## Kafka Plugins

Chia-Ping Tsai

## Kafka Plugin Configs

client.quota.callback.class key.serializer remote.log.storage.manager.class.name value.serializer remote.log.metadata.manager.class.name interceptor.classes create.topic.policy.class.name partitioner.class alter.config.policy.class.name key.deserializer authorizer.class.name value.deserializer group.consumer.assignors metric.reporters replica.selector.class

connector.client.config.override.policy

me
replication.policy.class

ame
topic.filter.class

group.filter.class

value-decoder-class

default.client.supplier
default.key.serde
default.timestamp.extractor
processing.exception.handler
default.value.serde
deserialization.exception.handler

default.deserialization.exception.handler

production.exception.handler

default.production.exception.handler

## What is the environment for Plugins?

your application

- 1 Consumer
- 2 Producer
- 3 Admin
- 4 Streams

your server

- 1 Controller
- 2 Broker
- 3 Worker

note the Dependency Hell

## No worries, let's just chat about the policy for plugins

## Num.0 All Plugins are Public APIs

## The Compatibility Policy

MINOR -> MINOR = Binary Compatibility

no re-compilation, no code change, everything work well

MAJOR -> MAJOR = Data Compatibility

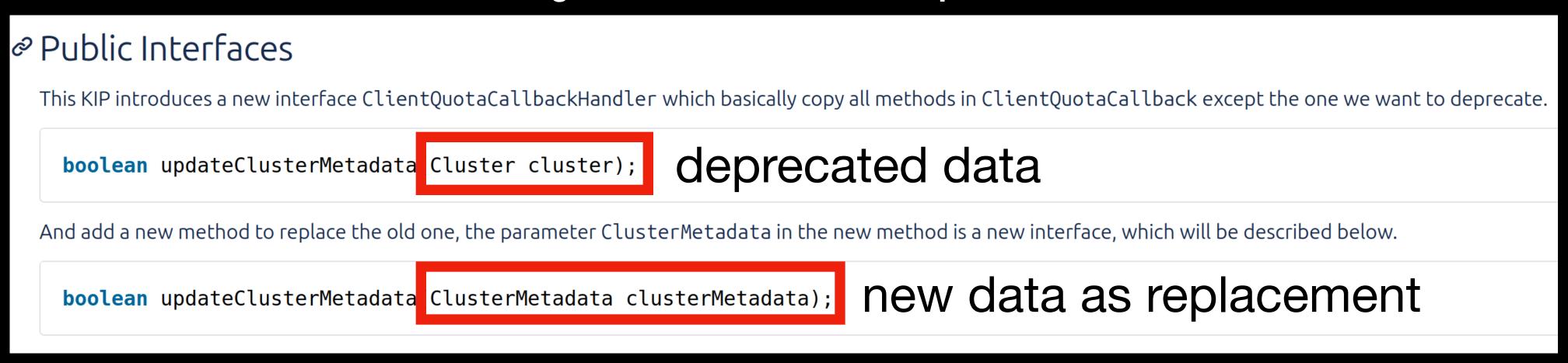
Only data is unbreakable
APIs, RPCs, metrics, and configs are breakable

### Which are Public APIs?

```
/<u>....</u>
 * Plug-able interface for selecting a preferred read replica given the current set of replicas for a partition
 * and metadata from the client.
 * Implement {@link org.apache.kafka.common.metrics.Monitorable} to enable the selector to register metrics.
 * The following tags are automatically added to all metrics registered: <code>config</code> set to
 * <code>replica.selector.class</code>, and <code>class</code> set to the ReplicaSelector class name.
 */
public interface ReplicaSelector extends
                                          Configurable
                                                        Closeable { 13 usages 4 implementations 🚨 David Arthur
    /**
     * Select the preferred replica a client should use for fetching. If no replica is available, this will return an
     * empty optional. -
            <ReplicaView>
                                  TopicPartition topicPartition, 2 implementations 😩 David Arthur
    Optional
                          select
                                  ClientMetadata DlientMetadata,
                                  PartitionView /artitionView);
    @Override 1 override 2 David Arthur
    default void close() throws IOException {
        // No-op by default
    @Override 1 override 2 David Arthur
    default void configure(Map<String, ?> configs) {
        // No-op by default
```

