[redacted]

# Securing Apache Kafka

## Security Overview

Tos

- Support since 0.9.0

- Wire encryption between client and broker
  - For cross data center mirroring

- Access control on resources such as topics
  - Enable sharing Kafka clusters

## Authentication Overview

Tos

- Broker support multiple ports
  - plain text (no wire encryption/authentication)
  - SSL (for wire encryption/authentication)
  - SASL (for Kerberos authentication)
  - SSL + SASL (SSL for wire encryption, SASL for authentication)

- Clients choose which port to use
  - need to provide required credentials through configs

# Performance impact with SSL

Tos

- r3.xlarge
  - 4 core, 30GB ram, 80GB ssd, moderate network (~90MB/s)

| | Throughput(MB/S) | CPU on client | CPU on broker |
|---|---|---|---|
| Producer(plaintext) | 83 | 12% | 30% |
| Producer (SSL) | 69 | 28% | 48% |
| Consumer (plaintext) | 83 | 8% | 2% |
| Consumer (SSL) | 69 | 27% | 24% |

- **Most overhead from encryption**

## Why Kerberos

Tos

- Secure single sign-on
  - An organization may provide multiple services
  - User just remember a single Kerberos password to use all services
- More convenient when there are many users
- Need Key Distribution Center (KDC)
  - Each service/user need a Kerberos principal in KDC

## Data transfer

Tos

- SASL_PLAINTEXT
  - No wire encryption
- SASL_SSL
  - Wire encryption over SSL

## Configuring Kerberos

**No client code change; just configuration change**

**Broker JAAS file**

```
KafkaServer {
    com.sun.security.auth.module
    .Krb5LoginModule required
    useKeyTab=true
    storeKey=true
    keyTab="/etc/security/keyt
    abs/kafka_server.keytab"
    principal="kafka/kafka1.ho
    stname.com@EXAMPLE.COM";
};
```

**Client JAAS file**

```
KafkaClient {
    com.sun.security.auth.module
    .Krb5LoginModule required
    useKeyTab=true
    storeKey=true
    keyTab="/etc/security/keyt
    abs/kafka_client.keytab"
    principal="kafka-client-
    1@EXAMPLE.COM";
};
```

