ALIBABA CLOUD

阿里云

消息队列 Kafka 版

生态对接

文档版本：20201102

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.生态对接概述

本文介绍消息队列Kafka版支持对接的生态。

阿里云生态

云数据库HBase版

- 基于HBase和Spark的数据处理平台

阿里云实时计算Flink

- 基于Flink的资讯场景实时数仓

大数据计算服务MaxCompute

- 接入MaxCompute

数据传输服务DTS

- 使用DTS同步MySQL

容器服务Kubernetes版

- 在Knative上实现Kafka消息推送

阿里云Elasticsearch

- 接入阿里云Elasticsearch

开源生态

Logstash
- 接入Logstash

Filebeat
- 接入Filebeat

MySQL
- 使用Kafka Connect同步MySQL

SQL Server
- 使用Kafka Connect同步SQL Server

2.阿里云生态

2.1. 搭配云HBase和Spark构建一体化数据处理平台

云HBase X-Pack是基于Apache HBase、Phoenix、Spark深度扩展，融合Solr检索等技术，支持海量数据的一站式存储、检索与分析。融合云Kafka+云HBase X-Pack能够构建一体化的数据处理平台，支持风控、推荐、检索、画像、社交、物联网、时空、表单查询、离线数仓等场景，助力企业数据智能化。

方案架构

下图是业界广泛应用的大数据中台架构。

> **说明** 其中HBase和Spark选择云HBase X-Pack。详情请参见X-pack Spark分析引擎。

- 消息流入：Flume、Logstash或者在线库的Binlog流入消息队列Kafka版。
- 实时计算：通过X-Pack Spark Streaming实时地消费消息队列Kafka版的消息，写入到云HBase中对外提供在线查询。
- 实时存储与检索：云HBase融合Solr以及Phoenix SQL层能够提供海量的实时存储，以及在线查询检索。
- 批处理、数仓及算法：在线存储HBase的数据可以自动归档到X-Pack Spark数仓。全量数据沉淀到Spark数仓（HiveMeta），做批处理、算法分析等复杂计算，结果回流到在线库对外提供查询。

更多信息

- 该套方案的实践操作，请参见Spark对接Kafka快速入门。

  > **注意** 在Spark接入消息队列Kafka版前，创建Topic和Consumer Group，详情请参见步骤三：创建资源。

- 公有云HBase和Spark的示例代码，请参见 Demo。

2.2. 基于Flink的资讯场景实时数仓

本文介绍如何针对资讯聚合类业务场景搭建基于消息队列Kafka版和实时计算Flink的实时数仓。

场景描述

本文首先介绍什么是实时数仓以及相关技术架构，接着介绍资讯聚合类业务的典型场景及其业务目标，并据此设计了相应的技术架构。然后介绍如何部署基础环境和搭建实时数仓，并介绍业务系统如何使用实时数仓。

解决的问题

- 通过消息队列Kafka版和实时计算Flink实现实时ETL和数据流。
- 通过消息队列Kafka版和实时计算Flink实现实时数据分析。
- 通过消息队列Kafka版和实时计算Flink实现事件触发。

部署架构图

选用的产品

- **消息队列Kafka版**
  消息队列Kafka版是阿里云基于Apache Kafka构建的高吞吐量、高可扩展性的分布式消息队列服务，广泛用于日志收集、监控数据聚合、流式数据处理、在线和离线分析等，是大数据生态中不可或缺的产品之一，阿里云提供全托管服务，免部署、免运维，更专业、更可靠、更安全。
  更多关于消息队列Kafka版的介绍，参见消息队列Kafka版产品详情页。
- **实时计算**
  实时计算（Alibaba Cloud Realtime Compute）是阿里云提供的基于Apache Flink构建的企业级大数据计算平台。在PB级别的数据集上可以支持亚秒级别的处理延时，赋能用户标准实时数据处理流程和行业解决方案；支持Datastream API作业开发，提供了批流统一的Flink SQL，简化BI场景下的开发；可与用户已使用的大数据组件无缝对接，更多增值特性助力企业实时化转型。
  更多关于实时计算的介绍，参见实时计算产品详情页。
- **DataV数据可视化**
  DataV旨在让更多的人看到数据可视化的魅力，帮助非专业的工程师通过图形化的界面轻松搭建专业水准的可视化应用，满足您会议展览、业务监控、风险预警、地理信息分析等多种业务的展示需求。
  更多关于阿里云DataV数据可视化的介绍，参见DataV数据可视化产品详情页。
- **专有网络VPC**
  专有网络VPC帮助您基于阿里云构建出一个隔离的网络环境，并可以自定义IP地址范围、网段、路由表和网关等；此外，也可以通过专线、VPN、GRE等连接方式实现云上VPC与传统IDC的互联，构建混合云业务。
  更多关于专有网络VPC的介绍，参见专有网络VPC产品详情页。
- **云数据库RDS**
  阿里云关系型数据库RDS（Relational Database Service）是一种稳定可靠、可弹性伸缩的在线数据库服务。基于阿里云分布式文件系统和SSD盘高性能存储，RDS支持MySQL、SQL Server、PostgreSQL、PPAS（Postgre Plus Advanced Server，高度兼容Oracle数据库）和MariaDB TX引擎，并且提供了容灾、备份、恢复、监控、迁移等方面的全套解决方案，彻底解决数据库运维的烦恼。
  更多关于云数据库RDS的介绍，参见云数据库RDS产品文档。
- **分析型数据库MySQL版**
  分析型数据库MySQL版（AnalyticDB for MySQL）是一种高并发低延时的PB级实时数据仓库，全面兼容MySQL协议以及SQL:2003语法标准，可以毫秒级针对万亿级数据进行即时的多维分析透视和业务探索。
  更多关于分析型数据库MySQL版的介绍，参见分析型数据库MySQL版产品详情页。
- **对象存储OSS**
  阿里云对象存储OSS（Object Storage Service），是阿里云提供的海量、安全、低成本、高可靠的云存储服务。
  更多关于对象存储OSS的介绍，参见对象存储OSS产品详情页。

详细信息

查看最佳实践详情

更多最佳实践

查看更多阿里云最佳实践

2.3. 将消息队列Kafka版的数据迁移至MaxCompute

本文介绍如何使用DataWorks数据同步功能，将消息队列Kafka版集群上的数据迁移至阿里云大数据计算服务MaxCompute，方便您对离线数据进行分析加工。

Kafka 大数据

前提条件

在开始本教程前，确保您已完成以下操作：

- 确保消息队列Kafka版集群运行正常。本文以部署在华东1（杭州）地域（Region）的集群为例。
- 开通MaxCompute。
- 开通DataWorks。
- 创建项目空间。本文以在华东1（杭州）地域创建名为bigdata_DOC的项目为例。

背景信息

大数据计算服务MaxCompute（原ODPS）是一种大数据计算服务，能提供快速、完全托管免运维的EB级云数据仓库解决方案。

DataWorks是基于MaxCompute计算和存储，提供工作流可视化开发、调度运维托管的一站式海量数据离线加工分析平台。在数加（一站式大数据平台）中，DataWorks控制台即为MaxCompute控制台。MaxCompute和DataWorks一起向用户提供完善的ETL和数仓管理能力，以及SQL、MR、Graph等多种经典的分布式计算模型，能够更快速地解决用户海量数据计算问题，有效降低企业成本，保障数据安全。

本教程旨在帮助您使用DataWorks，将消息队列Kafka版中的数据导入至MaxCompute，来进一步探索大数据的价值。

步骤一：准备Kafka数据

1. 登录消息队列Kafka版控制台创建Topic和Consumer Group，分别命名为testkafka和console-consumer。具体步骤参见步骤三：创建资源。本示例中，Consumer Group console-consumer将用于消费Topic testkafka中的数据。
2. 向Topic testkafka中写入数据。由于消息队列Kafka版用于处理流式数据，您可以持续不断地向其中写入数据。为保证测试结果，建议您写入10条以上的数据。您可以直接在控制台使用发送消息功能来写入数据，也可以使用消息队列Kafka版的SDK收发消息。详情参见使用SDK收发消息。
3. 为验证写入数据生效，您可以在控制台查询消息，看到之前写入Topic中的数据。

步骤二：创建DataWorks表

您需创建DataWorks表，以保证大数据计算服务MaxCompute可以顺利接收消息队列Kafka版数据。本例中为测试便利，使用非分区表。

1. 登录DataWorks控制台，在**工作空间**区域，单击目标工作空间的**进入数据开发**。
2. 在左侧导航栏单击**表管理**，然后单击新建图标。

3. 在**新建表**对话框，输入表名testkafk，然后单击**提交**。
4. 在创建的表页面，单击**DDL模式**。
5. 在**DDL模式**对话框，输入以下建表语句，单击**生成表结构**。

```
CREATE TABLE `testkafka` (
  `key` string,
  `value` string,
  `partition1` string,
  `timestamp1` string,
  `offset` string,
  `t123` string,
  `event_id` string,
  `tag` string
);
```

建表语句中的每一列对应DataWorks数据集成Kafka Reader的默认列。

- key：表示消息的Key。
- value：表示消息的完整内容 。
- partition：表示当前消息所在分区。
- headers：表示当前消息headers信息。
- offset：表示当前消息的偏移量。
- timestamp：表示当前消息的时间戳。

