Red Hat Streams for Apache Kafka 2.7

Apache Kafka API 参考流

在 OpenShift Container Platform 中为 Apache Kafka 2.7 配置流部署

Last Updated: 2024-06-28

Red Hat Streams for Apache Kafka 2.7 Apache Kafka API 参考流

在 OpenShift Container Platform 中为 Apache Kafka 2.7 配置流部署

法律通告

摘要

使用 Apache Kafka API 的 Streams 的配置属性来微调您的部署。

目录

前言

对红帽文档提供反馈

我们感谢您对我们文档的反馈。

要改进，创建一个 JIRA 问题并描述您推荐的更改。提供尽可能多的详细信息，以便我们快速解决您的请求。

前提条件

- 您有红帽客户门户网站帐户。此帐户可让您登录到 Red Hat Jira Software 实例。
  如果您没有帐户，系统会提示您创建一个帐户。

流程

1. 点以下内容： Create issue。
2. 在 **Summary** 文本框中输入问题的简短描述。
3. 在 **Description** 文本框中提供以下信息：
   - 找到此问题的页面的 URL。
   - 有关此问题的详细描述。
     您可以将信息保留在任何其他字段中的默认值。
4. 添加 reporter 名称。
5. 点 **Create** 将 JIRA 问题提交到文档团队。

感谢您花时间来提供反馈。

第 1 章 使用模式属性配置自定义资源

自定义资源可以使用配置属性管理和微调 Apache Kafka 组件的流操作。本参考指南描述了适用于多个自定义资源的通用配置属性，以及利用 Apache Kafka 的 Streams 的每个自定义资源模式可用的配置属性。适当地，扩展了属性的描述以及如何提供它们的示例。

为每个模式定义的属性提供了一种结构化和组织方式，用于指定自定义资源的配置。无论是调整资源分配还是指定访问控制，模式中的属性都允许精细配置。例如，您可以使用 **KafkaClusterSpec** 模式的属性为 Kafka 集群指定存储类型，或者添加为 Kafka 代理提供安全访问的监听程序。

模式中的一些属性选项可能会受限制，如属性描述中所述。这些限制定义了可分配给这些属性的值的具体选项或限制。约束可确保使用有效和适当的值配置自定义资源。

第 2 章 常见配置属性

使用 Common 配置属性为 Apache Kafka 自定义资源配置 Streams。您可以在自定义资源中添加常见配置属性，如该资源的任何其他支持的配置一样。

2.1. REPLICAS

使用 **replicas** 属性配置副本。

复制类型取决于资源。

- **KafkaTopic** 使用复制因素来配置 Kafka 集群中每个分区的副本数。
- Kafka 组件使用副本在部署中配置 pod 数量，以提供更好的可用性和可扩展性。

注意

在 OpenShift 上运行 Kafka 组件时，可能不需要运行多个副本来实现高可用性。当部署组件的节点崩溃时，OpenShift 会自动将 Kafka 组件 pod 重新调度到不同的节点。但是，使用多个副本运行 Kafka 组件可以提供更快的故障切换时间，因为其他节点将启动并运行。

2.2. BOOTSTRAPSERVERS

使用 **bootstrapServers** 属性配置 bootstrap 服务器列表。

bootstrap 服务器列表可以引用没有在同一 OpenShift 集群中部署的 Kafka 集群。它们也可以引用不是由 Apache Kafka Streams 部署的 Kafka 集群。

如果在同一 OpenShift 集群中，每个列表必须最好包含名为 ***CLUSTER-NAME*-kafka-bootstrap 和端口号的 Kafka 集群 bootstrap** 服务。如果由 Streams for Apache Kafka 部署但在不同的 OpenShift 集群中，列表内容取决于用于公开集群的方法(routes、ingress、nodeports 或 loadbalancers)。

当在不是由 Apache Kafka 的 Streams 管理的 Kafka 集群中使用 Kafka 时，您可以根据给定集群的配置指定 bootstrap 服务器列表。

2.3. SSL （支持的 TLS 版本和密码套件）

您可以纳入 SSL 配置和密码套件规格，以进一步保护客户端应用程序和 Kafka 集群之间的基于 TLS 的通信。除了标准 TLS 配置外，您还可以在 Kafka 代理配置中指定受支持的 TLS 版本并启用密码套件。如果要限制其使用的 TLS 版本和密码套件，您还可以将配置添加到您的客户端。客户端上的配置必须使用在代理上启用的协议和密码套件。

密码套件是用于安全连接和数据传输的一组安全机制。例如，密码套件 **TLS_AES_256_GCM_SHA384** 由以下机制组成，它们与 TLS 协议一起使用：

- AES （高级加密标准）加密(256 位密钥)
- GCM (Galois/Counter Mode)验证加密
- SHA384 （安全哈希算法）数据完整性保护

该组合封装在 **TLS_AES_256_GCM_SHA384** 密码套件规格中。

ssl.enabled.protocols 属性指定可用于保护集群及其客户端之间的通信的可用 TLS 版本。 **ssl.protocol**

属性为所有连接设置默认 TLS 版本，且必须从启用的协议中选择。使用 **ssl.endpoint.identification.algorithm** 属性启用或禁用主机名验证（仅适用于 Kafka 客户端 - Kafka Connect, MirrorMaker 1/2 和 Kafka Bridge）。

SSL 配置示例

```
# ...
config:
  ssl.cipher.suites: TLS_AES_256_GCM_SHA384,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (1)
  ssl.enabled.protocols: TLSv1.3, TLSv1.2 (2)
  ssl.protocol: TLSv1.3 (3)
  ssl.endpoint.identification.algorithm: HTTPS (4)
# ...
```

1. 启用密码套件规格。
2. 支持的 TLS 版本。
3. 默认 TLS 版本为 **TLSv1.3**。如果客户端只支持 TLSv1.2，它仍然可以连接到代理并使用该支持的版本进行通信，如果配置位于客户端，并且代理只支持 TLSv1.2，则反之亦然。
4. 通过设置为 **HTTPS** 来启用主机名验证。一个空字符串会禁用验证。

2.4. TRUSTEDCERTIFICATES

设置 **tls** 来配置 TLS 加密后，使用 **trustedCertificates** 属性为 secret 列表提供证书以 X.509 格式存储的密钥名称。

您可以使用 Cluster Operator 为 Kafka 集群创建的 secret，也可以创建自己的 TLS 证书文件，然后从文件中创建 **Secret** ：

```
oc create secret generic MY-SECRET \
--from-file=MY-TLS-CERTIFICATE-FILE.crt
```

TLS 加密配置示例

```
tls:
  trustedCertificates:
    - secretName: my-cluster-cluster-cert
      certificate: ca.crt
    - secretName: my-cluster-cluster-cert
      certificate: ca2.crt
```

如果证书存储在同一 secret 中，则可以多次列出。

如果要启用 TLS 加密，但使用 Java 附带的默认公共证书颁发机构集合，您可以将 **trustedCertificates** 指定为空数组：

使用默认 Java 证书启用 TLS 的示例

```
tls:
  trustedCertificates: []
```

有关配置 mTLS 身份验证的详情，请参考 **KafkaClientAuthenticationTls** 模式参考。

2.5. RESOURCES

配置 *资源请求和限值*，以控制 Apache Kafka 容器流的资源。您可以为 **内存和 cpu** 资源指定请求和限值。请求应该足以确保 Kafka 的稳定性能。

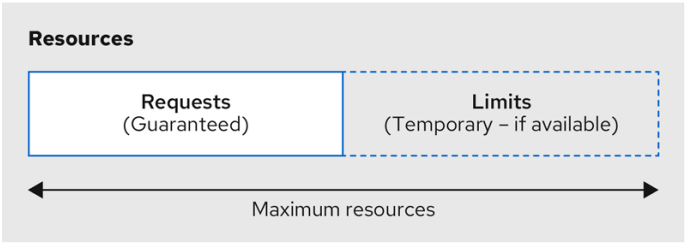
如何在生产环境中配置资源取决于多个因素。例如，应用可能会共享 OpenShift 集群中的资源。

对于 Kafka，部署的以下方面可能会影响您需要的资源：

- 信息吞吐量和大小
- 处理消息的网络线程数量
- 生成者和消费者的数量
- 主题和分区数量

为资源请求指定的值保留，并始终可供容器使用。资源限值指定给定容器可消耗的最大资源。请求和限制之间的数量不会被保留，且可能并不总是可用。容器只能在资源可用时最多使用限制。资源限值是临时的，可以重新分配。

资源请求和限值



212_Streams_0322

如果您在设置限制时没有设置请求，或反之，OpenShift 会将相同的值用于限制和请求。为资源设置相等的请求和限值保证服务质量，因为 OpenShift 不会终止容器，除非超过其限值。

您可以为一个或多个支持的资源配置资源请求和限值。

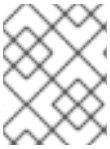
资源配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    #...
    resources:
      requests:
        memory: 64Gi
        cpu: "8"
```

```
    limits:
      memory: 64Gi
      cpu: "12"
  entityOperator:
    #...
    topicOperator:
      #...
      resources:
        requests:
          memory: 512Mi
          cpu: "1"
        limits:
          memory: 512Mi
          cpu: "1"
```

Topic Operator 和 User Operator 的资源请求和限值在 **Kafka** 资源中设置。

如果资源请求超过 OpenShift 集群中的可用可用资源，则不会调度该容器集。



注意

Apache Kafka 的流使用 OpenShift 语法来指定 **内存和 cpu** 资源。有关在 OpenShift 中管理计算资源的更多信息，请参阅管理容器的计算资源。

内存资源

在配置内存资源时，请考虑组件的总要求。
Kafka 在 JVM 中运行，并使用操作系统页面缓存在写入磁盘前存储消息数据。Kafka 的内存请求应该适合 JVM 堆和页面缓存。您可以配置 **jvmOptions** 属性 来控制最小和最大堆大小。

其他组件不依赖于页面缓存。您可以在不配置 **jvmOptions** 的情况下，控制堆大小的情况下配置内存资源。

内存请求和限值以兆字节、GB、兆字节和千兆字节指定。在规格中使用以下后缀：

- **M** 表示 MB
- **G** 表示 GB
- **Mi** 代表 mebibytes
- **Gi** 代表 gibibytes

使用不同内存单元的资源示例

```
# ...
resources:
  requests:
    memory: 512Mi
  limits:
    memory: 2Gi
# ...
```

有关内存规格和其他支持的单元的详情，请参阅收集 内存。

CPU 资源

CPU 请求应该足以随时提供可靠的性能。CPU 请求和限制以 *cores* 或 *millicpus/millicores* 的形式指定。
CPU 内核指定为整数(**5** 个 CPU 内核)或十进制(**2.5** 个 CPU 内核)。1000 *millicore* 与 **1** 个 CPU 内核相同。

CPU 单元示例

```
# ...
resources:
  requests:
    cpu: 500m
  limits:
    cpu: 2.5
# ...
```

1 个 CPU 内核的计算能力可能会因部署 OpenShift 的平台而异。

有关 CPU 规格的更多信息，请参阅 CPU 机制。

2.6. IMAGE

使用 **image** 属性配置组件使用的容器镜像。

只有在需要使用不同的容器 registry 或自定义镜像时，才建议在特殊情况下覆盖容器镜像。

例如，如果您的网络不允许访问 Apache Kafka 的 Streams 使用的容器存储库，您可以复制 Apache Kafka 镜像的流或从源构建它们。但是，如果配置的镜像与 Apache Kafka 镜像的 Streams 不兼容，则它可能无法正常工作。

容器镜像的副本也可以自定义并用于调试。

您可以使用以下资源中的 **image** 属性指定要用于组件的容器镜像：

- **Kafka.spec.kafka**
- **Kafka.spec.zookeeper**
- **Kafka.spec.entityOperator.topicOperator**
- **Kafka.spec.entityOperator.userOperator**
- **Kafka.spec.entityOperator.tlsSidecar**
- **Kafka.spec.cruiseControl**
- **Kafka.spec.kafkaExporter**
- **Kafka.spec.kafkaBridge**
- **KafkaConnect.spec**
- **KafkaMirrorMaker.spec**
- **KafkaMirrorMaker2.spec**

- **KafkaBridge.spec**

注意

更改 Kafka 镜像版本不会自动为其他 Kafka 组件更新镜像版本，如 Kafka Exporter。这些组件不依赖于版本，因此在更新 Kafka 镜像版本时不需要额外的配置。

为 **Kafka、Kafka Connect 和 Kafka MirrorMaker 配置image 属性**

Kafka、Kafka Connect 和 Kafka MirrorMaker 支持多个 Kafka 版本。每个组件都需要自己的镜像。不同 Kafka 版本的默认镜像在以下环境变量中配置：

- **STRIMZI_KAFKA_IMAGES**
- **STRIMZI_KAFKA_CONNECT_IMAGES**
- **STRIMZI_KAFKA_MIRROR_MAKER2_IMAGES**
- （已弃用） **STRIMZI_KAFKA_MIRROR_MAKER_IMAGES**

这些环境变量包含 Kafka 版本和对应镜像之间的映射。映射与 **image** 和 **version** 属性一起使用，以确定所使用的镜像：

- 如果自定义资源中没有给出镜像或 **版本**，则 **版本** 默认为 Cluster Operator 的默认 Kafka 版本，使用的镜像是环境变量中与此版本对应的镜像。
- **如果指定了 image 但没有指定 version，则指定的镜像会被使用，其 version 被假设为 Cluster Operator 的默认 Kafka 版本。**
- **如果给出 version，但没有提供 image，则使用与环境变量中给定版本对应的镜像。**
- **如果同时提供了 version 和 image，则使用给定的镜像。假设镜像包含带有给定版本的 Kafka 镜像。**

组件的 镜像和 版本 可以在以下属性中配置：

- 对于 **spec.kafka.image 和 spec.kafka.version 中的 Kafka。**
- 对于 **spec.image 和 spec.version 中的 Kafka Connect 和 Kafka MirrorMaker。**

警告

建议您仅提供 version，并不指定 image 属性。这可减少配置自定义资源时出错的机会。如果您需要更改用于不同 Kafka 版本的镜像，最好配置 Cluster Operator 的环境变量。

在其他资源中配置 image 属性

对于其他组件的自定义资源中的 image 属性，在部署期间使用给定值。如果没有设置 image 属性，则使用在 Cluster Operator 配置中指定为环境变量的容器镜像。如果在 Cluster Operator 配置中没有定义镜像名称，则会使用默认值。

如需有关镜像环境变量的更多信息，请参阅配置 Cluster Operator。

表 2.1. 镜像环境变量和默认值

组件	环境变量	默认镜像
Topic Operator	STRIMZI_DEFAULT_TOPIC_OPERATOR_IMAGE	registry.redhat.io/amq-streams/strimzi-rhel9-operator:2.7.0
User Operator	STRIMZI_DEFAULT_USER_OPERATOR_IMAGE	registry.redhat.io/amq-streams/strimzi-rhel9-operator:2.7.0
Entity Operator TLS sidecar	STRIMZI_DEFAULT_TLS_SIDECAR_ENTITY_OPERATOR_IMAGE	registry.redhat.io/amq-streams/kafka-37-rhel9:2.7.0
Kafka Exporter	STRIMZI_DEFAULT_KAFKA_EXPORTER_IMAGE	registry.redhat.io/amq-streams/kafka-37-rhel9:2.7.0
Sything Control	STRIMZI_DEFAULT_CRUISE_CONTROL_IMAGE	registry.redhat.io/amq-streams/kafka-37-rhel9:2.7.0
Kafka Bridge	STRIMZI_DEFAULT_KAFKA_BRIDGE_IMAGE	registry.redhat.io/amq-streams/bridge-rhel9:2.7.0

组件	环境变量	默认镜像
Kafka initializer	**STRIMZI_DEFAULT_KAFKA_INIT_IMAGE**	**registry.redhat.io/amq-streams/strimzi-rhel9-operator:2.7.0**

容器镜像配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
    image: my-org/my-image:latest
    # ...
  zookeeper:
    # ...
```

2.7. LIVENESSPROBE 和 READINESSPROBE 状况检查

使用 livenessProbe 和 readinessProbe 属性为 Apache Kafka 配置流中支持的健康检查探测。

健康检查是定期测试，用于验证应用程序的健康状况。当 Healthcheck 探测失败时，OpenShift 会假定应用不健康并尝试修复它。

有关探测的详情，请参阅配置存活度和就绪度探测。

livenessProbe 和 readinessProbe 都支持以下选项：

- initialDelaySeconds
- timeoutSeconds

- periodSeconds
- successThreshold
- failureThreshold

存活度和就绪度探测配置示例

```
# ...
readinessProbe:
  initialDelaySeconds: 15
  timeoutSeconds: 5
livenessProbe:
  initialDelaySeconds: 15
  timeoutSeconds: 5
# ...
```

有关 livenessProbe 和 readinessProbe 选项的更多信息，请参阅 探测架构参考。

2.8. METRICSCONFIG

使用 metricsConfig 属性启用和配置 Prometheus 指标。

metricsConfig 属性包含对具有 Prometheus JMX Exporter 的 ConfigMap 的额外配置的引用。Apache Kafka 的流支持使用 Prometheus JMX exporter 的 Prometheus 指标将 Apache Kafka 和 ZooKeeper 支持的 JMX 指标转换为 Prometheus 指标。

要在不进一步配置的情况下启用 Prometheus metrics 导出，您可以在 metricsConfig.valueFrom.configMapKeyRef.key 下引用包含空文件的 ConfigMap。当引用空文件时，所有指标都会公开，只要它们尚未重命名。

带有 Kafka 指标配置的 ConfigMap 示例

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: my-configmap
data:
  my-key: |
    lowercaseOutputName: true
    rules:
    # Special cases and very specific rules
    - pattern: kafka.server<type=(.+), name=(.+), clientId=(.+), topic=(.+), partition=(.*)><>Value
      name: kafka_server_$1_$2
      type: GAUGE
      labels:
       clientId: "$3"
       topic: "$4"
       partition: "$5"
    # further configuration
```

Kafka 的指标配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
    metricsConfig:
      type: jmxPrometheusExporter
      valueFrom:
        configMapKeyRef:
          name: my-config-map
          key: my-key
    # ...
  zookeeper:
    # ...
```

启用指标后，会在端口 9404 公开它们。

当资源中没有定义 metricsConfig （或已弃用的 metrics）属性时，Prometheus 指标会被禁用。

有关设置和部署 Prometheus 和 Grafana 的更多信息，请参阅 将指标引入到 Kafka。

2.9. JVMOPTIONS

以下 Apache Kafka 组件的流在 Java 虚拟机(JVM)中运行：

- Apache Kafka
- Apache ZooKeeper
- Apache Kafka Connect
- Apache Kafka MirrorMaker
- Apache Kafka Bridge 的流

要在不同平台和构架中优化其性能，您可以在以下资源中配置 jvmOptions 属性：

- Kafka.spec.kafka
- Kafka.spec.zookeeper
- Kafka.spec.entityOperator.userOperator
- Kafka.spec.entityOperator.topicOperator
- Kafka.spec.cruiseControl

- KafkaNodePool.spec
- KafkaConnect.spec
- KafkaMirrorMaker.spec
- KafkaMirrorMaker2.spec
- KafkaBridge.spec

您可以在配置中指定以下选项：

-Xms

JVM 启动时的最小初始分配堆大小

-Xmx

最大堆大小

-XX

JVM 的高级运行时选项

javaSystemProperties

其他系统属性

gcLoggingEnabled

启用垃圾收集器日志记录

注意

JVM 设置接受的单元（如 -Xmx 和 -Xms ）是对应镜像中 JDK java 二进制文件所接受的单元。因此，1g 或 1G 表示 1,073,741,824 字节，Gi 不是有效的单位后缀。这与用于内存请求和限值 的单元不同，其遵循 OpenShift 约定，其中 1G 表示 1,000,000,000 字节，1Gi 表示 1,073,741,824 字节。

-Xms 和 -Xmx 选项

除了为容器设置内存请求和限制值外，您还可以使用 -Xms 和 -Xmx JVM 选项为您的 JVM 设置特定的堆大小。使用 -Xms 选项设置初始堆大小和 -Xmx 选项，以设置最大堆大小。

指定堆大小，以更好地控制分配给 JVM 的内存。堆大小应充分利用 容器的内存限值（和请求），而不超过它。堆大小以及任何其他内存要求都需要在指定的内存限值中容纳。如果您没有在配置中指定堆大小，但配置内存资源限制（和请求），Cluster Operator 会自动实施默认的堆大小。Cluster Operator 根据内存资源配置百分比设置默认的最大值和最小堆值。

下表显示了默认的堆值。

表 2.2. 组件的默认堆设置

组件	分配给堆的可用内存百分比	最大限制
Kafka	50%	5 GB
ZooKeeper	75%	2 GB
Kafka Connect	75%	None
MirrorMaker 2	75%	None
MirrorMaker	75%	None
Sything Control	75%	None
Kafka Bridge	50%	31 Gi

如果没有指定内存限制（和请求），则 JVM 的最小堆大小设置为 128M。JVM 的最大堆大小没有定义，以允许内存根据需要增加。这是测试和开发中的单一节点环境的理想选择。

设置适当的内存请求可能会阻止以下内容：

- 如果节点上运行其他容器集的内存压力，OpenShift 会终止容器。
- OpenShift 将容器调度到内存不足的节点。如果将 -Xms 设置为 -Xmx，则容器将立即崩溃；如果没有，则容器将稍后崩溃。

在本例中，JVM 将 2 GiB (=2,147,483,648 字节)用于其堆。JVM 内存用量总量可能比最大堆大小多得多。

示例 -Xmx 和 -Xms 配置

```
# ...
jvmOptions:
  "-Xmx": "2g"
  "-Xms": "2g"
# ...
```

为 initial (-Xms)和最大(-Xmx)堆大小设置相同的值，可以避免 JVM 启动后必须分配内存，但成本可能比实际需要的堆更大。

重要

执行大量磁盘 I/O （如 Kafka 代理容器）的容器需要可用内存用作操作系统页面缓存。对于这样的容器，请求的内存应显著高于 JVM 使用的内存。

-XX 选项

-XX 选项用于配置 Apache Kafka 的 KAFKA_JVM_PERFORMANCE_OPTS 选项。

示例 -XX 配置

```
jvmOptions:
  "-XX":
    "UseG1GC": "true"
    "MaxGCPauseMillis": "20"
    "InitiatingHeapOccupancyPercent": "35"
    "ExplicitGCInvokesConcurrent": "true"
```

由 -XX 配置生成的 JVM 选项

```
-XX:+UseG1GC -XX:MaxGCPauseMillis=20 -XX:InitiatingHeapOccupancyPercent=35 -
XX:+ExplicitGCInvokesConcurrent -XX:-UseParNewGC
```

注意

如果没有指定 -XX 选项，则使用 KAFKA_JVM_PERFORMANCE_OPTS 的默认 Apache Kafka 配置。

javaSystemProperties

javaSystemProperties 用于配置额外的 Java 系统属性，如调试实用程序。

javaSystemProperties 配置示例

```
jvmOptions:
  javaSystemProperties:
    - name: javax.net.debug
      value: ssl
```

有关 jvmOptions 的更多信息，请参阅 JvmOptions 模式参考。

2.10. 垃圾收集器日志记录

jvmOptions 属性还允许您启用和禁用垃圾收集器(GC)日志记录。默认情况下禁用 GC 日志记录。要启用它，请设置 gcLoggingEnabled 属性，如下所示：

GC 日志记录配置示例

```
# ...
jvmOptions:
  gcLoggingEnabled: true
# ...
```

第 3 章 KAFKA 模式参考

属性	属性类型	描述
spec	**KafkaSpec**	Kafka 和 ZooKeeper 集群的规格，以及 Topic Operator。
status	**KafkaStatus**	Kafka 和 ZooKeeper 集群的状态，以及 Topic Operator。

第 4 章 KAFKASPEC 模式参考

使用于： Kafka

属性	属性类型	描述
kafka	KafkaClusterSpec	配置 Kafka 集群。
zookeeper	ZookeeperClusterSpec	ZooKeeper 集群的配置。运行基于 ZooKeeper 的 Apache Kafka 集群时需要此部分。
entityOperator	EntityOperatorSpec	配置实体 Operator。
clusterCa	certificateAuthority	配置集群证书颁发机构。
clientsCa	certificateAuthority	配置客户端证书颁发机构。
cruiseControl	CruiseControlSpec	配置 Cruise Control 部署。在指定时部署 Cruise Control 实例。
jmxTrans	JmxTransSpec	**jmxTrans 属性已弃用。**jmxtrans 已被弃用，并在 Apache Kafka 2.5 的 Streams 中删除相关资源。从 Apache Kafka 2.5 的 Streams 开始，JMXTrans 不再被支持，这个选项将被忽略。
kafkaExporter	KafkaExporterSpec	配置 Kafka Exporter。Kafka Exporter 可以提供额外的指标，例如在 topic/partition 的消费者组的滞后。
maintenanceTimeWindows	字符串数组	用于维护任务的时间窗列表（即证书续订）。每个时间窗都由 cron 表达式定义。

第 5 章 KAFKACLUSTERSPEC 模式参考

使用于： KafkaSpec

KafkaClusterSpec 模式属性的完整列表

配置 Kafka 集群。

5.1. 监听器

使用 listeners 属性配置监听程序来提供 Kafka 代理的访问。

没有身份验证的普通（未加密）侦听器配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
spec:
  kafka:
    # ...
    listeners:
      - name: plain
        port: 9092
        type: internal
        tls: false
    # ...
  zookeeper:
    # ...
```

5.2. CONFIG

使用 config 属性将 Kafka 代理选项配置为密钥。

这些值可以是以下 JSON 类型之一：

- 字符串
- Number
- 布尔值

例外

您可以指定并配置 Apache Kafka 文档 中列出的选项。

但是，Apache Kafka 的流负责配置和管理与以下内容相关的选项，而这无法更改：

- 安全性（加密、身份验证和授权）
- 侦听器配置
- 代理 ID 配置
- 配置日志数据目录
- Inter-broker 通信
- ZooKeeper 连接

无法设置具有以下前缀的属性：

- advertised.

- 授权者.
- broker.
- controller
- cruise.control.metrics.reporter.bootstrap.
- cruise.control.metrics.topic
- host.name
- inter.broker.listener.name
- 侦听器.
- 监听器.
- log.dir
- 密码。
- 端口
- process.roles
- SASL。

- 安全性。
- servers,node.id
- ssl.
- super.user
- zookeeper.clientCnxnSocket
- zookeeper.connect
- zookeeper.set.acl
- zookeeper.ssl

如果 config 属性包含一个无法更改的选项，它将被忽略，并将警告信息记录到 Cluster Operator 日志文件。所有其他支持选项都转发到 Kafka，包括对 Apache Kafka 的 Streams 配置的选项的以下例外：

- 支持的 TLS 版本和密码套件的任何 ssl 配置
- zookeeper.connection.timeout.ms 属性的配置，以设置建立 ZooKeeper 连接的最大时间
- Cruise Control 指标属性：
  - cruise.control.metrics.topic.num.partitions
  - cruise.control.metrics.topic.replication.factor

- cruise.control.metrics.topic.retention.ms
- cruise.control.metrics.topic.auto.create.retries
- cruise.control.metrics.topic.auto.create.timeout.ms
- cruise.control.metrics.topic.min.insync.replicas

- 控制器属性：
  - controller.quorum.election.backoff.max.ms
  - controller.quorum.election.timeout.ms
  - controller.quorum.fetch.timeout.ms

Kafka 代理配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
    config:
      num.partitions: 1
      num.recovery.threads.per.data.dir: 1
      default.replication.factor: 3
      offsets.topic.replication.factor: 3
      transaction.state.log.replication.factor: 3
      transaction.state.log.min.isr: 1
      log.retention.hours: 168
      log.segment.bytes: 1073741824
      log.retention.check.interval.ms: 300000
      num.network.threads: 3
      num.io.threads: 8
      socket.send.buffer.bytes: 102400
```

```
    socket.receive.buffer.bytes: 102400
    socket.request.max.bytes: 104857600
    group.initial.rebalance.delay.ms: 0
    zookeeper.connection.timeout.ms: 6000
  # ...
```

5.3. BROKERRACKINITIMAGE

启用机架感知后，Kafka 代理 pod 使用 init 容器从 OpenShift 集群节点收集标签。用于此容器的容器镜像可以使用 brokerRackInitImage 属性进行配置。当缺少 brokerRackInitImage 字段时，会按照优先级顺序使用以下镜像：

1. 在 Cluster Operator 配置中的 STRIMZI_DEFAULT_KAFKA_INIT_IMAGE 环境变量中指定的容器镜像。
2. registry.redhat.io/amq-streams/strimzi-rhel9-operator:2.7.0 container image.

brokerRackInitImage 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
    rack:
      topologyKey: topology.kubernetes.io/zone
    brokerRackInitImage: my-org/my-image:latest
    # ...
```

注意

只有在特殊情况下，才建议使用覆盖容器镜像，其中需要使用不同的容器 registry。例如，因为您的网络不允许访问由 Apache Kafka 的 Streams 使用的容器 registry。在这种情况下，您应该复制 Apache Kafka 镜像的 Streams，或者从源构建它们。如果配置的镜像与 Apache Kafka 镜像的 Streams 不兼容，则它可能无法正常工作。

5.4. LOGGING

Kafka 都有自己的可配置的日志记录器，其中包括：

- log4j.logger.org.I0Itec.zkclient.ZkClient
- log4j.logger.org.apache.zookeeper
- log4j.logger.kafka
- log4j.logger.org.apache.kafka
- log4j.logger.kafka.request.logger
- log4j.logger.kafka.network.Processor
- log4j.logger.kafka.server.KafkaApis
- log4j.logger.kafka.network.RequestChannel$
- log4j.logger.kafka.controller
- log4j.logger.kafka.log.LogCleaner

- log4j.logger.state.change.logger
- log4j.logger.kafka.authorizer.logger

Kafka 使用 Apache log4j 日志记录器实现。

使用 logging 属性配置日志记录器和日志记录器级别。

您可以通过直接指定日志记录器和级别（在线）或使用自定义（外部） ConfigMap 来设置日志级别。如果使用 ConfigMap，您可以将 logging.valueFrom.configMapKeyRef.name 属性设置为包含外部日志记录配置的 ConfigMap 的名称。在 ConfigMap 中，日志记录配置使用 log4j.properties 描述。logging.valueFrom.configMapKeyRef.name 和 logging.valueFrom.configMapKeyRef.key 属性都是强制的。使用指定的确切日志记录配置的 ConfigMap 会在 Cluster Operator 运行时使用自定义资源创建，然后在每次协调后重新创建。如果没有指定自定义 ConfigMap，则使用默认日志记录设置。如果没有设置特定的日志记录器值，则会为该日志记录器继承上级日志记录器设置。有关日志级别的更多信息，请参阅 Apache 日志记录服务。

在这里，我们看到 内联 和 外部日志记录 的示例。inline 日志记录指定根日志记录器级别。您还可以通过将特定类或日志记录器添加到 loggers 属性来设置日志级别。

内联日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
spec:
  # ...
  kafka:
    # ...
    logging:
      type: inline
      loggers:
        kafka.root.logger.level: INFO
        log4j.logger.kafka.coordinator.transaction: TRACE
        log4j.logger.kafka.log.LogCleanerManager: DEBUG
        log4j.logger.kafka.request.logger: DEBUG
        log4j.logger.io.strimzi.kafka.oauth: DEBUG
        log4j.logger.org.openpolicyagents.kafka.OpaAuthorizer: DEBUG
  # ...
```

注意

将日志级别设置为 DEBUG 可能会导致大量日志输出，并可能会影响性能。

外部日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
spec:
  # ...
  logging:
    type: external
    valueFrom:
      configMapKeyRef:
        name: customConfigMap
        key: kafka-log4j.properties
  # ...
```

任何未配置的可用日志记录器将其级别设置为 OFF。

如果使用 Cluster Operator 部署 Kafka，则动态应用对 Kafka 日志记录级别的更改。

如果使用外部日志记录，则会在日志附加程序更改时触发滚动更新。

垃圾收集器(GC)

垃圾回收收集器日志记录也可以使用 jvmOptions 属性启用 （或禁用）。

5.5. KAFKACLUSTERSPEC 模式属性

属性	属性类型	描述

属性	属性类型	描述
version	string	Kafka 代理版本。默认为最新版本。请参阅用户文档了解升级或降级版本所需的流程。
metadataVersion	string	在 Streams for Apache Kafka 2.7 中添加。Kafka 集群使用的 KRaft 元数据版本。在 ZooKeeper 模式下运行时，会忽略此属性。如果没有设置属性，则默认为与 **version** 属性对应的元数据版本。
replicas	整数	集群中的 pod 数量。当节点池没有使用时，需要此属性。
image	string	用于 Kafka pod 的容器镜像。如果没有设置属性，则默认 Kafka 镜像版本会根据 **版本** 配置决定。镜像名称被专门映射到 Cluster Operator 配置中的对应版本。更改 Kafka 镜像版本不会自动为其他组件更新镜像版本，如 Kafka Exporter。
监听器	**GenericKafkaListener** 数组	配置 Kafka 代理的监听程序。

属性	属性类型	描述
config	map	无法设置具有以下前缀的 Kafka 代理配置属性： listeners, advertised., broker., listener., host.name, port, inter.broker.listener.name, sasl., ssl., security., password., log.dir, zookeeper.connect, zookeeper.set.acl, zookeeper.ssl, zookeeper.clientCnxnSocket, authorizer., super.user, cruise.control.metrics.topic, cruise.control.metrics.reporter.bootstrap.servers, node.id, process.roles, controller., metadata.log.dir, zookeeper.metadata.migration.enable （带有 zookeeper.connection.timeout.ms, sasl.server.max.receive.size, sasl.server.max.receive.size, zookeeper.connection.timeout.ms, sasl.server.max.receive.size, ssl.cipher.suites, ssl.protocol, ssl.enabled.protocols, ssl.secure.random.implementation, cruise.control.metrics.topic.num.partitions, cruise.control.metrics.topic.replication.factor, cruise.control.metrics.topic.retention.ms, cruise.control.metrics.topic.create.retries, cruise.control.metrics.topic.auto.create.timeout.ms, cruise.control.metrics.topic.min.insync.replicas, controller.quorum.election.backoff.max.ms, controller.quorum.election.timeout.ms, controller.quorum.fetch.timeout.ms）。
storage	**EphemeralStorage**, **PersistentClaimStorage**, **JbodStorage**	存储配置（磁盘）。无法更新。当节点池没有使用时，需要此属性。
授权	**KafkaAuthorizationSimple**, **KafkaAuthorizationOpa**, **KafkaAuthorizationKeycloak**, **KafkaAuthorizationCustom**	Kafka 代理的授权配置。
rack	**Rack**	配置 **broker.rack** 代理配置。
brokerRackInitImage	string	用于初始化 **broker.rack** 的 init 容器的镜像。
livenessProbe	**probe**	Pod 存活度检查。

属性	属性类型	描述
readinessProbe	**probe**	Pod 就绪度检查。
jvmOptions	**JvmOptions**	pod 的 JVM 选项。
jmxOptions	**KafkaJmxOptions**	Kafka 代理的 JMX 选项。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
metricsConfig	**JmxPrometheusExporterMetrics**	指标配置。
logging	**InlineLogging**, **ExternalLogging**	Kafka 的日志记录配置。
模板	**KafkaClusterTemplate**	Kafka 集群资源的模板。该模板允许用户指定如何生成 OpenShift 资源。
tieredStorage	**TieredStorageCustom**	为 Kafka 代理配置分层存储功能。

第 6 章 GENERICKAFKALISTENER 模式参考

使用于： KafkaClusterSpec

GenericKafkaListener 模式属性的完整列表

配置监听程序，以连接到 OpenShift 内部和外部的 Kafka 代理。

您可以在 Kafka 资源中配置监听程序。

显示监听程序配置的 Kafka 资源示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    #...
    listeners:
      - name: plain
        port: 9092
        type: internal
        tls: false
      - name: tls
        port: 9093
        type: internal
        tls: true
        authentication:
          type: tls
      - name: external1
        port: 9094
        type: route
        tls: true
      - name: external2
        port: 9095
        type: ingress
        tls: true
        authentication:
          type: tls
        configuration:
          bootstrap:
            host: bootstrap.myingress.com
          brokers:
          - broker: 0
```

```
      host: broker-0.myingress.com
    - broker: 1
      host: broker-1.myingress.com
    - broker: 2
      host: broker-2.myingress.com
  #...
```