**Broker JVM**

```
Djava.security.auth.lo
gin.config=/etc/kafka/
kafka_server_jaas.conf
```

**ClientJVM**

```
-
Djava.security.auth.lo
gin.config=/etc/kafka/
kafka_client_jaas.conf
```

**Broker config**

```
security.inter.broker.protocol=
SASL_PLAINTEXT(SASL_SSL)
sasl.kerberos.service.name=kafka
```

**Client config**

```
security.protocol=SA
SL_PLAINTEXT(SASL_SSL)
sasl.kerberos.servic
e.name=kafka
```

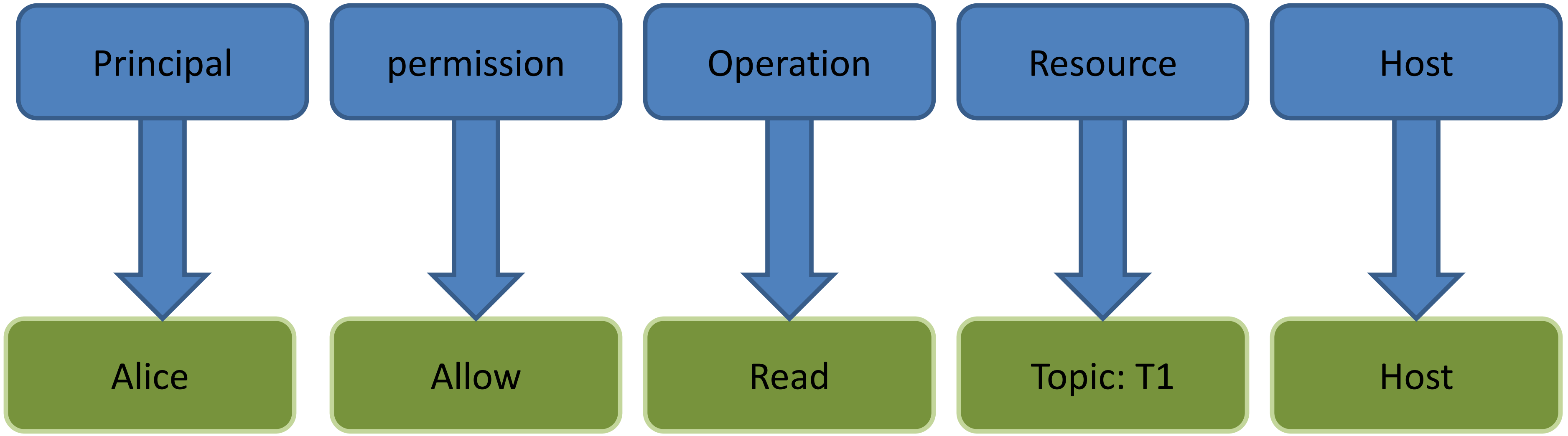
## Authorization

Tos

- Control which permission each authenticated principal has
- Pluggable with a default implementation

## ACL

**Alice is Allowed to Read from topic T1 from Host1**



## Operations and Resources

Tos

- Operations
  - Read, Write, Create, Describe, ClusterAction, All
- Resources
  - Topic, Cluster and ConsumerGroup

| **Operations** | **Resources** |
|---|---|
| Read, write, Describe (Read, Write implies Describe) | Topic |
| Read | Consumer Group |
| Create, ClusterAction(communication between controller and brokers) | Cluster |

