Flume

实验目的

掌握Flume的工作机制，熟练配置相关文件

实验内容

一、安装Flume

- 1.1首先验证flume安装包的完整性

```
chen@ubuntu:~/big_data_tools$ sha512sum -c apache-flume-1.9.0-bin.tar.gz.sha512
apache-flume-1.9.0-bin.tar.gz: OK
chen@ubuntu:~/big_data_tools$
```

- 1.2安装

```
# 复制到/apps下
cp apache-flume-1.9.0-bin.tar.gz /apps/
cd /apps
# 解压
tar -xzvf apache-flume-1.9.0-bin.tar.gz
# 删除包
rm -rf apache-flume-1.9.0-bin.tar.gz
# 重命名
mv apache-flume-1.9.0-bin flume
```

- 1.3 设置环境变量

```
vim ~/.bashrc
# 添加如下内容
export FLUME_HOME=/apps/flume
export PATH=$FLUME_HOME/bin:$PATH
####################################
# 使其生效
source ~/.bashrc
```

```
135 # Flume
136 export FLUME_HOME=/apps/flume
137 export PATH=$FLUME_HOME/bin:$PATH
```

- 1.4 修改配置文件

```
cd /apps/flume/conf
cp flume-env.sh.template flume-env.sh
vim flume-env.sh
# 修改22行为
export JAVA_HOME=/apps/java
```

```
22 export JAVA_HOME=/apps/java
```

- 1.5 测试

```
flume-ng version
```

```
chen@ubuntu:/apps/flume/conf$ flume-ng version
/apps/hadoop/bin/../libexec/hadoop-functions.sh: line 2326: HADOOP_ORG.APACHE.FLUME.TOO
LS.GETJAVAPROPERTY_USER: bad substitution
/apps/hadoop/bin/../libexec/hadoop-functions.sh: line 2421: HADOOP_ORG.APACHE.FLUME.TOO
LS.GETJAVAPROPERTY_OPTS: bad substitution
Error: Could not find or load main class org.apache.flume.tools.GetJavaProperty
Flume 1.9.0
Source code repository: https://git-wip-us.apache.org/repos/asf/flume.git
Revision: d4fcab4f501d41597bc616921329a4339f73585e
Compiled by fszabo on Mon Dec 17 20:45:25 CET 2018
From source with checksum 35db629a3bda49d23e9b3690c80737f9
```

解决办法

将/apps/hbase/conf/hbase-env.sh文件中对环境变量HBASE_CLASSPATH的设置注释掉

```
138 export JAVA_HOME=/apps/java
139 export HBASE_MANAGES_ZK=true
140 #export HBASE_CLASSPATH=/apps/hbase/conf
```

再测试，版本号为1.9.0

```
chen@ubuntu:/apps/hbase/conf$ flume-ng version
/apps/hadoop/bin/../libexec/hadoop-functions.sh: line 2326: HADOOP_ORG.APACHE.FLUME.TOO
LS.GETJAVAPROPERTY_USER: bad substitution
/apps/hadoop/bin/../libexec/hadoop-functions.sh: line 2421: HADOOP_ORG.APACHE.FLUME.TOO
LS.GETJAVAPROPERTY_OPTS: bad substitution
Flume 1.9.0
Source code repository: https://git-wip-us.apache.org/repos/asf/flume.git
Revision: d4fcab4f501d41597bc616921329a4339f73585e
Compiled by fszabo on Mon Dec 17 20:45:25 CET 2018
From source with checksum 35db629a3bda49d23e9b3690c80737f9
chen@ubuntu:/apps/hbase/conf$
```

二、Flume使用

一个source可以输出给多个channels，但一个sink只能接收一个channel

- 2. 1Avro Source

监听 Avro 端口来接受来自外部 Avro 客户端的 event 流。当与另一个 Flume agent 内 置的 Avro Sink 配对时，它可以创建分层收集结构。 Avro 是一个数据序列化系统，设计用于支持大批量数据交换的应用。它的主要特点是 支持二进制序列化方式，可以便捷，快速地处理大量数据；动态语言友好，Avro 提供的机制使动态语言可以方便地处理 Avro 数据 。