6.1. 监听器

您可以使用 Kafka 资源中的 listeners 属性配置 Kafka 代理监听程序。监听器被定义为数组。

侦听器配置示例

```
listeners:
  - name: plain
    port: 9092
    type: internal
    tls: false
```

在 Kafka 集群中，名称和端口必须是唯一的。通过为每个监听程序指定唯一的名称和端口，您可以配置多个监听程序。名称长度最多可 25 个字符，由小写字母和数字组成。

6.2. PORT

端口号是 Kafka 集群中使用的端口，这可能不是客户端用来访问的端口。

- LoadBalancer 侦听程序使用指定的端口号，如 internal 和 cluster-ip 监听程序
- Ingress 和 路由 监听程序使用端口 443 访问
- NodePort 侦听程序使用 OpenShift 分配的端口号

对于客户端连接，将地址和端口用于监听器的 bootstrap 服务。您可以从 Kafka 资源的状态中检索它。

检索客户端连接的地址和端口的命令示例

```
oc get kafka <kafka_cluster_name> -o=jsonpath='{.status.listeners[?
(@.name=="<listener_name>")].bootstrapServers}{"\n"}'
```

重要

当为客户端配置监听程序对代理的访问时，您可以使用端口 9092 或更高版本(9093、9094 等)，但有一些例外。侦听程序无法配置为使用为 interbroker 通信(9090 和 9091)、Prometheus 指标(9404)和 JMX (Java 管理扩展)监控(9999)保留的端口。

6.3. TYPE

类型设置为 internal，或对于外部监听程序，设置为 route, loadbalancer, nodeport, ingress 或 cluster-ip。您还可以配置 cluster-ip 侦听器，这是可用于构建自定义访问机制的内部监听程序类型。

internal

您可以使用 tls 属性使用或不加密配置内部监听程序。

内部监听程序 配置示例

```
#...
spec:
  kafka:
    #...
    listeners:
      #...
      - name: plain
        port: 9092
        type: internal
        tls: false
      - name: tls
        port: 9093
        type: internal
```

```
      tls: true
      authentication:
        type: tls
    #...
```

route

配置外部监听程序，以使用 OpenShift Routes 和 HAProxy 路由器公开 Kafka。

为每个 Kafka 代理 pod 创建一个专用 Route。创建额外 Route 以作为 Kafka bootstrap 地址。Kafka 客户端可以使用这些 Routes 连接到端口 443 上的 Kafka。客户端通过端口 443 （默认路由器端口）连接，但流量会路由到您配置的端口，本例中为 9094。

路由 监听程序配置示例

```
#...
spec:
  kafka:
    #...
    listeners:
      #...
      - name: external1
        port: 9094
        type: route
        tls: true
    #...
```

ingress

配置外部监听程序，以使用 Kubernetes Ingress 和 Ingress NGINX Controller for Kubernetes 公开 Kafka。

为每个 Kafka 代理 pod 创建一个专用 Ingress 资源。创建额外的 Ingress 资源，以用作 Kafka bootstrap 地址。Kafka 客户端可以使用这些 Ingress 资源在端口 443 上连接到 Kafka。客户端通过端口 443 （默认控制器端口）连接，但流量会路由到您配置的端口，以下示例中为 9095。

您必须使用 GenericKafkaListenerConfigurationBootstrap 和 GenericKafkaListenerConfigurationBroker 属性指定 bootstrap 和 per-broker 服务使用的主机

名。

ingress 侦听器配置示例

```
#...
spec:
  kafka:
    #...
    listeners:
      #...
      - name: external2
        port: 9095
        type: ingress
        tls: true
        authentication:
          type: tls
        configuration:
          bootstrap:
            host: bootstrap.myingress.com
          brokers:
          - broker: 0
            host: broker-0.myingress.com
          - broker: 1
            host: broker-1.myingress.com
          - broker: 2
            host: broker-2.myingress.com
  #...
```

注意

使用 Ingress 的外部监听程序当前只通过 Ingress NGINX Controller for Kubernetes 测试。

LoadBalancer

配置外部监听程序，以使用 Loadbalancer 类型 Service 来公开 Kafka。

为每个 Kafka 代理 pod 创建一个新的 loadbalancer 服务。创建了一个额外的负载均衡器来充当 Kafka *bootstrap* 地址。loadBalancers 侦听指定的端口号，本例中为端口 9094。

您可以使用 loadBalancerSourceRanges 属性配置 源范围 来限制对指定 IP 地址的访问。

loadbalancer 侦听器配置示例

```
#...
spec:
  kafka:
    #...
    listeners:
      - name: external3
        port: 9094
        type: loadbalancer
        tls: true
        configuration:
          loadBalancerSourceRanges:
            - 10.0.0.0/8
            - 88.208.76.87/32
    #...
```

nodeport

配置外部监听程序，以使用 NodePort 类型 Service 来公开 Kafka。

Kafka 客户端直接连接到 OpenShift 的节点。创建额外的 NodePort 服务类型作为 Kafka bootstrap 地址。

为 Kafka 代理 pod 配置公告地址时，Apache Kafka 的 Streams 使用运行给定 pod 的节点的地址。您可以使用 preferredNodePortAddressType 属性将 检查的第一个地址类型配置为节点地址。

nodeport 侦听器配置示例

```
#...
spec:
  kafka:
    #...
    listeners:
      #...
      - name: external4
        port: 9095
        type: nodeport
```

```
    tls: false
    configuration:
      preferredNodePortAddressType: InternalDNS
  #...
```

注意

目前，在使用节点端口公开 Kafka 集群时不支持 TLS 主机名验证。

cluster-ip

配置内部监听程序，以使用每个代理的 ClusterIP 类型 Service 来公开 Kafka。

侦听器不使用无头服务及其 DNS 名称将流量路由到 Kafka 代理。在使用无头服务时，您可以使用这类监听程序公开 Kafka 集群。您可以将其与自定义访问机制一起使用，例如使用特定 Ingress 控制器或 OpenShift 网关 API 的机制。

为每个 Kafka 代理 pod 创建一个新的 ClusterIP 服务。该服务被分配一个 ClusterIP 地址，来作为带有每个代理端口号的 Kafka *bootstrap* 地址。例如，您可以将监听程序配置为通过带有 TCP 端口配置的 Nginx Ingress Controller 公开 Kafka 集群。

cluster-ip 侦听器配置示例

```
#...
spec:
  kafka:
    #...
    listeners:
      - name: clusterip
        type: cluster-ip
        tls: false
        port: 9096
    #...
```

6.4. TLS

TLS 属性是必需的。

要启用 TLS 加密，请将 tls 属性设置为 true。对于 route 和 ingress 类型监听程序，必须始终启用 TLS 加密。

6.5. 身份验证

监听器的身份验证可指定为：

- mTLS (tls)
- SCRAM-SHA-512 (scram-sha-512)
- 基于令牌的 OAuth 2.0 (oauth)
- Custom (自定义)

6.6. NETWORKPOLICYPEERS

使用 networkPolicyPeers 来配置网络策略，以限制对网络级别的监听程序的访问。以下示例显示了普通 和 tls 侦听器的 networkPolicyPeers 配置。

在以下示例中：

- 只有与标签 app: kafka-sasl-consumer 和 app: kafka-sasl-producer 匹配的应用程序 pod 可以连接到 纯文本。应用程序 pod 必须与 Kafka 代理在同一命名空间中运行。
- 只有在与标签 project: myproject and project: myproject2 的命名空间中运行的应用程序 pod 才可以连接到 tls 侦听程序。

networkPolicyPeers 属性的语法与 NetworkPolicy 资源中的 from 属性相同。

网络策略配置示例

```
listeners:
  #...
  - name: plain
    port: 9092
    type: internal
    tls: true
    authentication:
      type: scram-sha-512
    networkPolicyPeers:
      - podSelector:
          matchLabels:
            app: kafka-sasl-consumer
      - podSelector:
          matchLabels:
            app: kafka-sasl-producer
  - name: tls
    port: 9093
    type: internal
    tls: true
    authentication:
      type: tls
    networkPolicyPeers:
      - namespaceSelector:
          matchLabels:
            project: myproject
      - namespaceSelector:
          matchLabels:
            project: myproject2
# ...
```

6.7. GENERICKAFKALISTENER 模式属性

属性	属性类型	描述
name	string	侦听器的名称。名称将用于识别侦听器和相关 OpenShift 对象。该名称必须在给定的 Kafka 集群中唯一。名称可由小写字符和数字组成，最多为 11 个字符。

属性	属性类型	描述
port	整数	Kafka 中监听器使用的端口号。端口号必须在给定的 Kafka 集群中唯一。允许端口号为 9092 及更高，但端口 9404 和 9999 除外，它们已经用于 Prometheus 和 JMX。根据监听器类型，端口号可能与连接 Kafka 客户端的端口号不同。
type	字符串([ingress, internal, route, loadbalancer, cluster-ip, nodeport] 之一)	侦听器的类型。支持的类型如下： • **内部** 类型仅在 OpenShift 集群中内部公开 Kafka。 • **路由类型** 使用 OpenShift Routes 来公开 Kafka。 • **LoadBalancer** 类型使用 LoadBalancer 类型服务来公开 Kafka。 • **NodePort** 类型使用 NodePort 类型服务来公开 Kafka。 • **ingress** 类型使用 OpenShift Nginx Ingress 来公开带有 TLS 透传的 Kafka。 • **cluster-ip** 类型使用每个代理的 **ClusterIP** 服务。
tls	布尔值	在监听器上启用 TLS 加密。这是必需属性。
身份验证	**KafkaListenerAuthenticationTls**, **KafkaListenerAuthenticationScramSha512**, **KafkaListenerAuthenticationOAuth**, **KafkaListenerAuthenticationCustom**	此侦听器的身份验证配置。
配置	**GenericKafkaListenerConfiguration**	其他监听程序配置。
networkPolicyPeers	NetworkPolicyPeer 数组	应该可以连接到此监听器的对等点列表。此列表中的对等点通过逻辑 OR 操作进行合并。如果此字段为空或缺失，则允许此监听器的所有连接。如果此字段存在并至少包含一个项目，则监听器仅允许与此列表中至少匹配某一项的流量。

第 7 章 KAFKALISTENERAUTHENTICATIONTLS 模式参考

使用 in: GenericKafkaListener

type 属性是一个辨别器，可区分来自 KafkaListenerAuthenticationScramSha512, KafkaListenerAuthenticationOAuth, KafkaListenerAuthenticationCustom 的 KafkaListenerAuthenticationTls 类型的使用。对于类型 KafkaListenerAuthenticationTls，它需要是值 tls。

属性	属性类型	描述
type	string	必须为 **tls**。

第 8 章 KAFKALISTENERAUTHENTICATIONSCRAMSHA512 SCHEMA REFERENCE

使用 in: GenericKafkaListener

type 属性是一种差异性，用于区分来自 KafkaListenerAuthenticationTls, KafkaListenerAuthenticationOAuth, KafkaListenerAuthenticationCustom 的 KafkaListenerAuthenticationScramSha512 类型。对于类型 KafkaListenerAuthenticationScramSha512，它需要值 scram-sha-512。

属性	属性类型	描述
type	string	必须是 **scram-sha-512**。

第 9 章 KAFKALISTENERAUTHENTICATIONOAUTH 模式参考

使用 in: GenericKafkaListener

type 属性是一种辨别器，它会区分来自 KafkaListenerAuthenticationTls, KafkaListenerAuthenticationScramSha512, KafkaListenerAuthenticationCustom 的 KafkaListenerAuthenticationOAuth 类型的使用。对于类型 KafkaListenerAuthenticationOAuth，它需要带有值 oauth。

属性	属性类型	描述
accessTokenIsJwt	布尔值	配置访问令牌是否被视为 JWT。如果授权服务器返回不透明令牌，则必须将其设置为 **false**。默认值为 **true**。
checkAccessTokenType	布尔值	配置是否执行访问令牌类型检查。如果授权服务器没有在 JWT 令牌中包含"typ"声明，则这应设为 **false**。默认值为 **true**。
checkAudience	布尔值	启用或禁用受众检查。受众检查可识别令牌的接收者。如果启用了 audience 检查，还必须使用 **clientId** 属性配置 OAuth 客户端 ID。Kafka 代理将拒绝在其 **aud** (audience) 声明中没有其 **clientId** 的令牌。默认值为 **false**。
checkIssuer	布尔值	启用或禁用签发者检查。默认情况下，使用由 **validIssuerUri** 配置的值来检查签发者。默认值为 **true**。
clientAudience	string	向授权服务器令牌端点发出请求时使用的受众。用于 Inter-broker 身份验证，并使用 **clientId** 和 **secret** 方法配置 OAuth 2.0。
clientId	string	Kafka 代理可用于对授权服务器进行身份验证的 OAuth 客户端 ID，并使用 introspection 端点 URI。
clientScope	string	向授权服务器令牌端点发出请求时使用的范围。用于 Inter-broker 身份验证，并使用 **clientId** 和 **secret** 方法配置 OAuth 2.0。
clientSecret	GenericSecretSource	链接到包含 Kafka 代理可用于向授权服务器进行身份验证的 OAuth 客户端 secret 的 OpenShift Secret，并使用 introspection 端点 URI。

属性	属性类型	描述
connectTimeoutSeconds	整数	连接到授权服务器时的连接超时（以秒为单位）。如果没有设置，则有效的连接超时为 60 秒。
customClaimCheck	string	要应用到 JWT 令牌或内省端点的响应以进行额外令牌验证的 jsonpath 过滤器查询。默认不设置。
disableTlsHostnameVerification	布尔值	启用或禁用 TLS 主机名验证。默认值为 **false**。
enableECDSA	布尔值	**enableECDSA 属性已弃用。**通过安装 BouncyCastle crypto provider 启用或禁用 ECDSA 支持。ECDSA 支持总是被启用。BouncyCastle 库不再与 Apache Kafka 的 Streams 一起打包。值将被忽略。
enableMetrics	布尔值	启用或禁用 OAuth 指标。默认值为 **false**。
enableOauthBearer	布尔值	启用或禁用 SASL_OAUTHBEARER 的 OAuth 身份验证。默认值为 **true**。
enablePlain	布尔值	启用或禁用 SASL_PLAIN 的 OAuth 身份验证。当使用此机制时，没有重新身份验证的支持。默认值为 **false**。
failFast	布尔值	启用或禁用 Kafka 代理进程的终止，因为启动过程中可能会恢复的运行时错误。默认值为 **true**。
fallbackUserNameClaim	string	如果 **userNameClaim** 指定的声明不存在，用于用户 id 的回退用户名声明。这仅在 **client_credentials** 身份验证生成在另一个声明中提供客户端 ID 时很有用。只有在设置了 **userNameClaim** 时，它才会生效。
fallbackUserNamePrefix	string	用于值 **fallbackUserNameClaim** 的前缀，以构造用户 id。这只有在 **fallbackUserNameClaim** 为 true 且声明存在值时才生效。将 username 和 client id 映射到同一用户 ID 空间对于防止名称冲突非常有用。
groupsClaim	string	用于在身份验证期间提取用户的组的 jsonpath 查询。提取的组可由自定义授权器使用。默认情况下，不会提取任何组。

属性	属性类型	描述
groupsClaimDelimiter	string	当以单个 String 值而不是 JSON 数组中提取时，用于解析组的分隔符。默认值为 ','(comma)。
httpRetries	整数	初始 HTTP 请求失败时尝试的最大重试次数。如果没有设置，则默认为不尝试任何重试。
httpRetryPauseMs	整数	重试失败的 HTTP 请求前需要暂停。如果没有设置，则默认为根本不暂停，而是立即重复请求。
includeAcceptHeader	布尔值	Accept 标头是否应该在请求中设置到授权服务器。默认值为 **true**。
introspectionEndpointUri	string	令牌内省端点的 URI，可用于验证不透明的非 JWT 令牌。
jwksEndpointUri	string	JWKS 证书端点的 URI，可用于本地 JWT 验证。
jwksExpirySeconds	整数	配置 JWKS 证书被视为有效的频率。到期间隔必须至少为 60 秒，然后在 **jwksRefreshSeconds** 中指定刷新间隔。默认为 360 秒。
jwksIgnoreKeyUse	布尔值	用于忽略 JWKS 端点响应中 **密钥** 声明的 'use' 属性。默认值为 **false**。
jwksMinRefreshPauseSeconds	整数	连续两个刷新之间的最小暂停。当遇到未知签名密钥时，将立即调度刷新，但总是等待这个最小暂停。默认值为 1 秒。
jwksRefreshSeconds	整数	配置刷新的 JWKS 证书的频率。刷新间隔必须至少为 60 秒，然后是 **jwksExpirySeconds** 中指定的到期间隔。默认值为 300 秒。
maxSecondsWithoutReauthentication	整数	在不重新身份验证的情况下，经过身份验证的会话保持有效的最大秒数。这可让 Apache Kafka 重新身份验证功能，并在访问令牌过期时导致会话过期。如果访问令牌在最大时间前过期，或者达到最大时间，客户端必须重新验证，否则服务器将丢弃连接。默认情况下不设置 - 经过身份验证的会话不会在访问令牌过期时过期。这个选项只适用于 SASL_OAUTHBEARER 身份验证机制（当 **enableOauthBearer** 为 **true**时）。

属性	属性类型	描述
readTimeoutSeconds	整数	连接到授权服务器时读取超时（以秒为单位）。如果没有设置，则有效读取超时为 60 秒。
tlsTrustedCertificates	**CertSecretSource** 数组	用于 TLS 连接到 OAuth 服务器的可信证书。
tokenEndpointUri	string	当客户端使用 **clientId** 和 **secret** 进行身份验证时，用于 SASL_PLAIN 机制的令牌端点的 URI。如果设置，客户端可以通过 SASL_PLAIN 进行身份验证，方法是将 **username** 设置为 **clientId**，并将 **password** 设置为客户端的 **secret**，或者将 **username** 设置为账户用户名，将 **password** 设置为带有 **$accessToken:** 前缀的访问令牌。如果没有设置这个选项，**密码** 总是被解释为访问令牌（没有前缀），**用户名作为帐户用户名**（称为 'no-client-credentials' 模式）。
type	string	必须是 **oauth**。
userInfoEndpointUri	string	当 Introspection 端点没有返回可用于用户 ID 的信息时，用作回退的 User Info Endpoint 的 URI，以获取用户 id。
userNameClaim	string	来自 JWT 身份验证令牌、Introspection Endpoint 响应或用户信息端点响应的声明名称，用于提取用户 ID。默认为 **sub**。
validIssuerUri	string	用于身份验证的令牌签发者的 URI。
validTokenType	string	Introspection Endpoint 返回的 **token_type** 属性的有效值。没有默认值，默认情况下不检查。

第 10 章 GENERICSECRETSOURCE 模式参考

用于： KafkaClientAuthenticationOAuth, KafkaListenerAuthenticationCustom, KafkaListenerAuthenticationOAuth

属性	属性类型	描述
key	string	secret 值存储在 OpenShift Secret 中的键。
secretName	string	包含 secret 值的 OpenShift Secret 名称。

第 11 章 CERTSECRETSOURCE 模式参考

用于： ClientTls, KafkaAuthorizationKeycloak, KafkaAuthorizationOpa, KafkaClientAuthenticationOAuth, KafkaListenerAuthenticationOAuth

属性	属性类型	描述
certificate	string	Secret 中文件证书的名称。
secretName	string	包含证书的 Secret 名称。

第 12 章 KAFKALISTENERAUTHENTICATIONCUSTOM SCHEMA REFERENCE

使用 in: GenericKafkaListener

KafkaListenerAuthenticationCustom schema 属性的完整列表

要配置自定义身份验证，请将 type 属性设置为 custom。

自定义身份验证允许使用任何类型的 Kafka 支持的身份验证。

自定义 OAuth 身份验证配置示例

```
spec:
  kafka:
    config:
      principal.builder.class: SimplePrincipal.class
    listeners:
      - name: oauth-bespoke
        port: 9093
        type: internal
        tls: true
        authentication:
          type: custom
          sasl: true
          listenerConfig:
            oauthbearer.sasl.client.callback.handler.class: client.class
            oauthbearer.sasl.server.callback.handler.class: server.class
            oauthbearer.sasl.login.callback.handler.class: login.class
            oauthbearer.connections.max.reauth.ms: 999999999
            sasl.enabled.mechanisms: oauthbearer
            oauthbearer.sasl.jaas.config: |
              org.apache.kafka.common.security.oauthbearer.OAuthBearerLoginModule required ;
          secrets:
            - name: example
```

生成协议映射，它使用 sasl 和 tls 值来确定要映射到监听程序的协议。

- sasl = True, TLS = True → SASL_SSL
- sasl = False, TLS = True → SSL
- sasl = True, TLS = False → SASL_PLAINTEXT
- sasl = False, TLS = False → PLAINTEXT

12.1. LISTENERCONFIG

使用 listenerConfig 指定的监听程序配置带有 listener.name . *<listener_name>-<port>* 前缀。例如，sasl.enabled.mechanisms 变为 listener.name. *<listener_name>-<port>*.sasl.enabled.mechanisms。

12.2. SECRETS

secret 被挂载到 Kafka 代理节点的 容器中的 /opt/kafka/custom-authn-secrets/custom-listener-*<listener_name>-<port>* / *<secret_name>* 中。

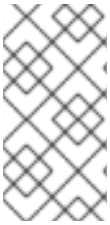
例如，示例配置中挂载的 secret (示例)将位于 /opt/kafka/custom-authn-secrets/custom-listener-oauth-bespoke-9093/example。

12.3. 主体构建器

您可以在 Kafka 集群配置中设置自定义主体构建器。但是，主体构建器有以下要求：

- 镜像上必须存在指定的主体构建器类。在自己构建*之前*，请检查是否已存在。您需要使用所需类重建 Apache Kafka 镜像的流。
- 没有其他监听程序使用 oauth 类型身份验证。这是因为 OAuth 侦听器将自己的原则构建器附加到 Kafka 配置中。
- 指定主体构建器与 Apache Kafka 的 Streams 兼容。

自定义主体构建器必须支持进行身份验证的对等证书，因为 Apache Kafka 的 Streams 使用它们来管理 Kafka 集群。



注意

Kafka 的默认主体构建器类 支持根据对等证书的名称构建主体。自定义主体构建器应使用 SSL peer 证书的名称提供类型为 user 的主体。

以下示例显示了一个自定义主体构建器，它满足 Apache Kafka 的 Streams 的 OAuth 要求。

自定义 OAuth 配置的主体构建器示例

```
public final class CustomKafkaPrincipalBuilder implements KafkaPrincipalBuilder {

    public KafkaPrincipalBuilder() {}

    @Override
    public KafkaPrincipal build(AuthenticationContext context) {
        if (context instanceof SslAuthenticationContext) {
            SSLSession sslSession = ((SslAuthenticationContext) context).session();
            try {
                return new KafkaPrincipal(
                    KafkaPrincipal.USER_TYPE, sslSession.getPeerPrincipal().getName());
            } catch (SSLPeerUnverifiedException e) {
                throw new IllegalArgumentException("Cannot use an unverified peer for
authentication", e);
            }
        }

        // Create your own KafkaPrincipal here
        ...
    }
}
```

12.4. KAFKALISTENERAUTHENTICATIONCUSTOM 模式属性

type 属性是一种辨别器，它会区分来自 KafkaListenerAuthenticationTls, KafkaListenerAuthenticationScramSha512, KafkaListenerAuthenticationOAuth 的

KafkaListenerAuthenticationCustom 类型的使用。对于类型 KafkaListenerAuthenticationCustom，它需要有值 custom。

属性	属性类型	描述
listenerConfig	map	用于特定监听程序的配置。所有值都以 listener.name. <*listener_name* > 前缀。
sasl	布尔值	在此监听器上启用或禁用 SASL。
secrets	**GenericSecretSource** 数组	要挂载到 /opt/kafka/custom-authn-secrets/custom-listener- <*listener_name*>-<*port*>/ <*secret_name* > 的 secret。
type	string	必须是 **custom**。

第 13 章 GENERICKAFKALISTENERCONFIGURATION 模式参考

使用 in: GenericKafkaListener

GenericKafkaListenerConfiguration 模式属性的完整列表

配置 Kafka 侦听程序。

13.1. BROKERCERTCHAINANDKEY

brokerCertChainAndKey 属性仅用于启用了 TLS 加密的监听程序。您可以使用属性来提供自己的 Kafka 侦听器证书。

启用 TLS 加密的 loadbalancer 外部监听程序配置示例

```
listeners:
  #...
  - name: external3
    port: 9094
    type: loadbalancer
    tls: true
    authentication:
      type: tls
    configuration:
      brokerCertChainAndKey:
        secretName: my-secret
        certificate: my-listener-certificate.crt
        key: my-listener-key.key
# ...
```

当 brokerCertChainAndKey secret 中的证书或密钥被更新时，Operator 会在下一个协调中自动检测它，并触发 Kafka 代理的滚动更新来重新载入证书。

13.2. EXTERNALTRAFFICPOLICY

externalTrafficPolicy 属性用于 loadbalancer 和 nodeport 侦听程序。在 OpenShift 外部公开 Kafka

时，您可以选择 Local 或 Cluster。Local 避免跃点到其他节点，并保留客户端 IP，而 Cluster 没有这些行为。默认值为 Cluster。

13.3. LOADBALANCERSOURCERANGES

loadBalancerSourceRanges 属性仅用于 loadbalancer 侦听程序。在 OpenShift 外部公开 Kafka 时，除了标签和注解外，还使用源范围来自定义服务创建方式。

为 loadbalancer 侦听器配置的源范围示例

```
listeners:
  #...
  - name: external3
    port: 9094
    type: loadbalancer
    tls: false
    configuration:
      externalTrafficPolicy: Local
      loadBalancerSourceRanges:
        - 10.0.0.0/8
        - 88.208.76.87/32
      # ...
# ...
```

13.4. CLASS

class 属性仅适用于 ingress 监听程序。您可以使用 class 属性配置 Ingress 类。

使用 Ingress 类 nginx-internal 的类型为 ingress 的外部监听程序示例

```
listeners:
  #...
  - name: external2
    port: 9094
    type: ingress
    tls: true
    configuration:
      class: nginx-internal
    # ...
# ...
```

-

13.5. PREFERREDNODEPORTADDRESSTYPE

preferredNodePortAddressType 属性仅用于 nodeport 监听程序。

使用监听器配置中的 preferredNodePortAddressType 属性来指定检查的第一个地址类型作为节点地址。例如，如果您的部署没有 DNS 支持，或者您只想通过内部 DNS 或 IP 地址在内部公开代理，则此属性很有用。如果找到了此类型的地址，则会使用它。如果没有找到首选地址类型，则 Apache Kafka 的 Streams 按照标准优先级顺序通过类型：

1. **ExternalDNS**
2. **ExternalIP**
3. **Hostname**
4. **InternalDNS**
5. **InternalIP**

使用首选节点端口地址类型配置的外部监听程序示例

```
listeners:
  #...
  - name: external4
    port: 9094
    type: nodeport
    tls: false
    configuration:
      preferredNodePortAddressType: InternalDNS
      # ...
# ...
```

13.6. USESERVICEDNSDOMAIN

useServiceDnsDomain 属性仅用于 internal 和 cluster-ip 侦听程序。它定义是否使用包含集群服务后缀（通常为 .cluster.local）的完全限定 DNS 名称。使用 useServiceDnsDomain 设置为 false 时，公告的地址在没有服务后缀的情况下生成；例如，my-cluster-kafka-0.my-cluster-kafka-brokers.myproject.svc。使用 useServiceDnsDomain 设置为 true 时，公告的地址使用服务后缀生成；例如，my-cluster-kafka-0.my-cluster-kafka-brokers.myproject.svc.cluster.local。默认为 false。

被配置为使用 Service DNS 域的内部监听程序示例

```
listeners:
  #...
  - name: plain
    port: 9092
    type: internal
    tls: false
    configuration:
      useServiceDnsDomain: true
      # ...
# ...
```

如果您的 OpenShift 集群使用与 .cluster.local 不同的服务后缀，您可以使用 Cluster Operator 配置中的 KUBERNETES_SERVICE_DNS_DOMAIN 环境变量配置后缀。

13.7. GENERICKAFKALISTENERCONFIGURATION 模式属性

属性	属性类型	描述
brokerCertChainAndKey	CertAndKeySecretSource	引用保存用于此监听器的证书和私钥对的 Secret。证书可以选择包含整个链。此字段只能与启用了 TLS 加密的监听程序一起使用。

属性	属性类型	描述
externalTrafficPolicy	字符串([本地、集群] 之一)	指定服务是否将外部流量路由到节点本地端点还是集群范围的端点。**集群** 可能会导致第二个跃点切换到另一节点，并模糊处理客户端源 IP。**local** 避免了 LoadBalancer 和 Nodeport 类型服务的第二个跃点，并保留客户端源 IP （当基础架构支持时）。如果未指定，OpenShift 将使用 **Cluster** 作为默认值。此字段只能用于 **loadbalancer** 或 **nodeport** 类型监听程序。
loadBalancerSourceRanges	字符串数组	客户端可以从中连接到负载均衡器类型的 CIDR 范围列表（如 **10.0.0.0/8** 或 **130.211.204.1/32**）。如果平台支持，则通过 loadbalancer 的流量将被限制为指定的 CIDR 范围。此字段仅适用于 loadbalancer 类型服务，如果云供应商不支持该功能，则忽略此字段。此字段只能用于 **loadbalancer** 类型监听程序。
bootstrap	**GenericKafkaListenerConfigurationBootstrap**	Bootstrap 配置。
代理(Broker)	**GenericKafkaListenerConfigurationBroker** 数组	每个代理配置。
ipFamilyPolicy	字符串（一个 [RequireDualStack, SingleStack, PreferDualStack]）	指定服务使用的 IP 系列策略。可用选项包括 **SingleStack**、**PreferDualStack** 和 **RequireDualStack**。**SingleStack** 用于单个 IP 系列。**PreferDualStack** 用于两个 IP 系列，用于双栈集群配置的集群或单堆栈集群上的单个 IP 系列。**RequireDualStack** 失败，除非双栈配置的集群中有两个 IP 系列。如果未指定，OpenShift 将根据服务类型选择默认值。
ipFamilies	字符串（一个或多个 [IPv6, IPv4]）数组	指定服务使用的 IP Families。可用选项包括 **IPv4** 和 **IPv6**。如果未指定，OpenShift 将根据 **ipFamilyPolicy** 设置选择默认值。
createBootstrapService	布尔值	是否创建 bootstrap 服务。默认情况下创建 bootstrap 服务（如果未指定），则默认创建。此字段可用于 **loadBalancer** 类型监听程序。

属性	属性类型	描述
class	string	为 **Ingress** 和 **LoadBalancer** 配置一个特定的类，用于定义将使用哪个控制器。此字段只能用于 **ingress** 和 **loadbalancer** 类型监听程序。如果没有指定，则使用默认控制器。对于 **ingress** 侦听程序，在 **Ingress** 资源中设置 **ingressClassName** 属性。对于 **loadbalancer** 侦听器，请在 **Service** 资源中设置 **loadBalancerClass** 属性。
finalizers	字符串数组	为此监听器创建的 **LoadBalancer** 类型服务配置的终结器列表。如果平台支持，finalizer **service.kubernetes.io/load-balancer-cleanup** 以确保外部负载均衡器与 service.For more 一起删除，请参阅 https://kubernetes.io/docs/tasks/access-application-cluster/create-external-load-balancer/#garbage-collecting-load-balancers。此字段只能用于 **loadbalancer** 类型监听程序。
maxConnectionCreationRate	整数	我们随时允许该监听器的最大连接创建率。如果达到限制，则将节流新的连接。
maxConnections	整数	在代理中，我们允许这个监听程序的最大连接数。如果达到限制，则阻止新的连接。
preferredNodePortAddressType	字符串([ExternalDNS, ExternalIP, Hostname, InternalIP, InternalDNS] 之一)	定义应将哪些地址类型用作节点地址。可用的类型是：**ExternalDNS**, **ExternalIP**, **InternalDNS**, **InternalIP** 和 **Hostname**。默认情况下，地址将按以下顺序使用（将使用第一个地址）： • **ExternalDNS** • **ExternalIP** • **InternalDNS** • **InternalIP** • **Hostname** 此字段用于选择首选地址类型，该类型会首先检查。如果没有找到这个地址类型的地址，则默认检查其他类型。此字段只能与 **nodeport** 类型监听程序一起使用。

