuthorizer

KRaft (no ZooKeeper)

Uses StandardAuthorizer by default

ACLs stored in __cluster_metadata topic

How Kafka applies ACLs

CONFLUENT Developer

Broker configuration (with ZooKeeper)

authorizer.class.name=kafka.security.authorizer.AclAuthorizer

KRaft (no ZooKeeper)

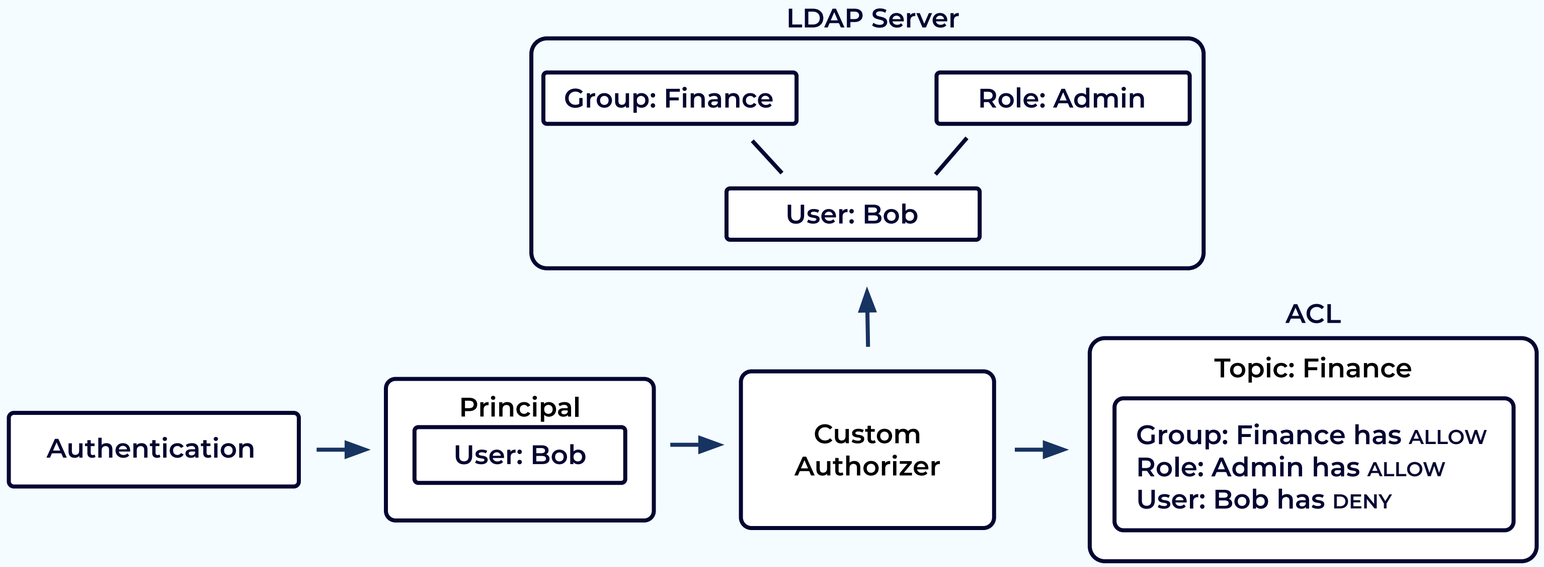
Uses StandardAuthorizer by default

ACLs stored in __cluster_metadata topic

Deny > Allow

Customizing access control

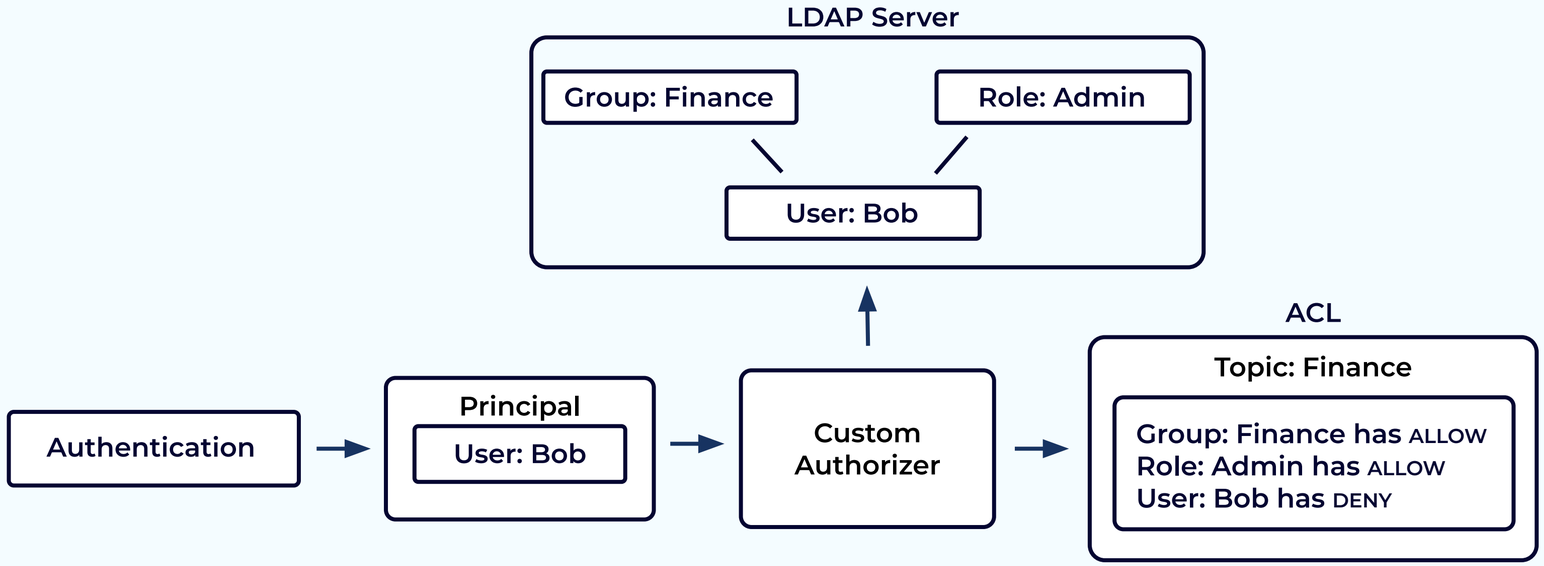
CONFLUENT Developer



Customizing access control

CONFLUENT Developer

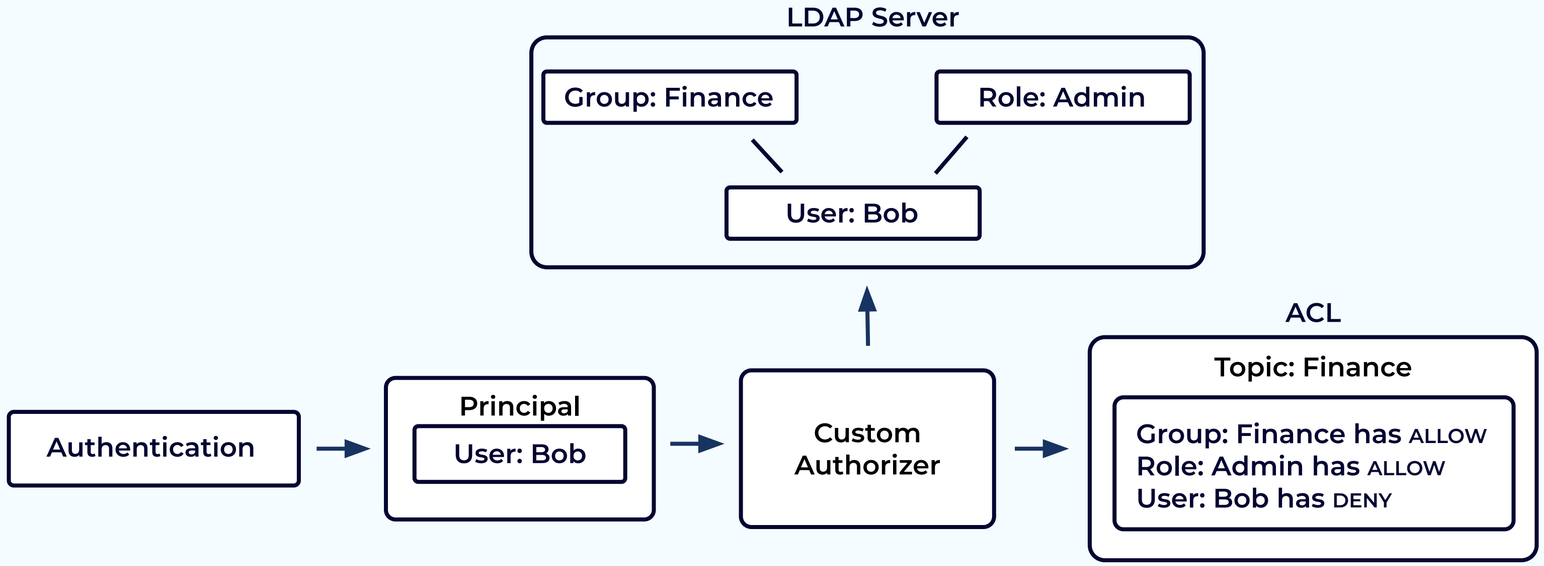
- Store user, group and role associations in e.g. an external LDAP server