在 /apps/flume/conf目录下新建文件single_avro.conf,写入如下内容

```
# 配置一个agent,agent名称可以自定义
# 指定agent的sources、sinks、channels

# 分别指定agent的sources,sinks,channels的名称
a1.sources=s1
a1.sinks=k1
a1.channels=c1

# 为sources和sinks绑定channels,注意channel的单复数
a1.sources.s1.channels=c1
a1.sinks.k1.channel=c1
```

```
# 配置source
a1.sources.s1.type=avro
a1.sources.s1.bind=localhost
a1.sources.s1.port=6666

# 配置channels
a1.channels.c1.type=memory

# 配置sink
a1.sinks.k1.type=logger
```

通过下面命令启动flume

```
flume-ng agent --conf conf \
--conf-file /apps/flume/conf/single_avro.conf \
--name a1 \
-Dflume.root.logger=DEBUG,console \
-Dorg.apache.flume.log.printconfig=true \
-Dorg.apache.flume.log.rawdata=true
```

```
2020-11-17 18:15:45,005 WARN conf.FlumeConfiguration: Agent configuration for 'a1' has
no configfilters.
2020-11-17 18:15:45,035 INFO conf.FlumeConfiguration: Post-validation flume configurati
on contains configuration for agents: [a1]
2020-11-17 18:15:45,035 INFO node.AbstractConfigurationProvider: Creating channels
2020-11-17 18:15:45,046 INFO channel.DefaultChannelFactory: Creating instance of channe
l c1 type memory
2020-11-17 18:15:45,050 INFO node.AbstractConfigurationProvider: Created channel c1
2020-11-17 18:15:45,051 INFO source.DefaultSourceFactory: Creating instance of source s
1, type avro
2020-11-17 18:15:45,062 INFO sink.DefaultSinkFactory: Creating instance of sink: k1, ty
pe: logger
2020-11-17 18:15:45,065 INFO node.AbstractConfigurationProvider: Channel c1 connected t
o [s1, k1]
2020-11-17 18:15:45,078 INFO node.Application: Starting new configuration:{ sourceRunne
rs:{s1=EventDrivenSourceRunner: { source:Avro source s1: { bindAddress: localhost, port
: 6666 } }} sinkRunners:{k1=SinkRunner: { policy:org.apache.flume.sink.DefaultSinkProce
ssor@53f12e86 counterGroup:{ name:null counters:{} } }} channels:{c1=org.apache.flume.c
hannel.MemoryChannel{name: c1}} }
2020-11-17 18:15:45,078 INFO node.Application: Starting Channel c1
2020-11-17 18:15:45,125 INFO instrumentation.MonitoredCounterGroup: Monitored counter g
roup for type: CHANNEL, name: c1: Successfully registered new MBean.
2020-11-17 18:15:45,126 INFO instrumentation.MonitoredCounterGroup: Component type: CHA
NNEL, name: c1 started
2020-11-17 18:15:45,126 INFO node.Application: Starting Sink k1
2020-11-17 18:15:45,127 INFO node.Application: Starting Source s1
2020-11-17 18:15:45,128 INFO source.AvroSource: Starting Avro source s1: { bindAddress:
 localhost, port: 6666 }...
2020-11-17 18:15:45,326 INFO instrumentation.MonitoredCounterGroup: Monitored counter g
roup for type: SOURCE, name: s1: Successfully registered new MBean.
2020-11-17 18:15:45,326 INFO instrumentation.MonitoredCounterGroup: Component type: SOU
RCE, name: s1 started
2020-11-17 18:15:45,332 INFO source.AvroSource: Avro source s1 started.
```