属性	属性类型	描述
useServiceDnsDomain	布尔值	配置是否应使用 OpenShift 服务 DNS 域。如果设置为 **true**，则生成的地址将包含服务 DNS 域后缀（默认为 **.cluster.local**，可以使用环境变量 **KUBERNETES_SERVICE_DNS_DOMAIN**）进行配置。默认为 **false**。此字段只能用于 **internal** 和 **cluster-ip** 类型监听程序。

第 14 章 CERTANDKEYSECRETSOURCE 模式参考

用于： GenericKafkaListenerConfiguration, KafkaClientAuthenticationTls

属性	属性类型	描述
certificate	string	Secret 中文件证书的名称。
key	string	Secret 中私钥的名称。
secretName	string	包含证书的 Secret 名称。

第 15 章 GENERICKAFKALISTENERCONFIGURATIONBOOTSTRAP 模式参考

used in: GenericKafkaListenerConfiguration

GenericKafkaListenerConfigurationBootstrap 模式属性的完整列表

与 nodePort, host, loadBalancerIP 和 annotations 属性相应的代理服务在 GenericKafkaListenerConfigurationBroker schema 中配置。

15.1. ALTERNATIVENAMES

您可以为 bootstrap 服务指定替代名称。名称添加到代理证书中，可用于 TLS 主机名验证。alternativeNames 属性适用于所有类型的监听程序。

使用额外 bootstrap 地址配置 的外部路由 监听程序示例

```
listeners:
  #...
  - name: external1
    port: 9094
    type: route
    tls: true
    authentication:
      type: tls
    configuration:
      bootstrap:
        alternativeNames:
          - example.hostname1
          - example.hostname2
# ...
```

15.2. 主机

host 属性用于 route 和 ingress 监听程序，以指定 bootstrap 和 per-broker 服务使用的主机名。

对于 ingress 监听程序，host 属性值是必需的，因为 Ingress 控制器不会自动分配任何主机名。确保

主机名解析为 Ingress 端点。Apache Kafka 的流不会执行任何请求的主机可用的验证，并正确路由到 Ingress 端点。

ingress 侦听程序的主机配置示例

```
listeners:
  #...
  - name: external2
    port: 9094
    type: ingress
    tls: true
    authentication:
      type: tls
    configuration:
      bootstrap:
        host: bootstrap.myingress.com
      brokers:
      - broker: 0
        host: broker-0.myingress.com
      - broker: 1
        host: broker-1.myingress.com
      - broker: 2
        host: broker-2.myingress.com
# ...
```

默认情况下，路由 监听程序主机由 OpenShift 自动分配。但是，您可以通过指定主机来覆盖分配的路由主机。

Apache Kafka 的流不会执行任何请求主机的验证可用。您必须确保它们可用且可以使用。

路由监听程序的主机配置示例

```
# ...
listeners:
  #...
  - name: external1
    port: 9094
    type: route
    tls: true
    authentication:
      type: tls
```

```
    configuration:
      bootstrap:
        host: bootstrap.myrouter.com
      brokers:
      - broker: 0
        host: broker-0.myrouter.com
      - broker: 1
        host: broker-1.myrouter.com
      - broker: 2
        host: broker-2.myrouter.com
# ...
```

15.3. NODEPORT

默认情况下，OpenShift 会自动分配用于 bootstrap 和 broker 服务的端口号。您可以通过指定请求的端口号来覆盖为 nodeport 侦听程序分配的节点端口。

Apache Kafka 的流不会在请求的端口上执行任何验证。您必须确保它们可用且可供使用。

配置有覆盖节点端口的外部监听程序示例

```
# ...
listeners:
  #...
  - name: external4
    port: 9094
    type: nodeport
    tls: true
    authentication:
      type: tls
    configuration:
      bootstrap:
        nodePort: 32100
      brokers:
      - broker: 0
        nodePort: 32000
      - broker: 1
        nodePort: 32001
      - broker: 2
        nodePort: 32002
# ...
```

15.4. LOADBALANCERIP

使用 loadBalancerIP 属性在创建 loadbalancer 时请求特定的 IP 地址。当您需要使用带有特定 IP 地址的 loadbalancer 时，请使用此属性。如果云供应商不支持该功能，则忽略 loadBalancerIP 字段。

带有特定负载均衡器 IP 地址请求的、类型为 loadbalancer 的外部监听程序示例

```
# ...
listeners:
  #...
  - name: external3
    port: 9094
    type: loadbalancer
    tls: true
    authentication:
      type: tls
    configuration:
      bootstrap:
        loadBalancerIP: 172.29.3.10
      brokers:
      - broker: 0
        loadBalancerIP: 172.29.3.1
      - broker: 1
        loadBalancerIP: 172.29.3.2
      - broker: 2
        loadBalancerIP: 172.29.3.3
# ...
```

15.5. ANNOTATIONS

使用 annotations 属性向与监听器相关的 OpenShift 资源添加注解。例如，您可以使用这些注解来检测 DNS 工具，如 外部 DNS，它们会自动为 loadbalancer 服务分配 DNS 名称。

类型为 loadbalancer 的使用 annotations 的一个外部监听程序的示例。

```
# ...
listeners:
```

```
    #...
    - name: external3
      port: 9094
      type: loadbalancer
      tls: true
      authentication:
        type: tls
      configuration:
        bootstrap:
          annotations:
            external-dns.alpha.kubernetes.io/hostname: kafka-bootstrap.mydomain.com.
            external-dns.alpha.kubernetes.io/ttl: "60"
        brokers:
        - broker: 0
          annotations:
            external-dns.alpha.kubernetes.io/hostname: kafka-broker-0.mydomain.com.
            external-dns.alpha.kubernetes.io/ttl: "60"
        - broker: 1
          annotations:
            external-dns.alpha.kubernetes.io/hostname: kafka-broker-1.mydomain.com.
            external-dns.alpha.kubernetes.io/ttl: "60"
        - broker: 2
          annotations:
            external-dns.alpha.kubernetes.io/hostname: kafka-broker-2.mydomain.com.
            external-dns.alpha.kubernetes.io/ttl: "60"
    # ...
```

15.6. GENERICKAFKALISTENERCONFIGURATIONBOOTSTRAP SCHEMA PROPERTIES

属性	属性类型	描述
alternativeNames	字符串数组	bootstrap 服务的其他替代名称。备用名称将添加到 TLS 证书的主题备用名称列表中。
主机	string	bootstrap 主机。此字段将在 Ingress 资源或 Route 资源中使用，以指定所需主机名。此字段只能用于 **路由** （可选）或 **ingress** （必需）类型监听程序。
nodePort	整数	bootstrap 服务的节点端口。此字段只能用于 **nodeport** 类型监听程序。
loadBalancerIP	string	使用此字段中指定的 IP 地址请求 loadbalancer。此功能取决于底层云供应商是否支持在创建负载均衡器时指定 **loadBalancerIP**。如果云供应商不支持这个功能，则此字段会被忽略。此字段只能用于 **loadbalancer** 类型监听程序。

属性	属性类型	描述
annotations	map	将添加到 **Ingress**、**Route** 或 **Service** 资源的注解。您可以使用此字段配置 DNS 供应商，如外部 DNS。此字段只能用于 **loadbalancer**、**nodeport**、**route** 或 **ingress** 类型监听程序。
labels	map	将添加到 **Ingress**、**Route** 或 **Service** 资源的标签。此字段只能用于 **loadbalancer**、**nodeport**、**route** 或 **ingress** 类型监听程序。

第 16 章 GENERICKAFKALISTENERCONFIGURATIONBROKER 模式参考

used in: GenericKafkaListenerConfiguration

GenericKafkaListenerConfigurationBroker 模式属性的完整列表

您可以在 GenericKafkaListenerConfigurationBootstrap schema 中看到 nodePort,host,loadBalancerIP 和 annotations 属性的示例配置，它配置 bootstrap 服务覆盖。

代理的公告地址

默认情况下，Apache Kafka 的 Streams 会尝试自动决定 Kafka 集群向其客户端公告的主机名和端口。在所有情况下都不够，因为运行 Apache Kafka 的基础架构可能无法提供可通过其访问 Kafka 的正确主机名或端口。

您可以指定代理 ID，并在监听程序的配置属性中自定义公告的主机名和端口。然后，Apache Kafka 的 Streams 会在 Kafka 代理中自动配置公告的地址，并将其添加到代理证书中，以便它可用于 TLS 主机名验证。覆盖公告的主机和端口适用于所有类型的监听程序。

配置 的外部路由监听程序的示例，带有覆盖公告的地址的覆盖

```
listeners:
  #...
  - name: external1
    port: 9094
    type: route
    tls: true
    authentication:
      type: tls
    configuration:
      brokers:
      - broker: 0
        advertisedHost: example.hostname.0
        advertisedPort: 12340
      - broker: 1
        advertisedHost: example.hostname.1
        advertisedPort: 12341
      - broker: 2
        advertisedHost: example.hostname.2
        advertisedPort: 12342
# ...
```

16.1. GENERICKAFKALISTENERCONFIGURATIONBROKER SCHEMA 属性

属性	属性类型	描述
broker	整数	kafka 代理的 ID （代理标识符）。代理 ID 从 0 开始，并与代理副本数对应。
advertisedHost	string	代理的 **advertised.listeners** 中使用的主机名。
advertisedPort	整数	代理的 **advertised.listeners** 中使用的端口号。
主机	string	代理主机。此字段将在 Ingress 资源或 Route 资源中使用，以指定所需主机名。此字段只能用于 **路由** （可选）或 **ingress** （必需）类型监听程序。
nodePort	整数	per-broker 服务的节点端口。此字段只能用于 **nodeport** 类型监听程序。
loadBalancerIP	string	使用此字段中指定的 IP 地址请求 loadbalancer。此功能取决于底层云供应商是否支持在创建负载均衡器时指定 **loadBalancerIP**。如果云供应商不支持这个功能，则此字段会被忽略。此字段只能用于 **loadbalancer** 类型监听程序。
annotations	map	将添加到 **Ingress** 或 **Service** 资源的注解。您可以使用此字段配置 DNS 供应商，如外部 DNS。此字段只能用于 **loadbalancer**、**nodeport** 或 **ingress** 类型监听程序。
labels	map	将添加到 **Ingress**、**Route** 或 **Service** 资源的标签。此字段只能用于 **loadbalancer**、**nodeport**、**route** 或 **ingress** 类型监听程序。

第 17 章 EPHEMERALSTORAGE SCHEMA 参考

用于：Jbo dStorage, KafkaClusterSpec, KafkaNodePoolSpec, ZookeeperClusterSpec

type 属性是一个辨别器，可区分来自 PersistentClaimStorage 的 EphemeralStorage 类型。对于类型 EphemeralStorage，值需要是 ephemeral。

属性	属性类型	描述
id	整数	存储标识号。只适用于在类型为"jbod"存储中定义的存储卷。
sizeLimit	string	当 type=ephemeral 时，定义此 EmptyDir 卷所需的本地存储总量（例如 1Gi）。
type	string	必须为 **临时**。

第 18 章 PERSISTENTCLAIMSTORAGE 模式参考

用于：Jbo dStorage, KafkaClusterSpec, KafkaNodePoolSpec, ZookeeperClusterSpec

type 属性是一个辨别器，可区分来自 EphemeralStorage 的 PersistentClaimStorage 类型。它必须具有类型为 PersistentClaimStorage 的 persistent-claim 值。

属性	属性类型	描述
type	string	必须是 **persistent-claim**。
size	string	当 **type=persistent-claim** 时，定义持久性卷声明的大小，如 100Gi。在 **type=persistent-claim** 时强制使用。
selector	map	指定要使用的特定持久性卷。它包含代表选择此类卷的 key:value 对。
deleteClaim	布尔值	指定在集群未部署时是否必须删除持久性卷声明。
class	string	用于动态卷分配的存储类。
id	整数	存储标识号。只适用于在类型为"jbod"存储中定义的存储卷。
overrides	PersistentClaimStorageOverride 数组	覆盖单个代理。**overrides** 字段允许为不同的代理指定不同的配置。

第 19 章 PERSISTENTCLAIMSTORAGEOVERRIDE 模式参考

used in: PersistentClaimStorage

属性	属性类型	描述
class	string	用于此代理的动态卷分配的存储类。
broker	整数	kafka 代理的 ID （代理标识符）。

第 20 章 JBODSTORAGE SCHEMA 参考

用于： KafkaClusterSpec, KafkaNodePoolSpec

type 属性用于区分使用的 JbodStorage 类型，包括 EphemeralStorage, PersistentClaimStorage。对于类型 JbodStorage，它需要是值 jbod。

属性	属性类型	描述
type	string	必须为 **jbod**。
卷	EphemeralStorage, PersistentClaimStorage 数组	卷列表作为代表 JBOD 存储阵列的 Storage 对象。

第 21 章 KAFKAAUTHORIZATIONSIMPLE 模式参考

使用于： KafkaClusterSpec

KafkaAuthorizationSimple 模式属性的完整列表

对于简单授权，Apache Kafka 的流使用 Kafka 的内置授权插件：KRaft 模式的 StandardAuthorizer 和基于 ZooKeeper 的集群管理的 AclAuthorizer。ACL 允许您定义哪些用户有权访问细致级别的资源。

配置 Kafka 自定义资源以使用简单授权。将 authorization 部分中的 type 属性设为值 simple，并配置超级用户列表。

为 KafkaUser 配置访问规则，如 ACLRule 模式参考 中所述。

21.1. SUPERUSERS

被视为超级用户的用户主体列表，以便在不查询 ACL 规则的情况下始终允许它们。

简单授权配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
  namespace: myproject
spec:
  kafka:
    # ...
    authorization:
      type: simple
      superUsers:
        - CN=client_1
        - user_2
        - CN=client_3
    # ...
```

注意

Kafka.spec.kafka 中的 config 配置属性中的 super.user 配置选项会被忽略。改为指定 authorization 属性中的超级用户。如需更多信息，请参阅 Kafka 代理配置。

21.2. KAFKAAUTHORIZATIONSIMPLE 模式属性

type 属性是一个辨别器，可区分来自 KafkaAuthorizationOpa, KafkaAuthorizationKeycloak, KafkaAuthorizationCustom 的 KafkaAuthorizationSimple 类型。对于类型 KafkaAuthorizationSimple，它需要有 simple 值。

属性	属性类型	描述
type	string	必须 **很简单**。
superUsers	字符串数组	超级用户列表。应包含用户主体列表，其应获得无限访问权限。

第 22 章 KAFKAAUTHORIZATIONOPA 模式参考

使用于： KafkaClusterSpec

KafkaAuthorizationOpa 模式属性的完整列表

要使用 Open Policy Agent 授权，请将 authorization 部分中的 type 属性设为值 opa，并根据需要配置 OPA 属性。Apache Kafka 的 Streams 使用 Open Policy Agent 插件作为授权器。有关输入数据和策略示例格式的更多信息，请参阅 Kafka 授权的 Open Policy Agent 插件。

22.1. URL

用于连接到 Open Policy Agent 服务器的 URL。URL 必须包含将由授权器查询的策略。必需。

22.2. ALLOWONERROR

定义当授权器无法查询 Open Policy Agent 时（例如，当 Kafka 客户端暂时不可用时）是否应默认允许或拒绝 Kafka 客户端。默认值为 false - 所有操作都将被拒绝。

22.3. INITIALCACHECAPACITY

授权器使用的本地缓存的初始容量，以避免为每个请求查询 Open Policy Agent。默认值为 5000。

22.4. MAXIMUMCACHESIZE

授权器使用的本地缓存的最大容量，以避免为每个请求查询 Open Policy Agent。默认值为 50000。

22.5. EXPIREAFTERMS

保存在本地缓存中记录的过期，以避免为每个请求查询 Open Policy Agent。定义从 Open Policy Agent 服务器重新加载缓存的授权决策的频率。以毫秒为单位。默认为 3600000 毫秒(1 小时)。

22.6. TLSTRUSTEDCERTIFICATES

用于 TLS 连接到 OPA 服务器的可信证书。

22.7. SUPERUSERS

被视为超级用户的用户主体列表，以便在不查询开放策略代理策略的情况下始终允许它们。

Open Policy Agent 授权器配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
  namespace: myproject
spec:
  kafka:
    # ...
    authorization:
      type: opa
      url: http://opa:8181/v1/data/kafka/allow
      allowOnError: false
      initialCacheCapacity: 1000
      maximumCacheSize: 10000
      expireAfterMs: 60000
      superUsers:
        - CN=fred
        - sam
        - CN=edward
    # ...
```

22.8. KAFKAAUTHORIZATIONOPA 模式属性

type 属性是一个辨别器，可区分来自 KafkaAuthorizationSimple, KafkaAuthorizationKeycloak, KafkaAuthorizationCustom 的 KafkaAuthorizationOpa 类型的使用。对于类型 KafkaAuthorizationOpa，它需要是值 opa。

属性	属性类型	描述
type	string	必须为 **opa**。

属性	属性类型	描述
url	string	用于连接到 Open Policy Agent 服务器的 URL。URL 必须包含将由授权器查询的策略。这个选项是必需的。
allowOnError	布尔值	定义当授权器无法查询 Open Policy Agent 时（例如，当 Kafka 客户端暂时不可用时）是否应默认允许或拒绝 Kafka 客户端。默认值为 **false** - 所有操作都将被拒绝。
initialCacheCapacity	整数	授权者使用的本地缓存的初始容量，以避免为每个请求查询 Open Policy Agent。默认为 **5000**。
maximumCacheSize	整数	授权器使用的本地缓存的最大容量，以避免为每个请求查询 Open Policy Agent。默认值为 **50000**。
expireAfterMs	整数	保存在本地缓存中记录的过期，以避免为每个请求查询 Open Policy Agent。定义从 Open Policy Agent 服务器重新加载缓存的授权决策的频率。以毫秒为单位。默认为 **3600000**。
tlsTrustedCertificates	**CertSecretSource** 数组	用于 TLS 连接到 OPA 服务器的可信证书。
superUsers	字符串数组	超级用户列表，特别是具有无限访问权限的用户主体列表。
enableMetrics	布尔值	定义 Open Policy Agent 授权器插件是否应提供指标。默认值为 **false**。

第 23 章 KAFKAAUTHORIZATIONKEYCLOAK 模式参考

使用于： KafkaClusterSpec

type 属性是一个差异性，可区分来自 KafkaAuthorizationSimple, KafkaAuthorizationOpa, KafkaAuthorizationCustom 的 KafkaAuthorizationKeycloak 类型的使用。对于类型 KafkaAuthorizationKeycloak，它需要有值 keycloak。

属性	属性类型	描述
type	string	必须是 **keycloak**。
clientId	string	Kafka 客户端 ID，用于向 OAuth 服务器进行身份验证并使用令牌端点 URI。
tokenEndpointUri	string	授权服务器令牌端点 URI。
tlsTrustedCertificates	**CertSecretSource** 数组	用于 TLS 连接到 OAuth 服务器的可信证书。
disableTlsHostnameVerification	布尔值	启用或禁用 TLS 主机名验证。默认值为 **false**。
delegateToKafkaAcls	布尔值	如果 Red Hat Single Sign-On Authorization Services 策略的 DENIED，则是否应将授权决定委派给 'Simple' 授权器。默认值为 **false**。
grantsRefreshPeriodSeconds	整数	连续两次的时间允许刷新运行（以秒为单位）。默认值为 60。
grantsRefreshPoolSize	整数	用于刷新活跃会话的线程数量。更多线程（更并行性）越多，因此作业完成越早。但是，使用更多线程会对授权服务器造成更高的负载。默认值为 5。
grantsGcPeriodSeconds	整数	连续运行作业之间的时间（以秒为单位）。默认值为 300。
grantsAlwaysLatest	布尔值	控制是否为新会话获取最新的授权。启用后，从 Red Hat Single Sign-On 检索并缓存该用户。默认值为 **false**。
superUsers	字符串数组	超级用户列表。应包含用户主体列表，其应获得无限访问权限。

属性	属性类型	描述
connectTimeoutSeconds	整数	连接到授权服务器时的连接超时（以秒为单位）。如果没有设置，则有效的连接超时为 60 秒。
readTimeoutSeconds	整数	连接到授权服务器时读取超时（以秒为单位）。如果没有设置，则有效读取超时为 60 秒。
httpRetries	整数	初始 HTTP 请求失败时尝试的最大重试次数。如果没有设置，则默认为不尝试任何重试。
enableMetrics	布尔值	启用或禁用 OAuth 指标。默认值为 **false**。
includeAcceptHeader	布尔值	Accept 标头是否应该在请求中设置到授权服务器。默认值为 **true**。
grantsMaxIdleTimeSeconds	整数	闲置授权可以从缓存中驱除的时间（以秒为单位）。默认值为 300。

第 24 章 KAFKAAUTHORIZATIONCUSTOM 模式参考

使用于： KafkaClusterSpec

KafkaAuthorizationCustom 模式属性的完整列表

要将自定义授权用于 Apache Kafka，您可以配置自己的 Authorizer 插件来定义访问控制列表(ACL)。

ACL 允许您定义哪些用户有权访问细致级别的资源。

配置 Kafka 自定义资源以使用自定义授权。将 authorization 部分中的 type 属性设为值 custom，并配置超级用户列表。

重要

自定义授权器必须实施 org.apache.kafka.server.authorizer.Authorizer 接口，并支持使用 super.users 配置属性配置超级用户。

24.1. AUTHORIZERCLASS

（必需）实施 org.apache.kafka.server.authorizer.Authorizer 接口的 Java 类，以支持自定义 ACL。

24.2. SUPERUSERS

被视为超级用户的用户主体列表，以便在不查询 ACL 规则的情况下始终允许它们。

注意

Kafka.spec.kafka 中的 config 配置属性中的 super.user 配置选项会被忽略。改为指定 authorization 属性中的超级用户。如需更多信息，请参阅 Kafka 代理配置。

24.3. 其他配置选项

您可以使用 Kafka.spec.kafka.config 添加额外的配置来初始化自定义授权器。

Kafka.spec下的自定义授权配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
  namespace: myproject
spec:
  kafka:
    # ...
    authorization:
      type: custom
      authorizerClass: io.mycompany.CustomAuthorizer
      superUsers:
        - CN=client_1
        - user_2
        - CN=client_3
    # ...
    config:
      authorization.custom.property1=value1
      authorization.custom.property2=value2
    # ...
```

24.4. 将自定义授权器 JAR 文件添加到容器镜像

除了 Kafka 自定义资源配置外，包含自定义授权器类及其依赖项的 JAR 文件还必须在 Kafka 代理的类路径上可用。

您可以从 source-code 为 Apache Kafka 构建流来添加它们。Apache Kafka 构建过程的 Streams 提供了将自定义第三方库添加到生成的 Kafka 代理容器镜像的机制，方法是将自定义第三方库作为依赖项添加到 docker-images/artifacts/kafka-thirdparty-libs 目录下的 pom.xml 文件中。目录包含不同 Kafka 版本的不同文件夹。选择适当的文件夹。在修改 pom.xml 文件前，第三方库必须在 Maven 存储库中可用，并且 Maven 存储库必须可以被 Apache Kafka 构建过程访问。

另外，您可以将 JAR 添加到 Apache Kafka 容器镜像的现有 Streams 中：

```
FROM registry.redhat.io/amq-streams/kafka-37-rhel9:2.7.0
USER root:root
```

```
COPY ./my-authorizer/ /opt/kafka/libs/
USER 1001
```

24.5. 使用带有 OAUTH 身份验证的自定义授权程序

当使用带有 groupsClaim 配置的 oauth 身份验证从 JWT 令牌中提取用户组信息时，可以在自定义授权调用中使用组信息。组可以在自定义授权调用过程中通过 OAuthKafkaPrincipal 对象访问，如下所示：

```
    public List<AuthorizationResult> authorize(AuthorizableRequestContext requestContext,
List<Action> actions) {

        KafkaPrincipal principal = requestContext.principal();
        if (principal instanceof OAuthKafkaPrincipal) {
            OAuthKafkaPrincipal p = (OAuthKafkaPrincipal) principal;

            for (String group: p.getGroups()) {
                System.out.println("Group: " + group);
            }
        }
    }
```

24.6. KAFKAAUTHORIZATIONCUSTOM 模式属性

type 属性是一个辨别器，可区分来自 KafkaAuthorizationSimple, KafkaAuthorizationOpa, KafkaAuthorizationKeycloak 的 KafkaAuthorizationCustom 类型。对于类型 KafkaAuthorizationCustom，它需要有值 custom。

属性	属性类型	描述
type	string	必须是 **custom**。
authorizerClass	string	授权实施类必须在 classpath 中提供。
superUsers	字符串数组	超级用户列表，它们是具有无限访问权限的用户主体。
supportsAdminApi	布尔值	指明自定义授权器是否支持使用 Kafka Admin API 管理 ACL 的 API。默认值为 **false**。

第 25 章 RACK 模式参考

used in: KafkaBridgeSpec, KafkaClusterSpec, KafkaConnectSpec, KafkaMirrorMaker2Spec

Rack 模式属性的完整列表

rack 选项配置机架感知。*机架* 可以代表一个可用性区域、数据中心或数据中心中的实际机架。*机架* 通过 topologyKey 配置。topologyKey 标识 OpenShift 节点上的标签，其值中包含拓扑名称。此类标签的示例是 topology.kubernetes.io/zone （或旧的 OpenShift 版本上的 failure-domain.beta.kubernetes.io/zone ），其中包含运行 OpenShift 节点的可用区的名称。您可以将 Kafka 集群配置为了解其运行的 *机架*，并启用额外的功能，如将分区副本分散到不同的机架中或使用最接近的副本的消息。

如需有关 OpenShift 节点标签的更多信息，请参阅 Well -Known Labels、Annotations 和 Taints。有关代表节点要部署到的区域或机架的节点标签，请参阅您的 OpenShift 管理员。

25.1. 在机架之间分散分区副本

当配置了机架感知时，Apache Kafka 的 Streams 将为每个 Kafka 代理设置 broker.rack 配置。broker.rack 配置为每个代理分配一个机架 ID。配置 broker.rack 时，Kafka 代理将尽可能将分区副本分散到不同的机架中。当副本分散到多个机架中时，多个副本的概率会同时失败，超过同一机架中。分散副本提高了弹性，对于可用性和可靠性至关重要。要在 Kafka 中启用机架感知，请在 Kafka 自定义资源的 .spec.kafka 部分添加 rack 选项，如下例所示。

Kafka 的 rack 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
    rack:
      topologyKey: topology.kubernetes.io/zone
    # ...
```

注意

当 pod 被删除或重启时，代理运行的 *机架* 在某些情况下可能会改变。因此，在不同机架中运行的副本可能会共享相同的机架。使用 Cruise Control 和带有 RackAwareGoal 的 KafkaRebalance 资源，以确保副本在不同机架之间保持分布。

当在 Kafka 自定义资源中启用机架感知时，Apache Kafka 的 Streams 将自动添加 OpenShift preferredDuringSchedulingIgnoredDuringExecution 关联性规则，以便在不同的机架之间分发 Kafka 代理。但是，*偏好规则* 不保证代理将被分散。根据确切的 OpenShift 和 Kafka 配置，您应该添加额外的关联性规则，或为 ZooKeeper 和 Kafka 配置 topologySpreadConstraints，以确保节点可以正确分布在多个机架中。如需更多信息，请参阅配置 pod 调度。

25.2. 使用来自最接近的副本的消息

机架感知也可用于消费者从最接近的副本获取数据。当 Kafka 集群跨越多个数据中心时，这可用于减少网络上的负载，并可在公有云中运行 Kafka 时降低成本。但是，可能会导致延迟增加。

为了能够从最接近的副本使用，必须在 Kafka 集群中配置机架感知，且必须启用 RackAwareReplicaSelector。replica selector 插件提供逻辑，使客户端能够从最接近的副本使用。默认实现使用 LeaderSelector 始终为客户端选择领导副本。为 replica.selector.class 指定 RackAwareReplicaSelector 来从默认的实现中切换。

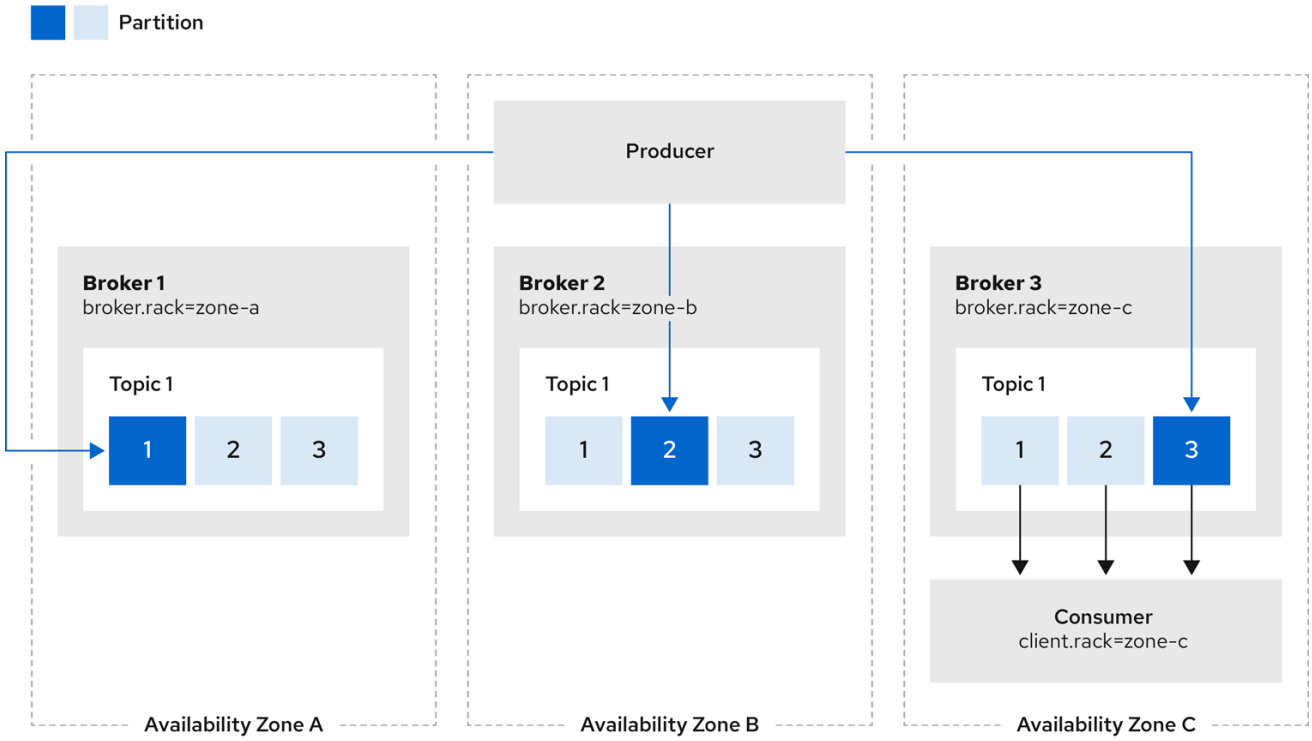
带有启用副本感知选择器的 rack 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
    rack:
      topologyKey: topology.kubernetes.io/zone
    config:
      # ...
      replica.selector.class: org.apache.kafka.common.replica.RackAwareReplicaSelector
    # ...
```

除了 Kafka 代理配置外，您还需要在消费者中指定 client.rack 选项。client.rack 选项应该指定运行使

用者的 *机架 ID*。RackAwareReplicaSelector 关联匹配的 broker.rack 和 client.rack ID，以查找最接近的副本并从其使用。如果同一机架中有多个副本，RackAwareReplicaSelector 始终选择最新的副本。如果没有指定机架 ID，或者找不到具有相同机架 ID 的副本，它将回退到领导副本。

图 25.1. 在同一可用区中显示来自副本的客户端示例



您还可以配置 Kafka Connect、MirrorMaker 2 和 Kafka Bridge，以便连接器使用最接近的副本的消息。您可以在 KafkaConnect、KafkaMirrorMaker2 和 KafkaBridge 自定义资源中启用机架感知。配置不会设置关联性规则，但您也可以配置 affinity 或 topologySpreadConstraints。如需更多信息，请参阅配置 pod 调度。

当使用 Streams for Apache Kafka 部署 Kafka Connect 时，您可以使用 KafkaConnect 自定义资源中的 rack 部分自动配置 client.rack 选项。

Kafka Connect 的 rack 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
# ...
spec:
  # ...
  rack:
    topologyKey: topology.kubernetes.io/zone
  # ...
```

当使用 Apache Kafka 的 Streams 部署 MirrorMaker 2 时，您可以使用 KafkaMirrorMaker2 自定义资源中的 rack 部分自动配置 client.rack 选项。

MirrorMaker 2 的 rack 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaMirrorMaker2
# ...
spec:
  # ...
  rack:
    topologyKey: topology.kubernetes.io/zone
  # ...
```

当使用 Streams for Apache Kafka 部署 Kafka Bridge 时，您可以使用 KafkaBridge 自定义资源中的 rack 部分自动配置 client.rack 选项。

Kafka Bridge 的 rack 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaBridge
# ...
spec:
  # ...
  rack:
    topologyKey: topology.kubernetes.io/zone
  # ...
```

25.3. RACK 模式属性

属性	属性类型	描述
topologyKey	string	与分配给 OpenShift 集群节点的标签匹配的键。标签的值用于设置代理的 **broker.rack** 配置，以及用于 Kafka Connect 或 MirrorMaker 2 的 **client.rack** 配置。

第 26 章 PROBE 模式参考

用于： CruiseControlSpec, EntityTopicOperatorSpec, EntityUserOperatorSpec, KafkaBridgeSpec, KafkaClusterSpec, KafkaConnectSpec, KafkaExporterSpec, KafkaMirrorMaker2Spec, KafkaMirrorMakerSpec, TlsSidecar, ZookeeperClusterSpec

属性	属性类型	描述
failureThreshold	整数	在成功后，探测被视为失败的连续最小失败。默认值为 3。最小值为 1。
initialDelaySeconds	整数	第一次检查健康状况前的初始延迟。默认为 15 秒。最小值为 0。
periodSeconds	整数	执行探测的频率（以秒为单位）。默认值为 10 秒。最小值为 1。
successThreshold	整数	在失败后，探测被视为成功的最低连续成功。默认为 1。对于存活度，必须为 1。最小值为 1。
timeoutSeconds	整数	每次尝试健康检查的超时时间。默认为 5 秒。最小值为 1。

