实验二：Flume整合Kafka（Exec Source 实时日志采集）

一、实验目的

通过本实验，学生将能够：

1. 理解 **Exec Source + Kafka Sink** 的实时日志采集流程。
2. 掌握使用 `tail -F` 监听日志文件实时追加内容的方法。
3. 学会编写 Flume 配置文件，将实时日志发送到 Kafka 主题。
4. 能够使用 Kafka 创建 Topic、启动消费者并验证数据流转。
5. 建立"实时日志 → Flume → Kafka → 消费者"完整数据链路意识。

二、实验背景与原理

1. 实验场景

企业中的 Web 系统、应用日志、访问日志是"**持续追加**"的。
例如：

- Java 应用运行日志
- 系统 audit 日志

这些日志不断增长，因此需要实时监听文件内容，并把新增日志发送到 Kafka。

本实验模拟：

> 使用 `tail -F` 实时监听 `messages` 文件的新增行，并通过 Flume 采集到 Kafka。

---

2. 数据流流程（Exec Source → Kafka）

```
实时追加日志（messages）
          ↓
Flume（Exec Source → Memory Channel → Kafka Sink）
          ↓
Kafka Topic（realtime）
          ↓
Kafka 消费者实时输出
```

- **Exec Source**
  - 执行系统命令，如 `tail -F 文件`
  - 持续监听文件新增行
  - 每新增一行就是一个 Event
- **Memory Channel**
  - 临时缓存 Event
- **Kafka Sink**

- 将 Event 的 body 写入 Kafka Topic

三、实验环境与前置条件

1. 三节点集群：`master、slave1、slave2`
2. 已安装：
   - Zookeeper 集群
   - Kafka 集群
   - Flume（安装于 slave1）
3. 所有节点主机名互通（SSH 已配置免密最好）。
4. Flume 监听目录与日志文件已准备。

四、实验步骤

> **建议顺序：先启动底层服务 → 创建 Topic 和消费者 → 编写 Flume → 实时写日志 → 验证。**

步骤 1：启动 Zookeeper 集群

在 master / slave1 / slave2 运行：

```
zkServer.sh start
```

步骤 2：启动 Kafka 集群

在 master / slave1 / slave2 运行：

```
kafka-server-start.sh -daemon /opt/apps/kafka/config/server.properties
```

步骤 3：在 Kafka 创建 Topic（realtime）

在 **master** 执行：

```
kafka-topics.sh --create \
--bootstrap-server master:9092,slave1:9092,slave2:9092 \
--topic realtime \
--partitions 1 \
--replication-factor 3
```

查看 Topic：

```
kafka-topics.sh --list \
--bootstrap-server master:9092,slave1:9092,slave2:9092
```

步骤 4：启动 Kafka 消费者（用于验证）

在 **master** 执行：

```
kafka-console-consumer.sh \
--bootstrap-server master:9092 \
--topic realtime \
--from-beginning
```

⚠️ **保持窗口开启，用于验证 Flume 是否成功发送消息。**

步骤 5：编写 Flume 配置文件（slave1）

路径：`/opt/apps/flume/conf/exec-to-kafka.conf`

创建文件：

```
cd /opt/apps/flume/conf
touch exec-to-kafka.conf
```

写入配置内容：

```
# 定义 Flume Agent 的组件
# a1 是该 Agent 的名称，用于后续配置中指代此 Agent
a1.sources = r1
# 定义 Channel 名称为 c1
a1.channels = c1
# 定义 Sink 名称为 k1
a1.sinks = k1

# Source 配置：使用 tail -F 监听文件
a1.sources.r1.type = exec
# 使用 tail 命令实时监控日志文件的新增内容
a1.sources.r1.command = tail -F /opt/apps/flume/logs/messages

# Channel 配置：内存通道
a1.channels.c1.type = memory
# Channel 的容量最多存储 1000 条事件
# 当事件数达到 1000 时，Channel 将暂时停止接收新事件，直到事件被消费腾出空间
a1.channels.c1.capacity = 1000
# Source 和 Sink 每次事务传输的最大事件数量为 100
a1.channels.c1.transactionCapacity = 100

# Sink 配置：Kafka Sink
a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
# 设置 Kafka 主题名称，将日志数据写入 Kafka 的 realtime 主题
a1.sinks.k1.topic = realtime
# 配置 Kafka 的 broker 列表，格式为 "主机名:端口号"
# master、slave1 和 slave2 是 Kafka 集群的节点
a1.sinks.k1.brokerList = master:9092,slave1:9092,slave2:9092
# 每次批量传输 20 条事件到 Kafka，优化性能
```

```
a1.sinks.k1.batchSize = 20

# 绑定 Source、Sink 和 Channel 之间的关系
# 将 Source r1 绑定到 Channel c1，数据从 Source 传递到 Channel
a1.sources.r1.channels = c1
# 将 Sink k1 绑定到 Channel c1，数据从 Channel 传递到 Sink
a1.sinks.k1.channel = c1
```