Customizing access control

CONFLUENT Developer

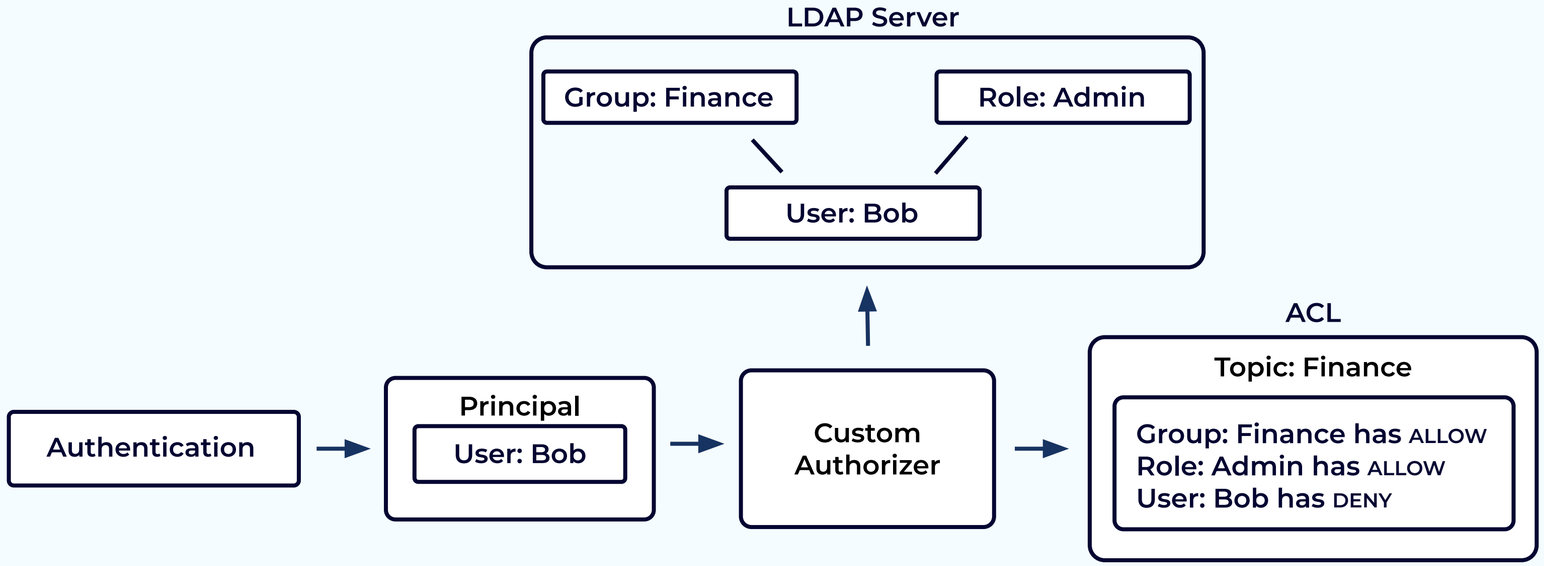
- Store user, group and role associations in e.g. an external LDAP server
- Add user, group and role permissions to the ACLs stored in ZooKeeper



Customizing access control

CONFLUENT Developer

- Store user, group and role associations in e.g. an external LDAP server
- Add user, group and role permissions to the ACLs stored in ZooKeeper
- Implement custom Authorizer that fetches a user's groups and roles from LDAP, and resolves against a resource's ACLs.



Considerations

- Use super users and allow.everyone.if.no.acl.found with caution – or not at all

Considerations

CONFLUENT Developer

- Use super users and allow.everyone.if.no.acl.found with caution – or not at all
- Don’t grant access to ANONYMOUS principal

Considerations

- Use super users and allow.everyone.if.no.acl.found with caution – or not at all
- Don’t grant access to ANONYMOUS principal
- Automate the process of creating user credentials and assigning ACLs for all environments

Considerations

- Use super users and allow.everyone.if.no.acl.found with caution – or not at all
- Don’t grant access to ANONYMOUS principal
- Automate the process of creating user credentials and assigning ACLs for all environments
- Adjust connections.max.reauth.ms to force connection to reauthenticate at intervals

Considerations

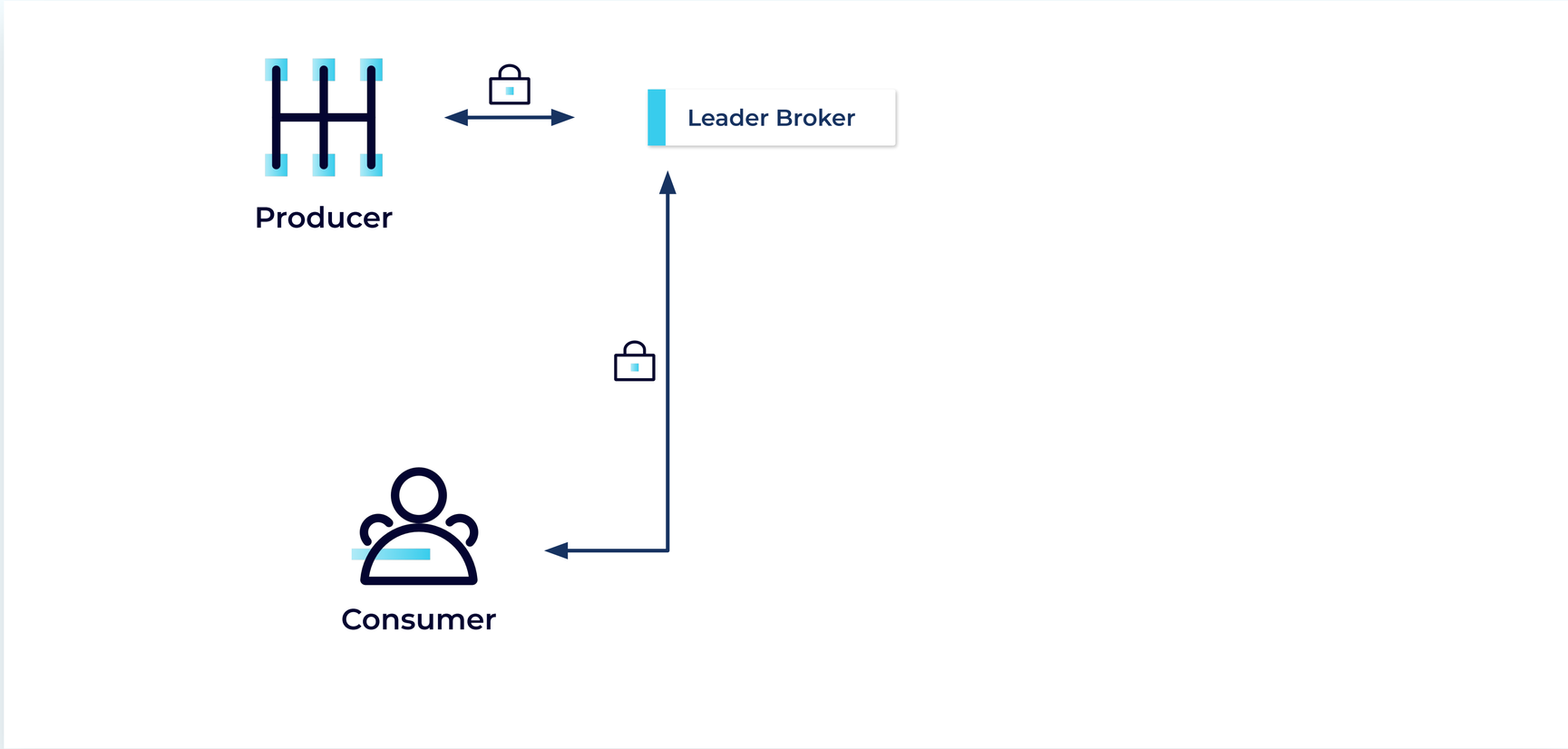
- Use super users and allow.everyone.if.no.acl.found with caution – or not at all
- Don’t grant access to ANONYMOUS principal
- Automate the process of creating user credentials and assigning ACLs for all environments
- Adjust connections.max.reauth.ms to force connection to reauthenticate at intervals
- Use Deny ACLs to prevent actions from compromised users