您还可以自主命名，详情参见Kafka Reader。

6. 单击**提交到生产环境**。 详情请参见管理表。

步骤三：同步数据

1. 新增自定义数据集成资源组。此处创建的ECS实例将用以完成数据同步任务。
2. 登录DataWorks控制台，在**工作空间**区域，单击目标工作空间的**进入数据开发**。
3. 在左侧导航栏，选择**数据开发 > 业务流程 > 数据迁移**。
4. 右键选择**数据集成 > 新建数据集成节点 > 数据同步**。
5. 在**新建节点**对话框，输入节点名称（即数据同步任务名称），然后单击**提交**。
6. 在创建的节点页面，选择**数据来源**的**数据源**为**Kafka**，选择**数据去向**的**数据源**为**ODPS**，选择您在步骤二：创建DataWorks表中创建的表。完成上述配置后，请单击框中的按钮，转换为脚本模式，如下图所示。

7. 配置脚本，示例如下。

```
{
  "type": "job",
  "steps": [
    {
      "stepType": "kafka",
      "parameter": {
        "server": "47.xxx.xxx.xxx:9092",
        "kafkaConfig": {
```

```
            "group.id": "console-consumer"
          },
          "valueType": "ByteArray",
          "column": [
            "__key__",
            "__value__",
            "__partition__",
            "__timestamp__",
            "__offset__",
            "'123'",
            "event_id",
            "tag.desc"
          ],
          "topic": "testkafka",
          "keyType": "ByteArray",
          "waitTime": "10",
          "beginOffset": "0",
          "endOffset": "3"
        },
        "name": "Reader",
        "category": "reader"
      },
      {
        "stepType": "odps",
        "parameter": {
          "partition": "",
          "truncate": true,
          "compress": false,
          "datasource": "odps_first",
          "column": [
            "key",
            "value",
            "partition1",
            "timestamp1",
            "offset",
            "t123",
            "event_id",
            "tag"
          ],
          "emptyAsNull": false,
          "table": "testkafka"
```

```
            table : "testkafka"
        },
        "name": "Writer",
        "category": "writer"
      }
    ],
    "version": "2.0",
    "order": {
      "hops": [
        {
          "from": "Reader",
          "to": "Writer"
        }
      ]
    },
    "setting": {
      "errorLimit": {
        "record": ""
      },
      "speed": {
        "throttle": false,
        "concurrent": 1
      }
    }
}
```

8. 在脚本页面，单击**配置任务资源组**，选择步骤1中创建的自定义资源组，然后单击**运行**图标。

执行结果

完成运行后，**运行日志**中显示运行成功。

后续步骤

您可以新建一个数据开发任务运行SQL语句，查看当前表中是否已存在从Kafka同步过的数据。本文以 `select * from testkafka` 为例，具体步骤如下：

1. 在左侧导航栏，选择**数据开发 > 业务流程**。
2. 右键选择**数据开发 > 新建数据开发节点 > ODPS SQL**。
3. 在**新建节点**对话框，输入节点名称，然后单击**提交**。
4. 在创建的节点页面，输入 `select * from testkafka` ，然后单击**运行**图标。

2.4. 从自建MySQL同步至消息队列Kafka版

通过数据传输服务DTS（Data Transmission Service），您可以将自建MySQL同步至消息队列Kafka版，扩展消息处理能力。

前提条件

您已完成以下操作：

- 自建MySQL数据库且数据库版本为5.1、5.5、5.6、5.7或8.0版本。
- 购买并部署消息队列Kafka版。详情请参见购买并部署实例。

> **说明** 消息队列Kafka版提供标准版和专业版，两种规格都支持数据同步。您可以根据自建Kafka集群迁移情况选择实例规格，详情请参见评估规格。

- 在目标实例中创建用于接收同步数据的Topic。详情请参见创建Topic。

> **注意**
>
> - Topic名称只能包含字母、数字、下划线（_）和短划线（-）。
> - Topic名称长度限制在3~64字符，长度超过64字符将被自动截取。
> - Topic名称创建后，将无法修改。

背景信息

消息队列Kafka版是阿里云提供的分布式、高吞吐、可扩展的消息队列服务，针对开源的Apache Kafka提供全托管服务，彻底解决开源产品长期以来的痛点，您只需专注于业务开发，无需部署运维。消息队列Kafka版广泛用于日志收集、监控数据聚合、流式数据处理、在线和离线分析等大数据领域，已成为大数据生态中不可或缺的部分。

注意事项

- DTS在执行全量数据初始化时将占用源库和目标库一定的读写资源，可能会导致数据库的负载上升，在数据库性能较差、规格较低或业务量较大的情况下（例如源库有大量慢SQL、存在无主键表或目标库存在死锁等），可能会加重数据库压力，甚至导致数据库服务不可用。因此您需要在执行数据同步前评估源库和目标库的性能，同时建议您在业务低峰期执行数据同步（例如源库和目标库的CPU负载在30%以下）。
- 如果源数据库没有主键或唯一约束，且所有字段没有唯一性，可能会导致目标数据库中出现重复数据。

功能限制

- 同步对象仅支持数据表，不支持非数据表的对象。
- 不支持自动调整同步对象，如果对同步对象中的数据表进行重命名操作，且重命名后的名称不在同步对象中，那么这部分数据将不再同步到目标Kafka集群中。如需将修改后的数据表继续数据同步至目标Kafka集群中，您需要进行**修改同步对象**操作，详情请参见新增同步对象。

支持同步的SQL操作

数据传输服务DTS支持同步的SQL操作包括INSERT、UPDATE、DELETE、REPLACE。

消息格式

同步到Kafka集群中的数据以avro格式存储，schema定义详情请参见DTS avro schema定义。

> **说明** 数据同步到Kafka集群后，您需要根据avro schema定义进行数据解析。

费用说明

数据传输服务DTS费用，请参见产品定价。

准备工作

为自建MySQL创建账号并设置binlog

操作步骤

1. 购买数据同步作业，详情请参见购买流程。

   > **说明** 购买时，选择源实例为MySQL，目标实例为Kafka，选择同步拓扑为**单向同步**。

2. 登录数据传输控制台。
3. 在左侧导航栏，单击**数据同步**。
4. 在**同步作业列表**页面顶部，选择同步的目标实例所属地域。

   

5. 定位至已购买的数据同步实例，单击**配置同步链路**。
6. 配置同步作业的源实例及目标实例信息。

   

项目	选项	说明
同步作业名称	无	DTS会自动生成一个同步作业名称，建议配置具有业务意义的名称（无唯一性要求），便于后续识别。
源实例信息	实例类型	根据源库部署位置，您可以选择**RDS实例**、**ECS上的自建数据库**或**通过专线、VPN网关、智能网关接入的自建数据库**。 本文以**ECS上的自建数据库**为例介绍配置流程，其他类型的配置流程与该案例类似。
	实例地区	购买数据同步实例时选择的源实例地域信息，不可变更。
	ECS实例ID	选择自建MySQL所属的ECS实例ID。
	数据库类型	固定为**MySQL**，不可变更。
	端口	填入自建MySQL的数据库服务端口。
	数据库账号	填入自建MySQL的数据库账号，该账号需具备REPLICATION SLAVE、REPLICATION CLIENT及所有同步对象的SELECT权限。
	数据库密码	填入该数据库账号对应的密码。

项目	选项	说明
目标实例信息	实例类型	选择**通过专线/VPN网关/智能网关接入的自建数据库**。 **说明** 由于DTS暂时不支持直接选择消息队列Kafka版，此处将其作为自建Kafka来配置数据同步。
	实例地区	购买数据同步实例时选择的目标实例地域信息，不可变更。
	对端专有网络	选择目标消息队列Kafka版实例所属的专有网络ID。您可以在消息队列Kafka版实例的**基本信息**页面中查看到专有网络ID。
	数据库类型	选择为**Kafka**。
	IP地址	填入消息队列Kafka版实例**默认接入点**中的任意一个IP地址。 **说明** 您可以在消息队列Kafka版实例的**基本信息**页面中，获取**默认接入点**对应的IP地址。
	端口	消息队列Kafka版实例的服务端口，默认为9092。
	数据库账号	填入消息队列Kafka版实例的用户名。 **说明** 如果消息队列Kafka版实例的实例类型为**VPC实例**，无需配置**数据库账号**和**数据库密码**。
	数据库密码	填入该用户名对应的密码。
	Topic	i. 单击右侧的**获取Topic列表**。 ii. 下拉选择具体的Topic名称。
	Kafka版本	根据消息队列Kafka版实例版本，选择对应的版本信息。

7. 单击页面右下角的**授权白名单并进入下一步**。
8. 配置同步对象信息。

配置	说明

配置	说明
同步对象	在**源库对象**区域框中，选择需要同步的对象（仅支持选择数据表），然后单击 将其移动到**已选对象**区域框中。 **说明** DTS会自动将表名映射为配置同步的源和目标实例信息时选择的Topic名称。如果需要更换同步的目标Topic（更换后的Topic需是消息队列Kafka版实例中真实存在的），您可以将鼠标指针放置在要进行名称映射的表上，并单击出现的**编辑**进行调整。

9. 上述配置完成后，单击页面右下角的**下一步**。
10. 配置同步初始化的高级配置信息。

说明 同步初始化类型细分为：结构初始化，全量数据初始化。选中**结构初始化**和**全量数据初始化**后，DTS会在增量数据同步之前，将源数据库中待同步对象的结构和存量数据，同步到目标数据库。