第 27 章 JVMOPTIONS 模式参考

用于： CruiseControlSpec, EntityTopicOperatorSpec, EntityUserOperatorSpec, KafkaBridgeSpec, KafkaClusterSpec, KafkaConnectSpec, KafkaMirrorMaker2Spec, KafkaMirrorMakerSpec, KafkaNodePoolSpec, ZookeeperClusterSpec

属性	属性类型	描述
-XX	map	JVM 的 -XX 选项映射。
-Xms	string	JVM 的 -Xms 选项。
-Xmx	string	JVM 的 -Xmx 选项。
gcLoggingEnabled	布尔值	指定是否启用 Garbage Collection 日志记录。默认值为 false。
javaSystemProperties	SystemProperty 数组	将利用 **-D** 选项传递给 JVM 的其他系统属性映射。

第 28 章 SYSTEMPROPERTY 模式参考

使用于： JvmOptions

属性	属性类型	描述
name	string	系统属性名称。
value	string	系统属性值。

第 29 章 KAFKAJMXOPTIONS 模式参考

用于： KafkaClusterSpec, KafkaConnectSpec, KafkaMirrorMaker2Spec, ZookeeperClusterSpec

KafkaJmxOptions 模式属性的完整列表

配置 JMX 连接选项。

通过连接到端口 9999，从 Kafka 代理、ZooKeei 节点、Kafka Connect 和 MirrorMaker 2 获取 JMX 指标。使用 jmxOptions 属性配置密码保护或未保护的 JMX 端口。使用密码保护可防止未授权的 pod 访问端口。

然后您可以获取有关组件的指标。

例如，对于每个 Kafka 代理，您可以从客户端获取字节每秒使用量数据，或者代理的网络请求率。

要为 JMX 端口启用安全性，请将 身份验证 字段中的 type 参数设置为 password。

Kafka 代理和 ZooKeeper 节点的受密码保护的 JMX 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
    jmxOptions:
      authentication:
        type: "password"
    # ...
  zookeeper:
    # ...
    jmxOptions:
      authentication:
        type: "password"
    #...
```

然后，您可以通过指定您要地址的代理，使用无头服务将 pod 部署到集群中，并使用无头服务获取 JMX 指标。

例如，从代理 *0* 中获取 JMX 指标，您可以指定：

```
"CLUSTER-NAME-kafka-0.CLUSTER-NAME-kafka-brokers"
```

CLUSTER-NAME-kafka-0 是代理 pod 的名称，*CLUSTER-NAME*-kafka-brokers 是无头服务的名称，以返回代理 pod 的 IP。

如果 JMX 端口安全，您可以通过从 Pod 部署中的 JMX Secret 引用用户名和密码来获取它们。

对于未受保护的 JMX 端口，请使用空对象 {} 在无头服务上打开 JMX 端口。您可以部署 pod 并获取与受保护端口相同的方法，但在这种情况下，任何 pod 都可以从 JMX 端口读取。

Kafka 代理和 ZooKeeper 节点的开放端口 JMX 示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
    jmxOptions: {}
    # ...
  zookeeper:
    # ...
    jmxOptions: {}
    # ...
```

其他资源

- 有关使用 JMX 公开的 Kafka 组件指标的更多信息，请参阅 Apache Kafka 文档。

29.1. KAFKAJMXOPTIONS 模式属性

属性	属性类型	描述
身份验证	KafkaJmxAuthentication Password	用于连接到 JMX 端口的身份验证配置。

第 30 章 KAFKAJMXAUTHENTICATIONPASSWORD 模式参考

用于： KafkaJmxOptions

type 属性是一个辨别器，它区分使用来自以后可能添加的其他子类型的 KafkaJmxAuthenticationPassword 类型。对于类型 KafkaJmxAuthenticationPassword，它需要是值 password。

属性	属性类型	描述
type	string	必须为 **password**。

第 31 章 JMXPROMETHEUSEXPORTERMETRICS SCHEMA REFERENCE

用于： CruiseControlSpec, KafkaClusterSpec, KafkaConnectSpec, KafkaMirrorMaker2Spec, KafkaMirrorMakerSpec, ZookeeperClusterSpec

type 属性是一个辨别器，它区分使用来自以后可能添加的其他子类型 JmxPrometheusExporterMetrics 类型。对于类型 JmxPrometheusExporterMetrics，它需要有值 jmxPrometheusExporter。

属性	属性类型	描述
type	string	必须是 **jmxPrometheusExporter**。
valueFrom	**ExternalConfigurationReference**	存储 Prometheus JMX 导出器配置的 ConfigMap 条目。

第 32 章 EXTERNALCONFIGURATIONREFERENCE 模式参考

用于： ExternalLogging, JmxPrometheusExporterMetrics

属性	属性类型	描述
configMapKeyRef	ConfigMapKeySelector	对包含配置的 ConfigMap 中的键的引用。

第 33 章 INLINELOGGING 模式参考

用于： CruiseControlSpec, EntityTopicOperatorSpec, EntityUserOperatorSpec, KafkaBridgeSpec, KafkaClusterSpec, KafkaConnectSpec, KafkaMirrorMaker2Spec, KafkaMirrorMakerSpec, ZookeeperClusterSpec

type 属性是一个辨别器，可区分来自 ExternalLogging 的 InlineLogging 类型的使用。对于类型 InlineLogging，它需要有 inline 值。

属性	属性类型	描述
type	string	必须是 **inline**。
loggers	map	从日志记录器名称到日志记录器级别的映射。

第 34 章 EXTERNALLOGGING 模式参考

用于： CruiseControlSpec, EntityTopicOperatorSpec, EntityUserOperatorSpec, KafkaBridgeSpec, KafkaClusterSpec, KafkaConnectSpec, KafkaMirrorMaker2Spec, KafkaMirrorMakerSpec, ZookeeperClusterSpec

type 属性是一个辨别器，可区分来自 InlineLogging 类型的 ExternalLogging 类型。对于类型 ExternalLogging，它需要的值是 external。

属性	属性类型	描述
type	string	必须是 **外部**。
valueFrom	ExternalConfigurationReference	存储日志记录配置的 **ConfigMap** 条目。

第 35 章 KAFKACLUSTERTEMPLATE 模式参考

使用于： KafkaClusterSpec

属性	属性类型	描述
statefulset	StatefulSetTemplate	**statefulset 属性已弃用。**在 Apache Kafka 2.5 的 Streams 中删除了对 StatefulSets 的支持。此属性将被忽略。Kafka **StatefulSet** 模板。
pod	PodTemplate	Kafka **Pod 的模板**。
bootstrapService	InternalServiceTemplate	Kafka bootstrap **Service** 的模板。
brokersService	InternalServiceTemplate	Kafka **代理服务** 的模板。
externalBootstrapService	ResourceTemplate	Kafka 外部 bootstrap **Service** 的模板。
perPodService	ResourceTemplate	用于从 OpenShift 外部访问的 Kafka 针对每个 **Services** 的模板。
externalBootstrapRoute	ResourceTemplate	Kafka 外部 bootstrap 路由 **的模板**。
perPodRoute	ResourceTemplate	用于从 OpenShift 外部访问的 Kafka 针对每个 **Routes** 的模板。
externalBootstrapIngress	ResourceTemplate	Kafka 外部 bootstrap **Ingress** 的模板。
perPodIngress	ResourceTemplate	用于从 OpenShift 外部访问的 Kafka 针对每个 **Ingress** 模板。
persistentVolumeClaim	ResourceTemplate	所有 Kafka **PersistentVolumeClaims** 的模板。
podDisruptionBudget	PodDisruptionBudgetTemplate	Kafka **PodDisruptionBudget** 的模板。
kafkaContainer	ContainerTemplate	Kafka 代理容器的模板。
initContainer	ContainerTemplate	Kafka init 容器的模板。
clusterCaCert	ResourceTemplate	使用 Kafka 集群证书公钥的 Secret 模板。
serviceAccount	ResourceTemplate	Kafka 服务帐户的模板。

属性	属性类型	描述
jmxSecret	**ResourceTemplate**	Kafka Cluster JMX 身份验证的 Secret 模板。
clusterRoleBinding	**ResourceTemplate**	Kafka ClusterRoleBinding 的模板。
podSet	**ResourceTemplate**	Kafka **StrimziPodSet** 资源的模板。

第 36 章 STATEFULSETTEMPLATE 模式参考

用于： **KafkaClusterTemplate**, **ZookeeperClusterTemplate**

属性	属性类型	描述
metadata	**MetadataTemplate**	应用到资源的元数据。
podManagementPolicy	字符串([OrderedReady、Parallel] 之一)	用于此 StatefulSet 的 PodManagementPolicy。有效值为 **Parallel** 和 **OrderedReady**。默认为 **Parallel**。

第 37 章 METADATATEMPLATE 模式参考

用于： BuildConfigTemplate, DeploymentTemplate, InternalServiceTemplate, PodDisruptionBudgetTemplate, PodTemplate, ResourceTemplate, StatefulSetTemplate

MetadataTemplate 模式属性的完整列表

Labels 和 Annotations 用于识别和组织资源，并在 metadata 属性中配置。

例如：

```
# ...
template:
  pod:
    metadata:
      labels:
        label1: value1
        label2: value2
      annotations:
        annotation1: value1
        annotation2: value2
# ...
```

labels 和 annotations 字段可以包含没有保留字符串 strimzi.io 的任何标签或注解。包含 strimzi.io 的标签和注解在内部被 Apache Kafka 使用，且无法配置。

37.1. METADATATEMPLATE 模式属性

属性	属性类型	描述
labels	map	标签添加到 OpenShift 资源。
annotations	map	注解添加到 OpenShift 资源。

第 38 章 PODTEMPLATE 模式参考

用于： CruiseControlTemplate, EntityOperatorTemplate, JmxTransTemplate, KafkaBridgeTemplate, KafkaClusterTemplate, KafkaConnectTemplate, KafkaExporterTemplate, KafkaMirrorMakerTemplate, KafkaNodePoolTemplate, ZookeeperClusterTemplate

PodTemplate 模式属性的完整列表

配置 Kafka pod 模板。

PodTemplate 配置示例

```
# ...
template:
  pod:
    metadata:
      labels:
        label1: value1
      annotations:
        anno1: value1
    imagePullSecrets:
      - name: my-docker-credentials
    securityContext:
      runAsUser: 1000001
      fsGroup: 0
    terminationGracePeriodSeconds: 120
# ...
```

38.1. HOSTALIASES

使用 hostAliases 属性指定主机和 IP 地址列表，这些列表注入到 pod 的 /etc/hosts 文件中。

当用户同时请求集群外的连接时，此配置对 Kafka Connect 或 MirrorMaker 特别有用。

hostAliases 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
#...
spec:
 # ...
 template:
  pod:
   hostAliases:
   - ip: "192.168.1.86"
     hostnames:
     - "my-host-1"
     - "my-host-2"
   #...
```

38.2. PODTEMPLATE 模式属性

属性	属性类型	描述
metadata	**MetadataTemplate**	应用到资源的元数据。
imagePullSecrets	LocalObjectReference 数组	同一命名空间中的 secret 的引用列表，用于拉取此 Pod 使用的任何镜像。当 Cluster Operator 中的 **STRIMZI_IMAGE_PULL_SECRETS** 环境变量和 **imagePullSecrets** 选项被指定时，只使用 **imagePullSecrets** 变量，并忽略 **STRIMZI_IMAGE_PULL_SECRETS** 变量。
securityContext	PodSecurityContext	配置 pod 级别的安全属性和通用容器设置。
terminationGracePeriodSeconds	整数	宽限期是 pod 中运行的进程发送终止信号后的时间（以秒为单位），以及进程通过 kill 信号强制停止的时间。将此值设置为比您的进程预期的清理时间长。值必须是非负整数。零值表示立即删除。您可能需要为非常大的 Kafka 集群增加宽限期，以便 Kafka 代理有足够的时间将其工作传送到另一个代理。默认值为 30 秒。
关联性	关联性	pod 的关联性规则。
容限（tolerations）	容限 数组	pod 的容限。
priorityClassName	string	用于为 pod 分配优先级的优先级类的名称。

属性	属性类型	描述
schedulerName	string	用于分配此 **Pod** 的调度程序的名称。如果没有指定，将使用默认调度程序。
hostAliases	HostAlias 数组	pod 的 HostAliases。hostAliases 是主机和 IP 的可选列表，如果指定，将注入到 Pod 的主机文件中。
tmpDirSizeLimit	string	定义临时 EmptyDir 卷 (/**tmp**) 所需的本地存储的总数量（如 **1Gi**）。默认值为 **5Mi**。
enableServiceLinks	布尔值	指明是否应将有关服务的信息注入到 Pod 的环境变量中。
topologySpreadConstraints	TopologySpreadConstraint array	pod 的拓扑分布限制。

第 39 章 INTERNALSERVICETEMPLATE 模式参考

用于： CruiseControlTemplate, KafkaBridgeTemplate, KafkaClusterTemplate, KafkaConnectTemplate, ZookeeperClusterTemplate

属性	属性类型	描述
metadata	MetadataTemplate	应用到资源的元数据。
ipFamilyPolicy	字符串（一个 [RequireDualStack, SingleStack, PreferDualStack]）	指定服务使用的 IP 系列策略。可用选项包括 **SingleStack**、**PreferDualStack** 和 **RequireDualStack**。**SingleStack** 用于单个 IP 系列。**PreferDualStack** 用于两个 IP 系列，用于双栈集群配置的集群或单堆栈集群上的单个 IP 系列。**RequireDualStack** 失败，除非双栈配置的集群中有两个 IP 系列。如果未指定，OpenShift 将根据服务类型选择默认值。
ipFamilies	字符串（一个或多个 [IPv6, IPv4]）数组	指定服务使用的 IP Families。可用选项包括 **IPv4** 和 **IPv6**。如果未指定，OpenShift 将根据 **ipFamilyPolicy** 设置选择默认值。

第 40 章 RESOURCETEMPLATE 模式参考

用于： CruiseControlTemplate, EntityOperatorTemplate, JmxTransTemplate, KafkaBridgeTemplate, KafkaClusterTemplate, KafkaConnectTemplate, KafkaExporterTemplate, KafkaMirrorMakerTemplate, KafkaNodePoolTemplate, KafkaUserTemplate, ZookeeperClusterTemplate

属性	属性类型	描述
metadata	MetadataTemplate	应用到资源的元数据。

第 41 章 PODDISRUPTIONBUDGETTEMPLATE 模式参考

用于： CruiseControlTemplate, KafkaBridgeTemplate, KafkaClusterTemplate, KafkaConnectTemplate, KafkaMirrorMakerTemplate, ZookeeperClusterTemplate

PodDisruptionBudgetTemplate 模式属性的完整列表

PodDisruptionBudget (PDB)是一个 OpenShift 资源，通过指定计划维护或升级过程中必须可用的最少 pod 数量来确保高可用性。Apache Kafka 的 Streams 为每个新 StrimziPodSet 或 Deployment 创建一个 PDB。默认情况下，PDB 仅允许一个容器集在任何给定时间不可用。您可以通过更改 maxUnavailable 属性的默认值来增加允许的不可用 pod 数量。

StrimziPodSet 自定义资源使用无法直接使用 maxUnavailable 值的自定义控制器管理 pod。相反，在创建 PDB 资源时，maxUnavailable 值会自动转换为 minAvailable 值，它有效地提供相同的目的，如下例所示：

- 如果在 Kafka 资源中有三个代理 pod，并且 maxUnavailable 属性被设置为 1，minAvailable 设置为 2，则允许一个 pod 不可用。
- 如果有三个代理 pod，并且 maxUnavailable 属性被设置为 0 (零)，minAvailable 设置为 3，则需要所有三个代理 pod 都可用，并允许零个 pod 不可用。

PodDisruptionBudget 模板配置示例

```
# ...
template:
  podDisruptionBudget:
    metadata:
      labels:
        key1: label1
        key2: label2
      annotations:
        key1: label1
        key2: label2
    maxUnavailable: 1
# ...
```

41.1. PODDISRUPTIONBUDGETTEMPLATE 模式属性

属性	属性类型	描述
metadata	MetadataTemplate	应用到 **PodDisruptionBudgetTemplate** 资源的元数据。
maxUnavailable	整数	允许自动 pod 驱除的最大不可用 pod 数量。当 **maxUnavailable** 数量或较少 pod 在驱除后不可用时，允许 Pod 驱除。将此值设置为 0 可防止所有自愿驱除，因此必须手动驱除 pod。默认为 1。

第 42 章 CONTAINERTEMPLATE 模式参考

用于： CruiseControlTemplate, EntityOperatorTemplate, JmxTransTemplate, KafkaBridgeTemplate, KafkaClusterTemplate, KafkaConnectTemplate, KafkaExporterTemplate, KafkaMirrorMakerTemplate, KafkaNodePoolTemplate, ZookeeperClusterTemplate

ContainerTemplate 模式属性的完整列表

您可以为容器设置自定义安全上下文和环境变量。

环境变量在 env 属性下定义，作为带有 name 和 value 字段的对象列表。以下示例显示了两个自定义环境变量，并为 Kafka 代理容器设置了自定义安全上下文：

```
# ...
template:
  kafkaContainer:
    env:
    - name: EXAMPLE_ENV_1
      value: example.env.one
    - name: EXAMPLE_ENV_2
      value: example.env.two
    securityContext:
      runAsUser: 2000
# ...
```

前缀为 KAFKA_ 的环境变量是 Apache Kafka 的 Streams 内部，应该避免。如果您设置了已在由 Streams for Apache Kafka 使用的自定义环境变量，它会被忽略，并在日志中记录警告信息。

42.1. CONTAINERTEMPLATE 模式属性

属性	属性类型	描述
env	ContainerEnvVar 数组	应应用到容器的环境变量。
securityContext	securityContext	容器的安全上下文。

第 43 章 CONTAINERENVVAR 模式参考

使用于： ContainerTemplate

属性	属性类型	描述
name	string	环境变量密钥。
value	string	环境变量值。

第 44 章 TIEREDSTORAGECUSTOM 模式参考

使用于： KafkaClusterSpec

TieredStorageCustom 模式属性的完整列表

为 Kafka 启用自定义分层存储。

如果要使用自定义分层存储，您必须首先通过构建自定义容器镜像将 Kafka 插件的分层存储添加到 Apache Kafka 镜像的 Streams 中。

自定义分层存储配置允许使用自定义 RemoteStorageManager 配置。RemoteStorageManager 是一个 Kafka 接口，用于管理 Kafka 和远程分层存储之间的交互。

如果启用了自定义分层存储，Apache Kafka 的流将 TopicBasedRemoteLogMetadataManager 用于 Remote Log Metadata Management (RLMM)。

警告

分层存储是一个早期访问 Kafka 功能，它也可用于 Apache Kafka 的流。由于其当前限制，不建议在生产环境中使用。

自定义分层存储配置示例

```
kafka:
  tieredStorage:
    type: custom
    remoteStorageManager:
      className: com.example.kafka.tiered.storage.s3.S3RemoteStorageManager
      classPath: /opt/kafka/plugins/tiered-storage-s3/*
      config:
        # A map with String keys and String values.
        # Key properties are automatically prefixed with `rsm.config.`
        # and appended to Kafka broker config.
```

```
      storage.bucket.name: my-bucket
config:
  ...
  # Additional RLMM configuration can be added through the Kafka config
  # under `spec.kafka.config` using the `rlmm.config.` prefix.
  rlmm.config.remote.log.metadata.topic.replication.factor: 1
```

44.1. TIEREDSTORAGECUSTOM 模式属性

type 属性是一个辨别器，可区分将 TieredStorageCustom 类型的使用与其他子类型（可能在以后添加）。对于类型 TieredStorageCustom，它需要是 custom 的值。

属性	属性类型	描述
remoteStorageManager	RemoteStorageManager	远程存储管理器的配置。
type	string	必须是 **custom**。

第 45 章 REMOTESTORAGEMANAGER 模式参考

使用于： TieredStorageCustom

属性	属性类型	描述
className	string	**RemoteStorageManager** 实现的类名称。
classpath	string	**RemoteStorageManager** 实现的类路径。
config	map	**RemoteStorageManager** 实现的额外配置映射。密钥将自动作为前缀 **rsm.config.**，并添加到 Kafka 代理配置中。

第 46 章 ZOOKEEPERCLUSTERSPEC 模式参考

使用于： KafkaSpec

ZookeeperClusterSpec 模式属性的完整列表

配置 ZooKeeper 集群。

46.1. CONFIG

使用 config 属性将 ZooKeeper 选项配置为键。

这些值可以是以下 JSON 类型之一：

- 字符串
- Number
- 布尔值

例外

您可以指定并配置 ZooKeeper 文档 中列出的选项。

但是，Apache Kafka 的流负责配置和管理与以下内容相关的选项，而这无法更改：

- 安全性（加密、身份验证和授权）
- 侦听器配置

- 配置数据目录
- zookeeper 集群组成

无法设置具有以下前缀的属性：

- 4lw.commands.whitelist
- authProvider
- clientPort
- dataDir
- dataLogDir
- quorum.auth
- reconfigEnabled
- requireClientAuthScheme
- secureClientPort
- 服务器。
- snapshot.trust.empty

- standaloneEnabled
- serverCnxnFactory
- ssl.
- sslQuorum

如果 config 属性包含一个无法更改的选项，它将被忽略，并将警告信息记录到 Cluster Operator 日志文件。所有其他支持选项都转发到 ZooKeeper，包括以下例外到 Apache Kafka 的 Streams 配置的选项：

- 支持的 TLS 版本和密码套件的任何 ssl 配置

ZooKeeper 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
spec:
  kafka:
    # ...
  zookeeper:
    # ...
    config:
      autopurge.snapRetainCount: 3
      autopurge.purgeInterval: 2
    # ...
```

46.2. LOGGING

ZooKeeper 具有可配置的日志记录器：

- zookeeper.root.logger

ZooKeeper 使用 Apache log4j 日志记录器实现。

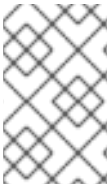
使用 logging 属性配置日志记录器和日志记录器级别。

您可以通过直接指定日志记录器和级别（在线）或使用自定义（外部） ConfigMap 来设置日志级别。如果使用 ConfigMap，您可以将 logging.valueFrom.configMapKeyRef.name 属性设置为包含外部日志记录配置的 ConfigMap 的名称。在 ConfigMap 中，日志记录配置使用 log4j.properties 描述。logging.valueFrom.configMapKeyRef.name 和 logging.valueFrom.configMapKeyRef.key 属性都是强制的。使用指定的确切日志记录配置的 ConfigMap 会在 Cluster Operator 运行时使用自定义资源创建，然后在每次协调后重新创建。如果没有指定自定义 ConfigMap，则使用默认日志记录设置。如果没有设置特定的日志记录器值，则会为该日志记录器继承上级日志记录器设置。有关日志级别的更多信息，请参阅 Apache 日志记录服务。

在这里，我们看到 内联 和 外部日志记录 的示例。inline 日志记录指定根日志记录器级别。您还可以通过将特定类或日志记录器添加到 loggers 属性来设置日志级别。

内联日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
spec:
  # ...
  zookeeper:
    # ...
    logging:
      type: inline
      loggers:
        zookeeper.root.logger: INFO
        log4j.logger.org.apache.zookeeper.server.FinalRequestProcessor: TRACE
        log4j.logger.org.apache.zookeeper.server.ZooKeeperServer: DEBUG
    # ...
```



注意

将日志级别设置为 DEBUG 可能会导致大量日志输出，并可能会影响性能。

外部日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
spec:
  # ...
  zookeeper:
    # ...
    logging:
      type: external
      valueFrom:
        configMapKeyRef:
          name: customConfigMap
          key: zookeeper-log4j.properties
  # ...
```

垃圾收集器(GC)

垃圾回收收集器日志记录也可以使用 jvmOptions 属性启用 （或禁用）。

46.3. ZOOKEEPERCLUSTERSPEC 模式属性

属性	属性类型	描述
replicas	整数	集群中的 pod 数量。
image	string	用于 ZooKeeper pod 的容器镜像。如果没有明确指定镜像名称，它会根据 **spec.kafka.version** 中设置的 Kafka 版本决定。镜像名称被专门映射到 Cluster Operator 配置中的对应版本。
storage	EphemeralStorage, PersistentClaimStorage	存储配置（磁盘）。无法更新。

属性	属性类型	描述
config	map	ZooKeeper 代理配置。无法设置带有以下前缀的属性：server., dataDir, dataLogDir, clientPort, authProvider, quorum.auth, requireClientAuthScheme, snapshot.trust.empty, standaloneEnabled, reconfigEnabled, 4lw.commands.whitelist, secureClientPort, ssl., serverCnxnFactory, sslQuorum （除 ssl.protocol 除外）ssl.quorum.protocol, ssl.enabledProtocols, ssl.quorum.enabledProtocols, ssl.ciphersuites, ssl.quorum.ciphersuites, ssl.hostnameVerification, ssl.quorum.hostnameVerification）。
livenessProbe	**probe**	Pod 存活度检查。
readinessProbe	**probe**	Pod 就绪度检查。
jvmOptions	**JvmOptions**	pod 的 JVM 选项。
jmxOptions	**KafkaJmxOptions**	Zookeeper 节点的 JMX 选项。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
metricsConfig	**JmxPrometheusExporterMetrics**	指标配置。
logging	**InlineLogging**, **ExternalLogging**	ZooKeeper 的日志记录配置。
模板	**ZookeeperClusterTemplate**	ZooKeeper 集群资源模板。该模板允许用户指定如何生成 OpenShift 资源。

第 47 章 ZOOKEEPERCLUSTERTEMPLATE 模式参考

使用于：Zoo keeperClusterSpec

属性	属性类型	描述
statefulset	StatefulSetTemplate	**statefulset 属性已弃用。**在 Apache Kafka 2.5 的 Streams 中删除了对 StatefulSets 的支持。此属性将被忽略。ZooKeeper **StatefulSet** 的模板。
pod	PodTemplate	ZooKeeper **Pod 的模板**。
clientService	InternalServiceTemplate	ZooKeeper 客户端服务 **模板**。
nodesService	InternalServiceTemplate	ZooKeeper 节点 **Service** 的模板。
persistentVolumeClaim	ResourceTemplate	所有 ZooKeeper **PersistentVolumeClaims** 的模板。
podDisruptionBudget	PodDisruptionBudgetTemplate	ZooKeeper **PodDisruptionBudget** 的模板。
zookeeperContainer	ContainerTemplate	ZooKeeper 容器的模板。
serviceAccount	ResourceTemplate	ZooKeeper 服务帐户的模板。
jmxSecret	ResourceTemplate	Zookeeper Cluster JMX 身份验证的 Secret 模板。
podSet	ResourceTemplate	ZooKeeper **StrimziPodSet** 资源的模板。

第 48 章 ENTITYOPERATORSPEC 模式参考

使用于： **KafkaSpec**

属性	属性类型	描述
topicOperator	**EntityTopicOperatorSpec**	配置主题 Operator。
userOperator	**EntityUserOperatorSpec**	配置 User Operator。
tlsSidecar	**TlsSidecar**	TLS sidecar 配置。
模板	**EntityOperatorTemplate**	Entity Operator 资源的模板。通过该模板，用户可以指定如何生成 **Deployment** 和 **Pod**。

第 49 章 ENTITYTOPICOPERATORSPEC 模式参考

用于： EntityOperatorSpec

EntityTopicOperatorSpec 模式属性的完整列表

配置主题 Operator。

49.1. LOGGING

主题 Operator 有一个可配置的日志记录器：

- rootLogger.level

主题 Operator 使用 Apache log4j2 日志记录器实现。

使用 Kafka 资源 Kafka 资源的 entityOperator.topicOperator 字段中的 logging 属性来配置日志记录器和日志记录器级别。

您可以通过直接指定日志记录器和级别（在线）或使用自定义（外部） ConfigMap 来设置日志级别。如果使用 ConfigMap，您可以将 logging.valueFrom.configMapKeyRef.name 属性设置为包含外部日志记录配置的 ConfigMap 的名称。在 ConfigMap 中，日志记录配置使用 log4j2.properties 来描述。logging.valueFrom.configMapKeyRef.name 和 logging.valueFrom.configMapKeyRef.key 属性都是强制的。使用指定的确切日志记录配置的 ConfigMap 会在 Cluster Operator 运行时使用自定义资源创建，然后在每次协调后重新创建。如果没有指定自定义 ConfigMap，则使用默认日志记录设置。如果没有设置特定的日志记录器值，则会为该日志记录器继承上级日志记录器设置。有关日志级别的更多信息，请参阅 Apache 日志记录服务。

在这里，我们看到 内联 和 外部日志记录 的示例。inline 日志记录指定根日志记录器级别。您还可以通过将特定类或日志记录器添加到 loggers 属性来设置日志级别。

内联日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
```

```
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
  zookeeper:
    # ...
  entityOperator:
    # ...
    topicOperator:
      watchedNamespace: my-topic-namespace
      reconciliationIntervalSeconds: 60
      logging:
        type: inline
        loggers:
          rootLogger.level: INFO
          logger.top.name: io.strimzi.operator.topic 1
          logger.top.level: DEBUG 2
          logger.toc.name: io.strimzi.operator.topic.TopicOperator 3
          logger.toc.level: TRACE 4
          logger.clients.level: DEBUG 5
  # ...
```

1

为 topic 软件包创建一个日志记录器。

2

设置 topic 软件包的日志记录级别。

3

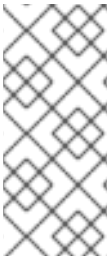
为 TopicOperator 类创建一个日志记录器。

4

设置 TopicOperator 类的日志记录级别。

5

更改默认 客户端 日志记录器的日志级别。客户端 日志记录器是流为 Apache Kafka 提供的日志记录配置的一部分。默认情况下，设置为 INFO。



注意

在调查操作器的问题时，通常足以将 rootLogger 更改为 DEBUG，以获取更详细的日志。但请注意，将日志级别设置为 DEBUG 可能会导致大量日志输出，并可能会影响性能。

外部日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
  zookeeper:
    # ...
  entityOperator:
    # ...
    topicOperator:
      watchedNamespace: my-topic-namespace
      reconciliationIntervalSeconds: 60
      logging:
        type: external
        valueFrom:
          configMapKeyRef:
            name: customConfigMap
            key: topic-operator-log4j2.properties
  # ...
```

垃圾收集器(GC)

垃圾回收收集器日志记录也可以使用 jvmOptions 属性启用 （或禁用）。

49.2. ENTITYTOPICOPERATORSPEC 模式属性

属性	属性类型	描述
watchedNamespace	string	主题 Operator 应该监视的命名空间。
image	string	用于主题 Operator 的镜像。

属性	属性类型	描述
reconciliationIntervalSeconds	整数	定期协调之间的间隔。
zookeeperSessionTimeoutSeconds	整数	ZooKeeper 会话的超时。
startupProbe	**probe**	Pod 启动检查。
livenessProbe	**probe**	Pod 存活度检查。
readinessProbe	**probe**	Pod 就绪度检查。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
topicMetadataMaxAttempts	整数	获取主题元数据时的尝试次数。
logging	**InlineLogging**, **ExternalLogging**	日志记录配置。
jvmOptions	**JvmOptions**	pod 的 JVM 选项。

第 50 章 ENTITYUSEROPERATORSPEC 模式参考

用于： EntityOperatorSpec

EntityUserOperatorSpec 模式属性的完整列表

配置 User Operator。

50.1. LOGGING

User Operator 有一个可配置的日志记录器：

- rootLogger.level

User Operator 使用 Apache log4j2 日志记录器实现。

使用 Kafka 资源的 entityOperator.userOperator 字段中的 logging 属性来配置日志记录器和日志记录器级别。

您可以通过直接指定日志记录器和级别（在线）或使用自定义（外部） ConfigMap 来设置日志级别。如果使用 ConfigMap，您可以将 logging.valueFrom.configMapKeyRef.name 属性设置为包含外部日志记录配置的 ConfigMap 的名称。在 ConfigMap 中，日志记录配置使用 log4j2.properties 来描述。logging.valueFrom.configMapKeyRef.name 和 logging.valueFrom.configMapKeyRef.key 属性都是强制的。使用指定的确切日志记录配置的 ConfigMap 会在 Cluster Operator 运行时使用自定义资源创建，然后在每次协调后重新创建。如果没有指定自定义 ConfigMap，则使用默认日志记录设置。如果没有设置特定的日志记录器值，则会为该日志记录器继承上级日志记录器设置。有关日志级别的更多信息，请参阅 Apache 日志记录服务。

在这里，我们看到 内联 和 外部日志记录 的示例。inline 日志记录指定 rootLogger.level。您还可以通过将特定类或日志记录器添加到 loggers 属性来设置日志级别。

内联日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
```

```
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
  zookeeper:
    # ...
  entityOperator:
    # ...
    userOperator:
      watchedNamespace: my-topic-namespace
      reconciliationIntervalSeconds: 60
      logging:
        type: inline
        loggers:
          rootLogger.level: INFO
          logger.uop.name: io.strimzi.operator.user (1)
          logger.uop.level: DEBUG (2)
          logger.abstractcache.name: io.strimzi.operator.user.operator.cache.AbstractCache (3)
          logger.abstractcache.level: TRACE (4)
          logger.jetty.level: DEBUG (5)

    # ...
```

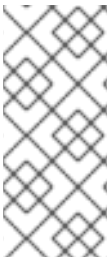
(1) 为 user 软件包创建一个日志记录器。

(2) 设置 user 软件包的日志级别。

(3) 为 AbstractCache 类创建一个日志记录器。

(4) 设置 AbstractCache 类的日志记录级别。

(5) 更改默认 jetty 日志记录器的日志级别。jetty logger 是 Apache Kafka 提供的日志记录配置的一部分。默认情况下，设置为 INFO。



注意

在调查操作器的问题时，通常足以将 rootLogger 更改为 DEBUG，以获取更详细的日志。但请注意，将日志级别设置为 DEBUG 可能会导致大量日志输出，并可能会影响性能。

外部日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    # ...
  zookeeper:
    # ...
  entityOperator:
    # ...
    userOperator:
      watchedNamespace: my-topic-namespace
      reconciliationIntervalSeconds: 60
      logging:
        type: external
        valueFrom:
          configMapKeyRef:
            name: customConfigMap
            key: user-operator-log4j2.properties
  # ...
```

垃圾收集器(GC)