CONFLUENT
Developer

Encryption

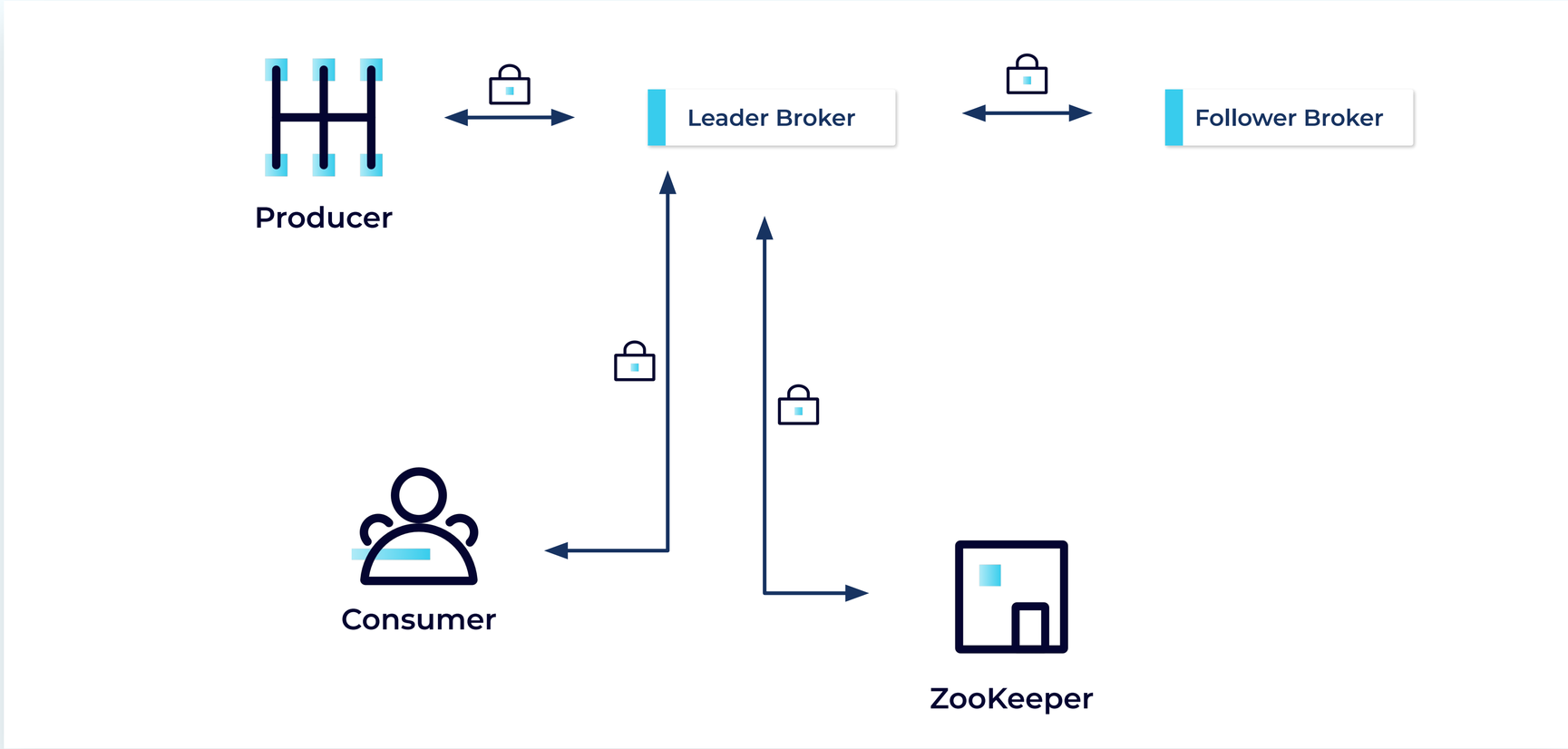
Encryption

CONFLUENT Developer



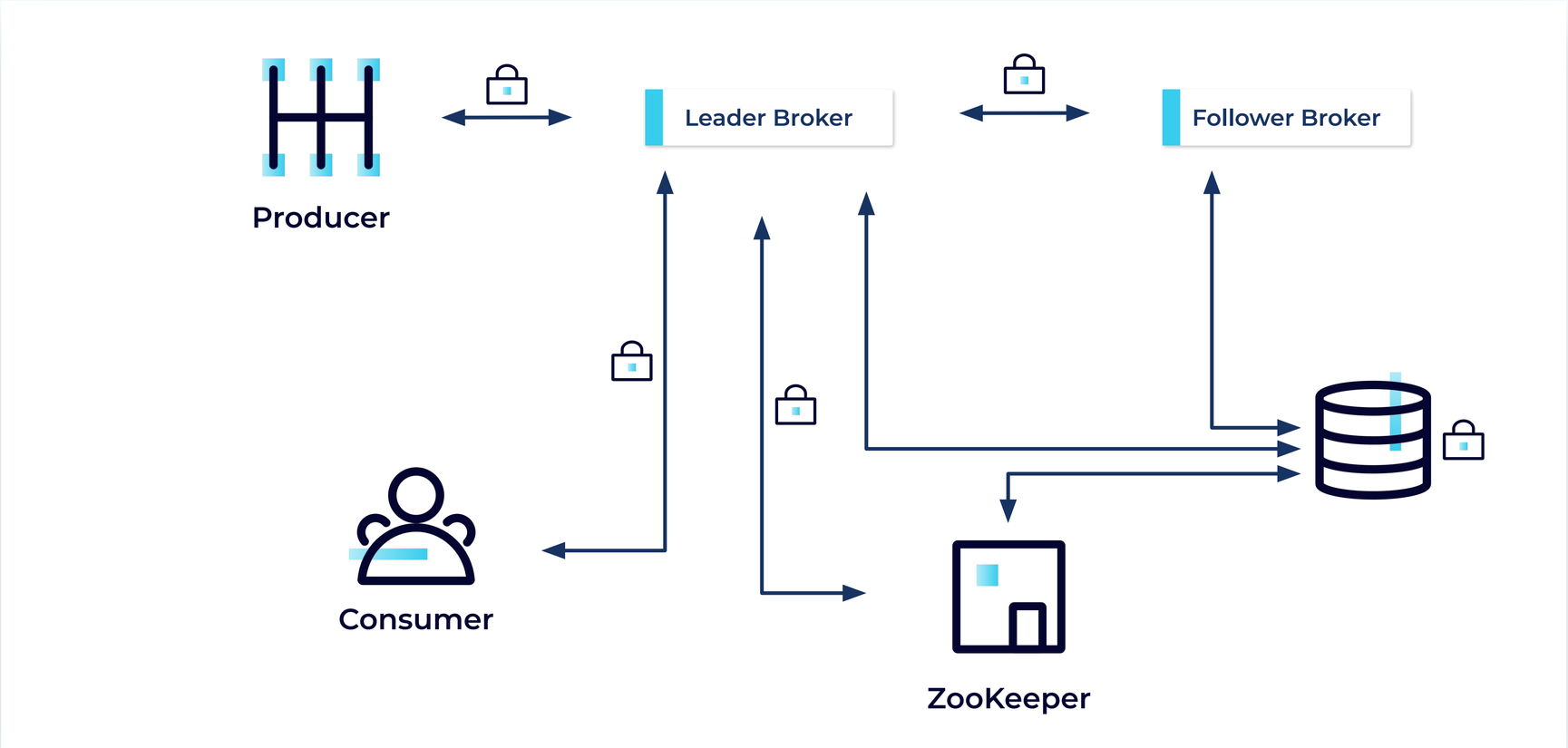
Encryption

CONFLUENT
Developer



Encryption

CONFLUENT Developer

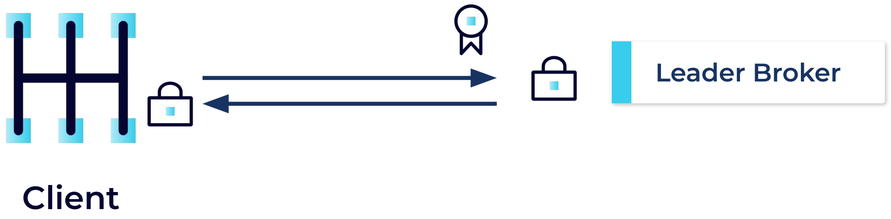


SSL

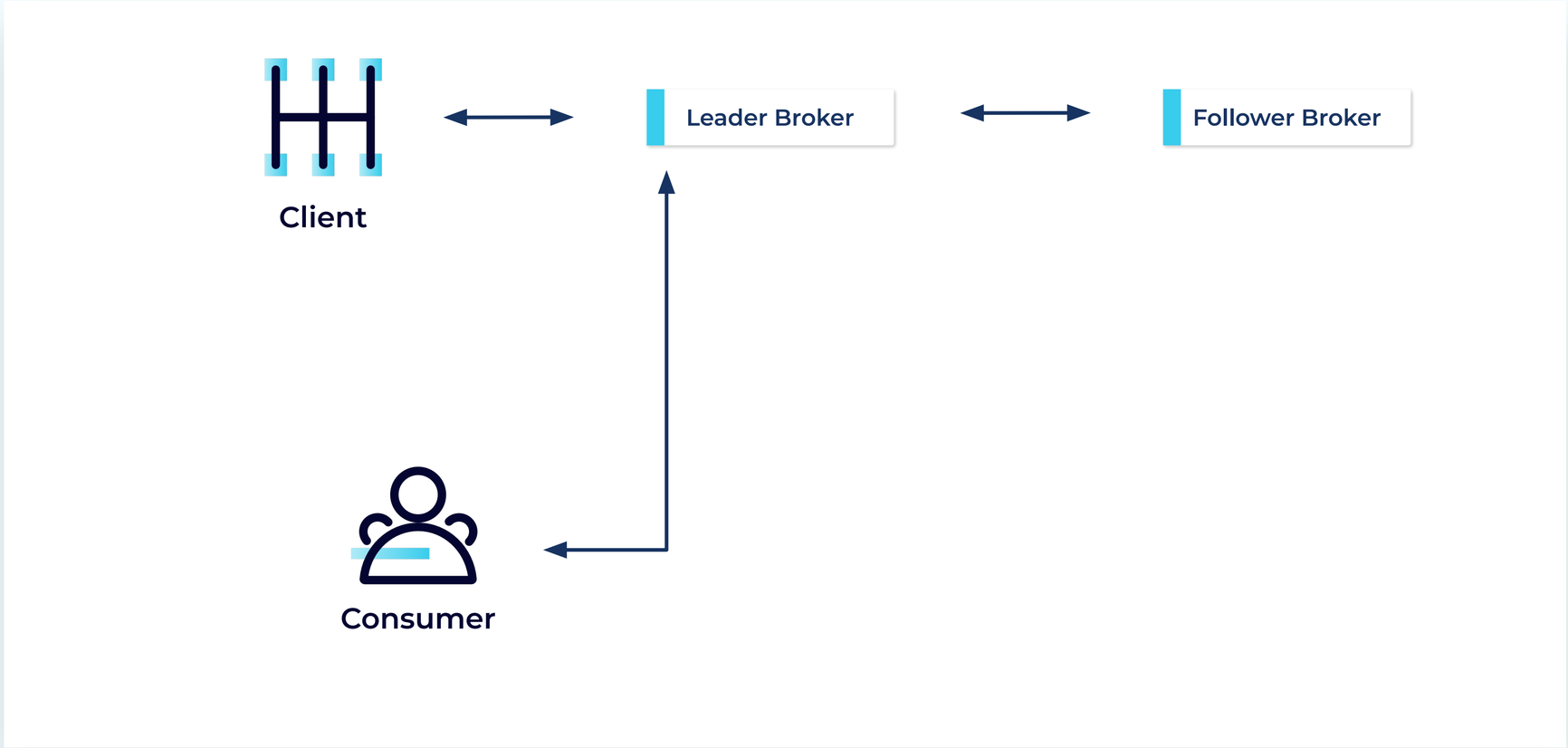