11. 上述配置完成后，单击页面右下角的**预检查并启动**。

说明
- 在数据同步作业正式启动之前，会先进行预检查。只有预检查通过后，才能成功启动数据同步作业。
- 如果预检查失败，单击具体检查项后的图标，查看失败详情。根据提示修复后，重新进行预检查。

12. 在**预检查**对话框中显示**预检查通过**后，关闭**预检查**对话框，同步作业将正式开始。
13. 等待同步作业的链路初始化完成，直至处于**同步中**状态。
您可以在**数据同步**页面，查看数据同步作业的状态。

2.5. 在Knative上实现Kafka消息推送

Knative已支持Kafka事件源，您可将Knative与消息队列Kafka版对接，在Knative上实现Kafka消息推送。

前提条件

- 您的消息队列Kafka版实例为包年包月专业版，且实例版本为2.0.0或以上。

说明
- Knative中的Kafka事件源目前仅支持2.0.0及以上版本。
- 目前仅包年包月专业版消息队列Kafka版实例支持升级为2.0.0或以上版本，详情请参见升级实例版本。

- 创建Topic
- 创建Consumer Group
- 部署Knative

背景信息

Knative是一款基于Kubernetes的Serverless框架，其目标是制定云原生、跨平台的Serverless编排标准。Knative系统分为三部分：

- Build：构建系统，用于将函数和应用构建成容器镜像。
- Serving：服务系统，用于配置应用的路由、升级策略、自动扩缩容等。
- Eventing：事件系统，用于自动完成事件的绑定和触发。

要让Eventing（事件系统）正常运行，就必须在Knative集群中实现Bus（内部事件存储层），目前支持的Bus实现方式包括Stub、Kafka。本文以消息队列Kafka版为例介绍如何实现Bus。

操作流程

在Knative上实现Kafka消息推送的操作流程如下图所示。

部署Kafka组件

1. 登录容器服务控制台。
2. 在左侧导航栏，选择 **Knative（公测） > 组件管理**。
3. 在**Knative组件管理**页面：
   i. 从**集群**列表，选择已部署Knative组件的集群。
   ii. 在**add-on组件**列表，找到**Kafka**，在其右侧**操作**列，单击**部署**。

   
4. 在**部署Kafka**对话框，单击**确定**。
   部署完成后，**Kafka**右侧的**状态**显示**已部署**。

   

创建event-display服务

1. 在左侧导航栏，选择 **Knative（公测） > 服务管理**。
2. 在**Knative服务管理**页面，单击**使用模板创建**。
3. 在**使用模板创建**：
   i. 从**集群**列表，选择已部署Knative组件的集群。
   ii. 从**命名空间**列表，选择**default**。
   iii. 从**示例模板**列表，选择**自定义**。

iv. 在**模板**区域，输入模板信息。

```
apiVersion: serving.knative.dev/v1
kind: Service
metadata:
 name: event-display
spec:
 template:
  spec:
   containers:
   - image: registry.cn-hangzhou.aliyuncs.com/knative-sample/eventing-sources-cmd-event_displ
ay:bf45b3eb1e7fc4cb63d6a5a6416cf696295484a7662e0cf9ccdf5c080542****
```

v. 单击**创建**。

创建完成后，**Knative服务管理**页面显示 event-display。

创建Kafka Source服务

1. 在左侧导航栏，选择**Knative（公测） > 服务管理**。
2. 在**Knative服务管理**页面，单击**使用模板创建**。
3. 在**使用模板创建**页面：
   i. 从**集群**列表，选择已部署Knative组件的集群
   ii. 从**命名空间**列表，选择**default**。
   iii. 从**示例模板**列表，选择**自定义**。
   iv. 在**模板**区域，输入模板信息。

```
apiVersion: sources.eventing.knative.dev/v1alpha1
kind: KafkaSource
metadata:
 name: alikafka-source
spec:
 consumerGroup: demo-consumer
 # Broker URL. Replace this with the URLs for your kafka cluster,
 # which is in the format of my-cluster-kafka-bootstrap.my-kafka-namespace:9092.
 bootstrapServers: xxx.xxx.x.xx:9092,xxx.xxx.x.xx:9092,xxx.xxx.x.xx:9092
 topics: demo-topic
 sink:
  apiVersion: serving.knative.dev/v1alpha1
  kind: Service
  name: event-display
```

v. 单击**创建**。

发送消息

1. 登录消息队列Kafka版控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**Topic管理**。
4. 在**Topic管理**页面，找到要发送消息的Topic，在其右侧**操作操作**列中，单击**发送消息发送消息**。

   

   - **Topic**：发送消息的Topic，例如demo-topic。
   - **Message Key**：消息的键，例如demo。
   - **Message Value**：消息的值，必须为JSON格式，例如 `{"key": "test"}` 。

结果验证

发送消息成功后，连接Kubernetes集群，执行命令 `kubectl logs {pod}` ，可查看发送消息的日志。

> **说明** 如需连接Kubernetes集群，请参见通过kubectl连接Kubernetes集群。

2.6. 将消息队列Kafka版接入阿里云Elasticsearch

随着时间的积累，消息队列Kafka版中的日志数据会越来越多。当您需要查看并分析庞杂的日志数据时，可通过阿里云Logstash将消息队列Kafka版中的日志数据导入阿里云Elasticsearch，然后通过Kibana进行可视化展示与分析。本文说明如何进行操作。
kafka elasticsearch vpc

前提条件

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见VPC接入。
- 创建阿里云Elasticsearch实例。详情请参见创建阿里云Elasticsearch实例。

  > **注意** 请注意保存创建阿里云Elasticsearch实例时设置的用户名及密码。该用户名及密码将用于步骤五：创建索引、步骤六：创建管道和步骤七：搜索数据。

- 创建阿里云Logstash实例。详情请参见创建阿里云Logstash实例。

背景信息

通过阿里云Logstash将数据从消息队列Kafka版导入阿里云Elasticsearch的过程如下图所示。

- 消息队列Kafka版

消息队列Kafka版是阿里云提供的分布式、高吞吐、可扩展的消息队列服务。消息队列Kafka版广泛用于日志收集、监控数据聚合、流式数据处理、在线和离线分析等大数据领域，已成为大数据生态中不可或缺的部分。详情请参见什么是消息队列Kafka版。

- 阿里云Elasticsearch
  Elasticsearch简称ES，是一个基于Lucene的实时分布式的搜索与分析引擎，是遵从Apache开源条款的一款开源产品，是当前主流的企业级搜索引擎。它提供了一个分布式服务，可以使您快速的近乎于准实时的存储、查询和分析超大数据集，通常被用来作为构建复杂查询特性和需求强大应用的基础引擎或技术。阿里云Elasticsearch支持5.5.3、6.3.2、6.7.0、6.8.0和7.4.0版本，并提供了商业插件X-Pack服务，致力于数据分析、数据搜索等场景服务。在开源Elasticsearch的基础上提供企业级权限管控、安全监控告警、自动报表生成等功能。详情请参见什么是阿里云Elasticsearch。
- 阿里云Logstash
  阿里云Logstash作为服务器端的数据处理管道，提供了100%兼容开源的Logstash功能。Logstash能够动态地从多个来源采集数据、转换数据，并且将数据存储到所选择的位置。通过输入、过滤和输出插件，Logstash可以对任何类型的事件加工和转换。详情请参见什么是阿里云Logstash。

步骤一：获取VPC环境接入点

阿里云Logstash通过消息队列Kafka版的接入点与消息队列Kafka版在VPC环境下建立连接。

1. 登录消息队列Kafka版控制台。
2. 在顶部菜单栏，选择地域，例如华东1（杭州）。
3. 在左侧导航栏，单击**实例详情**。
4. 在**实例详情**页面，选择要将数据导入阿里云Elasticsearch的实例。
5. 在**基本信息**区域，获取实例的VPC环境接入点。

   消息队列Kafka版支持以下VPC环境接入点：
   - 默认接入点：端口号为9092。
   - SASL接入点：端口号为9094。如需使用SASL接入点，请开启ACL。您可以提交工单申请开启ACL。

   详情请参见接入点对比。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**页面，输入Topic信息，然后单击**创建**。

步骤三：发送消息

向创建的Topic发送消息。

1. 在消息队列Kafka版控制台的**Topic管理**页面，找到创建的Topic，在其右侧**操作**列，单击**发送消息**。
2. 在**发送消息**对话框，输入消息信息，然后单击**发送**。

步骤四：创建Consumer Group

创建阿里云Elasticsearch所属的Consumer Group。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Consumer Group管理**。
2. 在**Consumer Group管理**页面，单击**创建Consumer Group**。
3. 在**创建Consumer Group**页面，输入Consumer Group信息，然后单击**创建**。

步骤五：创建索引

通过阿里云Elasticsearch创建索引，接收消息队列Kafka版的数据。

1. 登录阿里云Elasticsearch控制台。
2. 在顶部菜单栏，选择地域。
3. 在**实例列表**页面，单击创建的实例。
4. 在左侧导航栏，单击**可视化控制**。
5. 在**Kibana**区域，单击**进入控制台**。
6. 在Kibana登录页面，输入Username和Password，然后单击**Log in**。

> **说明**
> - Username为您创建阿里云Elasticsearch实例时设置的用户名。
> - Password为您创建阿里云Elasticsearch实例时设置的密码。

7. 在Kibana控制台的左侧导航栏，单击**Dev Tools**。
8. 执行以下命令创建索引。

```
PUT /elastic_test
{}
```