垃圾回收收集器日志记录也可以使用 jvmOptions 属性启用 （或禁用）。

50.2. ENTITYUSEROPERATORSPEC 模式属性

属性	属性类型	描述
watchedNamespace	string	User Operator 应该监视的命名空间。
image	string	用于 User Operator 的镜像。

属性	属性类型	描述
reconciliationIntervalSeconds	整数	定期协调之间的间隔。
zookeeperSessionTimeoutSeconds	整数	**zookeeperSessionTimeoutSeconds 属性已弃用。**此属性已弃用，因为 User Operator 不再使用 ZooKeeper。ZooKeeper 会话的超时。
secretPrefix	string	要添加为 Secret 名称的 KafkaUser 名称中的前缀。
livenessProbe	**probe**	Pod 存活度检查。
readinessProbe	**probe**	Pod 就绪度检查。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
logging	**InlineLogging**, **ExternalLogging**	日志记录配置。
jvmOptions	**JvmOptions**	pod 的 JVM 选项。

第 51 章 TLSSIDECAR 模式参考

用于： CruiseControlSpec, EntityOperatorSpec

TlsSidecar 模式属性的完整列表

配置 TLS sidecar，这是在 pod 中运行但满足支持目的的容器。在 Apache Kafka 的 Streams 中，TLS sidecar 使用 TLS 来加密和解密组件和 ZooKeeper 之间的通信。

TLS sidecar 在 Entity Operator 中使用。

TLS sidecar 使用 Kafka.spec.entityOperator 中的 tlsSidecar 属性进行配置。

TLS sidecar 支持以下附加选项：

- image
- 资源
- logLevel
- readinessProbe
- livenessProbe

resources 属性指定为 TLS sidecar 分配的 内存和 CPU 资源。

image 属性配置将使用的容器镜像。

readinessProbe 和 livenessProbe 属性为 TLS sidecar 配置 健康检查探测。

logLevel 属性指定日志级别。支持以下日志记录级别：

- emerg
- alert
- crit
- err
- warning
- 注意
- info
- debug

默认值为 *notice*。

TLS sidecar 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  # ...
  entityOperator:
    # ...
```

```
tlsSidecar:
  resources:
    requests:
      cpu: 200m
      memory: 64Mi
    limits:
      cpu: 500m
      memory: 128Mi
# ...
```

51.1. TLSSIDECAR 模式属性

属性	属性类型	描述
image	string	容器的 docker 镜像。
livenessProbe	**probe**	Pod 存活度检查。
logLevel	字符串([emerg, debug, crit, err, alert, warning, notice, info])	TLS sidecar 的日志级别。默认值为 **notice**。
readinessProbe	**probe**	Pod 就绪度检查。
resources	ResourceRequirements	要保留的 CPU 和内存资源。

第 52 章 ENTITYOPERATORTEMPLATE 模式参考

用于： **EntityOperatorSpec**

属性	属性类型	描述
部署	**DeploymentTemplate**	Entity Operator **Deployment** 的模板。
pod	**PodTemplate**	实体 Operator **Pod** 的模板。
topicOperatorContainer	**ContainerTemplate**	Entity Topic Operator 容器的模板。
userOperatorContainer	**ContainerTemplate**	Entity User Operator 容器的模板。
tlsSidecarContainer	**ContainerTemplate**	Entity Operator TLS sidecar 容器的模板。
serviceAccount	**ResourceTemplate**	Entity Operator 服务帐户的模板。
entityOperatorRole	**ResourceTemplate**	Entity Operator 角色的模板。
topicOperatorRoleBinding	**ResourceTemplate**	Entity Topic Operator RoleBinding 的模板。
userOperatorRoleBinding	**ResourceTemplate**	Entity Topic Operator RoleBinding 的模板。

第 53 章 DEPLOYMENTTEMPLATE 模式参考

用于： CruiseControlTemplate, EntityOperatorTemplate, JmxTransTemplate, KafkaBridgeTemplate, KafkaConnectTemplate, KafkaExporterTemplate, KafkaMirrorMakerTemplate

DeploymentTemplate 模式属性的完整列表

使用 deploymentStrategy 指定在部署配置更改时将旧 pod 替换为新 pod 的策略。

使用以下值之一：

- RollingUpdate ：Pod 在不停机的情况下重启。
- 重新创建 ：Pod 在创建新 Pod 前被终止。

使用 Recreate 部署策略具有不需要备用资源的优势，但其缺点是应用程序停机。

显示将部署策略设置为 Recreate 的示例。

```
# ...
template:
  deployment:
    deploymentStrategy: Recreate
# ...
```

这个配置更改不会导致滚动更新。

53.1. DEPLOYMENTTEMPLATE 模式属性

属性	属性类型	描述
metadata	**MetadataTemplate**	应用到资源的元数据。
deploymentStrategy	字符串([RollingUpdate, Recreate] 之一)	部署配置更改的 Pod 替换策略。有效值为 **RollingUpdate** 和 **Recreate**。默认为 **RollingUpdate**。

第 54 章 CERTIFICATEAUTHORITY 模式参考

使用于： KafkaSpec

配置在集群中使用 TLS 证书的配置。这适用于用于集群中内部通信的证书，以及用于通过 Kafka.spec.kafka.listeners.tls 进行客户端访问的证书。

属性	属性类型	描述
generateCertificateAuthority	布尔值	如果为 true，则会自动生成证书颁发机构证书。否则，用户需要为 Secret 提供 CA 证书。默认为 true。
generateSecretOwnerReference	布尔值	如果为 **true**，Cluster 和 Client CA Secrets 会配置 **ownerReference** 设置为 **Kafka** 资源。如果为 **true** 时删除了 **Kafka** 资源，则 CA Secret 也会被删除。如果为 **false**，则禁用 **ownerReference**。如果 **Kafka** 资源在 **false** 时被删除，则保留 CA Secret 并可供重复使用。默认为 **true**。
validityDays	整数	生成的证书的天数应有效。默认值为 365。
renewalDays	整数	证书续订周期中的天数。这是证书过期前可以执行续订操作的天数。当 **generateCertificateAuthority** 为 true 时，这会导致生成新证书。当 **generateCertificateAuthority** 为 true 时，这会导致额外的日志记录级别在 WARN 级别有关待处理证书过期的情况。默认值为 30。
certificateExpirationPolicy	字符串([replace-key, renew-certificate] 之一)	在生成 **CertificateAuthority=true 时，如何处理 CA 证书过期**。默认值是生成的新 CA 证书，以使用现有的私钥。

第 55 章 CRUISECONTROLSPEC 模式参考

使用于： KafkaSpec

CruiseControlSpec 模式属性的完整列表

配置一个 Cruise Control 集群。

配置选项与以下内容相关：

- 目标配置
- 资源分布目标的容量限制

55.1. CONFIG

使用 config 属性将控制选项配置为密钥。

这些值可以是以下 JSON 类型之一：

- 字符串
- Number
- 布尔值

例外

您可以指定并配置 Cruise Control 文档 中列出的选项。

但是，Apache Kafka 的流负责配置和管理与以下内容相关的选项，而这无法更改：

- 安全性（加密、身份验证和授权）
- 连接到 Kafka 集群
- 客户端 ID 配置
- ZooKeeper 连接
- Web 服务器配置
- 自我修复

无法设置具有以下前缀的属性：

- bootstrap.servers
- capacity.config.file
- client.id
- failed.brokers.zk.path
- kafka.broker.failure.detection.enable
- metric.reporter.sampler.bootstrap.servers

- 网络。
- request.reason.required
- 安全性。
- self.healing.
- ssl.
- topic.config.provider.class
- two.step.
- webserver.accesslog.
- webserver.api.urlprefix
- webserver.http.
- webserver.session.path
- zookeeper.

如果 config 属性包含一个无法更改的选项，它将被忽略，并将警告信息记录到 Cluster Operator 日志文件。所有其他支持选项都转发到 Cruise Control，包括对 Apache Kafka 的 Streams 配置的选项的以下例外：

- 支持的 TLS 版本和密码套件的任何 ssl 配置

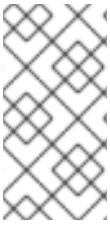
- 配置 webserver 属性以启用 Cross-Origin Resource Sharing (CORS)

Cruise Control 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  # ...
  cruiseControl:
    # ...
    config:
      # Note that `default.goals` (superset) must also include all `hard.goals` (subset)
      default.goals: >
        com.linkedin.kafka.cruisecontrol.analyzer.goals.RackAwareGoal,
        com.linkedin.kafka.cruisecontrol.analyzer.goals.ReplicaCapacityGoal
      hard.goals: >
        com.linkedin.kafka.cruisecontrol.analyzer.goals.RackAwareGoal
      cpu.balance.threshold: 1.1
      metadata.max.age.ms: 300000
      send.buffer.bytes: 131072
      webserver.http.cors.enabled: true
      webserver.http.cors.origin: "*"
      webserver.http.cors.exposeheaders: "User-Task-ID,Content-Type"
    # ...
```

55.2. 交叉资源共享(CORS)

cross-Origin Resource Sharing (CORS)是一种 HTTP 机制，用于控制对 REST API 的访问。限制可以是访问方法或客户端应用的原始 URL。您可以使用 config 中的 webserver.http.cors.enabled 属性通过 Cruise Control 启用 CORS。启用后，CORS 允许从不同于 Apache Kafka 的原始 URL 的应用程序读取对 Cruise Control REST API 的访问。这允许来自指定源中的应用程序通过 Cruise Control API 来使用 GET 请求获取 Kafka 集群的信息。例如，应用程序可以获取当前集群负载或最新优化建议的信息。不允许 POST 请求。



注意

有关使用 Cruise Control 的 CORS 的更多信息，请参阅 Cruise Control Wiki 中的 REST API。

为 Cruise Control 启用 CORS

您可以在 Kafka.spec.cruiseControl.config 中启用和配置 CORS。

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  # ...
  cruiseControl:
    # ...
    config:
      webserver.http.cors.enabled: true (1)
      webserver.http.cors.origin: "*" (2)
      webserver.http.cors.exposeheaders: "User-Task-ID,Content-Type" (3)

    # ...
```

(1)

启用 CORS。

(2)

为 Access-Control-Allow-Origin HTTP 响应标头指定允许的源。您可以使用通配符，或者将单个源指定为 URL。如果您使用通配符，则会在任何来源中返回响应。

(3)

为 Access-Control-Expose-Headers HTTP 响应标头公开指定的标头名称。允许的来源中的应用程序可以使用指定的标头读取响应。

55.3. CRUISE CONTROL REST API 安全性

Cruise Control REST API 使用 HTTP 基本身份验证和 SSL 保护集群，以保护集群免受潜在的破坏性 Cruise Control 操作，如弃用 Kafka 代理。我们建议 仅在启用了这些设置 的 Apache Kafka 中使用 Cruise Control for Apache Kafka。

但是，可以通过指定以下 Cruise Control 配置来禁用这些设置：

- 要禁用内置的 HTTP 基本身份验证，请将 webserver.security.enable 设置为 false。

- 要禁用内置的 SSL，请将 webserver.ssl.enable 设置为 false。

Cruise Control 配置来禁用 API 授权、身份验证和 SSL

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  # ...
  cruiseControl:
    config:
      webserver.security.enable: false
      webserver.ssl.enable: false
# ...
```

55.4. BROKERCAPACITY

Cruise Control 使用容量限制来确定资源容量限制的优化目标是否被破坏。这个类型有 4 个目标：

- DiskCapacityGoal - 磁盘使用率容量
- CpuCapacityGoal - CPU 使用率容量
- NetworkInboundCapacityGoal - 网络入站利用率容量
- NetworkOutboundCapacityGoal - 网络出站利用率容量

您可以在 Kafka.spec.cruiseControl 的 brokerCapacity 属性中为 Kafka 代理资源指定容量限制。它们默认是启用的，您可以更改它们的默认值。可以为以下代理资源设置容量限制：

- CPU - CPU 资源（毫秒或）或 CPU 内核（默认：1）

- inboundNetwork - 入站网络吞吐量（字节/每秒）（默认值：10000KiB/s）
- outboundNetwork - 出站网络吞吐量（字节/每秒）（默认值：10000KiB/s）

对于网络吞吐量，请使用带有标准 OpenShift 字节单元 (K、M、G) 或其 bibyte （指数 2）的等效值 (Ki, Mi, Gi) 的整数值。

注意

磁盘和 CPU 容量限制由 Apache Kafka 的 Streams 自动生成，因此您不需要设置它们。为了保证在使用 CPU 目标时准确的重新平衡提议，您可以在 Kafka.spec.kafka.resources 中设置与 CPU 限值相等的 CPU 请求。这样，所有 CPU 资源都会保留前期，并且始终可用。此配置允许 Cruise Control 在准备基于 CPU 目标的重新平衡建议时，正确评估 CPU 利用率。如果您在 Kafka.spec.kafka.resources 中无法设置 CPU 请求等于 CPU 限值，则可以为相同的准确性手动设置 CPU 容量。

使用 bibyte 单位的 Cruise Control brokerCapacity 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  # ...
  cruiseControl:
    # ...
    brokerCapacity:
      cpu: "2"
      inboundNetwork: 10000KiB/s
      outboundNetwork: 10000KiB/s
    # ...
```

55.5. 容量覆盖

代理可能会在带有异构网络或 CPU 资源的节点上运行。如果是这种情况，请指定 覆盖 为每个代理设置网络容量和 CPU 限制。覆盖可确保代理之间准确重新平衡。可以为以下代理资源设置覆盖容量限制：

- CPU - CPU 资源（毫秒或）或 CPU 内核（默认：1）
- inboundNetwork - 入站网络吞吐量（字节/每秒）（默认值：10000KiB/s）
- outboundNetwork - 出站网络吞吐量（字节/每秒）（默认值：10000KiB/s）

Cruise Control capacity 的示例使用 bibyte 单位覆盖配置

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
spec:
  # ...
  cruiseControl:
    # ...
    brokerCapacity:
      cpu: "1"
      inboundNetwork: 10000KiB/s
      outboundNetwork: 10000KiB/s
      overrides:
      - brokers: [0]
        cpu: "2.755"
        inboundNetwork: 20000KiB/s
        outboundNetwork: 20000KiB/s
      - brokers: [1, 2]
        cpu: 3000m
        inboundNetwork: 30000KiB/s
        outboundNetwork: 30000KiB/s
```

CPU 容量使用以下优先级顺序确定的配置值，首先优先使用最高的优先级：

1. Kafka.spec.cruiseControl.brokerCapacity.overrides.cpu 来为单个代理定义自定义 CPU 容量限制
2. kafka .cruiseControl.brokerCapacity.cpu，为 kafka 集群中的所有代理定义自定义 CPU 容量限制

3. Kafka.spec.kafka.resources.requests.cpu，用于定义 Kafka 集群中每个代理保留的 CPU 资源。

4. Kafka.spec.kafka.resources.limits.cpu，用于定义 Kafka 集群中每个代理可以使用的最大 CPU 资源。

这个优先级顺序是决定 Kafka 代理的实际容量限制时需要考虑不同配置值的顺序。例如，特定于代理的覆盖优先于所有代理的容量限制。如果没有指定任何 CPU 容量配置，则 Kafka 代理的默认 CPU 容量被设置为 1 个 CPU 内核。

如需更多信息，请参阅 BrokerCapacity 模式参考。

55.6. LOGGING

Cruise Control 本身有可配置的日志记录器：

- rootLogger.level

Cruise Control 使用 Apache log4j2 日志记录器实现。

使用 logging 属性配置日志记录器和日志记录器级别。

您可以通过直接指定日志记录器和级别（在线）或使用自定义（外部） ConfigMap 来设置日志级别。如果使用 ConfigMap，您可以将 logging.valueFrom.configMapKeyRef.name 属性设置为包含外部日志记录配置的 ConfigMap 的名称。在 ConfigMap 中，日志记录配置使用 log4j.properties 描述。logging.valueFrom.configMapKeyRef.name 和 logging.valueFrom.configMapKeyRef.key 属性都是强制的。使用指定的确切日志记录配置的 ConfigMap 会在 Cluster Operator 运行时使用自定义资源创建，然后在每次协调后重新创建。如果没有指定自定义 ConfigMap，则使用默认日志记录设置。如果没有设置特定的日志记录器值，则会为该日志记录器继承上级日志记录器设置。

在这里，我们看到 内联 和 外部日志记录 的示例。inline 日志记录指定根日志记录器级别。您还可以通过将特定类或日志记录器添加到 loggers 属性来设置日志级别。

内联日志记录

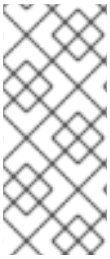
```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
# ...
spec:
  cruiseControl:
    # ...
    logging:
      type: inline
      loggers:
        rootLogger.level: INFO
        logger.exec.name: com.linkedin.kafka.cruisecontrol.executor.Executor 1
        logger.exec.level: TRACE 2
        logger.go.name: com.linkedin.kafka.cruisecontrol.analyzer.GoalOptimizer 3
        logger.go.level: DEBUG 4
    # ...
```

1 为 Cruise Control Executor 类创建一个日志记录器。

2 设置 Executor 类的日志级别。

3 为 Cruise Control GoalOptimizer 类创建一个日志记录器。

4 为 GoalOptimizer 类设置日志记录级别。



注意

在调查 Cruise Control 的问题时，通常足以将 rootLogger 更改为 DEBUG，以获得更详细的日志。但请注意，将日志级别设置为 DEBUG 可能会导致大量日志输出，并可能会影响性能。

外部日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
# ...
spec:
  cruiseControl:
    # ...
    logging:
      type: external
      valueFrom:
        configMapKeyRef:
          name: customConfigMap
          key: cruise-control-log4j.properties
    # ...
```

垃圾收集器(GC)

垃圾回收收集器日志记录也可以使用 jvmOptions 属性启用 （或禁用）。

55.7. CRUISECONTROLSPEC 模式属性

属性	属性类型	描述
image	string	用于 Cruise Control pod 的容器镜像。如果没有明确指定镜像名称，则镜像名称对应于 Cluster Operator 配置中指定的名称。如果在 Cluster Operator 配置中没有定义镜像名称，则会使用默认值。
tlsSidecar	**TlsSidecar**	**tlsSidecar 属性已弃用。**TLS sidecar 配置。
resources	ResourceRequirements	为 Cruise Control 容器保留的 CPU 和内存资源。
livenessProbe	**probe**	对 Cruise Control 容器进行 Pod 存活度检查。
readinessProbe	**probe**	对 Cruise Control 容器进行 Pod 就绪度检查。
jvmOptions	**JvmOptions**	Cruise Control 容器的 JVM 选项。
logging	**InlineLogging**, **ExternalLogging**	用于 Cruise Control 的日志记录配置(Log4j 2)

属性	属性类型	描述
模板	**CruiseControlTemplate**	模板来指定如何生成 Cruise Control 资源、**Deployment** 和 **Pod**。
brokerCapacity	**BrokerCapacity**	Cruise Control **brokerCapacity** 配置。
config	map	Cruise Control 配置。有关配置选项的完整列表，请参阅 https://github.com/linkedin/cruise-control/wiki/Configurations。请注意，带有以下前缀的属性无法设置：bootstrap.servers, client.id, zookeeper., network., security., failed.brokers.zk.path,webserver.http., webserver.api.urlprefix, webserver.session.path, webserver.accesslog., two.step., request.reason.required,metric.reporter.sampler.bootstrap.servers, capacity.config.file, self.healing., ssl., kafka.broker.failure.detection.enable, topic.config.provider.class （除：ssl.cipher.suites, ssl.protocol, ssl.enabled.protocols, webserver.http.cors.enabled, webserver.http.cors.origin, webserver.http.cors.exposeheaders, webserver.security.enable, webserver.ssl.enable）。
metricsConfig	**JmxPrometheusExporterMetrics**	指标配置。

第 56 章 CRUISECONTROLTEMPLATE 模式参考

使用于： **CruiseControlSpec**

属性	属性类型	描述
部署	**DeploymentTemplate**	用于 Cruise Control **Deployment** 的模板。
pod	**PodTemplate**	用于控制 **Pod** 的模板。
apiService	**InternalServiceTemplate**	Cruise Control API **Service** 的模板。
podDisruptionBudget	**PodDisruptionBudgetTemplate**	用于控制 **PodDisruptionBudget** 的模板。
cruiseControlContainer	**ContainerTemplate**	Cruise Control 容器的模板。
tlsSidecarContainer	**ContainerTemplate**	**tlsSidecarContainer 属性已弃用。** Cruise Control TLS sidecar 容器的模板。
serviceAccount	**ResourceTemplate**	Cruise Control 服务帐户的模板。

第 57 章 BROKERCAPACITY 模式参考

使用于： CruiseControlSpec

属性	属性类型	描述
disk	string	**disk 属性已弃用。**Cruise Control 磁盘容量设置已弃用，忽略，并将在以后的磁盘 Broker 容量中删除（以字节为单位）。使用带有标准 OpenShift 字节单位 (K、M、G 或 T) 的数字值、其 bibyte （指数 2）的等效值 (Ki、Mi、Gi 或 Ti)，或者带有或没有 E 表示法的字节值。例如：100000M、100000Mi、104857600000 或 1e+11。
cpuUtilization	整数	**cpuUtilization 属性已弃用。**Cruise Control CPU 容量设置已弃用，忽略，并将在以后的 CPU 资源使用率中的 Broker 容量作为一个百分比(0 - 100)删除。
cpu	string	内核或 millicores 的 CPU 资源的代理容量。例如：1, 1.500, 1500m。有关有效 CPU 资源单元的详情请参考 https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#meaning-of-cpu。
inboundNetwork	string	入站网络吞吐量（以字节为单位）的代理容量（以字节/秒为单位）。请使用带有标准 OpenShift 字节单元 (K、M、G) 或其 bibyte （指数 2）的等效值 (Ki, Mi, Gi) 的整数值。例如，10000KiB/s。
outboundNetwork	string	用于出站网络吞吐量的代理容量（以字节为单位）。请使用带有标准 OpenShift 字节单元 (K、M、G) 或其 bibyte （指数 2）的等效值 (Ki, Mi, Gi) 的整数值。例如，10000KiB/s。
overrides	BrokerCapacityOverride 数组	覆盖单个代理。**overrides** 属性允许您为不同的代理指定不同的容量配置。

第 58 章 BROKERCAPACITYOVERRIDE 模式参考

使用 in: BrokerCapacity

属性	属性类型	描述
代理(Broker)	整数数组	Kafka 代理列表（代理标识符）。
cpu	string	内核或 millicores 的 CPU 资源的代理容量。例如： 1, 1.500, 1500m。有关有效 CPU 资源单元的详情请参考 https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#meaning-of-cpu。
inboundNetwork	string	入站网络吞吐量（以字节为单位）的代理容量（以字节/秒为单位）。请使用带有标准 OpenShift 字节单元 (K、M、G) 或其 bibyte （指数 2）的等效值 (Ki, Mi, Gi) 的整数值。例如，10000KiB/s。
outboundNetwork	string	用于出站网络吞吐量的代理容量（以字节为单位）。请使用带有标准 OpenShift 字节单元 (K、M、G) 或其 bibyte （指数 2）的等效值 (Ki, Mi, Gi) 的整数值。例如，10000KiB/s。

第 59 章 JMXTRANSSPEC 模式参考

类型 JmxTransSpec 已被弃用。

使用于： KafkaSpec

属性	属性类型	描述
image	string	用于 JmxTrans 的镜像。
outputDefinitions	**JmxTransOutputDefinitionTemplate** 数组	定义稍后引用的输出主机。有关这些属性的更多信息，请参阅 **JmxTransOutputDefinitionTemplate** 模式参考。
logLevel	string	设置 JmxTrans 部署的日志级别。如需更多信息，请参阅 JmxTrans 日志记录级别。
kafkaQueries	**JmxTransQueryTemplate** 数组	发送到 Kafka 代理的查询，以定义应从每个代理读取哪些数据。有关这些属性的更多信息，请参阅 **JmxTransQueryTemplate** 模式参考。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
模板	**JmxTransTemplate**	JmxTrans 资源的模板。

第 60 章 JMXTRANSOUTPUTDEFINITIONTEMPLATE 模式参考

使用于：J mxTransSpec

属性	属性类型	描述
outputType	string	用于设置要推送的数据格式的模板。如需更多信息，请参阅 JmxTrans OutputWriters。
主机	string	数据推送到的远程主机的 DNS/hostname。
port	整数	数据推送到的远程主机的端口。
flushDelayInSeconds	整数	JmxTrans 在推送新数据集之前等待的时间。
typeNames	字符串数组	用于响应通配符查询中包含的过滤数据的模板。如需更多信息，请参阅 JmxTrans 查询。
name	string	用于设置输出定义名称的模板。这用于识别发送查询结果的位置。

第 61 章 JMXTRANSQUERYTEMPLATE 模式参考

使用于：J mxTransSpec

属性	属性类型	描述
targetMBean	string	如果使用通配符而不是特定的 MBean，则会从多个 MBeans 收集数据。否则，如果指定 MBean，则会从指定的 MBean 收集数据。
属性	字符串数组	确定应该包含目标 MBean 的属性。
输出	字符串数组	在 spec.kafka.jmxTrans.outputDefinitions 中指定的输出定义名称列表，用于定义 JMX 指标推送到的位置，以及数据格式。

第 62 章 JMXTRANSTEMPLATE 模式参考

使用于：J mxTransSpec

属性	属性类型	描述
部署	**DeploymentTemplate**	JmxTrans **部署** 模板。
pod	**PodTemplate**	JmxTrans **Pod 的模板**。
container	**ContainerTemplate**	JmxTrans 容器的模板。
serviceAccount	**ResourceTemplate**	JmxTrans 服务帐户的模板。

第 63 章 KAFKAEXPORTERSPEC 模式参考

使用于： **KafkaSpec**

属性	属性类型	描述
image	string	用于 Kafka Exporter pod 的容器镜像。如果没有明确指定镜像名称，则镜像名称对应于 Cluster Operator 配置中指定的版本。如果在 Cluster Operator 配置中没有定义镜像名称，则会使用默认值。
groupRegex	string	指定要收集的消费者组的正则表达式。默认值为 **.***。
topicRegex	string	指定收集哪些主题的正则表达式。默认值为 **.***。
groupExcludeRegex	string	指定要排除的消费者组的正则表达式。
topicExcludeRegex	string	指定要排除哪些主题的正则表达式。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
logging	string	仅记录具有指定严重性或更高严重性的日志消息。有效级别：[**info**,**debug**,**trace**]。默认日志级别为 **info**。
enableSaramaLogging	布尔值	启用 Sarama 日志记录，这是 Kafka Exporter 使用的 Go 客户端库。
showAllOffsets	布尔值	是否显示所有消费者组的 offset/lag，否则仅显示连接的消费者组。
模板	**KafkaExporterTemplate**	自定义部署模板和 pod。
livenessProbe	**probe**	Pod 存活度检查。
readinessProbe	**probe**	Pod 就绪度检查。

第 64 章 KAFKAEXPORTERTEMPLATE 模式参考

使用于： KafkaExporterSpec

属性	属性类型	描述
部署	DeploymentTemplate	Kafka Exporter **Deployment** 的模板。
pod	PodTemplate	Kafka Exporter **Pod 的模板**。
service	ResourceTemplate	**service 属性已弃用。**Kafka Exporter 服务已被删除。Kafka 导出器服务 **的模板**。
container	ContainerTemplate	Kafka Exporter 容器的模板。
serviceAccount	ResourceTemplate	Kafka Exporter 服务帐户的模板。

第 65 章 KAFKASTATUS 模式参考

使用于： Kafka

属性	属性类型	描述
conditions	**condition** 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
监听器	**ListenerStatus** 数组	内部和外部监听程序的地址。
kafkaNodePools	**UsedNodePoolStatus** 数组	此 Kafka 集群使用的 KafkaNodePools 列表。
clusterId	string	Kafka 集群 Id。
operatorLastSuccessfulVersion	string	Apache Kafka Cluster Operator 的 Streams 版本，执行最后一次成功协调。
kafkaVersion	string	当前在集群中部署的 Kafka 版本。
kafkaMetadataVersion	string	Kafka 集群当前使用的 KRaft metadata.version。
kafkaMetadataState	字符串([PreKRaft, ZooKeeper, KRaftMigration, KRaftDualWriting, KRaftPostMigration, KRaft] 之一)	定义存储集群元数据的位置。可能的值有：如果元数据存储在 ZooKeeper 中；如果控制器连接到 ZooKeeper，则 KRaftMigration 会在启用了 Zookeeper 迁移时滚动代理，并将连接信息发送到控制器，且元数据迁移过程正在运行；如果元数据迁移过程完成且集群处于双写模式；如果代理完全通过 KRaft-based，但控制器被部署到从 ZooKeeper 断开连接，KRaftPostMigration。 PreKRaft 如果代理和控制器完全基于 KRaft，则元数据存储在 KRaft 中，但必须删除 ZooKeeper；如果元数据存储在 KRaft 中，KRaft。

第 66 章 CONDITION 模式参考

用于： KafkaBridgeStatus, KafkaConnectorStatus, KafkaConnectStatus, KafkaMirrorMaker2Status, KafkaMirrorMakerStatus, KafkaNodePoolStatus, KafkaRebalanceStatus, kafkaStatus, KafkaTopicStatus, KafkaUserStatus, StrimziPodSetStatus

属性	属性类型	描述
type	string	条件的唯一标识符，用于区分资源中的其他条件。
status	string	条件的状态，可以是 True、False 或 Unknown。
lastTransitionTime	string	最后一次类型条件从一个状态变为另一个状态的时间。所需的格式为 UTC 时区为 'yyyy-MM-ddTHH:mm:ssZ'。
reason	string	条件最后一次转换的原因(CamelCase 中的单个单词)。
message	string	人类可读的消息，指示条件最后一次转换的详细信息。

第 67 章 LISTENERSTATUS 模式参考

用于： **KafkaStatus**

属性	属性类型	描述
type	string	**type 属性已弃用。type** 属性不再被使用。使用具有相同值的 **name** 属性。侦听器的名称。
name	string	侦听器的名称。
addresses	**ListenerAddress** 数组	此监听器的地址列表。
bootstrapServers	string	以逗号分隔的 **host:port** 对列表，用于使用此监听程序连接到 Kafka 集群。
证书	字符串数组	TLS 证书列表，可用于在连接到给定监听程序时验证服务器的身份。仅为 **tls** 和 **external** 监听程序设置。

第 68 章 LISTENERADDRESS 模式参考

使用于： ListenerStatus

属性	属性类型	描述
主机	string	Kafka bootstrap 服务的 DNS 名称或 IP 地址。
port	整数	Kafka bootstrap 服务的端口。

第 69 章 USEDNODEPOOLSTATUS 模式参考

用于： KafkaStatus

属性	属性类型	描述
name	string	此 Kafka 资源使用的 KafkaNodePool 的名称。

第 70 章 KAFKACONNECT 模式参考

属性	属性类型	描述
spec	**KafkaConnectSpec**	Kafka Connect 集群的规格。
status	**KafkaConnectStatus**	Kafka Connect 集群的状态。

第 71 章 KAFKACONNECTSPEC 模式参考

used in: KafkaConnect

KafkaConnectSpec 模式属性的完整列表

配置 Kafka Connect 集群。

71.1. CONFIG

使用 config 属性将 Kafka Connect 选项配置为密钥。

这些值可以是以下 JSON 类型之一：

- 字符串
- Number
- 布尔值

某些选项具有默认值：

- group.id 带有默认值 connect-cluster
- 带有默认值 connect-cluster-offsets 的 offset.storage.topic
- config.storage.topic 带有默认值 connect-cluster-configs
- status.storage.topic 带有默认值 connect-cluster-status

- key.converter 带有默认值 org.apache.kafka.connect.json.JsonConverter
- value.converter，默认值为 org.apache.kafka.connect.json.JsonConverter

如果 KafkaConnect.spec.config 属性中不存在这些选项，则这些选项会被自动配置。

例外

您可以指定并配置 Apache Kafka 文档 中列出的选项。

但是，Apache Kafka 的流负责配置和管理与以下内容相关的选项，而这无法更改：

- Kafka 集群 bootstrap 地址
- 安全性（加密、身份验证和授权）
- 侦听器和 REST 接口配置
- 插件路径配置

无法设置具有以下前缀的属性：

- bootstrap.servers
- consumer.interceptor.classes
- 监听器.

- plugin.path
- producer.interceptor.classes
- REST。
- SASL。
- 安全性。
- ssl.

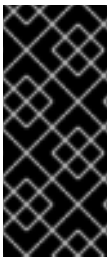
如果 config 属性包含一个无法更改的选项，它将被忽略，并将警告信息记录到 Cluster Operator 日志文件。所有其他支持选项都转发到 Kafka Connect，包括对流为 Apache Kafka 配置的选项的以下例外：

- 支持的 TLS 版本和密码套件的任何 ssl 配置

Kafka Connect 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
metadata:
  name: my-connect
spec:
  # ...
  config:
    group.id: my-connect-cluster
    offset.storage.topic: my-connect-cluster-offsets
    config.storage.topic: my-connect-cluster-configs
    status.storage.topic: my-connect-cluster-status
    key.converter: org.apache.kafka.connect.json.JsonConverter
    value.converter: org.apache.kafka.connect.json.JsonConverter
    key.converter.schemas.enable: true
    value.converter.schemas.enable: true
    config.storage.replication.factor: 3
```

```
    offset.storage.replication.factor: 3
    status.storage.replication.factor: 3
  # ...
```



重要

Cluster Operator 不会密钥或 config 对象中提供的值。如果提供了无效的配置，Kafka Connect 集群可能无法启动，或者可能会不稳定。在这种情况下，修复配置，以便 Cluster Operator 可将新配置部署到所有 Kafka Connect 节点。

71.2. LOGGING

Kafka Connect 具有自己的可配置的日志记录器：

- connect.root.logger.level
- log4j.logger.org.reflections

根据运行的 Kafka Connect 插件，添加了更多日志记录器。

使用 curl 请求从任何 Kafka 代理 pod 获取运行 Kafka Connect 日志记录器的完整列表：

```
curl -s http://<connect-cluster-name>-connect-api:8083/admin/loggers/
```