CONFLUENT Developer

Broker configuration

ssl.client.auth=required

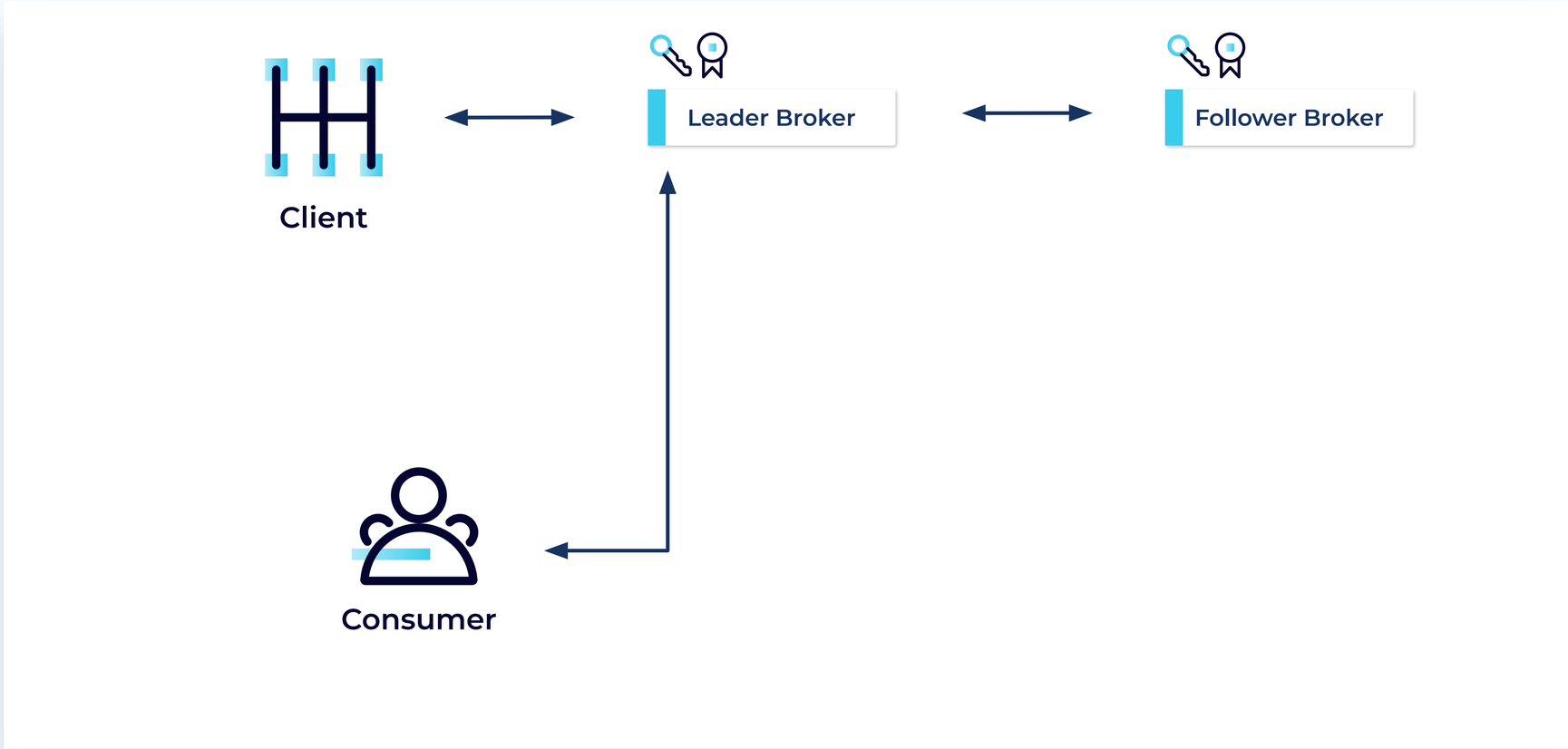


TLS

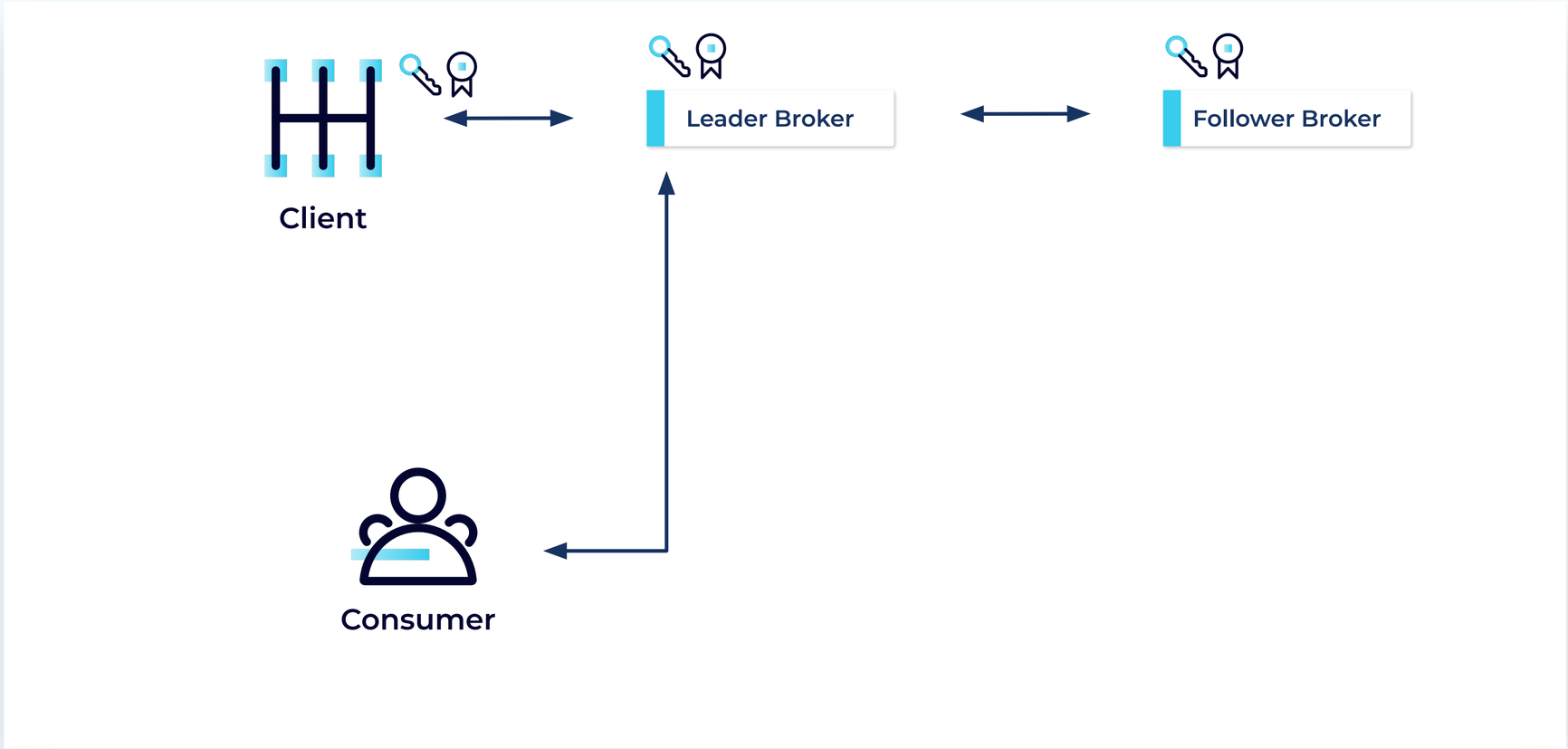


TLS

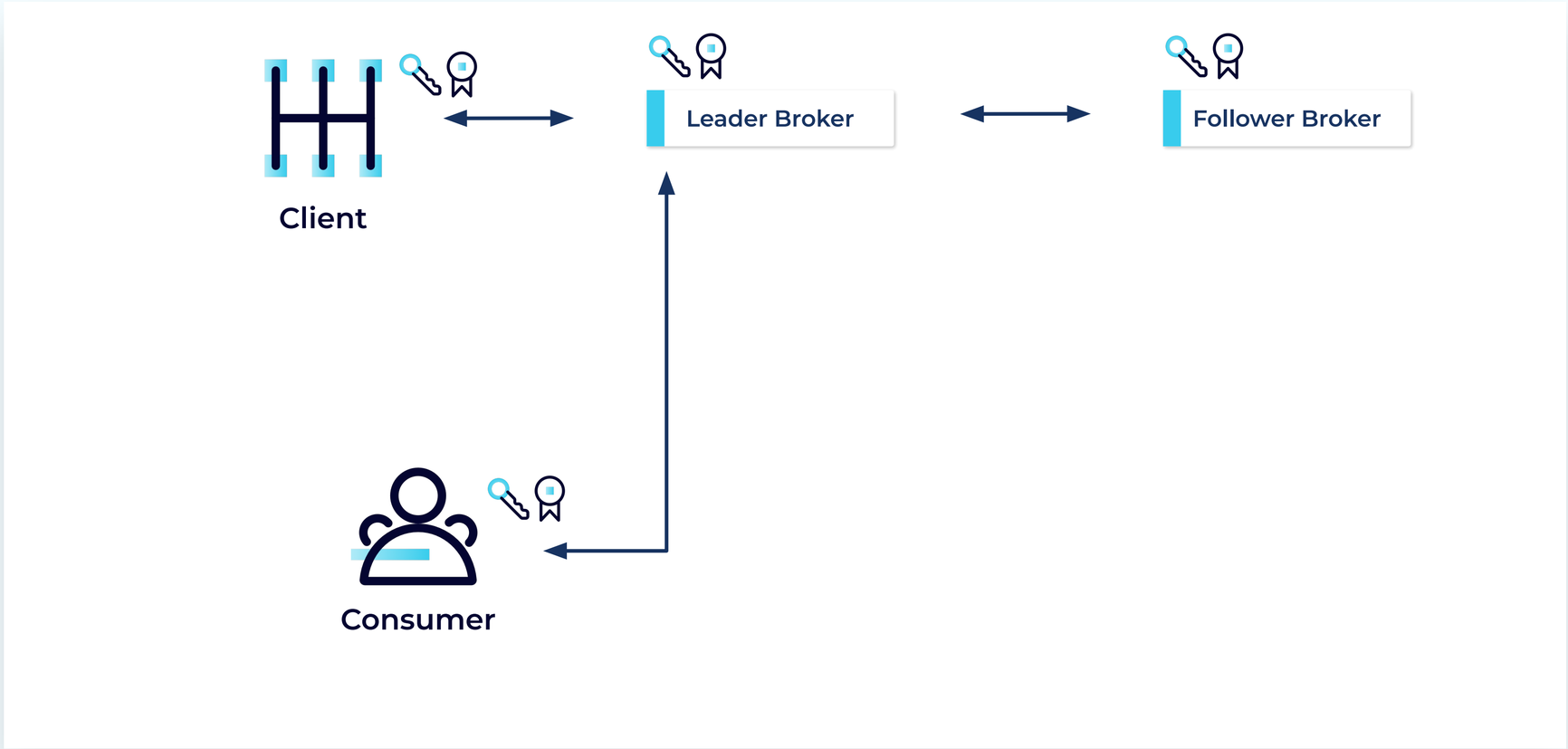
CONFLUENT
Developer



TLS



TLS



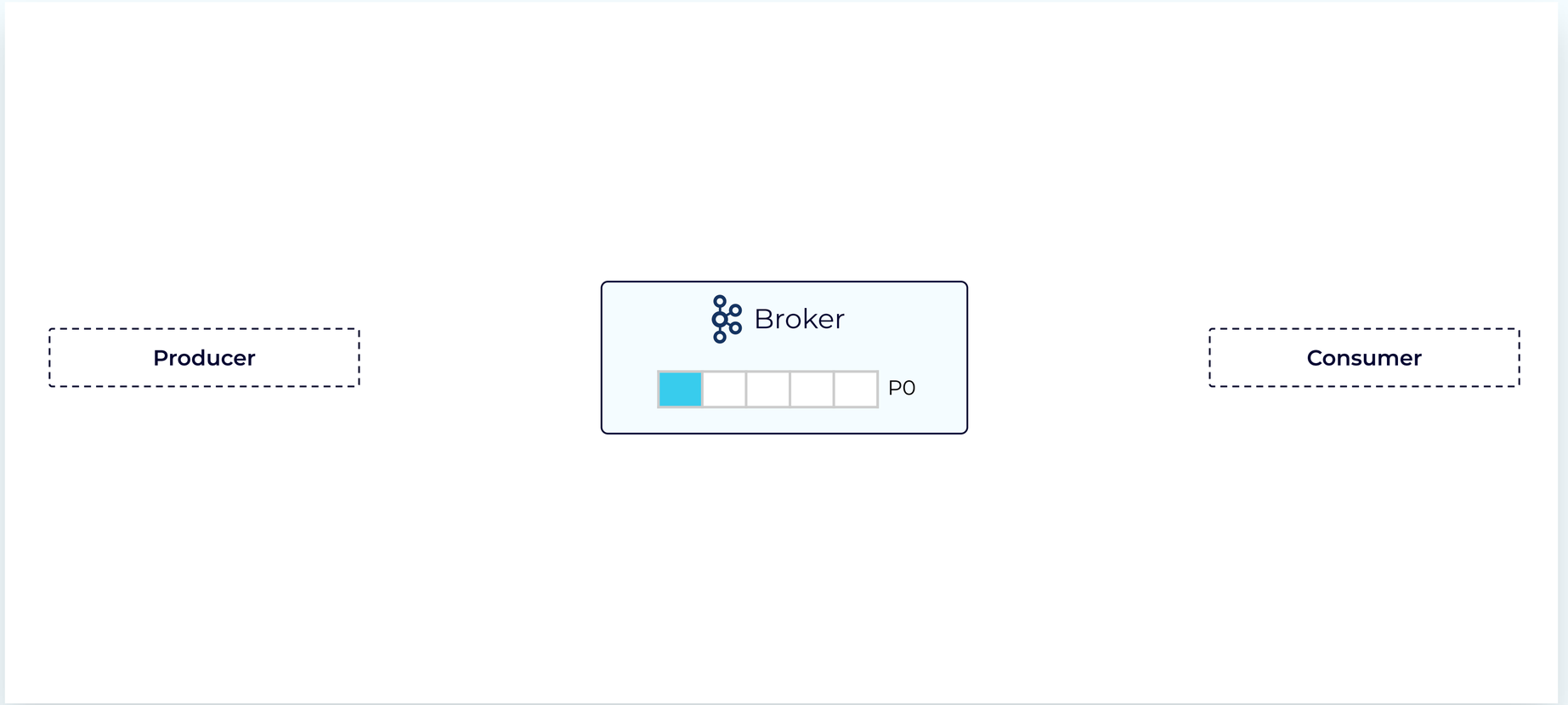