步骤六：创建管道

通过阿里云Logstash创建管道。管道部署后，将源源不断地从消息队列Kafka版导入数据进阿里云Elasticsearch。

1. 登录阿里云Logstash控制台。
2. 在顶部菜单栏，选择地域。
3. 在**实例列表**页面，单击创建的实例。
4. 在左侧导航栏，单击**管道管理**。
5. 在**管道列表**区域，单击**创建管道**。
6. 在**Config配置**中，输入配置。 配置示例如下。

```
input {
  kafka {
  bootstrap_servers => ["192.168.xx.xx:9092,192.168.xx.xx:9092,192.168.xx.xx:9092"]
  group_id => "elastic_group"
  topics => ["elastic_test"]
  consumer_threads => 12
  decorate_events => true
  }
}
output {
  elasticsearch {
  hosts => ["http://es-cn-o40xxxxxxxxxxxxwm.elasticsearch.aliyuncs.com:9200"]
  index => "elastic_test"
  password => "XXX"
  user => "elastic"
  }
}
```

input参数说明

参数	描述	示例值
bootstrap_servers	消息队列Kafka版的VPC环境接入点。	192.168.xx.xx:9092,192.168.xx.xx:9092,192.168.xx.xx:9092
group_id	Consumer Group的名称。	elastic_group
topics	Topic的名称。	elastic_test
consumer_threads	消费线程数。建议与Topic的分区数保持一致。	12
decorate_events	是否包含消息元数据。默认值为false。	true

output参数说明

参数	描述	示例值
hosts	阿里云Elasticsearch服务的访问地址。您可在阿里云Elasticsearch实例的**基本信息**页面获取。	http://es-cn-o40xxxxxxxxxxxxwm.elasticsearch.aliyuncs.com:9200
index	索引的名称。	elastic_test

参数	描述	示例值
password	访问阿里云Elasticsearch服务的密码。您在创建阿里云Elasticsearch实例时设置的密码。	XXX
user	访问阿里云Elasticsearch服务的用户名。您在创建阿里云Elasticsearch实例时设置的用户名。	elastic

7. 在**管道参数配置**中，输入配置信息，然后单击**保存并部署**。
8. 在**提示**对话框，单击**确认**。

步骤七：搜索数据

您可以在Kibana控制台搜索通过管道导入阿里云Elasticsearch的数据，确认数据是否导入成功。

1. 登录阿里云Elasticsearch控制台。
2. 在顶部菜单栏，选择地域。
3. 在**实例列表**页面，单击创建的实例。
4. 在左侧导航栏，单击**可视化控制**。
5. 在**Kibana**区域，单击**进入控制台**。
6. 在Kibana登录页面，输入Username和Password，然后单击**Log in**。

   > **说明**
   > - Username为您创建阿里云Elasticsearch实例时设置的用户名。
   > - Password为您创建阿里云Elasticsearch实例时设置的密码。

7. 在Kibana控制台的左侧导航栏，单击**Dev Tools**图标。
8. 执行以下命令搜索数据。

   ```
   GET /elastic_test/_search
   {}
   ```

   返回结果如下。

3.开源生态

3.1. Logstash

3.1.1. 接入Logstash

本文介绍如何将消息队列Kafka版接入Logstash。
kafka logstash

Logstash

Logstash是开源的服务器端数据处理管道，能够同时从多个数据源采集数据，然后对数据进行转换，并将数据写入指定的存储中。Logstash的数据处理流程如下：

1. 输入：采集各种格式、大小和来源的数据。在实际业务中，数据往往以各种各样的形式分散或集中地存储在多个系统中，Logstash支持多种数据输入方式，可以在同一时间从多种数据源采集数据。Logstash能够以连续的流式传输方式从日志、Web应用、数据存储等采集数据。
2. 过滤：实时解析和转换数据。数据从源传输到目标存储的过程中，Logstash过滤器能够解析各个事件，识别已命名的字段来构建结构，并将它们转换成通用格式，通过更轻松、快速的方式分析数据来实现商业价值。
3. 输出：导出数据。Logstash提供多种数据输出方向，灵活解锁众多下游用例。

更多关于Logstash的介绍，请参见Logstash简介。

接入优势

消息队列Kafka版接入Logstash可以带来以下优势：

- 异步处理：提高运行效率，防止突发流量影响用户体验。
- 应用解耦：当应用上下游中有一方存在异常情况，另一方仍能正常运行。
- 减少开销：减少Logstash的资源开销。

接入方案

消息队列Kafka版支持以下方式接入Logstash：

- VPC
  - 作为Input接入
  - 作为Output接入
- 公网
  - 作为Input接入
  - 作为Output接入

3.1.2. VPC

3.1.2.1. 作为Input接入

消息队列Kafka版可以作为Input接入Logstash。本文说明如何在VPC环境下通过Logstash从消息队列Kafka版消费消息。

前提条件

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见VPC接入。
- 下载并安装Logstash。详情请参见Download Logstash。
- 下载并安装JDK 8。详情请参见Download JDK 8。

步骤一：获取接入点

Logstash通过消息队列Kafka版的接入点与消息队列Kafka版建立连接。

1. 登录消息队列Kafka版控制台。
2. 在左侧导航栏，单击**实例详情**。
3. 在**实例详情**页面，选择要作为Input接入Logstash的实例。
4. 在**基本信息**区域，获取实例的接入点。

> **说明** 不同接入点的差异，请参见接入点对比。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**对话框，输入Topic信息，然后单击**创建**。

步骤三：发送消息

向创建的Topic发送消息。

1. 在消息队列Kafka版控制台的**Topic管理**页面，找到创建的Topic，在其右侧**操作**列，单击**发送消息**。
2. 在**发送消息**对话框，输入消息信息，然后单击**发送**。

步骤四：创建Consumer Group

创建Logstash所属的Consumer Group。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Consumer Group管理**。
2. 在**Consumer Group管理**页面，单击**创建Consumer Group**。
3. 在**创建Consumer Group**页面，输入Consumer Group信息，然后单击**创建**。

步骤五：Logstash消费消息

在安装了Logstash的机器上启动Logstash，从创建的Topic中消费消息。

1. 执行**cd**命令切换到logstash的*bin*目录。
2. 创建*input.conf*配置文件。
   i. 执行命令 `vim input.conf` 创建空的配置文件。
   ii. 按键进入插入模式。

iii. 输入以下内容。

```
input {
 kafka {
   bootstrap_servers => "192.168.XXX.XXX:9092,192.168.XXX.XXX:9092,192.168.XXX.XXX:9092"
   group_id => "logstash_group"
   topics => ["logstash_test"]
   consumer_threads => 12
   auto_offset_reset => "earliest"
 }
}
output {
 stdout{codec=>rubydebug}
}
```

参数	描述	示例值
bootstrap_servers	消息队列Kafka版提供以下VPC接入点： ■ 默认接入点 ■ SASL接入点	192.168.XXX.XXX:9092,192.168.XXX.XXX:9092,192.168.XXX.XXX:9092
group_id	Cosumer Group的名称。	logstash_group
topics	Topic的名称。	logstash_test
consumer_threads	消费线程数。建议与Topic的分区数保持一致。	12
auto_offset_reset	重置偏移量。取值： ■ earliest：读取最早的消息。 ■ latest：读取最新的消息。	earliest

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

3. 执行以下命令消费消息。

```
./logstash -f input.conf
```

返回结果如下。

更多信息

更多参数设置，请参考Kafka input plugin。

3.1.2.2. 作为Output接入

消息队列Kafka版可以作为Output接入Logstash。本文说明如何在VPC环境下通过Logstash向消息队列Kafka版发送消息。

前提条件

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见VPC接入。
- 下载并安装Logstash。详情请参见Download Logstash。
- 下载并安装JDK 8。详情请参见Download JDK 8。

步骤一：获取接入点

Logstash通过消息队列Kafka版的接入点与消息队列Kafka版建立连接。

1. 登录消息队列Kafka版控制台。
2. 在左侧导航栏，单击**实例详情**。
3. 在**实例详情**页面，选择要作为Output接入Logstash的实例。
4. 在**基本信息**区域，获取实例的接入点。

> **说明** 不同接入点的差异，请参见接入点对比。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**页面，输入Topic信息，然后单击**创建**。

步骤三：Logstash发送消息

在安装了Logstash的机器上启动Logstash，向创建的Topic发送消息。

1. 执行**cd**命令切换到logstash的*bin*目录。
2. 创建*output.conf*配置文件。
   i. 执行命令 `vim output.conf` 创建空的配置文件。
   ii. 按键进入插入模式。

iii. 输入以下内容。

```
input {
  input {
   stdin{}
 }
}

output {
  kafka {
     bootstrap_servers => "192.168.XXX.XXX:9092,192.168.XXX.XXX:9092,192.168.XXX.XXX:9092"
     topic_id => "logstash_test"
    }
}
```

参数	描述	示例值
bootstrap_servers	消息队列Kafka版提供以下VPC接入点： ■ 默认接入点 ■ SASL接入点	192.168.XXX.XXX:9092,192.168.XXX.XXX:9092,192.168.XXX.XXX:9092
topic_id	Topic的名称。	logstash_test

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

3. 向创建的Topic发送消息。

i. 执行 `./logstash -f output.conf` 。

ii. 输入*test*，然后按回车键。
返回结果如下。

步骤四：查看Topic分区

查看消息发送到Topic的情况。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，选择作为Output接入Logstash的实例，找到发送消息的Topic，在其右侧**操作**列单击**分区状态**。
3. 在**分区状态**页面，单击**刷新**。
发送的消息的分区ID和位点信息如下图所示。