### Breaking Change with Replacement

#### KIP-1162: Redesign ClientQuotaCallback#updateClusterMetadata



noted: not all APIs are replaceable

## Exception is a part of Signature

#### Please throw the exception defined by signature

```
* @throws PolicyViolationException if the request parameters do not satisfy this policy.
*/
void validate(RequestMetadata requestMetadata) throws PolicyViolationException; 3 implementa
```

noted: Kafka prefers RuntimeException type

## What If I throw other Exception?

```
@Override new*
public void validate(RequestMetadata requestMetadata) throws PolicyViolationException {
   if (requestMetadata.topic().equals("chia"))
        throw new PolicyViolationException("chia is not allowed");
   if (requestMetadata.topic().equals("ikea"))
        throw new RuntimeException("Ikea is not allowed in runtime");
}
public void validate(RequestMetadata requestMetadata) throws PolicyViolationException {
        if (requestMetadata.topic().equals("ikea"))
        throw new RuntimeException("Ikea is not allowed in runtime");
}
```

#### Your Log will be flooded with error messages

```
[2025-05-29 12:40:46,049] ERROR Encountered nonFatalFaultHandler fault: createTopics: event failed with R ntimeException (treated as UnknownServerException) at epoch 1 i java.lang.RuntimeException: Ikea is not allowed in runtime

at org.apache.kafka.controller.ReplicationControlManager.maybeCheckCreateTopicPolicy(ReplicationControlManager.java:871) ~[kafka-metadata-4.1.0-SNAPSHOT.jar:?]

at org.apache.kafka.controller.ReplicationControlManager.createTopics(ReplicationControlManager.java:799) ~[kafka-metadata-4.1.0-SNAPSHOT.jar:?]

at org.apache.kafka.controller.ReplicationControlManager.createTopics(ReplicationControlManager.java:666) ~[kafka-metadata-4.1.0-SNAPSHOT.jar:?]

at org.apache.kafka.controller.QuorumController.lambda$createTopics$7(QuorumController.java:1777) ~[kafka-metadata-4.1.0-SNAPSHOT.jar:?]

at org.apache.kafka.controller.QuorumController$ControllerWriteEvent.run(QuorumController.java:785) ~[kafka-metadata-4.1.0-SNAPSHOT.jar:?]

at org.apache.kafka.queue.KafkaEventQueue$EventContext.run(KafkaEventQueue.java:133) ~[kafka-server-common-4.1.0-SNAPSHOT.jar:?]

at org.apache.kafka.queue.KafkaEventQueue$EventHandler.run(KafkaEventQueue.java:187) ~[kafka-server-common-4.1.0-SNAPSHOT.jar:?]

at org.apache.kafka.queue.KafkaEventQueue$EventHandler.run(KafkaEventQueue.java:187) ~[kafka-server-common-4.1.0-SNAPSHOT.jar:?]

at org.apache.kafka.queue.KafkaEventQueue$EventHandler.run(KafkaEventQueue.java:187) ~[kafka-server-common-4.1.0-SNAPSHOT.jar:?]

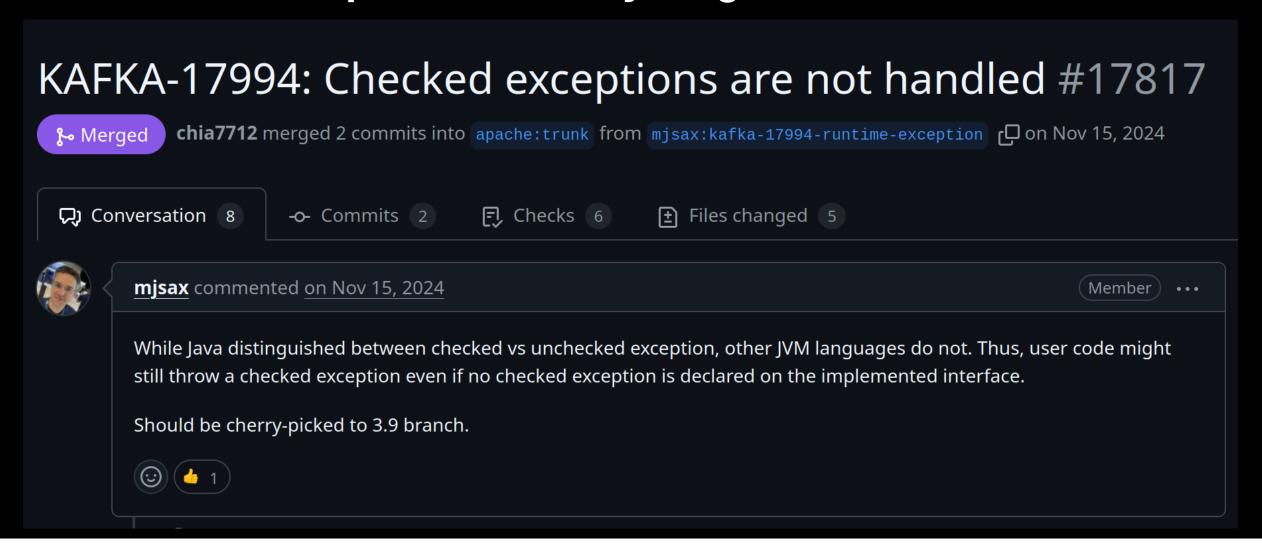
at java.base/java.lang.Thread.run(Thread.java:1583) [?:?]
```