Inter-broker TLS Communication

security.inter.broker.protocol=SSL

Encrypting data at rest

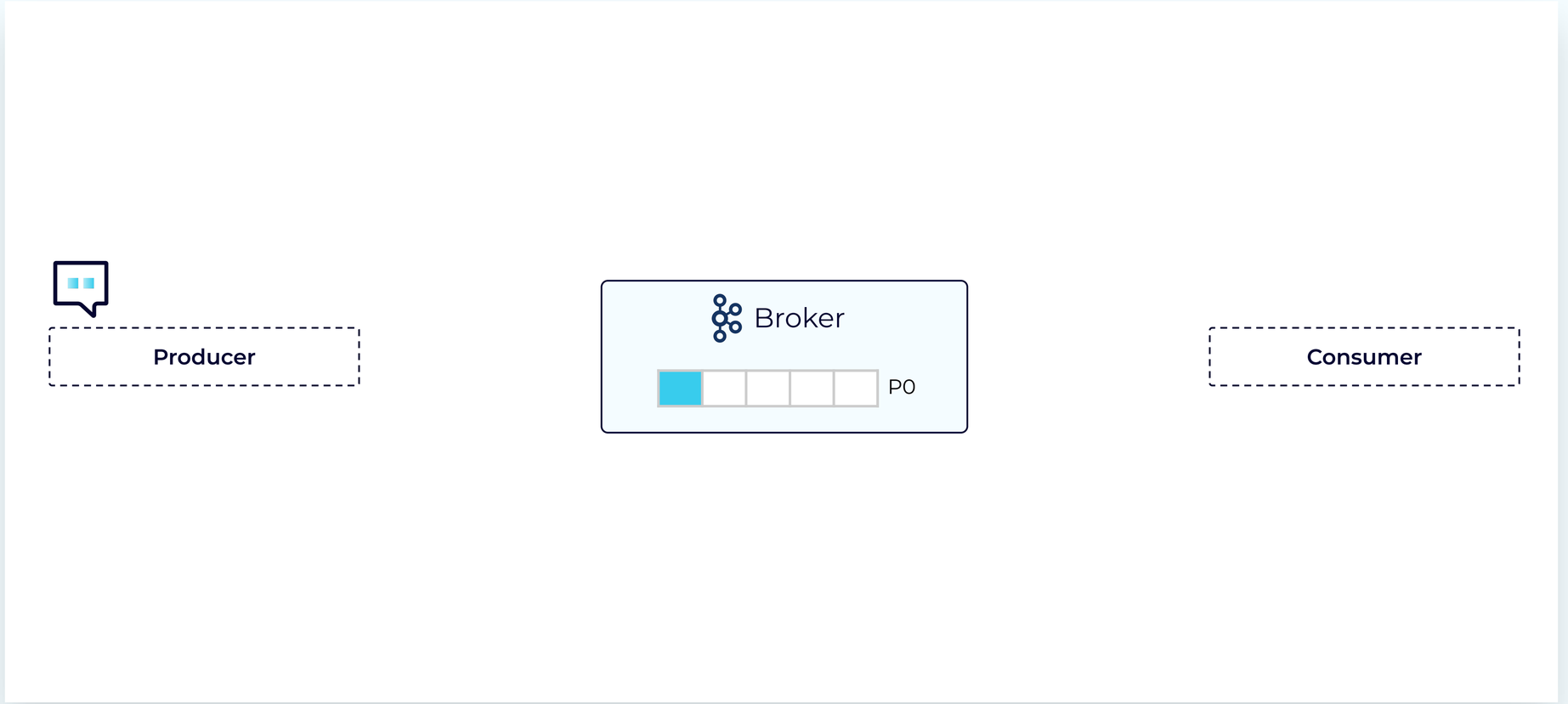
- Whole disk or volume encryption
  - Filesystem ACLs to restrict access
- Platform capabilities
  - E.g. Encrypted AWS EBS volumes
- Appliances
  - E.g. Vormetric, Gemalto

