Kafka Authorization Command Line Interface

Introduction

Kafka ships with a pluggable Authorizer and an out-of-box authorizer implementation that uses zookeeper to store all the acls. Kafka acls are defined in the general format of "Principal P is [Allowed/Denied] Operation O From Host H On Resource R". You can read more about the acl structure on KIP-11. In order to add, remove or list acls you can use the Kafka authorizer CLI.

Command Line interface

Kafka Authorization management CLI can be found under bin directory with all the other CLIs. The CLI script is called kafka-acls.sh. Following lists all the options that the script supports.

Option	Description	Default	Option type
--add	Indicates to the script that user is trying to add an acl.		Action
--remove	Indicates to the script that user is trying to remove an acl.		Action
--list	Indicates to the script that user is trying to list acls.		Action
--authorizer	Fully qualified class name of the authorizer.	kafka.security.auth.SimpleAclAuthorizer	Configuration
--authorizer-properties	comma separated key=val pairs that will be passed to authorizer for initialization.		Configuration
--cluster	Specifies cluster as resource.		Resource
--topic <topic-name>	Specifies the topic as resource.		Resource
--consumer-group <consumer-group>	Specifies the consumer-group as resource.		Resource
--allow-principal	Principal is in PrincipalType:name format. These principals will be used to generate an ACL with Allow permission. Multiple principals can be specified in a single command by specifying this option multiple times, i.e. --allow-principal User:test1@EXAMPLE.COM --allow-principal User:test2@EXAMPLE.COM		Principal
--deny-principal	Principal is in PrincipalType:name format. These principals will be used to generate an ACL with Deny permission. Multiple principals can be specified in the same way as described in --allow-principal option.		Principal
--allow-hosts	Comma separated list of hosts from which principals listed in --allow-principals will have access.	if --allow-principals is specified defaults to * which translates to "all hosts"	Host
--deny-hosts	Comma separated list of hosts from which principals listed in --deny-principals will be denied access.	if --deny-principals is specified defaults to * which translates to "all hosts"	Host
--operations	Comma separated list of operations. Valid values are : Read, Write, Create, Delete, Alter, Describe, ClusterAction, All	All	Operation
--producer	Convenience option to add/remove acls for producer role. This will generate acls that allows WRITE, DESCRIBE on topic and CREATE on cluster.		Convenience
--consumer	Convenience option to add/remove acls for consumer role. This will generate acls that allows READ, DESCRIBE on topic and READ on consumer-group.		Convenience

Examples

Adding Acls

Suppose you want to add an acl "Principals User:Bob and User:Alice are allowed to perform Operation Read and Write on Topic Test-Topic from Host1 and Host2". You can do that by executing the CLI with following options:

```
bin/kafka-acls.sh --authorizer kafka.security.auth.SimpleAclAuthorizer --authorizer-properties zookeeper.
connect=localhost:2181 --add --allow-principal User:Bob --allow-principal User:Alice --allow-hosts Host1,Host2
--operations Read,Write --topic Test-topic
```

By default all principals that don't have an explicit acl that allows access for an operation to a resource are denied. In rare cases where an allow acl is defined that allows access to all but some principal we will have to use the --deny-principals and --deny-host option. For example, if we want to allow all users to Read from Test-topic but only deny User:BadBob from host bad-host we can do so using following commands:

```
bin/kafka-acls.sh --authorizer kafka.security.auth.SimpleAclAuthorizer --authorizer-properties zookeeper.
connect=localhost:2181 --add --allow-principal User:* --allow-hosts * --deny-principal User:BadBob --deny-hosts
bad-host --operations Read--topic Test-topic
```

Above examples add acls to a topic by specifying --topic <topic-name> as the resource option. Similarly user can add acls to cluster by specifying --cluster and to a consumer group by specifying --consumer-group <group-name>.

Removing Acls

Removing acls is pretty much same , the only difference is instead of --add option users will have to specify --remove option. To remove the acls added by the first example above we can execute the CLI with following options:

```
bin/kafka-acls.sh --authorizer kafka.security.auth.SimpleAclAuthorizer --authorizer-properties zookeeper.
connect=localhost:2181 --remove --allow-principal User:Bob --allow-principal User:Alice --allow-hosts Host1,
Host2 --operations Read,Write --topic Test-topic
```

List Acls

We can list acls for any resource by specifying the --list option with the resource. To list all acls for Test-topic we can execute the CLI with following options:

```
bin/kafka-acls.sh --authorizer kafka.security.auth.SimpleAclAuthorizer --authorizer-properties zookeeper.
connect=localhost:2181 --list --topic Test-topic
```

Adding or removing a principal as producer or consumer

The most common use case for acl management are adding/removing a principal as producer or consumer so we added convenience options to handle these cases. In order to add User:Bob as a producer of Test-topic we can execute the following command:

```
bin/kafka-acls.sh --authorizer kafka.security.auth.SimpleAclAuthorizer --authorizer-properties zookeeper.
connect=localhost:2181 --add --allow-principal User:Bob --producer --topic Test-topic
```

Similarly to add Alice as a consumer of Test-topic with consumer group Group-1 we just have to pass --consumer option:

```
 bin/kafka-acls.sh --authorizer kafka.security.auth.SimpleAclAuthorizer --authorizer-properties zookeeper.
connect=localhost:2181 --add --allow-principal User:Bob --consumer --topic test-topic --consumer-group Group-1
```

Note that for consumer option we must also specify the consumer group.

In order to remove a principal from producer or consumer role we just need to pass --remove option.