Kafka安全配置：

Zookeeper:

1、zoo.cfg文件配置添加如下配置：

authProvider.1=org.apache.zookeeper.server.auth.SASLAuthenticationProvider

requireClientAuthScheme=sasl

jaasLoginRenew=3600000

2、编写JAAS文件这个文件定义需要链接到Zookeeper服务器的用户名和密码。

JAAS配置节默认为Server：

Server {

org.apache.kafka.common.security.plain.PlainLoginModule required

username="admin"

password="admin-secret"

user\_kafka="kafka-sec"

user\_producer="prod-sec";

};

配置文件我命名为zk\_server\_jaas.conf，并放在部署目录的/usr/zookeeper/zookeeper-3.4.10/conf/下。文件中定义了身份认证类(org.apache.kafka.common.security.plain.PlainLoginModule)，可以看到这个认证类是kafka命名空间，也就是需要加入kafka的插件，所以下面一步非常重要。这个文件中定义了两个用户，一个是kafka，一个是producer(user\_可以定义多个用户，等于的值就是用户密码)，这些用user\_配置出来的用户都可以提供给生产者程序和消费者程序认证使用。还有两个属性，username和password，其中username是配置Zookeeper节点之间内部认证的用户名，password是对应的密码。

1. 向zookeeper添加Kafka认证插件由于Zookeeper的认证机制是使用插件，这个插件只要支持JAAS即可。

Kafka需要链接到Zookeeper，直接使用Kafka的认证插件。这个插件类也包含在kafka-clients中(Maven项目)。将依赖的几个jar加入Zookeeper启动的classpath即可。如下是kafka-clients-0.10.0.1相关jar，可能你的版本和这个不一样，没关系，只要是名字相同就可以，jar在kafka的lib中查找，不需要网上下载。包括其依赖：

kafka-clients-0.10.0.1.jar

lz4-1.3.0.jar

slf4j-api-1.7.21.jar

slf4j-log4j12-1.7.21.jar

snappy-java-1.1.2.6.jar

我的做法比较直接，在Zookeeper部署根目录下创建一个路径for\_sasl，将上述所有jar文件复制到这个路径下，再修改bin/zkEnv.sh配置文件，这个文件主要负责加载一些启动Zookeeper有关的环境变量，输入参数。

for i in "$ZOOBINDIR"/../for\_sasl/\*.jar;

do CLASSPATH="$i:$CLASSPATH"

done

SERVER\_JVMFLAGS=" -Djava.security.auth.login.config=$ZOOCFGDIR/zk\_server\_jaas.conf "

逻辑比较简单，先将for\_sasl目录下的所有jar文件追加到CLASSPATH变量，再设置一个JVM参数给SERVER\_JVMFLAGS变量，这两个变量都会在Zookeeper启动时传给JVM。具体可以查看脚本[源码](https://www.2cto.com/ym/" \t "https://www.2cto.com/net/201708/_blank)。

4、配置其他节点照1到3步骤配置剩余的zookeeper节点。

5、启动所有节点将所有zookeeper节点的Quorum进程开启

$：bin/zkServer.sh start，查看zookeeper日志，看看之后所有节点是否都能稳定运行，再试试bin/zkCli.sh链接所有节点。是否都能通。

三、 Kafka集群配置SASL确保以上zookeeper配置成功后，开始配置Kafka。Kafka所有节点也是对等的，所以下面步骤的配置在所有节点上都相同。

1、创建JAAS配置文件定义链接Kafka Broker时所需要的用户名密码及broker各个节点之间相互通信的用户名密码，这部分配置定义在KafkaServer节，文件如下：

KafkaServer {

org.apache.kafka.common.security.plain.PlainLoginModule required

username="admin"

password="admin-sec"

user\_admin="admin-sec"

user\_producer="prod-sec"

user\_consumer="cons-sec";

};

Client {

org.apache.kafka.common.security.plain.PlainLoginModule required

username="kafka"

password="kafka-sec";

};

配置文件命名为：kafka\_server\_jaas.conf，放置在/usr/kafka/kafka\_2.11-0.11.0.0/config。下另外Client配置节主要配置了broker到Zookeeper的链接用户名密码。先解释KafkaServer，使用user\_来定义多个用户，供客户端程序(生产者、消费者程序)认证使用，可以定义多个，后续配置可能还可以根据不同的用户定义ACL，这部分内容超出本文范围。这是目前我对配置的理解。上例我定义了三个用户，一个是admin，一个是producer，一个是consumer，等号后面是对应用户的密码(如user\_producer定义了用户名为producer，密码为prod-sec的用户)。再选择一个用户，用于Kafka内部的各个broker之间通信，这里我选择admin用户，对应的密码是admin-sec。Client配置节则容易理解得多，主要是broker链接到zookeeper，从上文的Zookeeper JAAS文件中选择一个用户，填写用户名和密码即可。