End-to-End Encryption



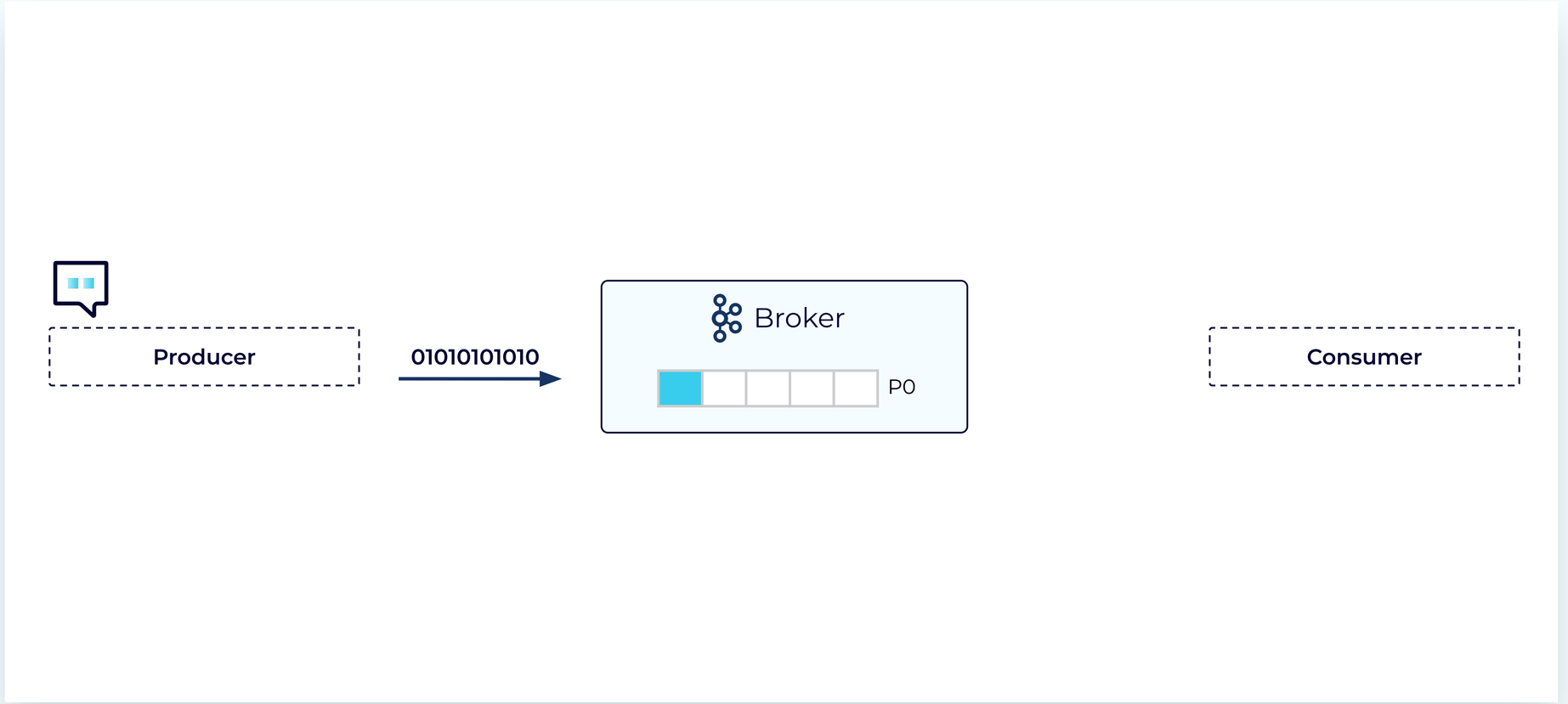
End-to-End Encryption

CONFLUENT Developer



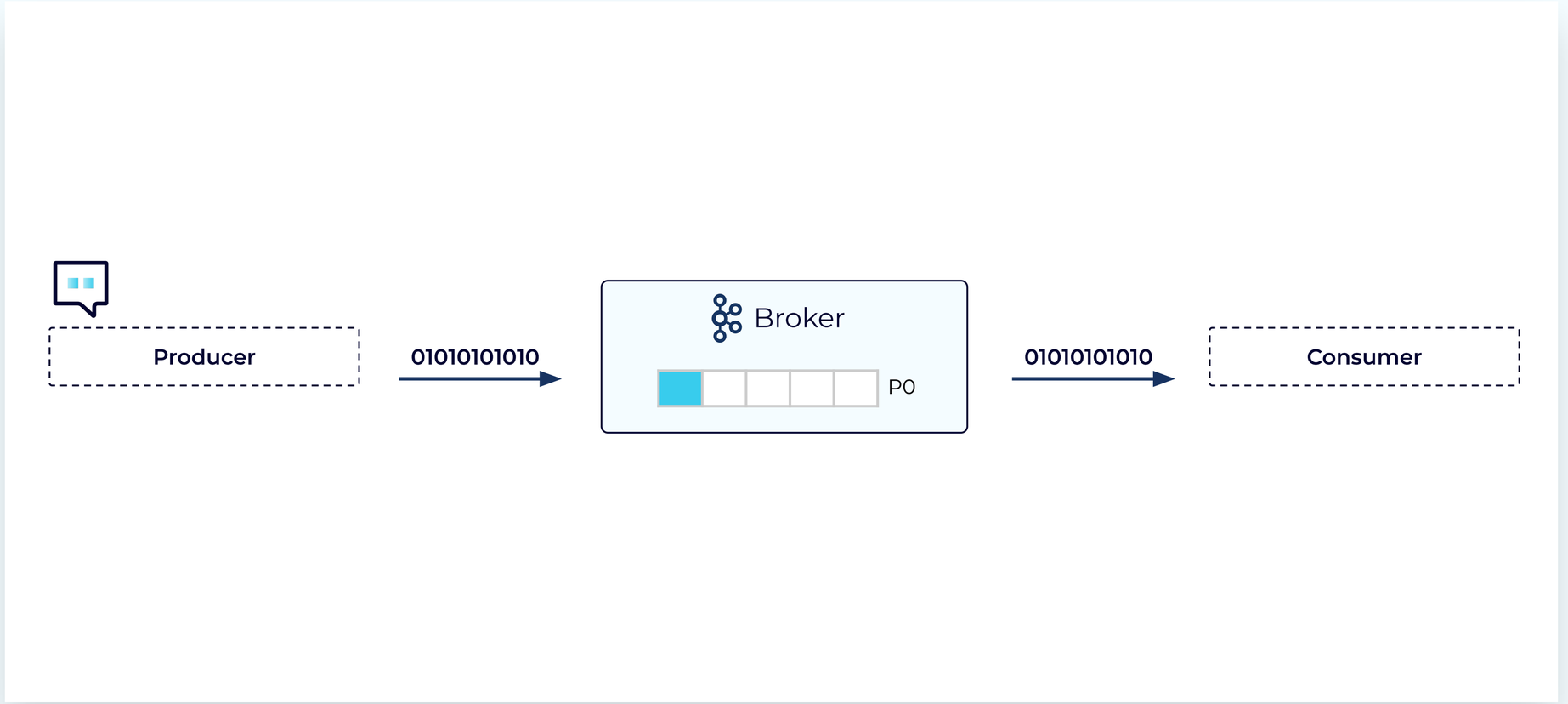
End-to-End Encryption

CONFLUENT Developer



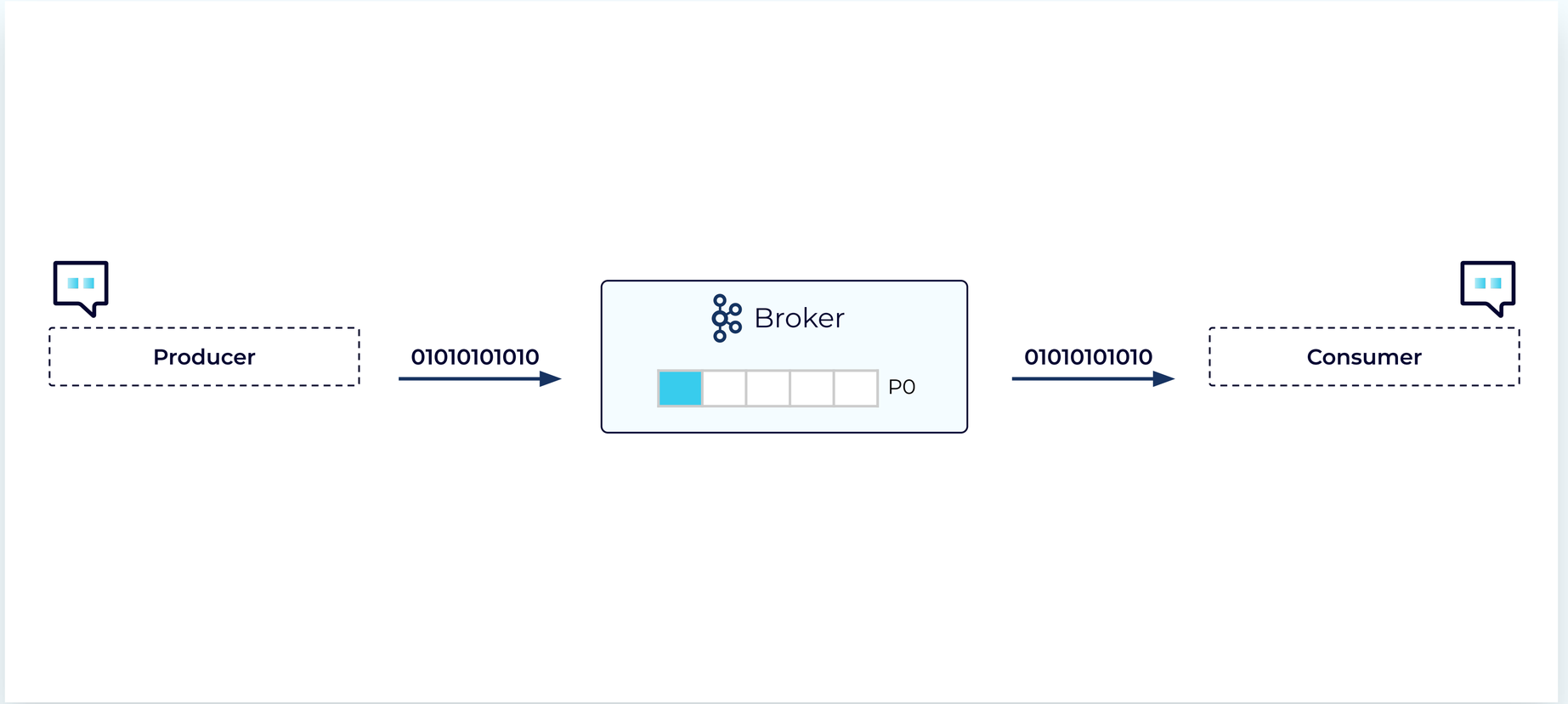
End-to-End Encryption

CONFLUENT Developer



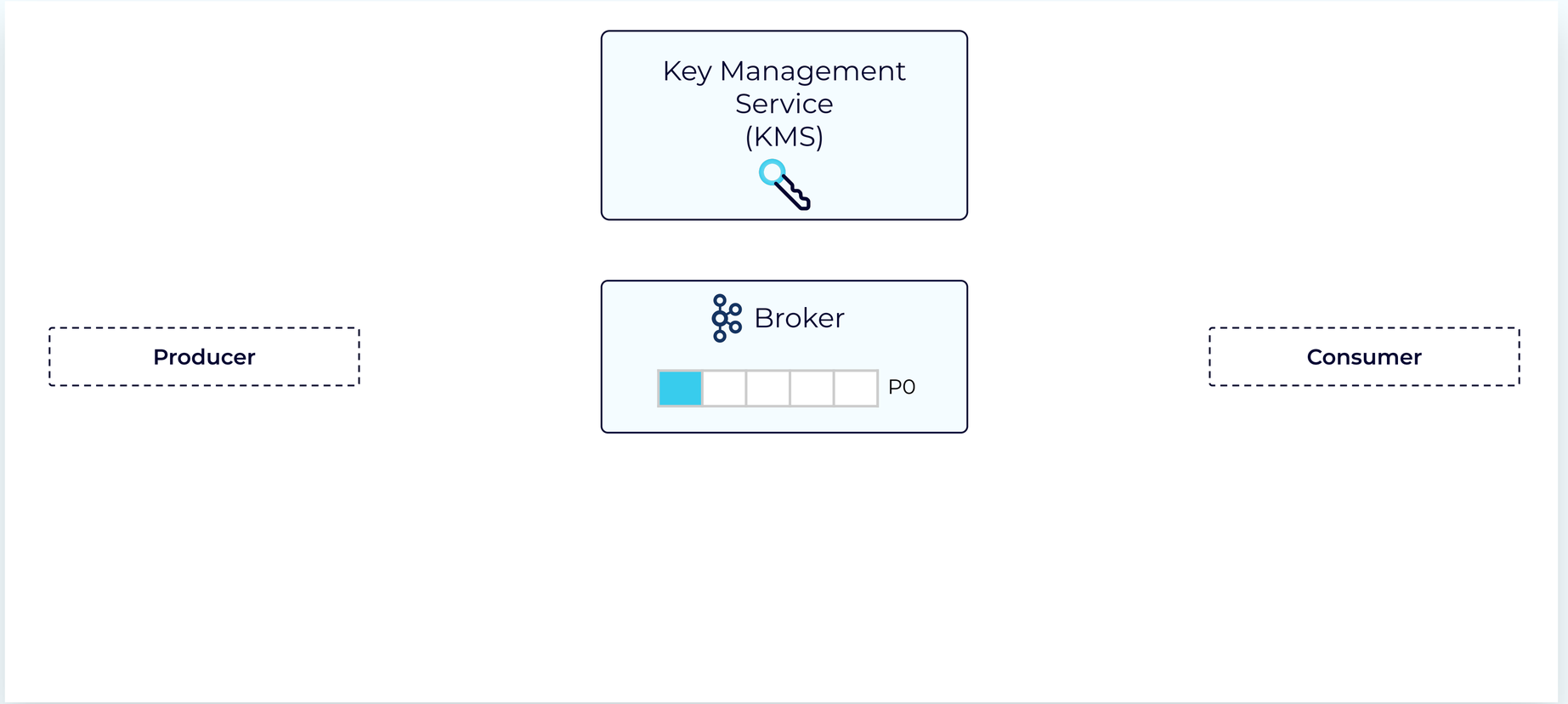
End-to-End Encryption

CONFLUENT Developer



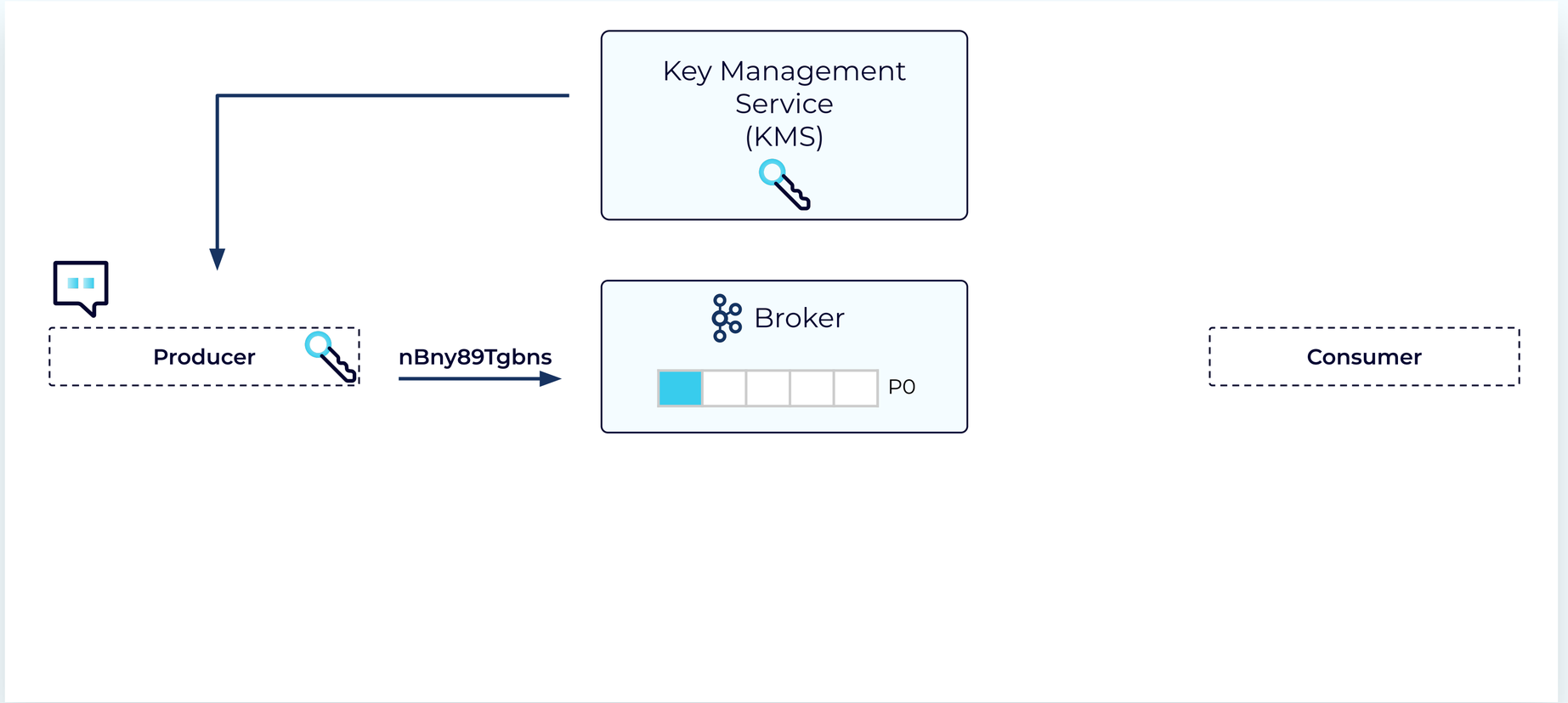
End-to-End Encryption

CONFLUENT Developer



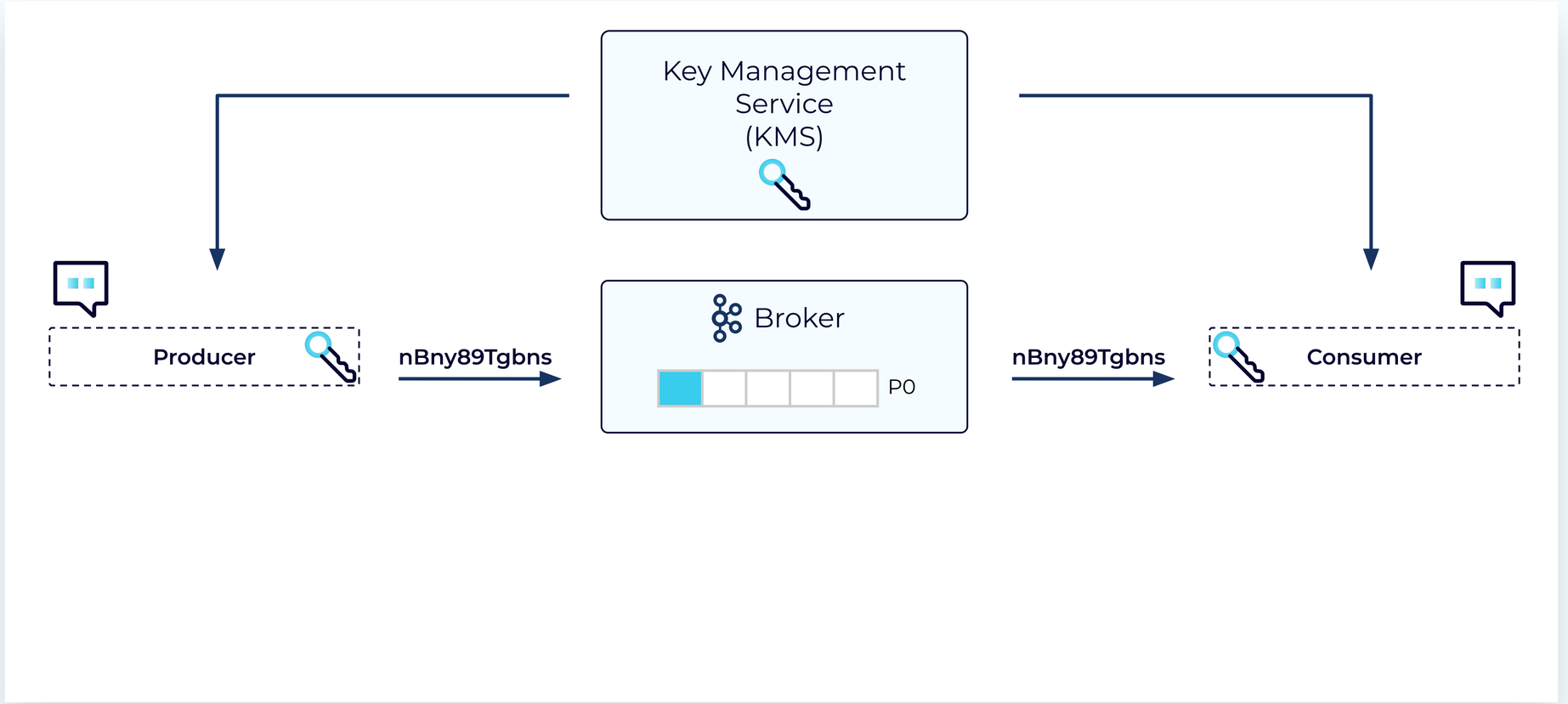
End-to-End Encryption

CONFLUENT Developer



End-to-End Encryption

CONFLUENT Developer



CONFLUENT
Developer

Securing ZooKeeper

Securing ZooKeeper

Set `zookeeper.set.acl` to `true`

SSL client authentication

SASL

Securing ZooKeeper

Set `zookeeper.set.acl` to `true`

SSL client authentication

- Each broker and CLI tool must use same Distinguished Name for *authorization*

SASL

Securing ZooKeeper

Set `zookeeper.set.acl` to `true`

SSL client authentication

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

SASL

Securing ZooKeeper

Set `zookeeper.set.acl` to `true`

SSL client authentication

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

SASL

- Integrate with Kerberos

Securing ZooKeeper

Set `zookeeper.set.acl` to `true`

SSL client authentication

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

CONFLUENT Developer

SASL

- Integrate with Kerberos
- Use TLS encryption
  - `ssl.clientAuth=none` in the ZooKeeper configuration

Securing ZooKeeper

Set `zookeeper.set.acl` to `true`

SSL client authentication

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

CONFLUENT Developer

SASL