步骤五：按位点查询消息

您可以根据发送的消息的分区ID和位点信息查询该消息。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**消息查询**。
2. 在**消息查询**页面，单击**按位点查询**页签。
3. 选择发送了消息的Topic，选择发送的消息的分区ID，选择发送的消息的位点，然后单击**搜索**。

4. （可选）在搜索结果右侧的**操作**列，单击**消息详情**。

更多信息

更多参数设置，请参考Kafka output plugin。

3.1.3. 公网

3.1.3.1. 作为Input接入

消息队列Kafka版可以作为Input接入Logstash。本文说明如何在公网环境下通过Logstash从消息队列Kafka版消费消息。

前提条件

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见公网+VPC接入。
- 下载并安装Logstash。详情请参见Download Logstash。
- 下载并安装JDK 8。详情请参见Download JDK 8。

步骤一：获取接入信息

Logstash通过消息队列Kafka版的接入点与消息队列Kafka版建立连接，并通过用户名及密码进行校验。

1. 登录消息队列Kafka版控制台。
2. 在左侧导航栏，单击**实例详情**。
3. 在**实例详情**页面，选择要作为Input接入Logstash的实例。
4. 在**基本信息**区域，获取实例的接入点。

> **说明** 不同接入点的差异，请参见接入点对比。

5. 在**安全配置**区域，获取实例的用户名和密码。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**页面，输入Topic信息，然后单击**创建**。

步骤三：发送消息

向创建的Topic发送消息。

1. 在消息队列Kafka版控制台的**Topic管理**页面，找到创建的Topic，在其右侧**操作**列，单击**发送消息**。
2. 在**发送消息**对话框，输入消息信息，然后单击**发送**。

步骤四：创建Consumer Group

创建Logstash所属的Consumer Group。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Consumer Group管理**。
2. 在**Consumer Group管理**页面，单击**创建Consumer Group**。
3. 在**创建Consumer Group**页面，输入Consumer Group信息，然后单击**创建**。

步骤五：Logstash消费消息

在安装了Logstash的机器上启动Logstash，从创建的Topic中消费消息。

1. 执行**cd**命令切换到logstash的*bin*目录。
2. 执行以下命令下载*kafka.client.truststore.jks*证书文件。

```
wget https://code.aliyun.com/alikafka/aliware-kafka-demos/raw/master/kafka-log-stash-demo/vpc-ssl/kafka.client.truststore.jks
```

3. 创建*jaas.conf*配置文件。
   i. 执行命令 `vim jaas.conf` 创建空的配置文件。
   ii. 按*i*键进入插入模式。
   iii. 输入以下内容。

```
KafkaClient {
 org.apache.kafka.common.security.plain.PlainLoginModule required
 username="XXX"
 password="XXX";
};
```

参数	描述	示例值
username	公网/VPC实例的用户名。	alikafka_pre-cn-v0h1***
password	公网/VPC实例的密码。	GQiSmqbQVe3b9hdKLDcIlkrBK6***

   iv. 按*Esc*键回到命令行模式。
   v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。
4. 创建*input.conf*配置文件。
   i. 执行命令 `vim input.conf` 创建空的配置文件。

ii. 按键进入插入模式。

iii. 输入以下内容。

```
input {
  kafka {
    bootstrap_servers => "121.XX.XX.XX:9093,120.XX.XX.XX:9093,120.XX.XX.XX:9093"
    topics => ["logstash_test"]

    security_protocol => "SASL_SSL"
    sasl_mechanism => "PLAIN"

    jaas_path => "/home/logstash-7.6.2/bin/jaas.conf"

    ssl_truststore_password => "KafkaOnsClient"
    ssl_truststore_location => "/home/logstash-7.6.2/bin/kafka.client.truststore.jks"

    ssl_endpoint_identification_algorithm => ""

    group_id => "logstash_group"
    consumer_threads => 3
    auto_offset_reset => "earliest"
  }
}

output {
  stdout {
    codec => rubydebug
  }
}
```

参数	描述	示例值
bootstrap_servers	消息队列Kafka版提供的公网接入点为SSL接入点。	121.XX.XX.XX:9093,120.XX.XX.XX:9093,120.XX.XX.XX:9093
topics	Topic的名称。	logstash_test
security_protocol	安全协议。默认为SASL_SSL，无需修改。	SASL_SSL
sasl_mechanism	安全认证机制。默认为PLAIN，无需修改。	PLAIN

参数	描述	示例值
jaas_path	*jaas.conf*配置文件位置。	/home/logstash-7.6.2/bin/jaas.conf
ssl_truststore_password	*kafka.client.truststore.jks*证书密码。默认值为KafkaOnsClient，无需修改。	KafkaOnsClient
ssl_truststore_location	*kafka.client.truststore.jks*证书位置。	/home/logstash-7.6.2/bin/kafka.client.truststore.jks
ssl_endpoint_identification_algorithm	6.x及以上版本Logstash需要加上该参数。	空值
group_id	Cosumer Group的名称。	logstash_group
consumer_threads	消费线程数。建议与Topic的分区数保持一致。	3
auto_offset_reset	重置偏移量。取值： ■ earliest：读取最早的消息。 ■ latest：读取最新的消息。	earliest

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

5. 执行以下命令消费消息。

```
./logstash -f input.conf
```

返回结果如下。

更多信息

更多参数设置，请参见Kafka input plugin。

3.1.3.2. 作为Output接入

消息队列Kafka版可以作为Output接入Logstash。本文说明如何在公网环境下通过Logstash向消息队列Kafka版发送消息。

前提条件

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见公网+VPC接入。
- 下载并安装Logstash。详情请参见Download Logstash。
- 下载并安装JDK 8。详情请参见Download JDK 8。

步骤一：获取接入点

Logstash通过消息队列Kafka版的接入点与消息队列Kafka版建立连接。

1. 登录消息队列Kafka版控制台。
2. 在左侧导航栏，单击**实例详情**。
3. 在**实例详情**页面，选择要作为Output接入Logstash的实例。
4. 在**基本信息**区域，获取实例的接入点。

> **说明** 不同接入点的差异，请参见接入点对比。

5. 在**安全配置**区域，获取实例的用户名和密码。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**页面，输入Topic信息，然后单击**创建**。

步骤三：Logstash发送消息

在安装了Logstash的机器上启动Logstash，向创建的Topic发送消息。

1. 执行**cd**命令切换到logstash的*bin*目录。
2. 执行以下命令下载*kafka.client.truststore.jks*证书文件。

```
wget https://github.com/AliwareMQ/aliware-kafka-demos/blob/master/kafka-log-stash-demo/vpc-ssl/kafka.client.truststore.jks
```

3. 创建*jaas.conf*配置文件。
   i. 执行命令 `vim jaas.conf` 创建空的配置文件。
   ii. 按键进入插入模式。
   iii. 输入以下内容。

```
KafkaClient {
 org.apache.kafka.common.security.plain.PlainLoginModule required
 username="XXX"
 password="XXX";
};
```

参数	描述	示例值
username	公网/VPC实例的用户名。	alikafka_pre-cn-v0h1***
password	公网/VPC实例的密码。	GQiSmqbQVe3b9hdKLDcIlkrBK6***

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

4. 创建*output.conf*配置文件。

i. 执行命令 `vim output.conf` 创建空的配置文件。

ii. 按*i*键进入插入模式。

iii. 输入以下内容。

```
input {
  stdin{}
}

output {
 kafka {
    bootstrap_servers => "121.40.XXX.XXX:9093,120.26.XXX.XXX:9093,120.26.XXX.XXX:9093"
    topic_id => "logstash_test"
    security_protocol => "SASL_SSL"
    sasl_mechanism => "PLAIN"
    jaas_path => "/home/logstash-7.6.2/bin/jaas.conf"
    ssl_truststore_password => "KafkaOnsClient"
    ssl_truststore_location => "/home/logstash-7.6.2/bin/kafka.client.truststore.jks"
    ssl_endpoint_identification_algorithm => ""
  }
}
```

参数	描述	示例值
bootstrap_servers	消息队列Kafka版提供的公网接入点为SSL接入点。	121.XX.XX.XX:9093,120.XX.XX.XX:9093,120.XX.XX.XX:9093
topic_id	Topic的名称。	logstash_test
security_protocol	安全协议。默认为SASL_SSL，无需修改。	SASL_SSL
sasl_mechanism	安全认证机制。默认为PLAIN，无需修改。	PLAIN
jaas_path	*jaas.conf*配置文件位置。	/home/logstash-7.6.2/bin/jaas.conf
ssl_truststore_password	*kafka.client.truststore.jks*证书密码。默认值为KafkaOnsClient，无需修改。	KafkaOnsClient

参数	描述	示例值
ssl_truststore_location	*kafka.client.truststore.jks*证书位置。	/home/logstash-7.6.2/bin/kafka.client.truststore.jks
ssl_endpoint_identification_algorithm	SSL接入点辨识算法。6.x及以上版本Logstash需要加上该参数。	空值

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

5. 向创建的Topic发送消息。

i. 执行 `./logstash -f output.conf` 。

ii. 输入*test*，然后按回车键。

步骤四：查看Topic分区

查看消息发送到Topic的情况。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，选择作为Output接入Logstash的实例，找到发送消息的Topic，在其右侧**操作**列单击**分区状态**。
3. 在**分区状态**页面，单击**刷新**。
   发送的消息的分区ID和位点信息如下图所示。