另外打开一个终端，通过 Flume 提供的 avro 客户端向指定机器指定端口发送日志信息

```
flume-ng avro-client -c ~/apps/flume/conf -H localhost -p 6666 -F hello.txt
```

```
2020-11-17 18:26:04,021 INFO ipc.NettyServer: [id: 0xdf1f8fca, /127.0.0.1:52408 => /127
.0.0.1:6666] OPEN
2020-11-17 18:26:04,022 INFO ipc.NettyServer: [id: 0xdf1f8fca, /127.0.0.1:52408 => /127
.0.0.1:6666] BOUND: /127.0.0.1:6666
2020-11-17 18:26:04,022 INFO ipc.NettyServer: [id: 0xdf1f8fca, /127.0.0.1:52408 => /127
.0.0.1:6666] CONNECTED: /127.0.0.1:52408
2020-11-17 18:26:04,210 INFO ipc.NettyServer: [id: 0xdf1f8fca, /127.0.0.1:52408 :> /127
.0.0.1:6666] DISCONNECTED
2020-11-17 18:26:04,210 INFO ipc.NettyServer: [id: 0xdf1f8fca, /127.0.0.1:52408 :> /127
.0.0.1:6666] UNBOUND
2020-11-17 18:26:04,210 INFO ipc.NettyServer: [id: 0xdf1f8fca, /127.0.0.1:52408 :> /127
.0.0.1:6666] CLOSED
2020-11-17 18:26:04,210 INFO ipc.NettyServer: Connection to /127.0.0.1:52408 disconnect
ed.
2020-11-17 18:26:05,408 INFO sink.LoggerSink: Event: { headers:{} body: 77 65 6C 63 6F
6D 65 20 74 6F 20 74 68 65 20 77 welcome to the w }
```

但是信息不全welcome to the world!

- 2.2 Exec Source

ExecSource 的配置就是设定一个 Linux 命令，然后通过这个命令不断输出数据。如果 进程退出，ExecSource 也一起退出，不会产生进一步的数据。

保存一下内容到/apps/flume/conf/case_exec.conf

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

# bind source and sink to channel
a1.sources.s1.channels=c1
a1.sinks.k1.channel=c1

# config sources
a1.sources.s1.type=exec
a1.sources.s1.command=tail -F /home/chen/flume/logs/test.log

# config sinks
a1.sinks.k1.type=logger

# config channel
a1.channels.c1.type=memory
a1.channels.c1.capacity=1000 # 能够存储的event的最大个数
a1.channels.c1.transactionCapacity=100 # channel每次从source取或每次传给sink的event
的最大个数
```

使用tail -F 命令一直读日志的尾部。

启动flume

```
flume-ng agent --conf conf \
--conf-file /apps/flume/conf/case_exec.conf \
--name a1 \
-Dflume.root.logger=DEBUG,console \
-Dorg.apache.flume.log.printconfig=true \
-Dorg.apache.flume.log.rawdata=true
```

```
2020-11-17 18:39:39,393 INFO node.Application: Starting Sink k1
2020-11-17 18:39:39,394 INFO node.Application: Starting Source s1
2020-11-17 18:39:39,395 INFO source.ExecSource: Exec source starting with command: tail
 -F /home/chen/flume/logs/test.log
2020-11-17 18:39:39,395 INFO instrumentation.MonitoredCounterGroup: Monitored counter g
roup for type: SOURCE, name: s1: Successfully registered new MBean.
2020-11-17 18:39:39,396 INFO instrumentation.MonitoredCounterGroup: Component type: SOU
RCE, name: s1 started
```

另开一个ssh，向日志文件中添加一些内容

```
echo "welcome to hadoop home!">>test.log
echo "welcome to the world!">>test.log
```