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

Securing ZooKeeper

SSL + SASL

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

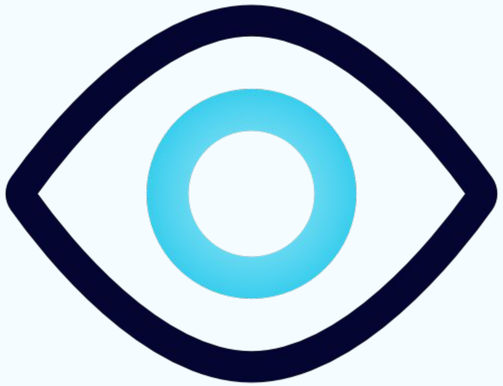
CONFLUENT
Developer

Audit Logs

Audit Logs

CONFLUENT Developer

1. **Insight** – Logging every attempted operation



Audit Logs

CONFLUENT Developer

1. **Insight** – Logging every attempted operation
2. **Security** – Monitoring and validating operations

Audit Logs

CONFLUENT Developer

1. **Insight** – Logging every attempted operation
2. **Security** – Monitoring and validating operations
3. **Impact** – Debugging client interactions

Audit Logs

CONFLUENT Developer

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

Audit Logs

- kafka.authorizer.logger
  - Used for authorization logs
- kafka.request.logger
  - Used for request logs

Log4j authorization logger

CONFLUENT Developer

log4j.properties

log4j.logger.kafka.authorizer.logger=DEBUG

output

DEBUG Principal = User:Alice is Allowed Operation = Write from host = 127.0.0.1 on resource = Topic:LITERAL:customerOrders for request = Produce with resourceRefCount = 1 (kafka.authorizer.logger)

INFO Principal = User:Mallory is Denied Operation = Describe from host = 10.0.0.13 on resource = Topic:LITERAL:customerOrders for request = Metadata with resourceRefCount = 1 (kafka.authorizer.logger)

Log4j request logger

CONFLUENT Developer

log4j.properties

log4j.logger.kafka.request.logger=DEBUG

output