步骤五：按位点查询消息

您可以根据发送的消息的分区ID和位点信息查询该消息。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**消息查询**。
2. 在**消息查询**页面，单击**按位点查询**页签。
3. 从**请输入Topic**列表，选择发送了消息的Topic，从**请选择分区**列表，选择发送的消息的分区ID，从**请输入位点**列表，选择发送的消息的位点，然后单击**搜索**。
4. （可选）在搜索结果右侧的**操作**列，单击**消息详情**。

更多信息

更多参数设置，请参见Kafka output plugin。

3.2. Filebeat

3.2.1. 接入Filebeat

本文介绍如何将消息队列Kafka版接入Filebeat。
kafka filebeat

Filebeat

Filebeat是用于转发和集中日志数据的轻量级传输程序。Filebeat可以监听指定的日志文件或位置，从中收集日志事件并将其转发到Elasticsearch或Logstash进行索引。Filebeat的工作原理如下：

1. Filebeat启动一个或多个Input，Input在指定的位置中查找日志数据。
2. Filebeat为每个找到的日志启动Harvester，Harvester读取日志并将日志数据发送到libbeat。
3. libbeat聚集数据，然后将聚集的数据发送到配置的Output。

接入优势

消息队列Kafka版接入Filebeat可以带来以下优势：

- 异步处理：防止突发流量。
- 应用解耦：当下游异常时，不会影响上游工作。
- 减少开销：减少Filebeat的资源开销。

接入方案

消息队列Kafka版支持以下方式接入Filebeat：

- VPC
  - 作为Input接入
  - 作为Output接入
- 公网
  - 作为Input接入
  - 作为Output接入

3.2.2. VPC

3.2.2.1. 作为Input接入

消息队列Kafka版可以作为Input接入Filebeat。本文说明如何在VPC环境下通过Filebeat从消息队列Kafka版消费消息。

背景信息

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见VPC接入。
- 下载并安装Filebeat。详情请参见Download Filebeat。
- 下载并安装JDK 8。详情请参见Download JDK 8。

步骤一：获取接入点

Filebeat通过消息队列Kafka版的接入点与消息队列Kafka版建立连接。

1. 登录消息队列Kafka版控制台。
2. 在左侧导航栏，单击**实例详情**。
3. 在**实例详情**页面，选择要作为Input接入Filebeat的实例。
4. 在**基本信息**区域，获取实例的接入点。

> **说明** 不同接入点的差异，请参见接入点对比。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**页面，输入Topic信息，然后单击**创建**。

步骤三：发送消息

向创建的Topic发送消息。

1. 在消息队列Kafka版控制台的**Topic管理**页面，找到创建的Topic，在其右侧**操作**列，单击**发送消息**。
2. 在**发送消息**对话框，输入消息信息，然后单击**发送**。

步骤四：创建Consumer Group

创建Filebeat所属的Consumer Group。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Consumer Group管理**。
2. 在**Consumer Group管理**页面，单击**创建Consumer Group**。
3. 在**创建Consumer Group**页面，输入Consumer Group信息，然后单击**创建**。

步骤五：Filebeat消费消息

在安装了Filebeat的机器上启动Filebeat，从创建的Topic中消费消息。

1. 执行**cd**命令切换到Filebeat的安装目录。
2. 创建*input.yml*配置文件。
   i. 执行命令 `vim input.yml` 创建空的配置文件。
   ii. 按i键进入插入模式。

iii. 输入以下内容。

```
filebeat.inputs:
- type: kafka
  hosts:
    - 192.168.XX.XX:9092
    - 192.168.XX.XX:9092
    - 192.168.XX.XX:9092
  topics: ["filebeat_test"]
  group_id: "filebeat_group"

output.console:
  pretty: true
```

参数	描述	示例值
type	Filebeat的Input类型。	kafka
hosts	消息队列Kafka版提供以下VPC接入点： ■ 默认接入点 ■ SASL接入点	- 192.168.XX.XX:9092 - 192.168.XX.XX:9092 - 192.168.XX.XX:9092
topics	Topic的名称。	filebeat_test
group_id	Consumer Group的名称。	filebeat_group

更多参数说明，请参见Kafka input plugin。

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

3. 执行以下命令消费消息。

```
./filebeat -c ./input.yml
```

3.2.2.2. 作为Output接入

消息队列Kafka版可以作为Output接入Filebeat。本文说明如何在公网环境下通过Filebeat向消息队列Kafka版发送消息。

前提条件

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见VPC接入。
- 下载并安装Filebeat。详情请参见Download Filebeat。
- 下载并安装JDK 8。详情请参见Download JDK 8。

步骤一：获取接入点

Filebeat通过消息队列Kafka版的接入点与消息队列Kafka版建立连接。

1. 登录消息队列Kafka版控制台。
2. 在左侧导航栏，单击**实例详情**。
3. 在**实例详情**页面，选择要作为Output接入Filebeat的实例。
4. 在**基本信息**区域，获取实例的接入点。

> **说明** 不同接入点的差异，请参见接入点对比。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**页面，输入Topic信息，然后单击**创建**。

步骤三：Filebeat发送消息

在安装了Filebeat的机器上启动Filebeat，向创建的Topic发送消息。

1. 执行**cd**命令切换到Filebeat的安装目录。
2. 创建*output.conf*配置文件。
   i. 执行命令 `vim output.conf` 创建空的配置文件。
   ii. 按键进入插入模式。
   iii. 输入以下内容。

```
filebeat.inputs:
- type: stdin

output.kafka:
 hosts: ["192.XX.XX.XX:9092", "192.XX.XX.XX:9092", "192.XX.XX.XX:9092"]

 topic: 'filebeat_test'

 required_acks: 1
 compression: none
 max_message_bytes: 1000000
```

参数	描述	示例值

参数	描述	示例值
hosts	消息队列Kafka版提供以下VPC接入点： ■ 默认接入点 ■ SASL接入点	192.168.XXX.XXX:9092,192.168.XXX.XXX:9092,192.168.XXX.XXX:9092
topic	Topic的名称。	filebeat_test
required_acks	ACK可靠性。取值： ■ 0：无响应 ■ 1：等待本地提交 ■ -1：等待所有副本提交 默认值为1。	1
compression	数据压缩编译码器。默认值为gzip。取值： ■ none：无 ■ snappy：用来压缩和解压缩的C++开发包 ■ lz4：着重于压缩和解压缩速度的无损数据压缩算法 ■ gzip：GNU自由软件的文件压缩程序	none
max_message_bytes	最大消息大小。单位为字节。默认值为1000000。该值应小于您配置的消息队列Kafka版最大消息大小。	1000000

更多参数说明，请参见Kafka output plugin。

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

3. 向创建的Topic发送消息。

i. 执行 `./filebeat -c ./output.yml` 。

ii. 输入*test*，然后按回车键。

步骤四：查看Topic分区

查看消息发送到Topic的情况。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，选择作为Output接入Filebeat的实例，找到发送消息的Topic，在其右侧**操作**列单击**分区状态**。
3. 在**分区状态**页面，单击**刷新**。
   发送的消息的分区ID和位点信息如下图所示。

步骤五：按位点查询消息

您可以根据发送的消息的分区ID和位点信息查询该消息。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**消息查询**。
2. 在**消息查询**页面，单击**按位点查询**页签。
3. 选择发送了消息的Topic，选择发送的消息的分区ID，选择发送的消息的位点，然后单击**搜索**。

4. （可选）在搜索结果右侧的**操作**列，单击**消息详情**。

3.2.3. 公网

3.2.3.1. 作为Input接入

消息队列Kafka版可以作为Input接入Filebeat。本文说明如何在公网环境下通过Filebeat从消息队列Kafka版消费消息。

背景信息

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见公网+VPC接入。
- 下载并安装Filebeat。详情请参见Download Filebeat。
- 下载并安装JDK 8。详情请参见Download JDK 8。

步骤一：获取接入点

Filebeat通过消息队列Kafka版的接入点与消息队列Kafka版建立连接。

1. 登录消息队列Kafka版控制台。
2. 在左侧导航栏，单击**实例详情**。
3. 在**实例详情**页面，选择要作为Input接入Filebeat的实例。
4. 在**基本信息**区域，获取实例的接入点。

> **说明** 不同接入点的差异，请参见接入点对比。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**页面，输入Topic信息，然后单击**创建**。

步骤三：发送消息

向创建的Topic发送消息。

1. 在消息队列Kafka版控制台的**Topic管理**页面，找到创建的Topic，在其右侧**操作**列，单击**发送消息**。

2. 在**发送消息**对话框，输入消息信息，然后单击**发送**。

步骤四：创建Consumer Group

创建Filebeat所属的Consumer Group。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Consumer Group管理**。
2. 在**Consumer Group管理**页面，单击**创建Consumer Group**。
3. 在**创建Consumer Group**页面，输入Consumer Group信息，然后单击**创建**。

步骤五：Filebeat消费消息

在安装了Filebeat的机器上启动Filebeat，从创建的Topic中消费消息。

1. 执行**cd**命令切换到Filebeat的安装目录。
2. 执行以下命令下载CA证书文件。

```
wget https://github.com/AliwareMQ/aliware-kafka-demos/blob/master/kafka-filebeat-demo/vpc-ssl/ca-cert
```