Kafka Connect 使用 Apache log4j 日志记录器实现。

使用 logging 属性配置日志记录器和日志记录器级别。

您可以通过直接指定日志记录器和级别（在线）或使用自定义（外部） ConfigMap 来设置日志级别。如果使用 ConfigMap，您可以将 logging.valueFrom.configMapKeyRef.name 属性设置为包含外部日志记录配置的 ConfigMap 的名称。在 ConfigMap 中，日志记录配置使用 log4j.properties 描述。logging.valueFrom.configMapKeyRef.name 和 logging.valueFrom.configMapKeyRef.key 属性都是强制的。使用指定的确切日志记录配置的 ConfigMap 会在 Cluster Operator 运行时使用自定义资源创建，然后在每次协调后重新创建。如果没有指定自定义 ConfigMap，则使用默认日志记录设置。如果

没有设置特定的日志记录器值，则会为该日志记录器继承上级日志记录器设置。有关日志级别的更多信息，请参阅 Apache 日志记录服务。

在这里，我们看到 内联 和 外部日志记录 的示例。inline 日志记录指定根日志记录器级别。您还可以通过将特定类或日志记录器添加到 loggers 属性来设置日志级别。

内联日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
spec:
  # ...
  logging:
    type: inline
    loggers:
      connect.root.logger.level: INFO
      log4j.logger.org.apache.kafka.connect.runtime.WorkerSourceTask: TRACE
      log4j.logger.org.apache.kafka.connect.runtime.WorkerSinkTask: DEBUG
  # ...
```

注意

将日志级别设置为 DEBUG 可能会导致大量日志输出，并可能会影响性能。

外部日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
spec:
  # ...
  logging:
    type: external
    valueFrom:
      configMapKeyRef:
        name: customConfigMap
        key: connect-logging.log4j
  # ...
```

任何未配置的可用日志记录器将其级别设置为 OFF。

如果使用 Cluster Operator 部署 Kafka Connect，则动态应用对 Kafka Connect 日志记录级别的更改。

如果使用外部日志记录，则会在日志附加程序更改时触发滚动更新。

垃圾收集器(GC)

垃圾回收收集器日志记录也可以使用 jvmOptions 属性启用 （或禁用）。

71.3. KAFKACONNECTSPEC 模式属性

属性	属性类型	描述
version	string	Kafka Connect 版本。默认为最新版本。请参阅用户文档了解升级或降级版本所需的流程。
replicas	整数	Kafka Connect 组中的 pod 数量。默认值为 **3**。
image	string	用于 Kafka Connect pod 的容器镜像。如果没有明确指定镜像名称，它会根据 **spec.version** 配置决定。镜像名称被专门映射到 Cluster Operator 配置中的对应版本。
bootstrapServers	string	要连接的引导服务器。这应该以逗号分隔的 *hostname*:_<port>_ 对列表提供。
tls	clientTLS	TLS 配置。

属性	属性类型	描述
身份验证	**KafkaClientAuthenticationTls**, **KafkaClientAuthenticationScramSha256**, **KafkaClientAuthenticationScramSha512**, **KafkaClientAuthenticationPlain**, **KafkaClientAuthenticationOAuth**	Kafka Connect 的身份验证配置。
config	map	Kafka Connect 配置。无法设置带有以下前缀的属性： ssl., sasl., security., listener, plugin.path, rest., bootstrap.servers, consumer.interceptor.classes, producer.interceptor.classes （例外：ssl.endpoint.identification.algorithm, ssl.cipher.suites, ssl.protocol, ssl.enabled.protocols）。
resources	ResourceRequirements	CPU 和内存资源和请求的初始资源的最大限制。
livenessProbe	**probe**	Pod 存活度检查。
readinessProbe	**probe**	Pod 就绪度检查。
jvmOptions	**JvmOptions**	pod 的 JVM 选项。
jmxOptions	**KafkaJmxOptions**	JMX 选项.
logging	**InlineLogging**, **ExternalLogging**	Kafka Connect 的日志记录配置。
clientRackInitImage	string	用于初始化 **client.rack** 的 init 容器的镜像。
rack	**Rack**	配置用作 **client.rack** 消费者配置的节点标签。
tracing	**jaegertracing**, **OpenTelemetryTracing**	在 Kafka Connect 中配置追踪。
模板	**KafkaConnectTemplate**	Kafka Connect 和 Kafka Mirror Maker 2 资源的模板。该模板允许用户指定如何生成 **Pod**、**Service** 和其他服务。

属性	属性类型	描述
externalConfiguration	**externalConfiguration**	将来自 Secret 或 ConfigMap 的数据传递给 Kafka Connect pod，并使用它们配置连接器。
build	**build**	配置如何构建 Connect 容器镜像。可选。
metricsConfig	**JmxPrometheusExporterMetrics**	指标配置。

第 72 章 CLIENTTLS 模式参考

used in: KafkaBridgeSpec, KafkaConnectSpec, KafkaMirrorMaker2ClusterSpec, KafkaMirrorMakerConsumerSpec, KafkaMirrorMakerProducerSpec

ClientTls 模式属性的完整列表

配置用于连接 KafkaConnect, KafkaBridge, KafkaMirror, KafkaMirrorMaker2 的 TLS 可信证书。

72.1. TRUSTEDCERTIFICATES

使用 trustedCertificates 属性提供 secret 列表。

72.2. CLIENT TLS 模式属性

属性	属性类型	描述
trustedCertificates	CertSecretSource 数组	TLS 连接的可信证书。

第 73 章 KAFKACLIENTAUTHENTICATIONTLS 模式参考

used in: KafkaBridgeSpec, KafkaConnectSpec, KafkaMirrorMaker2ClusterSpec, KafkaMirrorMakerConsumerSpec, KafkaMirrorMakerProducerSpec

KafkaClientAuthenticationTls 模式属性的完整列表

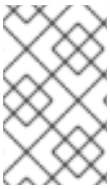
要配置 mTLS 身份验证，请将 type 属性设置为 tls。mTLS 使用 TLS 证书进行身份验证。

73.1. CERTIFICATEANDKEY

证书在 certificateAndKey 属性中指定，始终从 OpenShift secret 加载。在 secret 中，证书必须以 X509 格式存储在两个不同的密钥下：public 和 private。

您可以使用 User Operator 创建的 secret，或者您可以创建自己的 TLS 证书文件，使用用于身份验证的密钥，然后从文件创建 Secret ：

```
oc create secret generic MY-SECRET \
--from-file=MY-PUBLIC-TLS-CERTIFICATE-FILE.crt \
--from-file=MY-PRIVATE.key
```



注意

mTLS 身份验证只能用于 TLS 连接。

mTLS 配置示例

```
authentication:
  type: tls
  certificateAndKey:
    secretName: my-secret
    certificate: my-public-tls-certificate-file.crt
    key: private.key
```

73.2. KAFKACLIENTAUTHENTICATIONTLS 模式属性

type 属性是一种差异性，可区分来自 KafkaClientAuthenticationScramSha256 的 KafkaClientAuthenticationTls 类型，KafkaClient AuthenticationScramSha512, KafkaClientAuthenticationPlain, KafkaClientAuthenticationOAuth。对于类型 KafkaClientAuthenticationTls，它需要是值 tls。

属性	属性类型	描述
certificateAndKey	CertAndKeySecretSource	引用保存证书和私钥对的 **Secret**。
type	string	必须为 **tls**。

第 74 章 KAFKACLIENTAUTHENTICATIONSCRAMSHA256 SCHEMA REFERENCE

used in: KafkaBridgeSpec, KafkaConnectSpec, KafkaMirrorMaker2ClusterSpec, KafkaMirrorMakerConsumerSpec, KafkaMirrorMakerProducerSpec

KafkaClientAuthenticationScramSha256 模式属性的完整列表

要配置基于 SASL 的 SCRAM-SHA-256 身份验证，请将 type 属性设置为 scram-sha-256。SCRAM-SHA-256 身份验证机制需要一个用户名和密码。

74.1. USERNAME

在 username 属性中指定 username。

74.2. PASSWORDSECRET

在 passwordSecret 属性中，指定到包含密码的 Secret 的链接。

您可以使用 User Operator 创建的 secret。

如果需要，您可以创建一个包含密码（明文）的文本文件来进行验证：

```
echo -n PASSWORD > MY-PASSWORD.txt
```

然后，您可以从文本文件创建 Secret，为密码设置您自己的字段名称（密钥）：

```
oc create secret generic MY-CONNECT-SECRET-NAME --from-file=MY-PASSWORD-FIELD-NAME=./MY-PASSWORD.txt
```

Kafka Connect 的 SCRAM-SHA-256 客户端身份验证的 Secret 示例

```
apiVersion: v1
kind: Secret
metadata:
```

```
  name: my-connect-secret-name
type: Opaque
data:
  my-connect-password-field: LFTIyFRFIMmU2N2Tm
```

secretName 属性包含 Secret 的名称，password 属性包含密码存储在 Secret 中的键名称。

重要

不要在 password 属性中指定实际密码。

Kafka Connect 的基于 SASL 的 SCRAM-SHA-256 客户端身份验证配置示例

```
authentication:
  type: scram-sha-256
  username: my-connect-username
  passwordSecret:
    secretName: my-connect-secret-name
    password: my-connect-password-field
```

74.3. KAFKACLIENTAUTHENTICATIONSCRAMSHA256 SCHEMA PROPERTIES

属性	属性类型	描述
passwordSecret	PasswordSecretSource	对包含密码的 **Secret** 的引用。
type	string	必须为 **scram-sha-256**。
username	string	用于身份验证的用户名。

第 75 章 PASSWORDSECRETSOURCE 架构参考

用于： KafkaClientAuthenticationOAuth, KafkaClientAuthenticationPlain, KafkaClientAuthenticationScramSha256, KafkaClientAuthenticationScramSha512

属性	属性类型	描述
password	string	存储密码的 Secret 中的密钥名称。
secretName	string	包含密码的 Secret 名称。

第 76 章 KAFKACLIENTAUTHENTICATIONSCRAMSHA512 模式参考

used in: KafkaBridgeSpec, KafkaConnectSpec, KafkaMirrorMaker2ClusterSpec, KafkaMirrorMakerConsumerSpec, KafkaMirrorMakerProducerSpec

KafkaClientAuthenticationScramSha512 模式属性的完整列表

要配置基于 SASL 的 SCRAM-SHA-512 身份验证，请将 type 属性设置为 scram-sha-512。SCRAM-SHA-512 身份验证机制需要一个用户名和密码。

76.1. USERNAME

在 username 属性中指定 username。

76.2. PASSWORDSECRET

在 passwordSecret 属性中，指定到包含密码的 Secret 的链接。

您可以使用 User Operator 创建的 secret。

如果需要，您可以创建一个包含密码（明文）的文本文件来进行验证：

```
echo -n PASSWORD > MY-PASSWORD.txt
```

然后，您可以从文本文件创建 Secret，为密码设置您自己的字段名称（密钥）：

```
oc create secret generic MY-CONNECT-SECRET-NAME --from-file=MY-PASSWORD-FIELD-NAME=./MY-PASSWORD.txt
```

Kafka Connect 的 SCRAM-SHA-512 客户端身份验证的 Secret 示例

```
apiVersion: v1
kind: Secret
metadata:
```

```
  name: my-connect-secret-name
type: Opaque
data:
  my-connect-password-field: LFTIyFRFIMmU2N2Tm
```

secretName 属性包含 Secret 的名称，password 属性包含密码存储在 Secret 中的键名称。

重要

不要在 password 属性中指定实际密码。

Kafka Connect 基于 SASL 的 SCRAM-SHA-512 客户端身份验证配置示例

```
authentication:
  type: scram-sha-512
  username: my-connect-username
  passwordSecret:
    secretName: my-connect-secret-name
    password: my-connect-password-field
```

76.3. KAFKACLIENTAUTHENTICATIONSCRAMSHA512 模式属性

属性	属性类型	描述
passwordSecret	PasswordSecretSource	对包含密码的 Secret 的引用。
type	string	必须是 scram-sha-512。
username	string	用于身份验证的用户名。

第 77 章 KAFKACLIENTAUTHENTICATIONPLAIN 模式参考

used in: KafkaBridgeSpec, KafkaConnectSpec, KafkaMirrorMaker2ClusterSpec, KafkaMirrorMakerConsumerSpec, KafkaMirrorMakerProducerSpec

KafkaClientAuthenticationPlain 模式属性的完整列表

要配置基于 SASL 的 PLAIN 身份验证，请将 type 属性设置为 plain。SASL PLAIN 身份验证机制需要一个用户名和密码。

警告

SASL PLAIN 机制将以明文形式通过网络传输用户名和密码。如果启用了 TLS 加密，只有使用 SASL PLAIN 身份验证。

77.1. USERNAME

在 username 属性中指定 username。

77.2. PASSWORDSECRET

在 passwordSecret 属性中，指定到包含密码的 Secret 的链接。

您可以使用 User Operator 创建的 secret。

如果需要，创建一个包含密码（明文）的文本文件，用于身份验证：

```
echo -n PASSWORD > MY-PASSWORD.txt
```

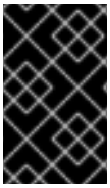
然后，您可以从文本文件创建 Secret，为密码设置您自己的字段名称（密钥）：

```
oc create secret generic MY-CONNECT-SECRET-NAME --from-file=MY-PASSWORD-FIELD-NAME=./MY-PASSWORD.txt
```

Kafka Connect 的 PLAIN 客户端身份验证的 Secret 示例

```
apiVersion: v1
kind: Secret
metadata:
  name: my-connect-secret-name
type: Opaque
data:
  my-password-field-name: LFTIyFRFlMmU2N2Tm
```

secretName 属性包含 Secret 的名称，password 属性包含密码存储在 Secret 中的键名称。



重要

不要在 password 属性中指定实际密码。

基于 SASL 的 PLAIN 客户端身份验证配置示例

```
authentication:
  type: plain
  username: my-connect-username
  passwordSecret:
    secretName: my-connect-secret-name
    password: my-password-field-name
```

77.3. KAFKACLIENTAUTHENTICATIONPLAIN 模式属性

type 属性是一种差异性，用于区分来自 KafkaClientAuthenticationTls, KafkaClientAuthenticationScramSha256, KafkaClientAuthenticationScramSha512, KafkaClientAuthenticationOAuth 的 KafkaClientAuthenticationPlain 类型。对于类型 KafkaClientAuthenticationPlain，它需要是值 plain。

属性	属性类型	描述
passwordSecret	PasswordSecretSource	对包含密码的 Secret 的引用。
type	string	必须为 纯。
username	string	用于身份验证的用户名。

第 78 章 KAFKACLIENTAUTHENTICATIONOAUTH 模式参考

used in: KafkaBridgeSpec, KafkaConnectSpec, KafkaMirrorMaker2ClusterSpec, KafkaMirrorMakerConsumerSpec, KafkaMirrorMakerProducerSpec

KafkaClientAuthenticationOAuth 模式属性的完整列表

要配置 OAuth 客户端身份验证，请将 type 属性设置为 oauth。

可以使用以下选项之一配置 OAuth 身份验证：

- 客户端 ID 和 secret
- 客户端 ID 和刷新令牌
- 访问令牌
- 用户名和密码
- TLS

客户端 ID 和 secret

您可以在 tokenEndpointUri 属性中配置授权服务器的地址，以及身份验证中使用的客户端 ID 和客户端 secret。OAuth 客户端将连接到 OAuth 服务器，使用客户端 ID 和 secret 进行身份验证，并获取用于与 Kafka 代理进行身份验证的访问令牌。在 clientSecret 属性中，指定到包含客户端 secret 的 Secret 的链接。

使用客户端 ID 和客户端 secret 的 OAuth 客户端身份验证示例

```
authentication:
  type: oauth
  tokenEndpointUri: https://sso.myproject.svc:8443/auth/realms/internal/protocol/openid-connect/token
```

```
  clientId: my-client-id
  clientSecret:
    secretName: my-client-oauth-secret
    key: client-secret
```

如果需要，可以指定 scope 今儿 audience。

客户端 ID 和刷新令牌

您可以在 tokenEndpointUri 属性中配置 OAuth 服务器的地址，以及 OAuth 客户端 ID 和刷新令牌。OAuth 客户端将连接到 OAuth 服务器，使用客户端 ID 进行身份验证并刷新令牌，并获取用于与 Kafka 代理进行身份验证的访问令牌。在 refreshToken 属性中，指定到包含刷新令牌的 Secret 的链接。

使用客户端 ID 和刷新令牌的 OAuth 客户端身份验证示例

```
authentication:
  type: oauth
  tokenEndpointUri: https://sso.myproject.svc:8443/auth/realms/internal/protocol/openid-
connect/token
  clientId: my-client-id
  refreshToken:
    secretName: my-refresh-token-secret
    key: refresh-token
```

访问令牌

您可以配置用于直接与 Kafka 代理进行身份验证的访问令牌。在这种情况下，您没有指定 tokenEndpointUri。在 accessToken 属性中，指定到包含访问令牌的 Secret 的链接。

仅使用访问令牌的 OAuth 客户端身份验证示例

```
authentication:
  type: oauth
  accessToken:
    secretName: my-access-token-secret
    key: access-token
```

用户名和密码

OAuth 用户名和密码配置使用 OAuth *Resource Owner Password Grant* 机制。机制已弃用，且只支持在无法使用客户端凭证(ID 和 secret)的环境中启用集成。如果您的访问管理系统不支持其他方法或用户帐户进行身份验证，您可能需要使用用户帐户。

典型的方法是在您的授权服务器中创建一个代表您的客户端应用程序的特殊用户帐户。然后，为帐户提供随机生成的密码，并且有非常有限的权限集。例如，帐户只能连接到您的 Kafka 集群，但不允许使用任何其他服务或登录到用户界面。

考虑首先使用刷新令牌机制。

您可以在 tokenEndpointUri 属性中配置授权服务器的地址，以及客户端 ID、用户名以及身份验证中使用的密码。OAuth 客户端将连接到 OAuth 服务器，使用用户名、密码、客户端 ID 进行验证，甚至客户端 secret 来获取它将用来与 Kafka 代理进行身份验证的访问令牌。

在 passwordSecret 属性中，指定到包含密码的 Secret 的链接。

通常，您还必须使用公共 OAuth 客户端配置 clientId。如果使用机密 OAuth 客户端，还必须配置 clientSecret。

使用带有公共客户端的用户名和密码的 OAuth 客户端身份验证示例

```
authentication:
  type: oauth
  tokenEndpointUri: https://sso.myproject.svc:8443/auth/realms/internal/protocol/openid-
connect/token
  username: my-username
  passwordSecret:
    secretName: my-password-secret-name
    password: my-password-field-name
  clientId: my-public-client-id
```

使用用户名和密码和机密客户端进行 OAuth 客户端身份验证的示例

```
authentication:
  type: oauth
  tokenEndpointUri: https://sso.myproject.svc:8443/auth/realms/internal/protocol/openid-
connect/token
  username: my-username
  passwordSecret:
    secretName: my-password-secret-name
    password: my-password-field-name
  clientId: my-confidential-client-id
  clientSecret:
    secretName: my-confidential-client-oauth-secret
    key: client-secret
```

如果需要，可以指定 scope 今儿 audience。

TLS

使用 HTTPS 协议访问 OAuth 服务器不需要任何其他配置，只要它使用的 TLS 证书由可信证书颁发机构签名，并且其主机名列在证书中。

如果您的 OAuth 服务器使用自签名证书，或者由不被信任的证书颁发机构签名，您可以在自定义资源中配置可信证书列表。tlsTrustedCertificates 属性包含存储证书的密钥名称的 secret 列表。证书必须以 X509 格式存储。

提供的 TLS 证书示例

```
authentication:
  type: oauth
  tokenEndpointUri: https://sso.myproject.svc:8443/auth/realms/internal/protocol/openid-
connect/token
  clientId: my-client-id
  refreshToken:
    secretName: my-refresh-token-secret
    key: refresh-token
  tlsTrustedCertificates:
    - secretName: oauth-server-ca
      certificate: tls.crt
```

OAuth 客户端默认将验证 OAuth 服务器的主机名是否与证书主题或其中一个替代 DNS 名称匹配。如果不需要，您可以禁用主机名验证。

禁用 TLS 主机名验证示例

```
authentication:
  type: oauth
  tokenEndpointUri: https://sso.myproject.svc:8443/auth/realms/internal/protocol/openid-connect/token
  clientId: my-client-id
  refreshToken:
    secretName: my-refresh-token-secret
    key: refresh-token
  disableTlsHostnameVerification: true
```

78.1. KAFKACLIENTAUTHENTICATIONOAUTH 模式属性

type 属性是一个差异性程序，它区分来自 KafkaClientAuthenticationTls, KafkaClientAuthenticationScramSha256, KafkaClientAuthenticationScramSha512, KafkaClientAuthenticationPlain 的 KafkaClientAuthenticationOAuth 类型。对于类型 KafkaClientAuthenticationOAuth，它需要是值 oauth。

属性	属性类型	描述
accessToken	GenericSecretSource	指向包含从授权服务器获取的访问令牌的 OpenShift Secret。
accessTokenIsJwt	布尔值	配置访问令牌是否应被视为 JWT。如果授权服务器返回不透明令牌，则这应设为 **false**。默认值为 **true**。
受众	string	对授权服务器进行身份验证时使用的 OAuth 受众。有些授权服务器需要明确设置 audience。可能的值取决于授权服务器的配置方式。默认情况下，在执行令牌端点请求时，不会指定 **audience**。

属性	属性类型	描述
clientId	string	Kafka 客户端 ID，用于向 OAuth 服务器进行身份验证并使用令牌端点 URI。
clientSecret	**GenericSecretSource**	链接到包含 Kafka 客户端 secret 的 OpenShift Secret，用于向 OAuth 服务器进行身份验证并使用令牌端点 URI。
connectTimeoutSeconds	整数	连接到授权服务器时的连接超时（以秒为单位）。如果没有设置，则有效的连接超时为 60 秒。
disableTlsHostnameVerification	布尔值	启用或禁用 TLS 主机名验证。默认值为 **false**。
enableMetrics	布尔值	启用或禁用 OAuth 指标。默认值为 **false**。
httpRetries	整数	初始 HTTP 请求失败时尝试的最大重试次数。如果没有设置，则默认为不尝试任何重试。
httpRetryPauseMs	整数	重试失败的 HTTP 请求前需要暂停。如果没有设置，则默认为根本不暂停，而是立即重复请求。
includeAcceptHeader	布尔值	Accept 标头是否应该在请求中设置到授权服务器。默认值为 **true**。
maxTokenExpirySeconds	整数	将访问令牌的生存时间设置为指定秒数。如果授权服务器返回不透明令牌，则应设置此项。
passwordSecret	**PasswordSecretSource**	对包含密码的 **Secret** 的引用。
readTimeoutSeconds	整数	连接到授权服务器时读取超时（以秒为单位）。如果没有设置，则有效读取超时为 60 秒。
refreshToken	**GenericSecretSource**	连接到包含刷新令牌的 OpenShift Secret，可用于从授权服务器获取访问令牌。
scope	string	对授权服务器进行身份验证时要使用的 OAuth 范围。有些授权服务器需要此设置。可能的值取决于授权服务器的配置方式。在执行令牌端点请求时，不指定 **范围**。
tlsTrustedCertificates	**CertSecretSource** 数组	用于 TLS 连接到 OAuth 服务器的可信证书。
tokenEndpointUri	string	授权服务器令牌端点 URI。

属性	属性类型	描述
type	string	必须是 **oauth**。
username	string	用于身份验证的用户名。

第 79 章 JAEGERTRACING 模式参考

类型 JaegerTracing 已被弃用。

used in: KafkaBridgeSpec, KafkaConnectSpec, KafkaMirrorMaker2Spec, KafkaMirrorMakerSpec

type 属性是一个辨别器，可区分来自 OpenTelemetryTracing 的 JaegerTracing 类型。对于类型 JaegerTracing，它需要是值 jaeger。

属性	属性类型	描述
type	string	必须为 **jaeger**。

第 80 章 OPENTELEMETRYTRACING 模式参考

used in: KafkaBridgeSpec, KafkaConnectSpec, KafkaMirrorMaker2Spec, KafkaMirrorMakerSpec

type 属性是一个辨别器，可区分 JaegerTracing 中的 OpenTelemetryTracing 类型。对于类型 OpenTelemetryTracing，它需要值 opentelemetry。

属性	属性类型	描述
type	string	必须为 **opentelemetry**。

第 81 章 KAFKACONNECTTEMPLATE 模式参考

used in: KafkaConnectSpec, KafkaMirrorMaker2Spec

属性	属性类型	描述
部署	DeploymentTemplate	**deployment 属性已弃用。**Kafka Connect 和 MirrorMaker 2 操作对象不再使用 **Deployment** 资源。此字段将被忽略。Kafka Connect **Deployment** 的模板。
podSet	ResourceTemplate	Kafka Connect **StrimziPodSet** 资源的模板。
pod	PodTemplate	Kafka Connect **Pod 的模板**。
apiService	InternalServiceTemplate	Kafka Connect API **Service** 的模板。
headlessService	InternalServiceTemplate	Kafka Connect 无头服务 **的模板**。
connectContainer	ContainerTemplate	Kafka Connect 容器的模板。
initContainer	ContainerTemplate	Kafka init 容器的模板。
podDisruptionBudget	PodDisruptionBudgetTemplate	Kafka Connect **PodDisruptionBudget** 的模板。
serviceAccount	ResourceTemplate	Kafka Connect 服务帐户的模板。
clusterRoleBinding	ResourceTemplate	Kafka Connect ClusterRoleBinding 的模板。
buildPod	PodTemplate	Kafka Connect 构建 **Pod 的模板**。构建 Pod 仅适用于 OpenShift。
buildContainer	ContainerTemplate	Kafka Connect Build 容器的模板。构建容器仅在 OpenShift 上使用。
buildConfig	BuildConfigTemplate	用于构建新容器镜像的 Kafka Connect BuildConfig 模板。BuildConfig 仅适用于 OpenShift。
buildServiceAccount	ResourceTemplate	Kafka Connect Build 服务帐户的模板。
jmxSecret	ResourceTemplate	Kafka Connect Cluster JMX 身份验证的 Secret 模板。

第 82 章 BUILDCONFIGTEMPLATE 模式参考

used in: KafkaConnectTemplate

属性	属性类型	描述
metadata	MetadataTemplate	应用到 **PodDisruptionBudgetTemplate** 资源的元数据。
pullSecret	string	带有用于拉取基础镜像的凭证的 Container Registry Secret。

第 83 章 EXTERNALCONFIGURATION 模式参考

used in: KafkaConnectSpec, KafkaMirrorMaker2Spec

ExternalConfiguration 模式属性的完整列表

配置外部存储属性，以定义 Kafka Connect 连接器的配置选项。

您可以将 ConfigMap 或 Secret 作为环境变量或卷挂载到 Kafka Connect pod 中。卷和环境变量在 KafkaConnect.spec 或 KafkaMirrorMaker2.spec 中的 externalConfiguration 属性中配置。

应用时，环境变量和卷可在开发连接器时使用。

如需更多信息，请参阅 从外部来源加载配置值。

83.1. EXTERNALCONFIGURATION 模式属性

属性	属性类型	描述
env	**ExternalConfigurationEnv** 数组	在 Kafka Connect pod 中作为环境变量提供 Secret 或 ConfigMap 的数据。
卷	**ExternalConfigurationVolumeSource** 数组	在 Kafka Connect pod 中作为卷提供 Secret 或 ConfigMap 的数据。

第 84 章 EXTERNALCONFIGURATIONENV 模式参考

用于： ExternalConfiguration

属性	属性类型	描述
name	string	将传递给 Kafka Connect pod 的环境变量名称。环境变量的名称不能以 **KAFKA_** 或 **STRIMZI_** 开头。
valueFrom	ExternalConfigurationEnvVarSource	将传递给 Kafka Connect pod 的环境变量值。它可以作为 Secret 或 ConfigMap 字段的引用传递。该字段必须正好指定一个 Secret 或 ConfigMap。

第 85 章 EXTERNALCONFIGURATIONENVVARSOURCE 模式参考

用于： **ExternalConfigurationEnv**

属性	属性类型	描述
configMapKeyRef	ConfigMapKeySelector	引用 ConfigMap 中的键。
secretKeyRef	SecretKeySelector	引用 Secret 中的密钥。

第 86 章 EXTERNALCONFIGURATIONVOLUMESOURCE 模式参考

用于： ExternalConfiguration

属性	属性类型	描述
configMap	ConfigMapVolumeSource	引用 ConfigMap 中的键。必须指定一个 Secret 或 ConfigMap。
name	string	要添加到 Kafka Connect pod 的卷名称。
secret	SecretVolumeSource	引用 Secret 中的密钥。必须指定一个 Secret 或 ConfigMap。

第 87 章 BUILD 架构参考

used in: KafkaConnectSpec

Build 架构属性的完整列表

为 Kafka Connect 部署配置额外的连接器。

87.1. OUTPUT

要使用其他连接器插件构建新容器镜像，Apache Kafka 的 Streams 需要可推送镜像的容器 registry，并从中拉取。Apache Kafka 的流不会运行自己的容器 registry，因此必须提供 registry。Apache Kafka 的流支持私有容器 registry 以及 Quay 或 Docker Hub 等公共 registry。容器 registry 在 KafkaConnect 自定义资源的 .spec.build.output 部分中配置。需要 的输出 配置支持两种类型： docker 和 imagestream。

使用 Docker registry

要使用 Docker registry，您必须将 type指定为 docker，image 字段为新容器镜像的完整名称。全名必须包含：

- registry 的地址
- 端口号（如果侦听非标准端口）
- 新容器镜像的标签

有效容器镜像名称示例：

- docker.io/my-org/my-image/my-tag
- quay.io/my-org/my-image/my-tag

- image-registry.image-registry.svc:5000/myproject/kafka-connect-build:latest

每个 Kafka Connect 部署都必须使用单独的镜像，这可能意味着最基本的级别的不同标签。

如果 registry 需要身份验证，请使用 pushSecret 使用 registry 凭证设置 Secret 名称。对于 Secret，请使用 kubernetes.io/dockerconfigjson 类型和一个 .dockerconfigjson 文件来包含 Docker 凭证。有关从私有 registry 中拉取镜像的更多信息，请参阅 基于现有 Docker 凭证创建 Secret。

output 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
metadata:
  name: my-connect-cluster
spec:
  #...
  build:
    output:
      type: docker 1
      image: my-registry.io/my-org/my-connect-cluster:latest 2
      pushSecret: my-registry-credentials 3
  #...
```

1 （必需） Streams 用于 Apache Kafka 的输出类型。

2 （必需）所用的镜像的全名，包括存储库和标签。

3 （可选）使用容器 registry 凭证的 secret 名称。

Using OpenShift ImageStream

您可以使用 OpenShift ImageStream 存储新的容器镜像，而不是 Docker。在部署 Kafka 连接前，必须手动创建 ImageStream。要使用 ImageStream，将 type 设置为 imagestream，并使用 image 属性

指定 ImageStream 的名称和使用的标签。例如，my-connect-image-stream:latest。

output 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
metadata:
  name: my-connect-cluster
spec:
  #...
  build:
    output:
      type: imagestream 1
      image: my-connect-build:latest 2
  #...
```

1

（必需） Streams 用于 Apache Kafka 的输出类型。

2

（必需）ImageStream 和 tag 的名称。

87.2. PLUGINS

连接器插件是一组文件，用于定义连接到某些类型的外部系统所需的实施。容器镜像所需的连接器插件必须使用 KafkaConnect 自定义资源的 .spec.build.plugins 属性进行配置。每个连接器插件都必须有一个在 Kafka Connect 部署中唯一的名称。另外，必须列出插件工件。这些工件由 Apache Kafka 的 Streams 下载，添加到新容器镜像中，并在 Kafka Connect 部署中使用。连接器插件工件也可以包含其他组件，如(de) serializers。每个连接器插件都下载到一个单独的目录中，以便不同的连接器和它们的依赖项被正确地*进行了沙盒处理*。每个插件必须配置至少一个 工件。

带有两个连接器插件的插件配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
metadata:
  name: my-connect-cluster
```

```
spec:
  #...
  build:
    output:
      #...
    plugins: 1
      - name: connector-1
        artifacts:
          - type: tgz
            url: <url_to_download_connector_1_artifact>
            sha512sum: <SHA-512_checksum_of_connector_1_artifact>
      - name: connector-2
        artifacts:
          - type: jar
            url: <url_to_download_connector_2_artifact>
            sha512sum: <SHA-512_checksum_of_connector_2_artifact>
  #...
```

1

（必需）连接器插件及其工件列表。

Apache Kafka 的流支持以下工件类型：

- 直接使用的 JAR 文件
- TGZ 归档（已下载并解包）
- ZIP 存档，它们已下载并解包
- Maven 工件（使用 Maven 协调）
- 其他直接下载和使用的工件

重要

Apache Kafka 的流不会对下载工件进行任何安全扫描。为了安全起见，您应该首先手动验证工件，并配置 checksum 验证，以确保在自动构建和 Kafka Connect 部署中使用相同的工件。

使用 JAR 工件

JAR 工件代表一个 JAR 文件，该文件已下载并添加到容器镜像中。要使用 JAR 工件，请将 type 属性设置为 jar，并使用 url 属性指定下载位置。

另外，您可以指定工件的 SHA-512 校验和。如果指定了，Apache Kafka 的 Streams 会在构建新容器镜像时验证工件的校验和。

JAR 工件示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
metadata:
  name: my-connect-cluster
spec:
  #...
  build:
    output:
      #...
    plugins:
      - name: my-plugin
        artifacts:
          - type: jar 1
            url: https://my-domain.tld/my-jar.jar 2
            sha512sum: 589...ab4 3
          - type: jar
            url: https://my-domain.tld/my-jar2.jar
  #...
```

1

（必需）工件的类型。

2

（必需）下载工件的 URL。

3

（可选） SHA-512 校验和来验证工件。

使用 TGZ 工件

TGZ 工件用于下载使用 Gzip 压缩压缩的 TAR 存档。TGZ 工件可以包含整个 Kafka Connect 连接器，即使由多个不同文件组成。在构建新容器镜像时，TGZ 工件由 Streams for Apache Kafka 自动下载并解包。要使用 TGZ 工件，请将 type 属性设置为 tgz，并使用 url 属性指定下载位置。