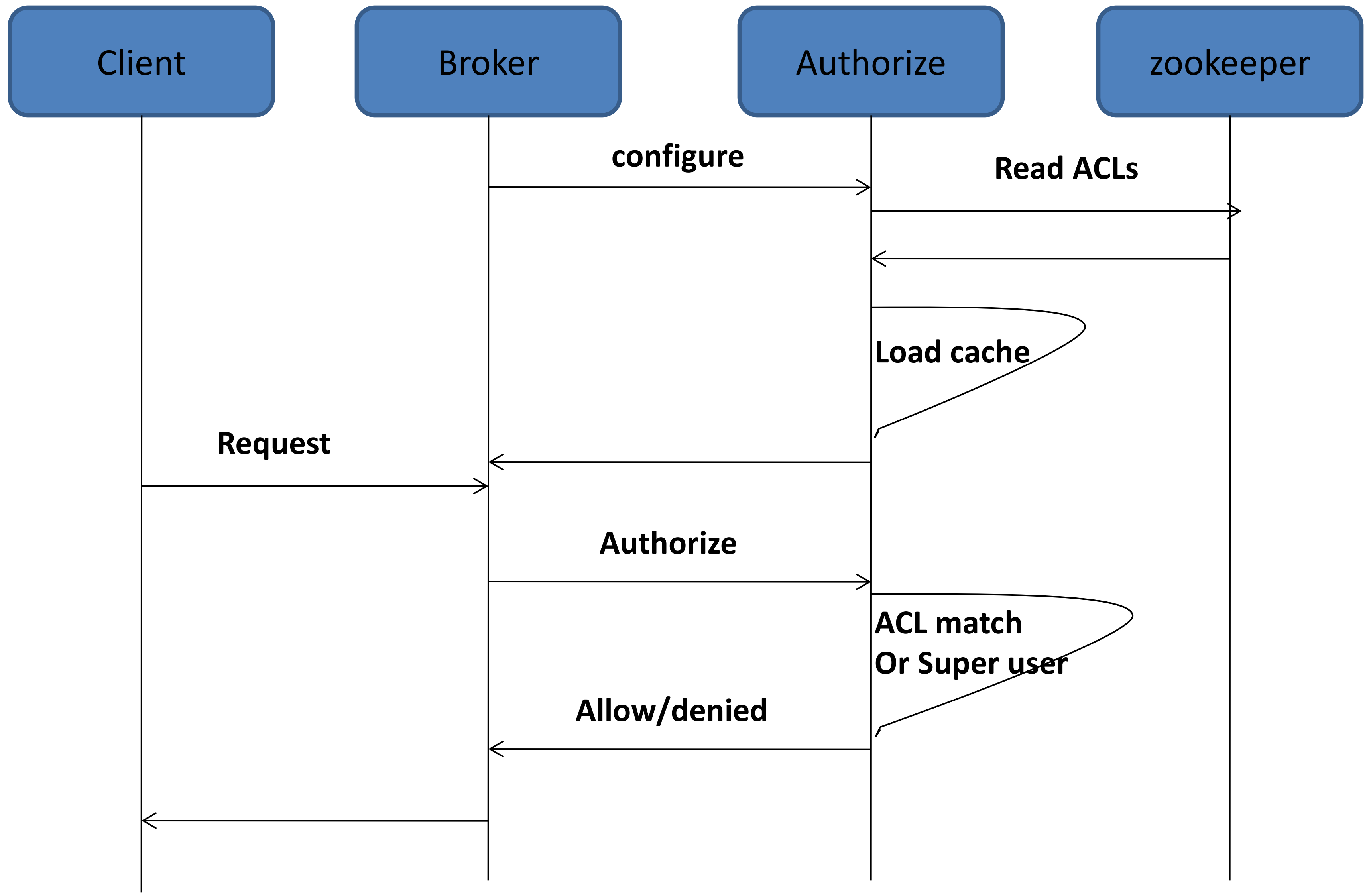
## SimpleAclAuthorizer

Tos

- Out of box authorizer implementation.
- CLI tool for adding/removing acls
- ACLs stored in zookeeper and propagated to brokers  asynchronously
- ACL cache in broker for better performance

# Authorizer Flow

Tos



## Configure broker ACL

- authorizer.class.name=kafka.security.auth.SimpleAclAuthorizer

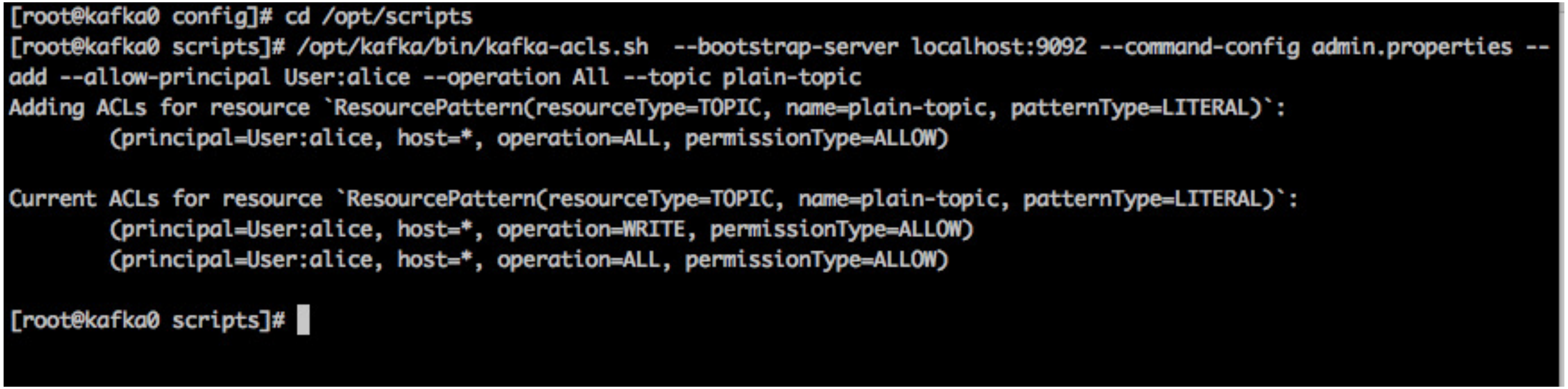
- Make Kafka principal super users
  - Or grant ClusterAction and Read all topics to Kafka principal

# Configure client ACL

Tos

- Producer
  - Grant Write on topic, Create on cluster (auto creation)
  - Or use --producer option in CLI

/opt/kafka/bin/kafka-acls.sh  --bootstrap-server kafka0:9092 --command-config admin.properties --add --allow-principal User:alice --operation All --topic plain-topic