2、配置server.properties

listeners=SASL\_PLAINTEXT://vubuntuez1:9092

security.inter.broker.protocol=SASL\_PLAINTEXT

sasl.enabled.mechanisms=PLAIN

sasl.mechanism.inter.broker.protocol=PLAIN

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

allow.everyone.if.no.acl.found=true

这里注意listeners配置项，将主机名部分(本例主机名是vubuntuez1)替换成当前节点的主机名。其他在各个节点的配置一致。注意，allow.everyone.if.no.acl.found这个配置项默认是false，若不配置成true，后续生产者、消费者无法正常使用Kafka。要么配置成true，这样必须对每个认证用户都配置ACL，比较麻烦，我直接设置true，当没有找到ACL配置时，允许所有的访问操作。

1. 配置KAFKA\_OPTS环境变量与伪分布式节点配置KAFKA\_OPTS变量相同

主要是为了传递JVM变量java.security.auth.login.config，指定JAAS配置文件。然后kafka-run-class.sh会负责将这个KAFKA\_OPTS变量传递给JVM。所以启动kafka broker的命令可以与这个设置变量操作写在同一个脚本下：

#!/bin/bashROOT=`dirname $0`

export KAFKA\_OPTS=" -Djava.security.auth.login.config=$ROOT/config/kafka\_server\_jaas.conf "

$ROOT/bin/kafka-server-start.sh -daemon $ROOT/config/server.properties

4、配置其他节点配置剩余的kafka broker节点，注意server.properties的listeners配置项，上文有提到。

5、启动所有的kafka节点启动完成后查看日志，看看是否能稳定运行，不怕抛出异常。运行正常后继续下面的操作。注意，一旦配置Kafka认证后，kafka自带的console-producer和console-consumer都无法使用了。

代码里面

KafkaClient {

org.apache.kafka.common.security.plain.PlainLoginModule required

username="kafka"

password="kafka-sec";

};

bin/writer-kafka-console-producer.sh --broker-list localhost:9092 --topic test --producer.config producer.config

**# 为用户 alice 在 test（topic）上添加读写的权限**

./kafka-acls.sh --authorizer-properties zookeeper.connect=mu3:2181/kafka-acl --add --allow-principal User:consumer --operation ALL --topic kafkaSasl

**# 对于 topic 为 test 的消息队列，拒绝来自 ip 为198.51.100.3账户为 BadBob 进行 read 操作，其他用户都允许**

bin/kafka-acls.sh --authorizer-properties zookeeper.connect=data-rt-dev02:2181/kafka\_test10 --add --allow-principal User:\* --allow-host \* --deny-principal User:BadBob --deny-host 198.51.100.3 --operation Read --topic test

**# 为bob 和 alice 添加all，以允许来自 ip 为198.51.100.0或者198.51.100.1的读写请求**

./kafka-acls.sh --authorizer-properties zookeeper.connect=data-rt-dev02:2181/kafka\_test10 --add --allow-principal User:bob --allow-principal User:alice --allow-host 198.51.100.0 --allow-host 198.51.100.1 --operation Read --operation Write --topic test

**# 列出 topic 为 test 的所有权限账户**

./kafka-acls.sh --authorizer-properties zookeeper.connect=mu3:2181/kafka-acl --list --topic kafkaSasl

**# 移除 acl**

./kafka-acls.sh --authorizer-properties zookeeper.connect=mu3:2181/kafka-acl --remove --allow-principal User:consumer --allow-principal User:consumer --allow-host localhost --allow-host 192.168.16.94 --operation Read --operation Write --topic kafkaSasl

**# producer**

bin/kafka-acls.sh --authorizer-properties zookeeper.connect=mu3:2181/kafka-acl --add --allow-principal User:consumer --producer --topic kafkaSasl

**#consumer**

./kafka-acls.sh --authorizer-properties zookeeper.connect=mu3:2181/kafka-acl --add --allow-principal User:consumer --consumer --topic kafkaSasl --group sasl-group