步骤 6：启动 Flume Agent（slave1）

```
# 在slave1启动flume
flume-ng agent \
-n a1 \
-c /opt/apps/flume/conf \
-f /opt/apps/flume/conf/exec-to-kafka.conf \
-Dflume.root.logger=INFO,console
```

注：保持flume所在窗口不要关闭

步骤 7：在 slave2 创建模拟实时写日志的脚本

（1）编写模拟实时写日志脚本

```
# 进入日志目录
cd /opt/data/flume/logs

# 创建 `data_writer.sh` —— 模拟实时写日志脚本
touch data_writer.sh
```

编写 `data_writer.sh` 脚本内容：

```
#!/bin/bash

# 本地要读取的文件（在 slave2 上）
LOGFILE="./messages"

# 远程目标主机（slave1）和目标文件路径
TARGET_HOST="slave1"
TARGET_FILE="/opt/apps/flume/logs/messages"

COUNT=1    # 计数器，从第1条开始

while true; do
    # 从本地 LOGFILE 一行一行读取数据
    while IFS= read -r line || [ -n "$line" ]; do

        # ① 生成当前时间戳：YYYY-MM-DD HH:MM:SS
        timestamp=$(date "+%Y-%m-%d %H:%M:%S")
```

```
        # ② 拼接“时间戳 + 内容”，并发送到 slave1
        newline="$timestamp,$line"
        printf '%s\n' "$newline" | ssh "$TARGET_HOST" "cat >> '$TARGET_FILE'"

        # ③ 在 slave2 终端输出你想要的格式
        echo "成功向 slave1 发送第 ${COUNT} 条数据：$newline"

        # ④ 累计条数 +1
        COUNT=$((COUNT + 1))

        # 每写一行停 1 秒，模拟实时输入
        sleep 1
    done < "$LOGFILE"
done
```

赋执行权即：

```
chmod 777 data_writer.sh
```

运行脚本：

```
./data_writer.sh
```

日志将**每秒写入一次** slave1 的 messages 文件。

（2）编写日志数据

在同路径下准备 messages文件

> **messages文件内容**
>
> 1,玛纳斯分公司（白酒销售）销售员,新疆五谷传奇酒业销售有限公司石...,玛纳斯,3800-10000元/月
> 2,仪表工,新疆雅澳科技有限责任公司,玛纳斯,5000-7000元/月
> 3,加液工,石河子市物源盛通商贸有限公司,北泉镇,3500-4500元/月
> 4,大锅饭师傅,石河子市天石肉类加工有限公司,石河子,3000-3300元/月
> 5,维修工,石河子市天石肉类加工有限公司,北泉镇,5000-6000元/月
> 6,车间统计员,石河子市天石肉类加工有限公司,北泉镇,4500-5000元/月
> 7,保洁,石河子开发区大河宴鱼馆,不限,3200-3500元/月
> 8,电工,石河子市天石肉类加工有限公司,北泉镇,4500-6000元/月
> .....

步骤 8：验证实验结果

回到 **master** 的 Kafka 消费者窗口，会看到不断刷出日志内容，例如：

```
2025-11-27 10:01:01 1,玛纳斯分公司（白酒销售）销售员,新疆五谷传奇酒业销售有限公司石...,玛纳斯,3800-10000元/月
2025-11-27 10:01:02 2,仪表工,新疆雅澳科技有限责任公司,玛纳斯,5000-7000元/月
...
```

如果看到了，就说明：

✔ Exec Source 正常工作
✔ Flume 正常传输
✔ Kafka 已成功接收
✔ 消费链路完整

五、重要注意事项（新手重点）

注意点	说明
🔧 日志文件必须提前存在	`tail -F` 只能监听已创建的文件
❗ Exec Source 不支持批量静态文件	需要 SpoolDir 来处理
🧠 适用于实时日志	如 Web 访问日志、系统日志
⚠️ SSH 写入需成功	data_writer.sh 必须能写入 slave1

六、实验小结

通过本实验，学生掌握了：

- Flume **Exec Source** 的使用方法
- 使用 `tail -F` 实时监听日志文件新增内容
- Flume 事件在 **Source → Channel → Sink** 的流转机制
- Kafka Topic 创建、消费者监听、数据验证
- 完成了一个典型"实时日志采集"场景的小型项目

本实验可作为 **实时流式采集** 的入门案例，适用于大数据平台项目实训。