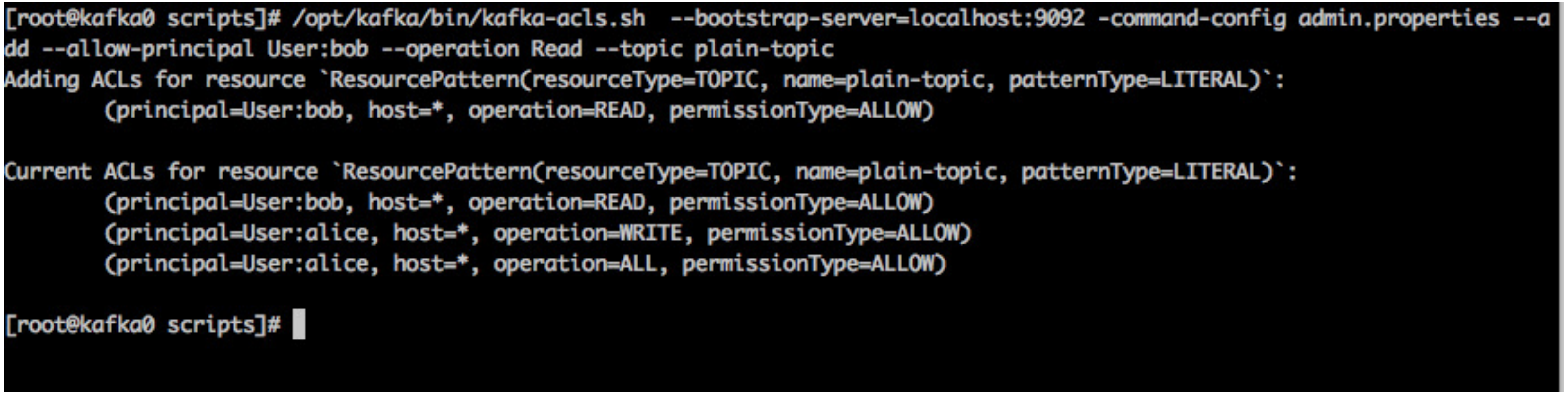
```
[root@kafka0 config]# cd /opt/scripts
[root@kafka0 scripts]# /opt/kafka/bin/kafka-acls.sh  --bootstrap-server localhost:9092 --command-config admin.properties --
add --allow-principal User:alice --operation All --topic plain-topic
Adding ACLs for resource `ResourcePattern(resourceType=TOPIC, name=plain-topic, patternType=LITERAL)`:
        (principal=User:alice, host=*, operation=ALL, permissionType=ALLOW)

Current ACLs for resource `ResourcePattern(resourceType=TOPIC, name=plain-topic, patternType=LITERAL)`:
        (principal=User:alice, host=*, operation=WRITE, permissionType=ALLOW)
        (principal=User:alice, host=*, operation=ALL, permissionType=ALLOW)

[root@kafka0 scripts]#
```

# Configure client ACL

Tos

- Consumer
  - Grant Read on topic, Read on consumer group
  - Or use --consumer option in CLI

```
/opt/kafka/bin/kafka-acls.sh  --bootstrap-server=kafka0:9092 -command-config admin.properties --add --
allow-principal  User:bob --operation Read --topic plain-topic
```



```
[root@kafka0 scripts]# /opt/kafka/bin/kafka-acls.sh  --bootstrap-server=localhost:9092 -command-config admin.properties --a
dd --allow-principal User:bob --operation Read --topic plain-topic
Adding ACLs for resource `ResourcePattern(resourceType=TOPIC, name=plain-topic, patternType=LITERAL)`:
        (principal=User:bob, host=*, operation=READ, permissionType=ALLOW)

Current ACLs for resource `ResourcePattern(resourceType=TOPIC, name=plain-topic, patternType=LITERAL)`:
        (principal=User:bob, host=*, operation=READ, permissionType=ALLOW)
        (principal=User:alice, host=*, operation=WRITE, permissionType=ALLOW)
        (principal=User:alice, host=*, operation=ALL, permissionType=ALLOW)

[root@kafka0 scripts]#
```