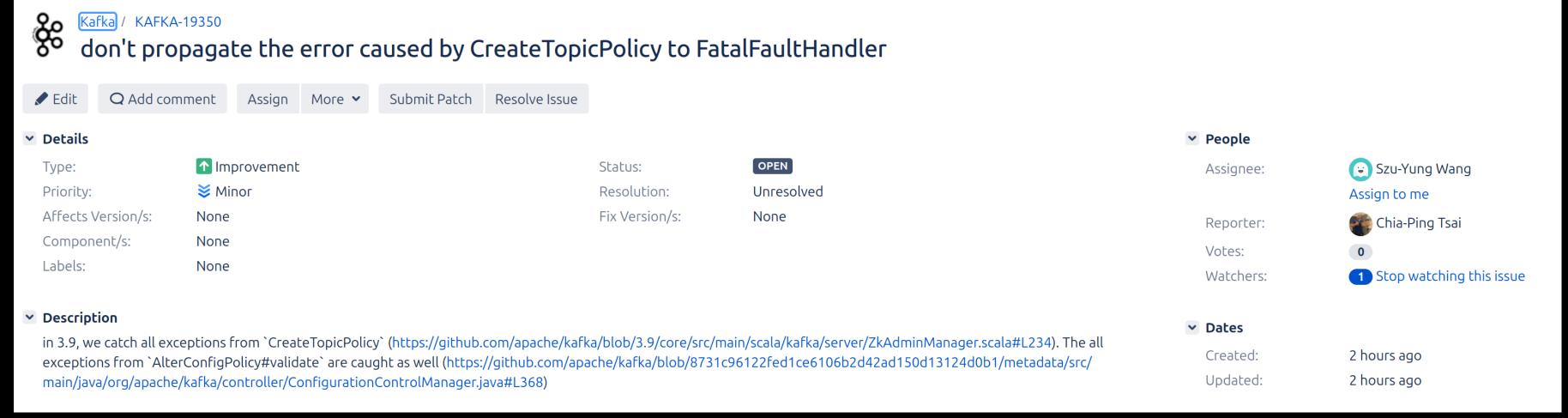
## Exceptions need to be Compatible

```
* In earlier versions a TimeoutException was thrown instead of this. To keep existing catch-clauses working
* this class extends TimeoutException.
*/
public class BufferExhaustedException extends TimeoutException { 10 usages 🚨 Jay Kreps +1
    private static final long seriαlVersionUID = 1L; no usages
    public BufferExhaustedException(String message) { super(message); }
```