另外，您可以指定工件的 SHA-512 校验和。如果指定了，Apache Kafka 的 Streams 会在解包并构建新容器镜像前验证 checksum。

TGZ 工件示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
metadata:
  name: my-connect-cluster
spec:
  #...
  build:
    output:
      #...
    plugins:
      - name: my-plugin
        artifacts:
          - type: tgz 1
            url: https://my-domain.tld/my-connector-archive.tgz 2
            sha512sum: 158...jg10 3
  #...
```

1

（必需）工件的类型。

2

3

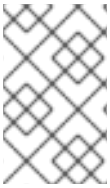
（可选） SHA-512 校验和来验证工件。

使用 ZIP 工件

ZIP 工件用于下载 ZIP 压缩存档。使用 ZIP 工件的方式与上一节中描述的 TGZ 工件相同。唯一的区别是指定 type: zip 而不是 type: tgz。

使用 Maven 工件

Maven 工件用于指定连接器插件工件作为 Maven 协调。Maven 协调识别插件工件和依赖项，以便可以从 Maven 存储库获取它们。



注意

连接器构建过程必须可以访问 Maven 存储库，才能将工件添加到容器镜像中。

Maven 工件示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
metadata:
  name: my-connect-cluster
spec:
  #...
  build:
    output:
      #...
    plugins:
      - name: my-plugin
        artifacts:
          - type: maven 1
            repository: https://mvnrepository.com 2
            group: <maven_group> 3
            artifact: <maven_artifact> 4
            version:  <maven_version_number> 5
  #...
```

1

（必需）工件的类型。

2

（可选）要从中下载工件的 Maven 存储库。如果没有指定存储库，则默认使用 Maven Central 存储库。

3

（必需）Maven 组 ID。

4

（必需）Maven 工件类型。

5

（必需）Maven 版本号。

使用其他 工件

其他 工件代表下载并添加到容器镜像中的任何类型的文件。如果要在生成的容器镜像中为工件使用特定名称，请使用 fileName 字段。如果没有指定文件名，则该文件会根据 URL 哈希命名。

另外，您可以指定工件的 SHA-512 校验和。如果指定了，Apache Kafka 的 Streams 会在构建新容器镜像时验证工件的校验和。

other 工件示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnect
metadata:
  name: my-connect-cluster
spec:
  #...
  build:
    output:
      #...
    plugins:
      - name: my-plugin
        artifacts:
          - type: other 1
```

```
        url: https://my-domain.tld/my-other-file.ext 2
        sha512sum: 589...ab4 3
        fileName: name-the-file.ext 4
#...
```

1

（必需）工件的类型。

2

（必需）下载工件的 URL。

3

（可选） SHA-512 校验和来验证工件。

4

（可选）文件存储在生成的容器镜像中的名称。

87.3. BUILD 架构属性

属性	属性类型	描述
output	**DockerOutput**, **ImageStreamOutput**	配置应存储新构建的镜像的位置。必需。
resources	ResourceRequirements	为构建保留的 CPU 和内存资源。
plugins	**插件** 数组	应该添加到 Kafka 连接中的连接器插件列表。必需。

第 88 章 DOCKEROUTPUT 模式参考

使用于： Build

type 属性是一个辨别器，可区分来自 ImageStreamOutput 的 DockerOutput 类型的使用。对于类型 DockerOutput，它需要是 docker 值。

属性	属性类型	描述
image	string	用于标记和推送新构建镜像的全名。例如 **quay.io/my-organization/my-custom-connect:latest**。必需。
pushSecret	string	带有用于推送新构建的镜像的凭证的 Container Registry Secret。
additionalKanikoOptions	字符串数组	配置在构建新连接镜像时将传递给 Kaniko executor 的附加选项。允许的选项包括：--customPlatform, --insecure, --insecure-pull, --insecure-registry, --log-format, --log-timestamp, --registry-mirror, --reproducible, --single-snapshot, --skip-tls-verify, --skip-tls-verify-pull, --skip-tls-verify-registry, --verbosity, --snapshotMode, --use-new-run.这些选项将仅用于使用 Kaniko executor 的 OpenShift。OpenShift 中会忽略它们。这些选项在 Kaniko GitHub 存储库中描述。更改此字段不会触发 Kafka Connect 镜像的新构建。
type	string	必须是 **docker**。

第 89 章 IMAGESTREAMOUTPUT 模式参考

使用于： Build

type 属性是一个辨别器，可区分来自 DockerOutput 的 ImageStreamOutput 类型的使用。类型 ImageStreamOutput 的值需要是 imagestream。

属性	属性类型	描述
image	string	推送新构建镜像的 ImageStream 的名称和标签。例如 **my-custom-connect:latest**。必需。
type	string	必须是 **镜像流**。

第 90 章 PLUGIN 架构参考

使用于： Build

属性	属性类型	描述
name	string	连接器插件的唯一名称。将用于生成要存储连接器工件的路径。名称在 KafkaConnect 资源中必须是唯一的。名称必须遵循以下模式：**^[a-z][-_a-z0-9]*[a-z]$**。必需。
工件	**JarArtifact**, **TgzArtifact**, **ZipArtifact**, MavenArtifact , **OtherArtifact** 数组	属于此连接器插件的工件列表。必需。

第 91 章 JARARTIFACT 模式参考

使用于：Plugin

属性	属性类型	描述
url	string	下载的工件的 URL。Apache Kafka 的流不会对下载工件的任何安全扫描。为安全起见，您应该首先手动验证工件并配置校验和验证，以确保在自动构建中使用相同的工件。**jar**、c**zip**、**tgz 和其他** 工件是必需的。不适用于 **maven** 工件类型。
sha512sum	string	工件的 SHA512 校验和。可选。如果指定，在构建新容器时将验证校验和。如果没有指定，则不会验证下载的工件。不适用于 **maven** 工件类型。
insecure	布尔值	默认情况下，验证使用 TLS 的连接，以检查它们是否安全。使用的服务器证书必须是有效、可信且包含服务器名称。通过将此选项设置为 **true**，则禁用所有 TLS 验证，并且下载工件，即使服务器被视为不安全的。
type	string	必须是 **jar**。

第 92 章 TGZARTIFACT 模式参考

使用于：Plugin

属性	属性类型	描述
url	string	下载的工件的 URL。Apache Kafka 的流不会对下载工件的任何安全扫描。为安全起见，您应该首先手动验证工件并配置校验和验证，以确保在自动构建中使用相同的工件。**jar**、**zip**、**tgz 和其他** 工件是必需的。不适用于 **maven** 工件类型。
sha512sum	string	工件的 SHA512 校验和。可选。如果指定，在构建新容器时将验证校验和。如果没有指定，则不会验证下载的工件。不适用于 **maven** 工件类型。
insecure	布尔值	默认情况下，验证使用 TLS 的连接，以检查它们是否安全。使用的服务器证书必须是有效、可信且包含服务器名称。通过将此选项设置为 **true**，则禁用所有 TLS 验证，并且下载工件，即使服务器被视为不安全的。
type	string	必须为 **tgz**。

第 93 章 ZIPARTIFACT 模式参考

使用于：Plugin

属性	属性类型	描述
url	string	下载的工件的 URL。Apache Kafka 的流不会对下载工件的任何安全扫描。为安全起见，您应该首先手动验证工件并配置校验和验证，以确保在自动构建中使用相同的工件。**jar**、c**zip**、**tgz 和其他** 工件是必需的。不适用于 **maven** 工件类型。
sha512sum	string	工件的 SHA512 校验和。可选。如果指定，在构建新容器时将验证校验和。如果没有指定，则不会验证下载的工件。不适用于 **maven** 工件类型。
insecure	布尔值	默认情况下，验证使用 TLS 的连接，以检查它们是否安全。使用的服务器证书必须是有效、可信且包含服务器名称。通过将此选项设置为 **true**，则禁用所有 TLS 验证，并且下载工件，即使服务器被视为不安全的。
type	string	必须为 **zip**。

第 94 章 MAVENARTIFACT 模式参考

使用于：Plugin

type 属性是一个辨别器，可以区分来自 JarArtifact, TgzArtifact, ZipArtifact, OtherArtifact 的 MavenArtifact 类型的使用。类型 MavenArtifact 的值需要是 maven。

属性	属性类型	描述
软件仓库	string	从中下载工件的 Maven 存储库。仅适用于 **maven** 工件类型。
group	string	Maven 组 ID.仅适用于 **maven** 工件类型。
工件	string	Maven 工件 ID.仅适用于 **maven** 工件类型。
version	string	Maven 版本号。仅适用于 **maven** 工件类型。
insecure	布尔值	默认情况下，验证使用 TLS 的连接，以检查它们是否安全。使用的服务器证书必须是有效、可信且包含服务器名称。通过将此选项设置为 **true**，则禁用所有 TLS 验证，并且下载工件，即使服务器被视为不安全的。
type	string	必须为 **maven**。

第 95 章 OTHERARTIFACT 模式参考

使用于：Plugin

属性	属性类型	描述
url	string	下载的工件的 URL。Apache Kafka 的流不会对下载工件的任何安全扫描。为安全起见，您应该首先手动验证工件并配置校验和验证，以确保在自动构建中使用相同的工件。**jar**、c**zip**、**tgz 和其他** 工件是必需的。不适用于 **maven** 工件类型。
sha512sum	string	工件的 SHA512 校验和。可选。如果指定，在构建新容器时将验证校验和。如果没有指定，则不会验证下载的工件。不适用于 **maven** 工件类型。
fileName	string	工件要存储的名称。
insecure	布尔值	默认情况下，验证使用 TLS 的连接，以检查它们是否安全。使用的服务器证书必须是有效、可信且包含服务器名称。通过将此选项设置为 **true**，则禁用所有 TLS 验证，并且下载工件，即使服务器被视为不安全的。
type	string	必须为 **其他**。

第 96 章 KAFKACONNECTSTATUS SCHEMA 参考

used in: KafkaConnect

属性	属性类型	描述
conditions	**condition** 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
url	string	管理和监控 Kafka Connect 连接器的 REST API 端点的 URL。
connectorPlugins	**ConnectorPlugin** 数组	此 Kafka Connect 部署中可用的连接器插件列表。
labelSelector	string	提供此资源的 pod 的标签选择器。
replicas	整数	用于提供此资源的当前 pod 数量。

第 97 章 CONNECTORPLUGIN 模式参考

用于： KafkaConnectStatus, KafkaMirrorMaker2Status

属性	属性类型	描述
type	string	连接器插件的类型。可用的类型是 **sink** 和 **source**。
version	string	连接器插件的版本。
class	string	连接器插件的类。

第 98 章 KAFKATOPIC 模式参考

属性	属性类型	描述
spec	**KafkaTopicSpec**	主题的规格。
status	**KafkaTopicStatus**	主题的状态。

第 99 章 KAFKATOPICSPEC 模式参考

使用于： KafkaTopic

属性	属性类型	描述
分区	整数	主题应具有的分区数量。这在创建主题后无法减少。它可能会在主题创建后增加，但务必要了解具有的结果，特别是语义分区的主题。如果不存在，则默认使用 **num.partitions** 的代理配置。
replicas	整数	主题应具有的副本数。如果不存在，则默认为 **default.replication.factor** 的代理配置。
config	map	主题配置。
topicName	string	主题的名称。如果不存在，则默认为主题的 metadata.name。建议不要设置此项，除非主题名称不是有效的 OpenShift 资源名称。

第 100 章 KAFKATOPICSTATUS 模式参考

使用于： **KafkaTopic**

属性	属性类型	描述
conditions	**condition** 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
topicName	string	主题名称。
topicId	string	主题的 id。对于具有 ready 条件的 KafkaTopic，只有在主题被删除并使用相同的名称重新创建时才更改。
replicasChange	**ReplicasChangeStatus**	复制因子更改状态。

第 101 章 REPLICASCHANGESTATUS 模式参考

用于： KafkaTopicStatus

属性	属性类型	描述
targetReplicas	整数	用户请求的目标副本数。当更改持续时，这可能与 .spec.replicas 不同。
state	字符串([ongoing, pending]之一)	副本更改操作的当前状态。当请求更改或 **持续** 时，当更改成功提交到 Cruise Control 时，这可能是 **待处理的**。
message	string	与副本更改请求相关的用户的消息。这可能包含在定期协调时可能会消失的临时错误消息。
sessionId	string	副本的会话标识符会更改与此 KafkaTopic 资源相关的请求。Topic Operator 用来跟踪 **持续** 副本更改操作的状态。

第 102 章 KAFKAUSER 模式参考

属性	属性类型	描述
spec	KafkaUserSpec	用户的规格。
status	KafkaUserStatus	Kafka 用户的状态。

第 103 章 KAFKAUSERSPEC 模式参考

使用于： KafkaUser

属性	属性类型	描述
身份验证	KafkaUserTlsClientAuthentication, KafkaUserTlsExternalClientAuthentication, KafkaUserScramSha512ClientAuthentication	为此 Kafka 用户启用验证机制。支持的身份验证机制包括 **scram-sha-512**、**tls** 和 **tls-external**。 • SCRAM **-sha-512** 生成带有 SASL SCRAM-SHA-512 凭证的 secret。 • **TLS** 为 mutual TLS 身份验证生成包含用户证书的 secret。 • **tls-external** 不生成用户证书。但是，使用在 User Operator 外部生成的用户证书，准备用户使用 mutual TLS 身份验证。为此用户设置的 ACL 和配额采用 **CN=<username>** 格式进行配置。 身份验证是可选的。如果没有配置身份验证，则不会生成凭证。为用户设置的 ACL 和配额以 < **username&** gt; 格式进行配置。
授权	KafkaUserAuthorizationSimple	此 Kafka 用户的授权规则。
配额	KafkaUserQuotas	控制客户端使用的代理资源的请求的配额。可以强制使用网络带宽和请求率配额。Kafka 用户配额文档可在 http://kafka.apache.org/documentation/#design_quotas 中找到。
模板	KafkaUserTemplate	模板来指定 Kafka 用户 **Secret 的** 生成方式。

第 104 章 KAFKAUSERTLSCLIENTAUTHENTICATION 模式参考

用于： KafkaUserSpec

type 属性是一个辨别器，它区分来自 KafkaUserTlsExternalClientAuthentication，KafkaUserScramSha512ClientAuthentication 的 KafkaUserTlsClientAuthentication 类型的使用。对于类型 KafkaUserTlsClientAuthentication，它需要是值 tls。

属性	属性类型	描述
type	string	必须为 **tls**。

第 105 章 KAFKAUSERTLSEXTERNALCLIENTAUTHENTICATION SCHEMA REFERENCE

用于： **KafkaUserSpec**

type 属性是一个辨别器，可以区分来自 KafkaUserTlsClientAuthentication，KafkaUserScramSha512ClientAuthentication 的 KafkaUserTlsExternalClientAuthentication 类型的使用。对于类型 KafkaUserTlsExternalClientAuthentication，它需要是值 tls-external。

属性	属性类型	描述
type	string	必须为 **tls-external**。

第 106 章 KAFKAUSERSCRAMSHA512CLIENTAUTHENTICATION 模式参考

用于： KafkaUserSpec

type 属性是一个辨别器，可区分来自 KafkaUserTlsClientAuthentication, KafkaUserTlsExternalClientAuthentication, KafkaUserScramSha512ClientAuthentication 类型的使用。对于类型 KafkaUserScramSha512ClientAuthentication，它需要是 scram-sha-512 值。

属性	属性类型	描述
password	密码	指定用户的密码。如果没有设置，则由 User Operator 生成新密码。
type	string	必须是 **scram-sha-512**。

第 107 章 PASSWORD 架构参考

用于： KafkaUserScramSha512ClientAuthentication

属性	属性类型	描述
valueFrom	**PasswordSource**	从中读取密码的机密。

第 108 章 PASSWORDSOURCE 模式参考

使用于： 密码

属性	属性类型	描述
secretKeyRef	SecretKeySelector	在资源命名空间中选择 Secret 的密钥。

第 109 章 KAFKAUSERAUTHORIZATIONSIMPLE 模式参考

用于： KafkaUserSpec

type 属性是一个辨别器，它区分了可能在以后添加的其他子类型中的 KafkaUserAuthorizationSimple 类型的使用。对于类型 KafkaUserAuthorizationSimple，它需要有 simple 值。

属性	属性类型	描述
type	string	必须 **很简单**。
ACL	**AclRule** 数组	应应用于此用户的 ACL 规则列表。

第 110 章 ACLRULE 模式参考

用于： KafkaUserAuthorizationSimple

AclRule 模式属性的完整列表

110.1. ACLRULE 模式属性

属性	属性类型	描述
主机	string	允许或拒绝 ACL 规则中描述的操作的主机。
operation	字符串([Read, Write, Delete, Alter, Describe, All, IdempotentWrite, ClusterAction, Create, AlterConfigs, DescribeConfigs])	**operation 属性已弃用，现在应使用 spec.authorization.acls[*].operations 进行配置。**将允许或拒绝的操作。支持的操作有： Read, Write, Create, Delete, Alter, Describe, ClusterAction, AlterConfigs, DescribeConfigs, IdempotentWrite 和 All。
操作	字符串（一个或多个 [Read, Write, Delete, Alter, Describe, All, IdempotentWrite, ClusterAction, Create, AlterConfigs, DescribeConfigs]）数组	将允许或拒绝的操作列表。支持的操作有：Read, Write, Create, Delete, Alter, Describe, ClusterAction, AlterConfigs, DescribeConfigs, IdempotentWrite 和 All。
resource	AclRuleTopicResource, AclRuleGroupResource, AclRuleClusterResource , AclRuleTransactionalIdResource	指明应用给定 ACL 规则的资源。
type	字符串（一个 [allow, deny]）	规则的类型。目前唯一支持的类型 **是允许的**。具有 **允许** 类型的 ACL 规则用于允许用户执行指定操作。默认值为 **allow**。

第 111 章 ACLRULETOPICRESOURCE 模式参考

使用于：Acl Rule

type 属性是一个辨别器，可以区分来自 AclRuleGroupResource, AclRuleClusterResource, AclRuleTransactionalIdResource 的 AclRuleTopicResource 类型的使用。对于类型 AclRuleTopicResource，它需要是值 topic。

属性	属性类型	描述
type	string	必须为 **主题**。
name	string	应用的 ACL 规则要应用到的资源名称。可以与 **patternType** 字段结合使用以使用前缀模式。
patternType	字符串（一个 [prefix, literal]）	描述资源字段中使用的模式。支持的类型是 **literal** 和 **prefix**。使用 **字面** 模式类型时，resource 字段将用作完整主题名称的定义。使用 **前缀** 模式类型时，资源名称将仅用作前缀。默认值为 **literal**。

第 112 章 ACLRULEGROUPRESOURCE 模式参考

使用于：Acl Rule

type 属性是一个辨别器，可区分来自 AclRuleTopicResource, AclRuleClusterResource, AclRuleTransactionalIdResource 的 AclRuleGroupResource 类型的使用。对于类型 AclRuleGroupResource，它需要是值 group。

属性	属性类型	描述
type	string	必须为 组。
name	string	应用的 ACL 规则要应用到的资源名称。可以与 **patternType** 字段结合使用以使用前缀模式。
patternType	字符串（一个 [prefix, literal]）	描述资源字段中使用的模式。支持的类型是 **literal** 和 **prefix**。使用 **字面** 模式类型时，resource 字段将用作完整主题名称的定义。使用 **前缀** 模式类型时，资源名称将仅用作前缀。默认值为 **literal**。

第 113 章 ACLRULECLUSTERRESOURCE 模式参考

使用于：Acl Rule

type 属性是一个辨别器，可区分来自 AclRuleTopicResource、AclRuleGroupResource、AclRuleTransactionalIdResource 的 AclRuleClusterResource 类型的使用。对于类型 AclRuleClusterResource，它需要值 cluster。

属性	属性类型	描述
type	string	必须是 **cluster**。

第 114 章 ACLRULETRANSACTIONALIDRESOURCE 模式参考

使用于：Acl Rule

type 属性是一个辨别器，可以区分来自 AclRuleTopicResource, AclRuleGroupResource, AclRuleClusterResource 的 AclRuleTransactionalIdResource 类型的使用。对于类型 AclRuleTransactionalIdResource，它必须是值 transactionalId。

属性	属性类型	描述
type	string	必须是 **transactionalId**。
name	string	应用的 ACL 规则要应用到的资源名称。可以与 **patternType** 字段结合使用以使用前缀模式。
patternType	字符串（一个 [prefix, literal]）	描述资源字段中使用的模式。支持的类型是 **literal** 和 **prefix**。使用 **字面** 模式类型时，resource 字段将用作全名的定义。使用 **前缀** 模式类型时，资源名称将仅用作前缀。默认值为 **literal**。

第 115 章 KAFKAUSERQUOTAS 模式参考

用于： KafkaUserSpec

KafkaUserQuotas 模式属性的完整列表

Kafka 允许用户 设置配额 来控制客户端使用资源。

115.1. 配额

您可以将客户端配置为使用以下类型的配额：

- *网络使用量* 配额指定共享配额的每个客户端组的字节速率阈值。
- *CPU 使用率* 配额为来自客户端的代理请求指定一个窗口。该窗口是客户端发出请求的时间百分比。客户端对代理的 I/O 线程和网络线程发出请求。
- *分区* 修改配额限制允许每秒进行客户端的分区修改数。

分区修改配额可防止 Kafka 集群被并发主题操作造成大量。根据以下类型的用户请求，发生分区修改：

- 为新主题创建分区
- 在现有主题中添加分区
- 从主题中删除分区

您可以配置分区修改配额来控制用户请求接受变异的速度。

在很多情况下，对 Kafka 客户端使用配额可能很有用。考虑一个错误配置的 Kafka producer，它以太

高的速度发送请求。这种错误配置可能会导致服务拒绝其他客户端，因此有问题的客户端不良。通过使用网络限制配额，有可能防止这种情况对其他客户端产生重大影响。

Apache Kafka 的流支持用户级配额，但不支持客户端级别的配额。

Kafka 用户配额配置示例

```
spec:
  quotas:
    producerByteRate: 1048576
    consumerByteRate: 2097152
    requestPercentage: 55
    controllerMutationRate: 10
```

如需有关 Kafka 用户配额的更多信息，请参阅 Apache Kafka 文档。

115.2. KAFKAUSERQUOTAS 模式属性

属性	属性类型	描述
consumerByteRate	整数	在组中客户端节流前，每个客户端组可以从代理获取的最大字节数的配额。基于每个代理定义。
controllerMutationRate	number	创建主题请求、创建分区请求和删除主题请求的速率配额。速率由创建或删除的分区数量计算。
producerByteRate	整数	每个客户端组在节流中的客户端前，每个客户端组可以发布到代理的最大字节数的配额。基于每个代理定义。
requestPercentage	整数	每个客户端组的最大 CPU 使用率配额，作为网络和 I/O 线程的百分比。

第 116 章 KAFKAUSERTEMPLATE 模式参考

用于： KafkaUserSpec

KafkaUserTemplate 模式属性的完整列表

为 User Operator 创建的 secret 指定额外标签和注解。

显示 KafkaUserTemplate的示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaUser
metadata:
  name: my-user
  labels:
    strimzi.io/cluster: my-cluster
spec:
  authentication:
    type: tls
  template:
    secret:
      metadata:
        labels:
          label1: value1
        annotations:
          anno1: value1
  # ...
```

116.1. KAFKAUSERTEMPLATE 模式属性

属性	属性类型	描述
secret	ResourceTemplate	KafkaUser 资源的模板。该模板允许用户指定如何生成带有密码或 TLS 证书的 Secret。

第 117 章 KAFKAUSERSTATUS 模式参考

使用于： **KafkaUser**

属性	属性类型	描述
conditions	**condition** 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
username	string	用户名。
secret	string	存储凭证的 **Secret** 名称。

第 118 章 KAFKAMIRRORMAKER 模式参考

类型 **KafkaMirrorMaker** 已被弃用。请改用 **KafkaMirrorMaker2**。

属性	属性类型	描述
spec	**KafkaMirrorMakerSpec**	Kafka MirrorMaker 的规格。
status	**KafkaMirrorMakerStatus**	Kafka MirrorMaker 的状态。

第 119 章 KAFKAMIRRORMAKERSPEC 模式参考

使用于： KafkaMirrorMaker

KafkaMirrorMakerSpec 模式属性的完整列表

配置 Kafka MirrorMaker。

119.1. INCLUDE

使用 include 属性配置 Kafka MirrorMaker 将源镜像到目标 Kafka 集群的主题列表。

属性允许使用单个主题名称到复杂模式，从最简单的情形中执行任何正则表达式。例如，您可以使用 A|B 镜像主题 A 和 B，或 * 镜像所有主题。您还可以将用逗号分开的多个正则表达式传递给 Kafka MirrorMaker。

119.2. KAFKAMIRRORMAKERCONSUMERSPEC AND KAFKAMIRRORMAKERPRODUCERSPEC

使用 KafkaMirrorMakerConsumerSpec 和 KafkaMirrorMakerProducerSpec 配置源(consumer)和 target (producer)集群。

Kafka MirrorMaker 始终与两个 Kafka 集群（源和目标）协同工作。要建立连接，源和目标 Kafka 集群的 bootstrap 服务器被指定为用逗号分开的 *HOSTNAME:PORT* 对列表。每个以逗号分隔的列表包含一个或多个 Kafka 代理，或一个 Kafka 代理指定的 Service 分区作为 *HOSTNAME:PORT* 对。

119.3. LOGGING

Kafka MirrorMaker 本身有可配置的日志记录器：

- mirrormaker.root.logger

MirrorMaker 使用 Apache log4j 日志记录器实现。

使用 logging 属性配置日志记录器和日志记录器级别。

您可以通过直接指定日志记录器和级别（在线）或使用自定义（外部） ConfigMap 来设置日志级别。如果使用 ConfigMap，您可以将 logging.valueFrom.configMapKeyRef.name 属性设置为包含外部日志记录配置的 ConfigMap 的名称。在 ConfigMap 中，日志记录配置使用 log4j.properties 描述。logging.valueFrom.configMapKeyRef.name 和 logging.valueFrom.configMapKeyRef.key 属性都是强制的。使用指定的确切日志记录配置的 ConfigMap 会在 Cluster Operator 运行时使用自定义资源创建，然后在每次协调后重新创建。如果没有指定自定义 ConfigMap，则使用默认日志记录设置。如果没有设置特定的日志记录器值，则会为该日志记录器继承上级日志记录器设置。有关日志级别的更多信息，请参阅 Apache 日志记录服务。

在这里，我们看到 内联 和 外部日志记录 的示例。inline 日志记录指定根日志记录器级别。您还可以通过将特定类或日志记录器添加到 loggers 属性来设置日志级别。

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaMirrorMaker
spec:
  # ...
  logging:
    type: inline
    loggers:
      mirrormaker.root.logger: INFO
      log4j.logger.org.apache.kafka.clients.NetworkClient: TRACE
      log4j.logger.org.apache.kafka.common.network.Selector: DEBUG
  # ...
```

注意

将日志级别设置为 DEBUG 可能会导致大量日志输出，并可能会影响性能。

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaMirrorMaker
spec:
  # ...
  logging:
    type: external
    valueFrom:
      configMapKeyRef:
        name: customConfigMap
        key: mirror-maker-log4j.properties
  # ...
```

垃圾收集器(GC)

垃圾回收收集器日志记录也可以使用 jvmOptions 属性启用 （或禁用）。

119.4. KAFKAMIRRORMAKERSPEC 模式属性

属性	属性类型	描述
version	string	Kafka MirrorMaker 版本。默认为最新版本。请参阅文档了解升级或降级版本所需的流程。
replicas	整数	**部署**中的 pod 数量。
image	string	用于 Kafka MirrorMaker Pod 的容器镜像。如果没有明确指定镜像名称，它会根据 **spec.version** 配置决定。镜像名称被专门映射到 Cluster Operator 配置中的对应版本。
consumer	**KafkaMirrorMakerConsumerSpec**	源集群配置。
producer	**KafkaMirrorMakerProducerSpec**	目标集群的配置。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
whitelist	string	**whitelist 属性已弃用，现在应使用 spec.include 配置。**包含用于镜像的主题列表。此选项允许使用 Java 样式的正则表达式。镜像名为 A 和 B 的两个主题通过使用表达式 **A|B** 实现。或者，作为特殊情况，您可以使用正则表达式 * 来镜像所有主题。您还可以指定多个正则表达式，用逗号分开。
Include	string	包含用于镜像的主题列表。此选项允许使用 Java 样式的正则表达式。镜像名为 A 和 B 的两个主题通过使用表达式 **A|B** 实现。或者，作为特殊情况，您可以使用正则表达式 * 来镜像所有主题。您还可以指定多个正则表达式，用逗号分开。
jvmOptions	**JvmOptions**	pod 的 JVM 选项。
logging	**InlineLogging**, **ExternalLogging**	MirrorMaker 的日志记录配置。
metricsConfig	**JmxPrometheusExporterMetrics**	指标配置。

属性	属性类型	描述
tracing	**jaegertracing**, **OpenTelemetryTracing**	Kafka MirrorMaker 中的追踪配置。
模板	**KafkaMirrorMakerTemplate**	模板来指定 Kafka MirrorMaker 资源、**Deployment** 和 **Pod** 如何生成。
livenessProbe	**probe**	Pod 存活度检查。
readinessProbe	**probe**	Pod 就绪度检查。

第 120 章 KAFKAMIRRORMAKERCONSUMERSPEC SCHEMA REFERENCE

用于： KafkaMirrorMakerSpec

KafkaMirrorMakerConsumerSpec 模式属性的完整列表

配置 MirrorMaker 消费者。

120.1. NUMSTREAMS

使用 consumer.numStreams 属性配置消费者的流数量。

您可以通过增加消费者线程的数量来增加镜像主题的吞吐量。使用者线程属于为 Kafka MirrorMaker 指定的使用者组。主题分区分配在消费者线程中，它会并行使用消息。

120.2. OFFSETCOMMITINTERVAL

使用 consumer.offsetCommitInterval 属性为消费者配置偏移 auto-commit 间隔。

您可以指定在 Kafka MirrorMaker 消耗源 Kafka 集群的数据后提交偏移的定期时间间隔。时间间隔以毫秒为单位设置，默认值为 60,000。

120.3. CONFIG

使用 consumer.config 属性将消费者的 Kafka 选项配置为密钥。

这些值可以是以下 JSON 类型之一：

- 字符串
- Number

- 布尔值

例外

您可以为用户指定并配置 Apache Kafka 配置文档中列出的选项。

但是，Apache Kafka 的流负责配置和管理与以下内容相关的选项，而这无法更改：

- Kafka 集群 bootstrap 地址
- 安全性（加密、身份验证和授权）
- 消费者组标识符
- 拦截器

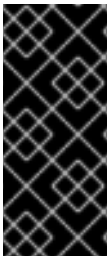
无法设置具有以下前缀的属性：

- bootstrap.servers
- group.id
- interceptor.classes
- SASL。
- 安全性。

- ssl.

如果 config 属性包含一个无法更改的选项，它将被忽略，并将警告信息记录到 Cluster Operator 日志文件。所有其他支持选项都转发到 MirrorMaker，包括对 Apache Kafka 的 Streams 配置的选项的以下例外：

- 支持的 TLS 版本和密码套件的任何 ssl 配置



重要

Cluster Operator 不会密钥或 config 对象中提供的值。如果提供了无效的配置，MirrorMaker 集群可能无法启动，或者可能不稳定。在这种情况下，修复配置，以便 Cluster Operator 可将新配置部署到所有 MirrorMaker 节点。

120.4. GROUPID

使用 consumer.groupId 属性为消费者配置消费者的消费者组标识符。

Kafka MirrorMaker 使用 Kafka 使用者来使用信息，与其他 Kafka 消费者客户端类似。从源 Kafka 集群使用的消息被镜像到目标 Kafka 集群。需要组标识符，因为消费者需要成为分配给分区的消费者组的一部分。

120.5. KAFKAMIRRORMAKERCONSUMERSPEC 模式属性

属性	属性类型	描述
numStreams	整数	指定要创建的消费者流线程数量。
offsetCommitInterval	整数	以 ms 为单位指定偏移 auto-commit 间隔。默认值为 60000。
bootstrapServers	string	用于建立到 Kafka 集群的初始连接的 host:port 对列表。
groupId	string	标识此消费者所属的消费者组的唯一字符串。

属性	属性类型	描述
身份验证	**KafkaClientAuthenticationTls**, **KafkaClientAuthenticationScramSha256**, **KafkaClientAuthenticationScramSha512**, **KafkaClientAuthenticationPlain**, **KafkaClientAuthenticationOAuth**	用于连接集群的身份验证配置。
config	map	MirrorMaker 消费者配置。无法设置带有以下前缀的属性：ssl., bootstrap.servers, group.id, sasl., security., interceptor.classes（除 ssl.endpoint.identification.algorithm, ssl.cipher.suites, ssl.protocol, ssl.enabled.protocols）。
tls	**clientTLS**	将 MirrorMaker 连接到集群的 TLS 配置。

第 121 章 KAFKAMIRRORMAKERPRODUCERSPEC SCHEMA REFERENCE

用于： KafkaMirrorMakerSpec

KafkaMirrorMakerProducerSpec 模式属性的完整列表

配置 MirrorMaker producer。

121.1. ABORTONSENDFAILURE

使用 producer.abortOnSendFailure 属性配置如何处理来自制作者的消息发送失败。

默认情况下，如果在将消息从 Kafka MirrorMaker 发送到 Kafka 集群时出现错误：

- Kafka MirrorMaker 容器在 OpenShift 中终止。
- 然后重新创建容器。

如果 abortOnSendFailure 选项被设置为 false，则消息发送错误将被忽略。

121.2. CONFIG

使用 producer.config 属性将制作者的 Kafka 选项配置为密钥。

这些值可以是以下 JSON 类型之一：

- 字符串
- Number

- 布尔值

例外

您可以为生成者指定并配置 Apache Kafka 配置文档中列出的选项。

但是，Apache Kafka 的流负责配置和管理与以下内容相关的选项，而这无法更改：

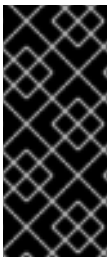
- Kafka 集群 bootstrap 地址
- 安全性（加密、身份验证和授权）
- 拦截器

无法设置具有以下前缀的属性：

- bootstrap.servers
- interceptor.classes
- SASL。
- 安全性。
- ssl.