3. 创建*input.yml*配置文件。
   i. 执行命令 `vim input.yml` 创建空的配置文件。
   ii. 按*i*键进入插入模式。

iii. 输入以下内容。

```
filebeat.inputs:
- type: kafka
  hosts:
    - 121.XX.XX.XX:9093
    - 120.XX.XX.XX:9093
    - 120.XX.XX.XX:9093
  username: "alikafka_pre-cn-v641e1dt***"
  password: "aeN3WLRoMPRXmAP2jvJuGk84Kuuo***"
  topics: ["filebeat_test"]
  group_id: "filebeat_group"
  ssl.certificate_authorities: ["/root/filebeat/filebeat-7.7.0-linux-x86_64/ca-cert"]
  ssl.verification_mode: none

output.console:
  pretty: true
```

参数	描述	示例值
hosts	消息队列Kafka版提供的公网接入点为SSL接入点。	- 121.XX.XX.XX:9093 - 120.XX.XX.XX:9093 - 120.XX.XX.XX:9093
username	公网/VPC实例的用户名。	alikafka_pre-cn-v641e1d***
password	公网/VPC实例的密码。	aeN3WLRoMPRXmAP2jvJuGk84Kuuo***
topics	Topic的名称。	filebeat_test
group_id	Consumer Group的名称。	filebeat_group
ssl.certificate_authorities	CA证书所在位置。	/root/filebeat/filebeat-7.7.0-linux-x86_64/ca-cert
ssl.verification_mode	认证模式。	none

更多参数设置，请参考Kafka input plugin。

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

4. 执行以下命令消费消息。

```
./filebeat -c ./input.yml
```

3.2.3.2. 作为Output接入

消息队列Kafka版可以作为Output接入Filebeat。本文说明如何在公网环境下通过Filebeat向消息队列Kafka版发送消息。

前提条件

在开始本教程前，请确保您已完成以下操作：

- 购买并部署消息队列Kafka版实例。详情请参见公网+VPC接入。
- 下载并安装Filebeat。详情请参见Download Filebeat。
- 下载并安装JDK 8。详情请参见Download JDK 8。

步骤一：获取接入点

Filebeat通过消息队列Kafka版的接入点与消息队列Kafka版建立连接。

1. 登录消息队列Kafka版控制台。
2. 在左侧导航栏，单击**实例详情**。
3. 在**实例详情**页面，选择要作为Output接入Filebeat的实例。
4. 在**基本信息**区域，获取实例的接入点。

> **说明** 不同接入点的差异，请参见接入点对比。

步骤二：创建Topic

创建用于存储消息的Topic。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，单击**创建Topic**。
3. 在**创建Topic**页面，输入Topic信息，然后单击**创建**。

步骤三：Filebeat发送消息

在安装了Filebeat的机器上启动Filebeat，向创建的Topic发送消息。

1. 执行**cd**命令切换到Filebeat的安装目录。
2. 执行以下命令下载CA证书文件。

```
wget https://code.aliyun.com/alikafka/aliware-kafka-demos/raw/master/kafka-filebeat-demo/vpc-ssl/ca-cert
```

3. 创建*output.conf*配置文件。
   i. 执行命令 `vim output.conf` 创建空的配置文件。
   ii. 按键进入插入模式。
   iii. 输入以下内容。

```
filebeat.inputs:
- type: stdin

output.kafka:
 hosts: ["121.XX.XX.XX:9093", "120.XX.XX.XX:9093", "120.XX.XX.XX:9093"]
 username: "alikafka_pre-cn-v641e1d***"
 password: "aeN3WLRoMPRXmAP2jvJuGk84Kuuo***"

 topic: 'filebeat_test'
 partition.round_robin:
  reachable_only: false
 ssl.certificate_authorities: ["/root/filebeat/filebeat-7.7.0-linux-x86_64/tasks/vpc_ssl/ca-cert"]
 ssl.verification_mode: none

 required_acks: 1
 compression: none
 max_message_bytes: 1000000
```

参数	描述	示例值
hosts	消息队列Kafka版提供的公网接入点为SSL接入点。	121.XX.XX.XX:9093, 120.XX.XX.XX:9093, 120.XX.XX.XX:9093
username	公网/VPC实例的用户名。	alikafka_pre-cn-v641e1d***
password	公网/VPC实例的密码。	aeN3WLRoMPRXmAP2jvJuGk84Kuuo***
topic	Topic的名称。	filebeat_test
reachable_only	消息是否只发送到可用的分区。取值： ■ true：如果主分区不可用，输出可能阻塞。 ■ false：即使主分区不可用，输出不被阻塞。	false
ssl.certificate_authorities	CA证书所在位置。	/root/filebeat/filebeat-7.7.0-linux-x86_64/ca-cert
ssl.verification_mode	认证模式。	none

参数	描述	示例值
required_acks	ACK可靠性。取值： ■ 0：无响应 ■ 1：等待本地提交 ■ -1：等待所有副本提交 默认值为1。	1
compression	数据压缩编译码器。默认值为gzip。取值： ■ none：无 ■ snappy：用来压缩和解压缩的C++开发包 ■ lz4：着重于压缩和解压缩速度的无损数据压缩算法 ■ gzip：GNU自由软件的文件压缩程序	none
max_message_bytes	最大消息大小。单位为字节。默认值为1000000。该值应小于您配置的消息队列Kafka版最大消息大小。	1000000

更多参数说明，请参见Kafka output plugin。

iv. 按*Esc*键回到命令行模式。

v. 按*：*键进入底行模式，输入*wq*，然后按回车键保存文件并退出。

4. 向创建的Topic发送消息。

i. 执行 `./filebeat -c ./output.yml` 。

ii. 输入*test*，然后按回车键。

步骤四：查看Topic分区

查看消息发送到Topic的情况。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**Topic管理**。
2. 在**Topic管理**页面，选择作为Output接入Filebeat的实例，找到发送消息的Topic，在其右侧**操作**列单击**分区状态**。
3. 在**分区状态**页面，单击**刷新**。
   发送的消息的分区ID和位点信息如下图所示。

步骤五：按位点查询消息

您可以根据发送的消息的分区ID和位点信息查询该消息。

1. 在消息队列Kafka版控制台的左侧导航栏，单击**消息查询**。
2. 在**消息查询**页面，单击**按位点查询**页签。

3. 从**请输入Topic**列表，选择发送了消息的Topic，从**请选择分区**列表，选择发送的消息的分区ID，从**请输入位点**列表，选择发送的消息的位点，然后单击**搜索**。

4. （可选）在搜索结果右侧的**操作**列，单击**消息详情**。

3.3. 使用Kafka Connect将MySQL数据同步至消息队列Kafka版

本教程介绍如何使用Kafka Connect的Source Connector将MySQL的数据同步至消息队列Kafka版。

背景信息

Kafka Connect主要用于将数据流输入和输出消息队列Kafka版。Kafka Connect主要通过各种Source Connector的实现，将数据从第三方系统输入到Kafka broker，通过各种Sink Connector实现，将数据从Kafka broker中导入到第三方系统。

前提条件

在开始本教程前，请确保您已完成以下操作：

- 已下载MySQL Source Connector。

  > **说明** 本教程以0.5.2版本的MySQL Source Connector为例。

- 已下载Kafka Connect。

  > **说明** 本教程以0.10.2.2版本的Kafka Connect为例。

- 已安装docker。

步骤一：配置Kafka Connect

1. 将下载完成的MySQL Connector解压到指定目录。
2. 在Kafka Connect的配置文件*connect-distributed.properties*中配置插件安装位置。

```
plugin.path=/kafka/connect/plugins
```

> **注意**
> Kafka Connect的早期版本不支持配置*plugin.path*，您需要在*CLASSPATH*中指定插件位置。
>
> ```
> export CLASSPATH=/kafka/connect/plugins/mysql-connector/*
> ```

步骤二：启动Kafka Connect

在配置好*connect-distributed.properties*后，执行以下命令启动Kafka Connect。

- 公网接入

i. 执行命令 `export KAFKA_OPTS="-Djava.security.auth.login.config=kafka_client_jaas.conf"` 设置*java.security.auth.login.config*。

ii. 执行命令 `bin/connect-distributed.sh config/connect-distributed.properties` 启动Kafka Connect。

- VPC接入
  执行命令 `bin/connect-distributed.sh config/connect-distributed.properties` 启动Kafka Connect。

步骤三：安装MySQL

1. 下载docker-compose-mysql.yaml。
2. 执行以下命令安装MySQL。

```
export DEBEZIUM_VERSION=0.5
docker-compose -f docker-compose-mysql.yaml up
```

步骤四：配置MySQL

1. 执行以下命令开启MySQL的binlog写入功能，并配置binlog模式为row。

```
[mysqld]
log-bin=mysql-bin
binlog-format=ROW
server_id=1
```

2. 执行以下命令设置MySQL的User权限。

```
GRANT SELECT, RELOAD, SHOW DATABASES, REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'debezium' IDENTIFIED BY 'dbz';
```

> **说明** 示例中MySQL的User为*debezium*，密码为*dbz*。

步骤五：启动MySQL Connector

1. 下载register-mysql.json。
2. 编辑*register-mysql.json*。
   - VPC接入

```
## Kafka接入点，通过控制台获取
## 您在控制台获取的默认接入点
"database.history.kafka.bootstrap.servers" : "kafka:9092",
## 需要提前在控制台创建同名Topic，在本例中创建topic：server1
## 所有Table的变更数据，会记录在server1.$DATABASE.$TABLE的Topic中，如 server1.inventory.products
## 因此用户需要提前在控制台中创建所有相关 Topic
"database.server.name": "server1",
## 记录schema变化信息将记录在这个Topic中
## 需要提前在控制台创建
"database.history.kafka.topic": "schema-changes-inventory"
```