### It's easy to overlook Exceptions need to be Compatible

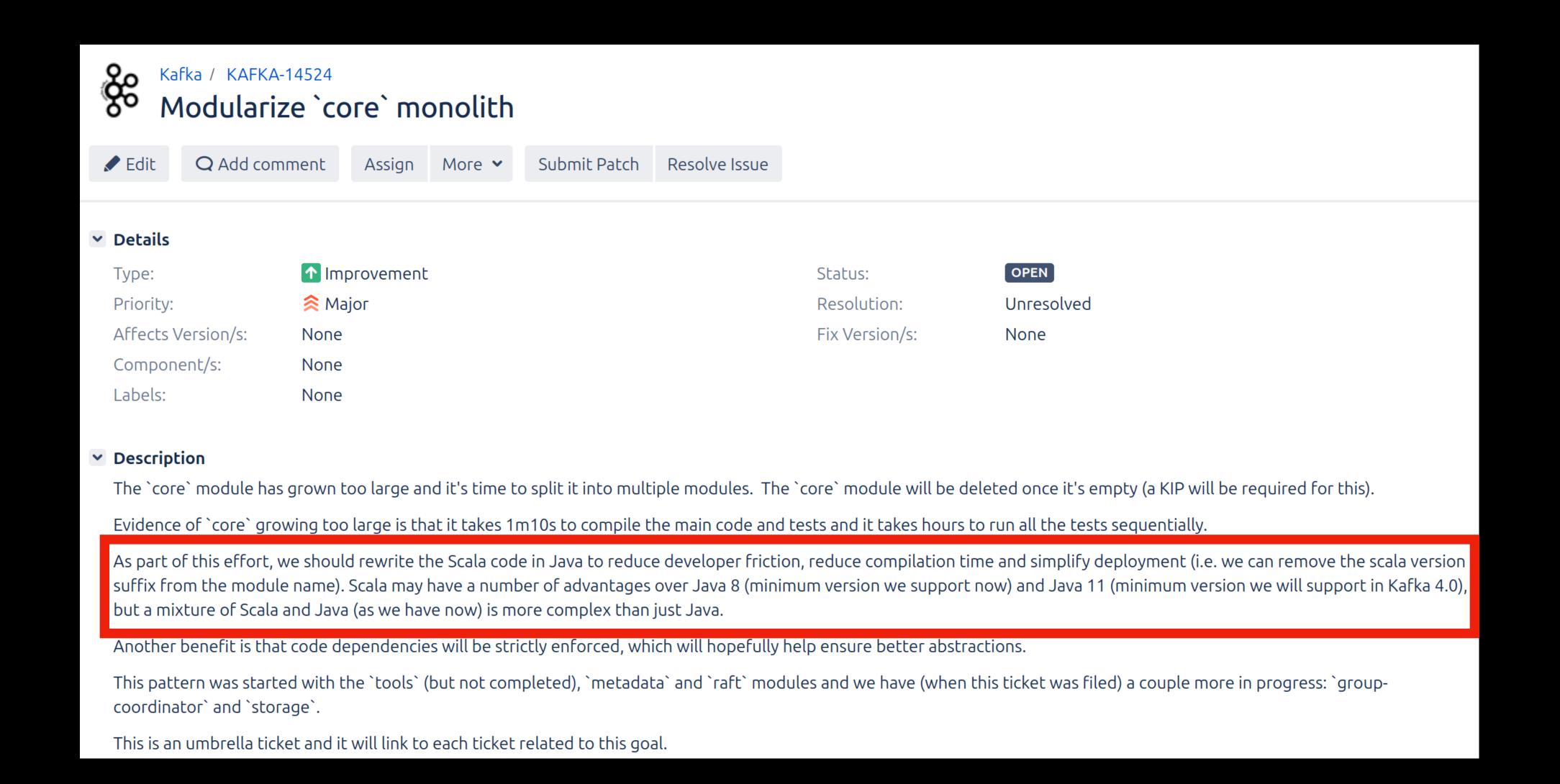
the exception thrown by Plugin is NOT handled





## Num.1 Each Plugin is a Java Interface

## Kafka don't love Scala anymore



## Server-side is still using Scala

```
// Add Scala version
25
       def defaultScala213Version = '2.13.16'
26
      if (hasProperty('scalaVersion')) {
27
        if (scalaVersion == '2.13') {
28
           versions["scala"] = defaultScala213Version
        } else {
30
           versions["scala"] = scalaVersion
31
32
      } else {
33
         versions["scala"] = defaultScala213Version
34
35
```

#### your server

- 1 Controller (o)
- ② Broker (o)
- ③ Worker (x)

noted not all "servers" have scala jars by default

## Mysterious Add-on Interfaces

```
public interface Monitorable {

    /**
    * Provides a {@link PluginMetrics} instance from the component that instantiates the plugin.
    * PluginMetrics can be used by the plugin to register and unregister metrics
    * at any point in their lifecycle prior to their close method being called.
    * Any metrics registered will be automatically removed when the plugin is closed.
    */
    void withPluginMetrics(PluginMetrics metrics);
}
```

```
public interface ClusterResourceListener {
    /**

    * A callback method that a user can implement to get updates for {@link ClusterResource}.
    * @param clusterResource cluster metadata
    */
    void onUpdate(ClusterResource clusterResource);
}
```

```
24
        * Interface for reconfigurable classes that support dynamic configuration.
25
26
       public interface Reconfigurable extends Configurable {
28
           /**
29
            * Returns the names of configs that may be reconfigured.
30
31
           Set<String> reconfigurableConfigs();
32
33
34
            * Validates the provided configuration. The provided map contains
35
            * all configs including any reconfigurable configs that may be different
36
            * from the initial configuration. Reconfiguration will be not performed
37
            * if this method throws any exception.
38
            * @throws ConfigException if the provided configs are not valid. The exception
39
                      message from ConfigException will be returned to the client in
40
                      the AlterConfigs response.
41
            * /
42
           void validateReconfiguration(Map<String, ?> configs) throws ConfigException;
43
44
45
            * Reconfigures this instance with the given key-value pairs. The provided
46
            * map contains all configs including any reconfigurable configs that
47
            * may have changed since the object was initially configured using
48
            * {@link Configurable#configure(Map)}. This method will only be invoked if
49
            * the configs have passed validation using {@link #validateReconfiguration(Map)}.
50
51
           void reconfigure(Map<String, ?> configs);
52
53
54
```

# Num.2 The common methods of Plugin 'configure' and 'close'

## Plugin Lifecycle

new instance noted: you must provide default "public" constructor

config: Map[String, Object]

configure noted: you should create resources in this phase



... do something



COSE noted: this method is always executed even if `configure` fails

「累了,先這樣吧,有任何疑問嘛」