如果 config 属性包含一个无法更改的选项，它将被忽略，并将警告信息记录到 Cluster Operator 日志文件。所有其他支持选项都转发到 MirrorMaker，包括对 Apache Kafka 的 Streams 配置的选项的以下例外：

- 支持的 TLS 版本和密码套件的任何 ssl 配置



重要

Cluster Operator 不会密钥或 config 对象中提供的值。如果提供了无效的配置，MirrorMaker 集群可能无法启动，或者可能不稳定。在这种情况下，修复配置，以便 Cluster Operator 可将新配置部署到所有 MirrorMaker 节点。

121.3. KAFKAMIRRORMAKERPRODUCERSPEC SCHEMA PROPERTIES

属性	属性类型	描述
bootstrapServers	string	用于建立到 Kafka 集群的初始连接的 host:port 对列表。
abortOnSendFailure	布尔值	将 MirrorMaker 设置为在发送失败时退出的标记。默认值为 **true**。
身份验证	**KafkaClientAuthenticationTls**, **KafkaClientAuthenticationScramSha256**, **KafkaClientAuthenticationScramSha512**, **KafkaClientAuthenticationPlain**, **KafkaClientAuthenticationOAuth**	用于连接集群的身份验证配置。
config	map	MirrorMaker producer 配置。无法设置带有以下前缀的属性： ssl., bootstrap.servers, sasl., security., interceptor.classes (except of: ssl.endpoint.identification.algorithm, ssl.cipher.suites, ssl.protocol, ssl.enabled.protocols)。
tls	**clientTLS**	将 MirrorMaker 连接到集群的 TLS 配置。

第 122 章 KAFKAMIRRORMAKERTEMPLATE 模式参考

用于： **KafkaMirrorMakerSpec**

属性	属性类型	描述
部署	**DeploymentTemplate**	Kafka MirrorMaker **Deployment** 模板。
pod	**PodTemplate**	Kafka MirrorMaker **Pod 的模板**。
podDisruptionBudget	**PodDisruptionBudgetTemplate**	Kafka MirrorMaker **PodDisruptionBudget** 模板。
mirrorMakerContainer	**ContainerTemplate**	Kafka MirrorMaker 容器的模板。
serviceAccount	**ResourceTemplate**	Kafka MirrorMaker 服务帐户的模板。

第 123 章 KAFKAMIRRORMAKERSTATUS 模式参考

使用于： KafkaMirrorMaker

属性	属性类型	描述
conditions	**condition** 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
labelSelector	string	提供此资源的 pod 的标签选择器。
replicas	整数	用于提供此资源的当前 pod 数量。

第 124 章 KAFKABRIDGE 模式参考

属性	属性类型	描述
spec	**KafkaBridgeSpec**	Kafka Bridge 的规格。
status	**KafkaBridgeStatus**	Kafka Bridge 的状态。

第 125 章 KAFKABRIDGESPEC 模式参考

used in: KafkaBridge

KafkaBridgeSpec 模式属性的完整列表

配置 Kafka Bridge 集群。

配置选项与以下内容相关：

- **Kafka 集群 bootstrap 地址**
- **安全性（加密、身份验证和授权）**
- **消费者配置**
- **生成者配置**
- **HTTP 配置**

125.1. LOGGING

Kafka Bridge 具有自己的可配置的日志记录器：

- **rootLogger.level**
- **logger.*<operation-id>***

您可以替换 logger.*<operation-id>* logger 中的 *<operation-id>* 来为特定操作设置日志级别：

- **createConsumer**
- **deleteConsumer**
- 订阅
- **unsubscribe**
- **poll**
- **分配**
- **commit**
- **send**
- **sendToPartition**
- **seekToBeginning**
- **seekToEnd**
- **seek**
- **健康**
- **ready**

- **openapi**

每个操作都由 OpenAPI 规格定义，并具有来自 HTTP 客户端的请求的对应 API 端点。您可以更改每个端点的日志级别，以创建有关传入和传出 HTTP 请求的细粒度日志记录信息。

每个日志记录器都必须配置为为它分配一个 name 作为 http.openapi.operation.*<operation-id>*。例如，为 发送 操作日志记录器配置日志级别意味着定义以下内容：

```
logger.send.name = http.openapi.operation.send
logger.send.level = DEBUG
```

Kafka Bridge 使用 Apache log4j2 日志记录器实现。日志记录器在 log4j2.properties 文件中定义，该文件对 healthy 和 ready 端点有以下默认配置：

```
logger.healthy.name = http.openapi.operation.healthy
logger.healthy.level = WARN
logger.ready.name = http.openapi.operation.ready
logger.ready.level = WARN
```

所有其他操作的日志级别默认设置为 INFO。

使用 logging 属性配置日志记录器和日志记录器级别。

您可以通过直接指定日志记录器和级别（在线）或使用自定义（外部） ConfigMap 来设置日志级别。如果使用 ConfigMap，您可以将 logging.valueFrom.configMapKeyRef.name 属性设置为包含外部日志记录配置的 ConfigMap 的名称。logging.valueFrom.configMapKeyRef.name 和 logging.valueFrom.configMapKeyRef.key 属性是必需的。如果没有设置 name 或 key，则会使用默认日志记录。在 ConfigMap 中，日志记录配置使用 log4j.properties 描述。有关日志级别的更多信息，请参阅 Apache 日志记录服务。

在这里，我们看到 内联 和 外部日志记录 的示例。

内联日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaBridge
spec:
```

```
# ...
logging:
  type: inline
  loggers:
    rootLogger.level: INFO
    # enabling DEBUG just for send operation
    logger.send.name: "http.openapi.operation.send"
    logger.send.level: DEBUG
# ...
```

外部日志记录

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaBridge
spec:
  # ...
  logging:
    type: external
    valueFrom:
      configMapKeyRef:
        name: customConfigMap
        key: bridge-logj42.properties
  # ...
```

任何未配置的可用日志记录器将其级别设置为 OFF。

如果使用 Cluster Operator 部署 Kafka Bridge，则动态应用对 Kafka Bridge 日志记录级别的更改。

如果使用外部日志记录，则会在日志附加程序更改时触发滚动更新。

垃圾收集器(GC)

垃圾回收收集器日志记录也可以使用 jvmOptions 属性启用 （或禁用）。

125.2. KAFKABRIDGESPEC 模式属性

属性	属性类型	描述
replicas	整数	**部署**中的 pod 数量。默认为**1**。
image	string	用于 Kafka Bridge pod 的容器镜像。如果没有明确指定镜像名称，则镜像名称对应于 Cluster Operator 配置中指定的镜像。如果在 Cluster Operator 配置中没有定义镜像名称，则会使用默认值。
bootstrapServers	string	用于建立到 Kafka 集群的初始连接的 host:port 对列表。
tls	**clientTLS**	将 Kafka Bridge 连接到集群的 TLS 配置。
身份验证	**KafkaClientAuthenticationTls**, **KafkaClientAuthenticationScramSha256**, **KafkaClientAuthenticationScramSha512**, **KafkaClientAuthenticationPlain**, **KafkaClientAuthenticationOAuth**	用于连接集群的身份验证配置。
http	**KafkaBridgeHttpConfig**	与 HTTP 相关的配置。
adminClient	**KafkaBridgeAdminClientSpec**	Kafka AdminClient 相关配置。
consumer	**KafkaBridgeConsumerSpec**	Kafka 消费者相关配置。
producer	**KafkaBridgeProducerSpec**	Kafka producer 相关配置。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
jvmOptions	**JvmOptions**	对于 pod，**目前不支持** JVM 选项。
logging	**InlineLogging**, **ExternalLogging**	Kafka Bridge 的日志记录配置。
clientRackInitImage	string	用于初始化 **client.rack** 的 init 容器的镜像。
rack	**Rack**	配置用作 client.rack 消费者配置的节点标签。
enableMetrics	布尔值	为 Kafka Bridge 启用指标。默认值为 false。

属性	属性类型	描述
livenessProbe	**probe**	Pod 存活度检查。
readinessProbe	**probe**	Pod 就绪度检查。
模板	**KafkaBridgeTemplate**	Kafka Bridge 资源的模板。通过该模板，用户可以指定如何生成 **Deployment** 和 **Pod**。
tracing	**jaegertracing**, **OpenTelemetryTracing**	Kafka Bridge 中的追踪配置。

第 126 章 KAFKABRIDGEHTTPCONFIG SCHEMA REFERENCE

used in: KafkaBridgeSpec

KafkaBridgeHttpConfig schema 属性的完整列表

为 Kafka Bridge 配置对 Kafka 集群的 HTTP 访问。

默认 HTTP 配置是 Kafka Bridge 侦听端口 8080。

126.1. CORS

除了启用对 Kafka 集群的 HTTP 访问外，HTTP 属性提供通过 Cross-Origin Resource Sharing (CORS)启用和定义 Kafka Bridge 的访问控制。CORS 是一种 HTTP 机制，它允许浏览器从多个来源访问所选资源。要配置 CORS，您可以定义允许的资源来源和 HTTP 访问方法的列表。对于起源，您可以使用 URL 或 Java 正则表达式。

Kafka Bridge HTTP 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaBridge
metadata:
  name: my-bridge
spec:
  # ...
  http:
    port: 8080
    cors:
      allowedOrigins: "https://strimzi.io"
      allowedMethods: "GET,POST,PUT,DELETE,OPTIONS,PATCH"
  # ...
```

126.2. KAFKABRIDGEHTTPCONFIG SCHEMA PROPERTIES

属性	属性类型	描述
port	整数	服务器侦听的端口。
CORS	KafkaBridgeHttpCors	HTTP Bridge 的 CORS 配置。

第 127 章 KAFKABRIDGEHTTPCORS 模式参考

used in: KafkaBridgeHttpConfig

属性	属性类型	描述
allowedOrigins	字符串数组	允许的源列表。可以使用 Java 正则表达式。
allowedMethods	字符串数组	允许的 HTTP 方法列表。

第 128 章 KAFKABRIDGEADMINCLIENTSPEC SCHEMA REFERENCE

used in: KafkaBridgeSpec

属性	属性类型	描述
config	map	用于网桥创建的 AdminClient 实例的 Kafka AdminClient 配置。

第 129 章 KAFKABRIDGECONSUMERSPEC 模式参考

used in: KafkaBridgeSpec

KafkaBridgeConsumerSpec 模式属性的完整列表

将 Kafka Bridge 的消费者选项配置为密钥。

这些值可以是以下 JSON 类型之一：

- 字符串
- Number
- 布尔值

例外

您可以为用户指定并配置 Apache Kafka 配置文档中列出的选项。

但是，Apache Kafka 的流负责配置和管理与以下内容相关的选项，而这无法更改：

- Kafka 集群 bootstrap 地址
- 安全性（加密、身份验证和授权）
- 消费者组标识符

无法设置具有以下前缀的属性：

- bootstrap.servers
- group.id
- SASL。
- 安全性。
- ssl.

如果 config 属性包含一个无法更改的选项，它将被忽略，并将警告信息记录到 Cluster Operator 日志文件。所有其他支持选项都转发到 Kafka Bridge，包括对流为 Apache Kafka 配置的选项的以下例外：

- 支持的 TLS 版本和密码套件的任何 ssl 配置

Kafka Bridge 消费者配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaBridge
metadata:
  name: my-bridge
spec:
  # ...
  consumer:
    config:
      auto.offset.reset: earliest
      enable.auto.commit: true
    # ...
```

重要

Cluster Operator 不会验证 config 对象中的键或值。如果提供了无效的配置，Kafka Bridge 部署可能无法启动，或者可能会不稳定。在这种情况下，修复配置，以便 Cluster Operator 可将新配置部署到所有 Kafka Bridge 节点。

129.1. KAFKABRIDGECONSUMERSPEC 模式属性

属性	属性类型	描述
config	map	用于网桥创建的消费者实例的 Kafka 使用者配置。无法设置带有以下前缀的属性：ssl., bootstrap.servers, group.id, sasl., security.（除：ssl.endpoint.identification.algorithm, ssl.cipher.suites, ssl.protocol, ssl.enabled.protocols）。

第 130 章 KAFKABRIDGEPRODUCERSPEC 模式参考

used in: KafkaBridgeSpec

KafkaBridgeProducerSpec 模式属性的完整列表

将 Kafka Bridge 的制作者选项配置为密钥。

这些值可以是以下 JSON 类型之一：

- 字符串
- Number
- 布尔值

例外

您可以为生成者指定并配置 Apache Kafka 配置文档中列出的选项。

但是，Apache Kafka 的流负责配置和管理与以下内容相关的选项，而这无法更改：

- Kafka 集群 bootstrap 地址
- 安全性（加密、身份验证和授权）
- 消费者组标识符

无法设置具有以下前缀的属性：

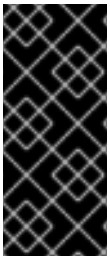
- bootstrap.servers
- SASL。
- 安全性。
- ssl.

如果 config 属性包含一个无法更改的选项，它将被忽略，并将警告信息记录到 Cluster Operator 日志文件。所有其他支持选项都转发到 Kafka Bridge，包括对流为 Apache Kafka 配置的选项的以下例外：

- 支持的 TLS 版本和密码套件的任何 ssl 配置

Kafka Bridge producer 配置示例

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaBridge
metadata:
  name: my-bridge
spec:
  # ...
  producer:
    config:
      acks: 1
      delivery.timeout.ms: 300000
    # ...
```



重要

Cluster Operator 不会验证 config 对象中的键或值。如果提供了无效的配置，Kafka Bridge 部署可能无法启动，或者可能会不稳定。在这种情况下，修复配置，以便 Cluster Operator 可将新配置部署到所有 Kafka Bridge 节点。

130.1. KAFKABRIDGEPRODUCERSPEC 模式属性

属性	属性类型	描述
config	map	用于网桥创建的制作者实例的 Kafka producer 配置。无法设置带有以下前缀的属性： ssl., bootstrap.servers, sasl., security.（除： ssl.endpoint.identification.algorithm, ssl.cipher.suites, ssl.protocol, ssl.enabled.protocols）。

第 131 章 KAFKABRIDGETEMPLATE 模式参考

used in: KafkaBridgeSpec

属性	属性类型	描述
部署	DeploymentTemplate	Kafka Bridge **Deployment** 的模板。
pod	PodTemplate	Kafka Bridge **Pod 的模板**。
apiService	InternalServiceTemplate	Kafka Bridge API **Service** 的模板。
podDisruptionBudget	PodDisruptionBudgetTemplate	Kafka Bridge **PodDisruptionBudget** 的模板。
bridgeContainer	ContainerTemplate	Kafka Bridge 容器的模板。
clusterRoleBinding	ResourceTemplate	Kafka Bridge ClusterRoleBinding 的模板。
serviceAccount	ResourceTemplate	Kafka Bridge 服务帐户的模板。
initContainer	ContainerTemplate	Kafka Bridge init 容器的模板。

第 132 章 KAFKABRIDGESTATUS SCHEMA REFERENCE

used in: KafkaBridge

属性	属性类型	描述
conditions	**condition** 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
url	string	外部客户端应用程序可以访问 Kafka Bridge 的 URL。
labelSelector	string	提供此资源的 pod 的标签选择器。
replicas	整数	用于提供此资源的当前 pod 数量。

第 133 章 KAFKACONNECTOR 模式参考

属性	属性类型	描述
spec	**KafkaConnectorSpec**	Kafka Connector 的规格。
status	**KafkaConnectorStatus**	Kafka Connector 的状态。

第 134 章 KAFKACONNECTORSPEC 模式参考

使用于： KafkaConnector

属性	属性类型	描述
class	string	Kafka Connector 的 Class。
tasksMax	整数	Kafka Connector 的最大任务数量。
autoRestart	AutoRestart	自动重启连接器和任务配置。
config	map	Kafka Connector 配置。无法设置以下属性：connector.class, tasks.max。
pause	布尔值	**pause 属性已弃用。**在 Apache Kafka 2.6 的 Streams 中弃用，使用状态。连接器是否应该暂停。默认为false。
state	字符串（一个 [running, paused, stopped]）	连接器应处于的状态。默认为 running。

第 135 章 AUTORESTART 模式参考

用于： KafkaConnectorSpec, KafkaMirrorMaker2ConnectorSpec

AutoRestart 模式属性的完整列表

为连接器和处于 FAILED 状态的任务配置自动重启。

启用后，后台算法会将自动重启应用到每个失败的连接器及其任务。使用公式 n * n + n 来计算增量后台间隔，其中 n 代表之前重启的数量。这个间隔上限为 60 分钟。因此，重启会立即发生，然后在 2、6、12、20、20、30、42、56 分钟后重启，然后以 60 分钟间隔重启。默认情况下，Apache Kafka 的 Streams 会无限期重启连接器及其任务。但是，您可以使用 maxRestarts 属性在重启数量上设置最大值。如果配置了 maxRestarts，并且连接器仍然会在最终重启尝试后失败，则必须手动重启连接器。

对于 Kafka Connect 连接器，使用 KafkaConnector 资源的 autoRestart 属性启用自动重启失败的连接器和任务。

为 Kafka Connect 启用自动重启失败的连接器

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnector
metadata:
  name: my-source-connector
spec:
  autoRestart:
    enabled: true
```

如果您愿意，您还可以为重启数量设置最大限制。

为带有有限重启的 Kafka Connect 启用自动重启失败的连接器

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaConnector
metadata:
```

```
  name: my-source-connector
spec:
  autoRestart:
    enabled: true
    maxRestarts: 10
```

对于 MirrorMaker 2，使用 KafkaMirrorMaker2 资源中的连接器的 autoRestart 属性启用自动重启失败的连接器和任务。

为 MirrorMaker 2 启用自动重启失败的连接器

```
apiVersion: kafka.strimzi.io/v1beta2
kind: KafkaMirrorMaker2
metadata:
  name: my-mm2-cluster
spec:
  mirrors:
  - sourceConnector:
      autoRestart:
        enabled: true
      # ...
    heartbeatConnector:
      autoRestart:
        enabled: true
      # ...
    checkpointConnector:
      autoRestart:
        enabled: true
      # ...
```

135.1. AUTORESTART 模式属性

属性	属性类型	描述
enabled	布尔值	是否应该启用或禁用为失败的连接器和任务重启。
maxRestarts	整数	Operator 将尝试的最大连接器数重启。如果在达到这个限制后连接器处于失败状态，用户必须手动重启它。默认为无限个重启。

第 136 章 KAFKACONNECTORSTATUS 模式参考

使用于： **KafkaConnector**

属性	属性类型	描述
conditions	**condition** 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
autoRestart	**AutoRestartStatus**	自动重启状态。
connectorStatus	map	连接器状态，如 Kafka Connect REST API 报告。
tasksMax	整数	Kafka Connector 的最大任务数量。
topics	字符串数组	Kafka Connector 使用的主题列表。

第 137 章 AUTORESTARTSTATUS 模式参考

用于： KafkaConnectorStatus, KafkaMirrorMaker2Status

属性	属性类型	描述
æ•°é‡	整数	连接器或任务重启的次数。
connectorName	string	正在重启的连接器的名称。
lastRestartTimestamp	string	最后一次尝试自动重启的时间。所需的格式为 UTC 时区中的 'yyyy-MM-ddTHH:mm:ssZ'。

第 138 章 KAFKAMIRRORMAKER2 模式参考

属性	属性类型	描述
spec	KafkaMirrorMaker2Spec	Kafka MirrorMaker 2 集群的规格。
status	KafkaMirrorMaker2Status	Kafka MirrorMaker 2 集群的状态。

第 139 章 KAFKAMIRRORMAKER2SPEC 模式参考

使用于： KafkaMirrorMaker2

属性	属性类型	描述
version	string	Kafka Connect 版本。默认为最新版本。请参阅用户文档了解升级或降级版本所需的流程。
replicas	整数	Kafka Connect 组中的 pod 数量。默认值为 **3**。
image	string	用于 Kafka Connect pod 的容器镜像。如果没有明确指定镜像名称，它会根据 **spec.version** 配置决定。镜像名称被专门映射到 Cluster Operator 配置中的对应版本。
connectCluster	string	用于 Kafka Connect 的集群别名。该值必须与 **spec.clusters 配置中指定的目标** Kafka 集群的别名匹配。目标 Kafka 集群由底层 Kafka Connect 框架用于其内部主题。
clusters	**KafkaMirrorMaker2ClusterSpec** array	用于镜像的 Kafka 集群。
mirrors	**KafkaMirrorMaker2MirrorSpec** array	配置 MirrorMaker 2 连接器。
resources	ResourceRequirements	CPU 和内存资源和请求的初始资源的最大限制。
livenessProbe	**probe**	Pod 存活度检查。
readinessProbe	**probe**	Pod 就绪度检查。
jvmOptions	**JvmOptions**	pod 的 JVM 选项。
jmxOptions	**KafkaJmxOptions**	JMX 选项.
logging	**InlineLogging**, **ExternalLogging**	Kafka Connect 的日志记录配置。
clientRackInitImage	string	用于初始化 **client.rack** 的 init 容器的镜像。
rack	**Rack**	配置用作 **client.rack** 消费者配置的节点标签。

属性	属性类型	描述
tracing	**jaegertracing**, **OpenTelemetryTracing**	在 Kafka Connect 中配置追踪。
模板	**KafkaConnectTemplate**	Kafka Connect 和 Kafka Mirror Maker 2 资源的模板。该模板允许用户指定如何生成 **Pod**、**Service** 和其他服务。
externalConfiguration	**externalConfiguration**	将来自 Secret 或 ConfigMap 的数据传递给 Kafka Connect pod，并使用它们配置连接器。
metricsConfig	**JmxPrometheusExporterMetrics**	指标配置。

第 140 章 KAFKAMIRRORMAKER2CLUSTERSPEC SCHEMA REFERENCE

使用于： KafkaMirrorMaker2Spec

KafkaMirrorMaker2ClusterSpec 模式属性的完整列表

为镜像配置 Kafka 集群。

140.1. CONFIG

使用 config 属性配置 Kafka 选项。

标准 Apache Kafka 配置可能会提供，仅限于不直接由 Apache Kafka 的 Streams 管理的属性。

对于使用 TLS 版本的特定 *密码套件* 的客户端连接，您可以配置允许的 ssl 属性。您还可以配置 ssl.endpoint.identification.algorithm 属性来启用或禁用主机名验证。

140.2. KAFKAMIRRORMAKER2CLUSTERSPEC 模式属性

属性	属性类型	描述
alias	string	用于引用 Kafka 集群的别名。
bootstrapServers	string	以逗号分隔的 **host:port** 对列表，用于建立与 Kafka 集群的连接。
tls	clientTLS	将 MirrorMaker 2 连接器连接到集群的 TLS 配置。
身份验证	KafkaClientAuthenticationTls, KafkaClientAuthenticationScramSha256, KafkaClientAuthenticationScramSha512, KafkaClientAuthenticationPlain, KafkaClientAuthenticationOAuth	用于连接集群的身份验证配置。

属性	属性类型	描述
config	map	MirrorMaker 2 集群配置。无法设置带有以下前缀的属性： ssl., sasl., security., listener, plugin.path, rest., bootstrap.servers, consumer.interceptor.classes, producer.interceptor.classes （例外： ssl.endpoint.identification.algorithm, ssl.cipher.suites, ssl.protocol, ssl.enabled.protocols）。

第 141 章 KAFKAMIRRORMAKER2MIRRORSPEC 模式参考

使用于： KafkaMirrorMaker2Spec

属性	属性类型	描述
sourceCluster	string	Kafka MirrorMaker 2 连接器使用的源集群的别名。别名必须与位于 **spec.clusters** 的列表中有一个集群匹配。
targetCluster	string	Kafka MirrorMaker 2 连接器使用的目标集群的别名。别名必须与位于 **spec.clusters** 的列表中有一个集群匹配。
sourceConnector	KafkaMirrorMaker2ConnectorSpec	Kafka MirrorMaker 2 源连接器的规格。
heartbeatConnector	KafkaMirrorMaker2ConnectorSpec	Kafka MirrorMaker 2 heartbeat 连接器规格。
checkpointConnector	KafkaMirrorMaker2ConnectorSpec	Kafka MirrorMaker 2 检查点连接器的规格。
topicsPattern	string	与要镜像的主题匹配的正则表达式，例如 "topic1|topic2|topic3"。也支持以逗号分隔的列表。
topicsBlacklistPattern	string	**topicsBlacklistPattern 属性已弃用，现在应使用 .spec.mirrors.topicsExcludePattern 进行配置。**与从镜像中排除的主题匹配的正则表达式。也支持以逗号分隔的列表。
topicsExcludePattern	string	与从镜像中排除的主题匹配的正则表达式。也支持以逗号分隔的列表。
groupsPattern	string	与要镜像的消费者组匹配的正则表达式。也支持以逗号分隔的列表。
groupsBlacklistPattern	string	**groupsBlacklistPattern 属性已弃用，现在应使用 .spec.mirrors.groupsExcludePattern 进行配置。**与消费者组匹配的正则表达式，以便从镜像中排除。也支持以逗号分隔的列表。
groupsExcludePattern	string	与消费者组匹配的正则表达式，以便从镜像中排除。也支持以逗号分隔的列表。

第 142 章 KAFKAMIRRORMAKER2CONNECTORSPEC SCHEMA REFERENCE

使用于： **KafkaMirrorMaker2MirrorSpec**

属性	属性类型	描述
tasksMax	整数	Kafka Connector 的最大任务数量。
config	map	Kafka Connector 配置。无法设置以下属性：connector.class, tasks.max。
autoRestart	**AutoRestart**	自动重启连接器和任务配置。
pause	布尔值	**pause 属性已弃用。**在 Apache Kafka 2.6 的 Streams 中弃用，使用状态。连接器是否应该暂停。默认为false。
state	字符串（一个 [running, paused, stopped]）	连接器应处于的状态。默认为 running。

第 143 章 KAFKAMIRRORMAKER2STATUS 模式参考

使用于： KafkaMirrorMaker2

属性	属性类型	描述
conditions	condition 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
url	string	管理和监控 Kafka Connect 连接器的 REST API 端点的 URL。
autoRestartStatuses	AutoRestartStatus 数组	MirrorMaker 2 连接器自动重启状态列表。
connectorPlugins	ConnectorPlugin 数组	此 Kafka Connect 部署中可用的连接器插件列表。
connectors	map 数组	Kafka Connect REST API 报告的 MirrorMaker 2 连接器状态列表。
labelSelector	string	提供此资源的 pod 的标签选择器。
replicas	整数	用于提供此资源的当前 pod 数量。

第 144 章 KAFKAREBALANCE 模式参考

属性	属性类型	描述
spec	KafkaRebalanceSpec	Kafka 重新平衡的规格。
status	KafkaRebalanceStatus	Kafka 重新平衡的状态。

第 145 章 KAFKAREBALANCESPEC 模式参考

使用于： KafkaRebalance

属性	属性类型	描述
模式	字符串（一个 [remove-brokers, full, add-brokers]）	运行重新平衡的模式。支持的模式是 **full**, **add-brokers**, **remove-brokers**。如果没有指定，则默认使用 **full** 模式。 • **full** 模式在集群中的所有代理中运行重新平衡。 • 在扩展集群后，可以使用 **add-brokers** 模式，将一些副本移到新添加的代理中。 • 在缩减集群前，可以使用 **remove-brokers** 模式，将副本移出要删除的代理。
代理(Broker)	整数数组	在扩展或要删除的代理时，如果缩减用于重新平衡，则新添加的代理列表。此列表只能用于重新平衡模式 **add-brokers** 和 **removed-brokers**。它以 **full** 模式忽略。
目标	字符串数组	目标列表按优先级降序排列，用于生成和执行重新平衡建议。支持的目标位于 https://github.com/linkedin/cruise-control#goals。如果提供了空目标列表，则使用 default.goals Cruise Control 配置参数中声明的目标。
skipHardGoalCheck	布尔值	是否允许 Kafka CR 中指定的硬目标在优化建议生成中跳过。当其中某些硬目标阻止找到平衡解决方案时，这非常有用。默认值为 false。
rebalanceDisk	布尔值	启用 intra-broker 磁盘平衡，平衡同一代理上磁盘之间的磁盘空间利用率。只适用于使用多个磁盘的 JBOD 存储的 Kafka 部署。启用后，禁用 inter-broker 平衡。默认值为 false。
excludedTopics	string	在计算优化提议时，将排除任何匹配主题的正则表达式。此表达式将由 java.util.regex.Pattern 类解析；有关支持的格式的更多信息，请参阅该类的文档。
concurrentPartitionMovementsPerBroker	整数	持续分区副本移动的上限会进入/移出每个代理。默认值为 5。

属性	属性类型	描述
concurrentIntraBrokerPartitionMovements	整数	每个代理内磁盘间持续分区副本移动的上限。默认值为 2。
concurrentLeaderMovements	整数	持续分区领导移动的上限。默认值为 1000。
replicationThrottle	整数	用于移动副本的带宽上的上限（以字节/秒为单位）。默认没有限制。
replicaMovementStrategies	字符串数组	用于决定生成的优化方案中副本移动的策略类名称列表。默认情况下，使用 BaseReplicaMovementStrategy，它将按照生成顺序执行副本移动。

第 146 章 KAFKAREBALANCESTATUS 模式参考

使用于： KafkaRebalance

属性	属性类型	描述
conditions	condition 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
sessionId	string	与这个 KafkaRebalance 资源相关的对 Cruise Control 的请求的会话标识符。Kafka Rebalance Operator 用来跟踪持续重新平衡操作的状态。
optimizationResult	map	描述优化结果的 JSON 对象。

第 147 章 KAFKANODEPOOL 模式参考

属性	属性类型	描述
spec	**KafkaNodePoolSpec**	KafkaNodePool 的规格。
status	**KafkaNodePoolStatus**	KafkaNodePool 的状态。

第 148 章 KAFKANODEPOOLSPEC SCHEMA REFERENCE

用于： **KafkaNodePool**

属性	属性类型	描述
replicas	整数	池中 pod 数量。
storage	**EphemeralStorage**, **PersistentClaimStorage**, **JbodStorage**	存储配置（磁盘）。无法更新。
roles	字符串（一个或多个 [controller, broker]）数组	当启用 KRaft 模式时，此池中的节点将具有的角色。支持的值是 'broker' 和 'controller'。此字段是必需的。当禁用 KRaft 模式时，如果 **broker**，唯一允许的值。
resources	ResourceRequirements	要保留的 CPU 和内存资源。
jvmOptions	**JvmOptions**	pod 的 JVM 选项。
模板	**KafkaNodePoolTemplate**	池资源的模板。该模板允许用户指定如何生成属于此池的资源。

第 149 章 KAFKANODEPOOLTEMPLATE SCHEMA REFERENCE

用于： KafkaNodePoolSpec

属性	属性类型	描述
podSet	ResourceTemplate	Kafka **StrimziPodSet** 资源的模板。
pod	PodTemplate	Kafka **Pod 的模板**。
perPodService	ResourceTemplate	用于从 OpenShift 外部访问的 Kafka 针对每个 **Services** 的模板。
perPodRoute	ResourceTemplate	用于从 OpenShift 外部访问的 Kafka 针对每个 **Routes** 的模板。
perPodIngress	ResourceTemplate	用于从 OpenShift 外部访问的 Kafka 针对每个 **Ingress** 模板。
persistentVolumeClaim	ResourceTemplate	所有 Kafka **PersistentVolumeClaims** 的模板。
kafkaContainer	ContainerTemplate	Kafka 代理容器的模板。
initContainer	ContainerTemplate	Kafka init 容器的模板。

第 150 章 KAFKANODEPOOLSTATUS 模式参考

用于： KafkaNodePool

属性	属性类型	描述
conditions	condition 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
nodeIds	整数数组	这个池中的 Kafka 节点使用的节点 ID。
clusterId	string	Kafka 集群 ID。
roles	字符串（一个或多个 [controller, broker]）数组	在 Streams for Apache Kafka 2.7 中添加。当前分配给这个池的角色。
replicas	整数	用于提供此资源的当前 pod 数量。
labelSelector	string	提供此资源的 pod 的标签选择器。

第 151 章 STRIMZIPODSET SCHEMA REFERENCE

StrimziPodSet 模式属性的完整列表

重要

StrimziPodSet 是 Apache Kafka 资源的内部流。提供信息仅供参考。不要创建、修改或删除 StrimziPodSet 资源，因为这可能导致错误。

151.1. STRIMZIPODSET SCHEMA PROPERTIES

属性	属性类型	描述
spec	StrimziPodSetSpec	StrimziPodSet 的规格。
status	StrimziPodSetStatus	StrimziPodSet 的状态。

第 152 章 STRIMZIPODSETSPEC SCHEMA REFERENCE

使用于：StrimziPod Set

属性	属性类型	描述
selector	LabelSelector	选择器(selector)是一个标签查询，它与此 **StrimziPodSet** 管理的所有 pod 匹配。仅支持 **matchLabels**。如果设置了 **matchExpressions**，它将会被忽略。
pods	map 数组	由此 StrimziPodSet 管理的 Pod。

第 153 章 STRIMZIPODSETSTATUS SCHEMA REFERENCE

使用于：StrimziPod Set

属性	属性类型	描述
conditions	**condition** 数组	状态条件列表。
observedGeneration	整数	Operator 最后协调的 CRD 的生成。
pods	整数	由此 **StrimziPodSet** 资源管理的 pod 数量。
readyPods	整数	由此 **StrimziPodSet** 资源管理的 pod 数量，该资源就绪。
currentPods	整数	由具有当前修订版本的此 **StrimziPodSet** 资源管理的 pod 数量。

附录 A. 使用您的订阅

Apache Kafka 的流通过软件订阅提供。要管理您的订阅，请访问红帽客户门户中的帐户。

访问您的帐户

1. 转至 access.redhat.com。
2. 如果您还没有帐户，请创建一个帐户。
3. 登录到您的帐户。

激活订阅

1. 转至 access.redhat.com。
2. 导航到 My Subscriptions。
3. 导航到 激活订阅 并输入您的 16 位激活号。

下载 Zip 和 Tar 文件

要访问 zip 或 tar 文件，请使用客户门户网站查找下载的相关文件。如果您使用 RPM 软件包，则不需要这一步。

1. 打开浏览器并登录红帽客户门户网站 产品下载页面，网址为 access.redhat.com/downloads。
2. 在 INTEGRATION AND AUTOMATION 目录中找到 Apache Kafka for Apache Kafka 的流。
3. 选择 Apache Kafka 产品所需的流。此时会打开 Software Downloads 页面。

4. 单击组件的 **Download** 链接。

使用 DNF 安装软件包

要安装软件包以及所有软件包的依赖软件包，请使用：

```
dnf install <package_name>
```

要从本地目录中安装之前下载的软件包，请使用：

```
dnf install <path_to_download_package>
```

更新于 2024-06-26