- 公网接入

```
## Kafka接入点，通过控制台获取。存储db中schema变化信息
## 您在控制台获取的SSL接入点
"database.history.kafka.bootstrap.servers" : "kafka:9092",
## 需要提前在控制台创建同名topic，在本例中创建Topic：server1
## 所有Table的变更数据，会记录在server1.$DATABASE.$TABLE的Topic中，如 server1.testDB.products
## 因此用户需要提前在控制台中创建所有相关 Topic
"database.server.name": "server1",
## schema变化信息将记录在这个Topic中
## 需要提前在控制台创建
"database.history.kafka.topic": "schema-changes-inventory",
## SSL公网方式访问配置
"database.history.producer.ssl.truststore.location": "kafka.client.truststore.jks",
"database.history.producer.ssl.truststore.password": "KafkaOnsClient",
"database.history.producer.security.protocol": "SASL_SSL",
"database.history.producer.sasl.mechanism": "PLAIN",
"database.history.consumer.ssl.truststore.location": "kafka.client.truststore.jks",
"database.history.consumer.ssl.truststore.password": "KafkaOnsClient",
"database.history.consumer.security.protocol": "SASL_SSL",
"database.history.consumer.sasl.mechanism": "PLAIN",
```

3. 配置好register-mysql.json后，您需要根据配置在控制台创建相应的Topic，相关操作步骤请参见步骤一：创建Topic。
   按照本教程中的方式安装的MySQL，您可以看到MySQL中已经提前创建好了*database：inventory*。其中有四张表：
   - *customers*
   - *orders*
   - *products*
   - *products_on_hand*

根据以上配置，您需要使用OpenAPI创建Topic：

- *server1*
- *server1.inventory.customers*
- *server1.inventory.orders*
- *server1.inventory.products*
- *server1.inventory.products_on_hand*

在*register-mysql.json*中，配置了将schema变化信息记录在*schema-changes-testDB*，因此您还需要使用OpenAPI创建Topic：*schema-changes-inventory*。使用OpenAPI创建Topic，请参见CreateTopic。

4. 执行以下命令启动MySQL Connector。

```
> curl -i -X POST -H "Accept:application/json" -H  "Content-Type:application/json" http://localhost:8083/
connectors/ -d @register-mysql.json
```

结果验证

按照以下步骤操作确认消息队列Kafka版能否接收到MySQL的变更数据。

1. 变更MySQL Table中的数据。
2. 在控制台的**消息查询**页面，查询变更数据。

3.4. 使用Kafka Connect将SQL Server数据同步至消息队列Kafka版

本教程介绍如何使用Kafka Connect的Source Connector将SQL Server的数据同步至消息队列Kafka版。

前提条件

在开始本教程前，请确保您已完成以下操作：

- 已下载SQL Server Source Connector。详情请参见SQL Server Source Connector。
- 已下载Kafka Connect。详情请参见Kafka Connect。

> **说明** SQL Server Source Connector目前只支持2.1.0及以上版本的Kafka Connect。

- 已下载Docker。详情请参见Docker。

步骤一：配置Kafka Connect

1. 将下载完成的SQL Server Connector解压到指定目录。
2. 在Kafka Connect的配置文件*connect-distributed.properties*中配置插件安装位置。

```
## 指定插件解压后的路径。
plugin.path=/kafka/connect/plugins
```

> 注意
> Kafka Connect的早期版本不支持配置*plugin.path*，您需要在*CLASSPATH*中指定插件位置。
>
> ```
> export CLASSPATH=/kafka/connect/plugins/sqlserver-connector/*
> ```

步骤二：启动Kafka Connect

配置好*connect-distributed.properties*后，执行以下命令启动Kafka Connect。

```
## 如果是公网接入，需先设置java.security.auth.login.config。
## 如果是VPC接入，可以跳过这一步。
export KAFKA_OPTS="-Djava.security.auth.login.config=kafka_client_jaas.conf"
## 启动Kafka Connect。
bin/connect-distributed.sh config/connect-distributed.properties
```

```
## 启动Kafka Connect。
bin/connect-distributed.sh config/connect-distributed.properties
```

步骤三：安装SQL Server

> 注意 SQL Server 2016 SP1以上版本支持CDC，因此您的SQL Server版本必须高于该版本。

1. 下载docker-compose-sqlserver.yaml。
2. 执行以下命令安装SQL Server。

   ```
   docker-compose -f docker-compose-sqlserver.yaml up
   ```

步骤四：配置SQL Server

1. 下载inventory.sql。
2. 执行以下命令初始化SQL Server中的测试数据。

   ```
   cat inventory.sql | docker exec -i tutorial_sqlserver_1 bash -c '/opt/mssql-tools/bin/sqlcmd -U sa -P $SA_PASSWORD'
   ```

3. （可选）如果您需要监听SQL Server中已有的数据表，请完成以下配置：
   i. 执行以下命令开启CDC配置。

   ```
   ## 开启CDC模板数据库
   USE testDB
   GO
   EXEC sys.sp_cdc_enable_db
   GO
   ```

ii. 执行以下命令开启指定Table的CDC配置。

```
## Enable a Table Specifying Filegroup Option Template
USE testDB
GO

EXEC sys.sp_cdc_enable_table
@source_schema = N'dbo',
@source_name   = N'MyTable',
@role_name     = N'MyRole',
@filegroup_name = N'MyDB_CT',
@supports_net_changes = 1
GO
```

iii. 执行以下命令确认是否有权限访问CDC Table。

```
EXEC sys.sp_cdc_help_change_data_capture
GO
```

> **说明** 如果返回结果为空，您需要确认是否有权限访问该表。

iv. 执行以下命令确认SQL Server Agent已开启。

```
EXEC master.dbo.xp_servicecontrol N'QUERYSTATE',N'SQLSERVERAGENT'
```

> **说明** 如果返回结果为 `Running`，则说明SQL Server Agent已开启。

步骤五：启动SQL Server Connector

1. 下载register-sqlserver.json。
2. 编辑*register-sqlserver.json*。
   - VPC接入

```
## 消息队列Kafka版实例的默认接入点，您可以在消息队列Kafka版控制台获取。
"database.history.kafka.bootstrap.servers" : "kafka:9092",
## 您需要提前在消息队列Kafka版控制台创建同名Topic，在本例中创建topic：server1。
## 所有table的变更数据，会记录在server1.$DATABASE.$TABLE的topic中，例如server1.testDB.products
。
## 因此您需要提前在消息队列Kafka版控制台中创建所有相关Topic。
"database.server.name": "server1",
## 记录schema变化信息将记录在该Topic中。
## 您需要提前在消息队列Kafka版控制台创建该Topic。
"database.history.kafka.topic": "schema-changes-inventory"
```

- 公网接入

```
## 消息队列Kafka版实例的SSL接入点，您可以在消息队列Kafka版控制台获取。
"database.history.kafka.bootstrap.servers" : "kafka:9092",
## 您需要提前在消息队列Kafka版控制台创建同名Topic，在本例中创建topic：server1。
## 所有table的变更数据，会记录在server1.$DATABASE.$TABLE的Topic中，例如server1.testDB.products。
## 因此您需要提前在消息队列Kafka版控制台中创建所有相关Topic。
"database.server.name": "server1",
## 记录schema变化信息将记录在该Topic中。
## 您需要提前在消息队列Kafka版控制台创建该Topic。
"database.history.kafka.topic": "schema-changes-inventory",
## 通过SSL接入点访问，还需要修改以下配置。
"database.history.producer.ssl.truststore.location": "kafka.client.truststore.jks",
"database.history.producer.ssl.truststore.password": "KafkaOnsClient",
"database.history.producer.security.protocol": "SASL_SSL",
"database.history.producer.sasl.mechanism": "PLAIN",
"database.history.consumer.ssl.truststore.location": "kafka.client.truststore.jks",
"database.history.consumer.ssl.truststore.password": "KafkaOnsClient",
"database.history.consumer.security.protocol": "SASL_SSL",
"database.history.consumer.sasl.mechanism": "PLAIN",
```

3. 完成*register-sqlserver.json*配置后，您需要根据配置在控制台创建相应的Topic，相关操作步骤请参见步骤一：创建Topic。
   按照本教程中的方式安装的SQL Server，您可以看到SQL Server中已经提前创建*db name：testDB*。其中有四张表：
   - *customers*
   - *orders*
   - *products*
   - *products_on_hand*

   根据以上*register-sqlserver.json*的配置，您需要使用OpenAPI创建Topic：
   - *server1*
   - *server1.testDB.customers*
   - *server1.testDB.orders*
   - *server1.testDB.products*
   - *server1.testDB.products_on_hand*

   在*register-sqlserver.json*中，配置了将schema变化信息记录在*schema-changes-testDB*，因此您还需要使用OpenAPI创建Topic：*schema-changes-inventory*，相关操作请参见CreateTopic。
4. 执行以下命令启动SQL Server。

```
> curl -i -X POST -H "Accept:application/json" -H  "Content-Type:application/json" http://localhost:8083/connectors/ -d @register-sqlserver.json
```

结果验证

确认消息队列Kafka版能否接收到SQL Server的变更数据：

1. 变更监听SQL Server中的数据。
2. 在控制台的**消息查询**页面，查询变更消息。具体操作步骤，请参见查询消息。