DEBUG Completed request:RequestHeader(apiKey=PRODUCE, apiVersion=8, clientId=producer-1, correlationId=6) -- {acks=-1,timeout=30000,partitionSizes=[customerOrders-0=15514]},response: {responses=[{topic=customerOrders,partition_responses=[{partition=0,error_code=0 ,base_offset=13,log_append_time=-1,log_start_offset=0,record_errors=[],error_mes sage=null}]}],throttle_time_ms=0} **from connection 127.0.0.1:9094-127.0.0.1:61040-0**;totalTime:2.42,requestQueueTime:0.112,local-Time:2.15,re moteTime:0.0,throttleTime:0,responseQueueTime:0.04,sendTime: 0.118,securityProtocol:SASL_SSL,**principal:User:Alice**,listener:SASL_SSL,clientInf ormation:ClientInformation(softwareName=apache-kafka-java, softwareVersion=2.7.0-SNAPSHOT) (kafka.request.logger)

Considerations

- Ensure you have sufficient disk space for each broker

Considerations

- Ensure you have sufficient disk space for each broker
- Set a retention policy for logs

Considerations

- Ensure you have sufficient disk space for each broker
- Set a retention policy for logs
- Capture and consolidate logs to view holistic issues

Considerations

- Ensure you have sufficient disk space for each broker
- Set a retention policy for logs
- Capture and consolidate logs to view holistic issues
- Use a visualization tool (e.g., ELK stack, Elasticsearch, Logstash, and Kibana)

CONFLUENT
Developer

Security Recommendations

Education

Start with security in mind

Encrypt the filesystem

Secure data in transit

Set up a system for administering ACLs

CONFLUENT
Developer

Rotate your keys

CONFLUENT
Developer

Dynamically update certificates

Enable Reauthentication

CONFLUENT
Developer

Protect ZooKeeper

CONFLUENT
Developer

Set up and monitor audit logs

Play, tinker, break things

CONFLUENT
Developer

Your Apache Kafka journey begins here

developer.confluent.io