Flume

第一章 简介

Apache Flume 是一个分布式，高可用的数据收集系统。它可以从不同的数据源收集数据，经过聚合后发送到存储系统中，通常用于日志数据的收集。Flume 分为 NG 和 OG (1.0 之前) 两个版本，NG 在 OG 的基础上进行了完全的重构，是目前使用最为广泛的版本。下面的介绍均以 NG 为基础。

第二章 架构和基本概念

基本架构

外部数据源以特定格式向 Flume 发送 `events` (事件)，当 `source` 接收到 `events` 时，它将其存储到一个或多个 `channel`，`channe` 会一直保存 `events` 直到它被 `sink` 所消费。`sink` 的主要功能从 `channel` 中读取 `events`，并将其存入外部存储系统或转发到下一个 `source`，成功后再从 `channel` 中移除 `events`。

基本概念

1. Event

`Event` 是Flume数据传输的基本单元。类似于 JMS 和消息系统中的消息。一个 `Event` 由标题和正文组成：前者是键/值映射，后者是任意字节数组。

2. Source

数据收集组件，从外部数据源收集数据，并存储到 Channel 中。

3. Channel

`Channel` 是源和接收器之间的管道，用于临时存储数据。可以是内存或持久化的文件系统：

- `Memory Channel` : 使用内存，优点是速度快，但数据可能会丢失 (如突然宕机)；
- `File Channel` : 使用持久化的文件系统，优点是能保证数据不丢失，但是速度慢。

4. Sink

`Sink` 的主要功能从 `Channel` 中读取 `Event`，并将其存入外部存储系统或将其转发到下一个 `Source`，成功后再从 `Channel` 中移除 `Event`。

5. Agent

是一个独立的 (JVM) 进程，包含 `Source`、`Channel`、`Sink` 等组件。

组件种类

Flume 中的每一个组件都提供了丰富的类型，适用于不同场景：

- Source 类型： 内置了几十种类型，如 `Avro Source`， `Thrift Source`， `Kafka Source`， `JMS Source`；
- Sink 类型： `HDFS Sink`， `Hive Sink`， `HBaseSinks`， `Avro Sink` 等；
- Channel 类型： `Memory Channel`， `JDBC Channel`， `Kafka Channel`， `File Channel` 等。

对于 Flume 的使用，除非有特别的需求，否则通过组合内置的各种类型的 Source，Sink 和 Channel 就能满足大多数的需求。在Flume 官网 上对所有类型组件的配置参数均以表格的方式做了详尽的介绍，并附有配置样例；同时不同版本的参数可能略有所不同，所以使用时建议选取官网对应版本的 User Guide 作为主要参考资料。

第三章 Flume构架模式

多agent顺序连接

Flume支持跨越多个Agent的数据传递，这要求前一个Agent的 Sink和下一个Agent的Source都必须是 `Avro` 类型，Sink 指向 Source 所在主机名 (或 IP 地址) 和端口。

多agent的复杂流

日志收集中常常存在大量的客户端（比如分布式 web 服务），Flume 支持使用多个 Agent 分别收集日志，然后通过一个或者多个 Agent 聚合后再存储到文件系统中。

多路复用

Flume 支持从一个 Source 向多个 Channel，也就是向多个 Sink 传递事件，这个操作称之为 `Fan Out` (扇出)。默认情况下 `Fan Out` 是向所有的 Channel 复制 `Event`，即所有 Channel 收到的数据都是相同的。同时 Flume 也支持在 `Source` 上自定义一个复用选择器 (multiplexing selector) 来实现自定义的路由规则。

第四章 Flume配置格式

Flume 配置通常需要以下两个步骤：

1. 分别定义好 Agent 的 Sources，Sinks，Channels，然后将 Sources 和 Sinks 与通道进行绑定。需要注意的是一个 Source 可以配置多个 Channel，但一个 Sink 只能配置一个 Channel。基本格式如下：

```
# 命名agent的source、sink、channel
<Agent>.sources = <Source>
<Agent>.sinks = <Sink>
<Agent>.channels = <Channel1> <Channel2>

# 绑定channel和source
<Agent>.sources.<Source>.channels = <Channel1> <Channel2> ...

# 绑定channel与sink
<Agent>.sinks.<Sink>.channel = <Channel1>
```

2. 分别定义 Source，Sink，Channel 的具体属性。基本格式如下：

```
# 配置source
<Agent>.sources.<Source>.<someProperty> = <someValue>

# 配置channel
<Agent>.channel.<Channel>.<someProperty> = <someValue>

# 配置sink
<Agent>.sources.<Sink>.<someProperty> = <someValue>
```

第五章 Flume整合Kafka

Flume 发送数据到 Kafka 上主要是通过 `KafkaSink` 来实现的。

启动Zk和Kf

```
# 启动Zookeeper
zkServer.sh start

# 启动kafka
bin/kafka-server-start.sh config/server.properties
```

创建主题

```
# 创建主题
bin/kafka-topics.sh --create \
--zookeeper master:2181 \
--replication-factor 1    \
--partitions 1 --topic flume-kafka

# 查看创建的主题
bin/kafka-topics.sh --zookeeper master:2181 --list
```

启动Kf消费者

启动一个消费者，监听我们刚才创建的 `flume-kafka` 主题：

```
bin/kafka-console-consumer.sh --bootstrap-server master:9092 --topic flume-kafka
```

配置Flume

新建配置文件 `exec-memory-kafka.properties`，文件内容如下。这里我们监听一个名为 `kafka.log` 的文件，当文件内容有变化时，将新增加的内容发送到 Kafka 的 `flume-kafka` 主题上。

```
a1.sources = s1
a1.channels = c1
a1.sinks = k1

a1.sources.s1.type=exec
a1.sources.s1.command=tail -F /tmp/kafka.log
a1.sources.s1.channels=c1

#设置Kafka接收器
a1.sinks.k1.type= org.apache.flume.sink.kafka.KafkaSink
#设置Kafka地址
a1.sinks.k1.brokerList=hadoop001:9092
#设置发送到Kafka上的主题
a1.sinks.k1.topic=flume-kafka
#设置序列化方式
a1.sinks.k1.serializer.class=kafka.serializer.StringEncoder
a1.sinks.k1.channel=c1

a1.channels.c1.type=memory
a1.channels.c1.capacity=10000
a1.channels.c1.transactionCapacity=100
```

启动Flume

```
flume-ng agent \
--c conf \
--f /opt/module/flume/job/exec-memory-kafka.properties \
--n a1 -Dflume.root.logger=INFO,console
```

测试

向监听的 `/tmp/kafka.log` 文件中追加内容，查看 Kafka 消费者的输出：

```
# 向文件中追加数据
echo "hello flume" >> kafka.log
echo "hello kafka" >> kafka.log
echo "hello flink" >> kafka.log
echo "hello storm" >> kafka.log
echo "hello SparkStreaming" >> kafka.log
```

可以看到 `flume-kafka` 主题的消费端已经收到了对应的消息。