```
2020-11-17 18:45:21,481 INFO sink.LoggerSink: Event: { headers:{} body: 77 65 6C 63 6F
6D 65 20 74 6F 20 68 61 64 6F 6F welcome to hadoo }
2020-11-17 18:45:25,484 INFO sink.LoggerSink: Event: { headers:{} body: 77 65 6C 63 6F
6D 65 20 74 6F 20 74 68 65 20 77 welcome to the w }
```

另一个终端可以接收到，但是不全

- 2.3 Spooling Directory Source

Spooling Directory Source 监视设置的目录，并将解析新文件的出现。Spool Source 有 2 个注意地方，第一个是拷贝到 spool 目录下的文件不可以再打开编辑，第二个是 spool 目录下不可包含相应的子目录。这个主要用途作为对日志的准实时监控。

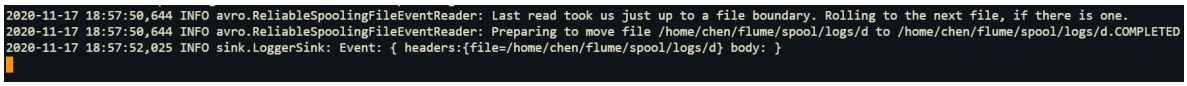
将下面的内容添加到文件/apps/flume/conf/case_spool.conf

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

# bind source and sink to channel
a1.sources.s1.channels=c1
a1.sinks.k1.channel=c1

# config sources
a1.sources.s1.type=spooldir
a1.sources.s1.spoolDir=/home/chen/flume/spool/logs  # 要提前创建该目录
a1.sources.s1.fileHeader=true

# config sinks
a1.sinks.k1.type=logger

# config channel
a1.channels.c1.type=memory
```

启动 Flume

```
flume-ng agent --conf conf \
--conf-file /apps/flume/conf/case_spool.conf \
--name a1 \
-Dflume.root.logger=INFO,console
```

另外打开一个终端，在监控目录/home/chen/flume/spool/logs 中新建三个文件 a,b,c,d，在 Flume 运行界 面会显示如下信息



- 2.4 NetCat Source

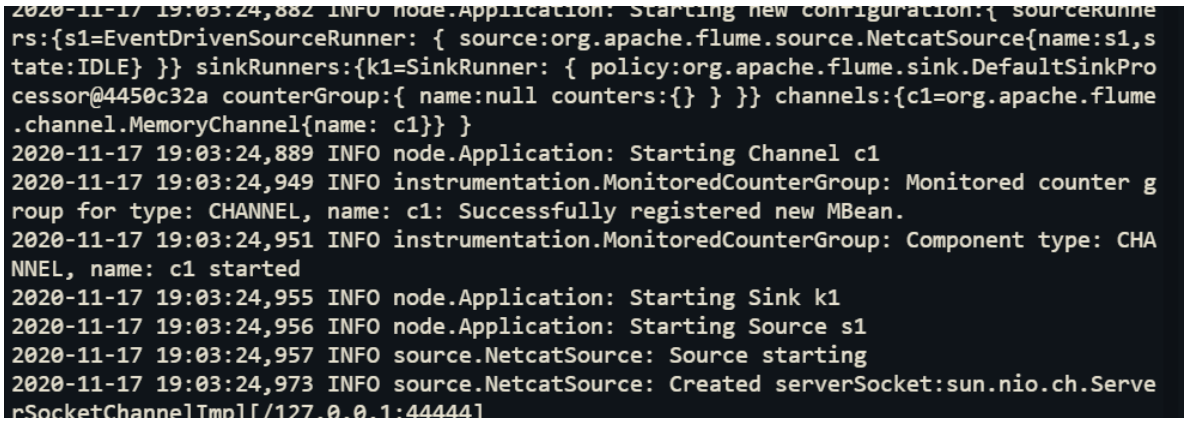
Netcat source 在某一端口上进行监听，并将接收到的数据每一行文字作为一个 event，也 就是数据是基于换行符分隔。并通过连接通道发送。

将下面的内容添加到文件/apps/flume/conf/case_netcat.conf

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

# bind source and sink to channel
a1.sources.s1.channels=c1
a1.sinks.k1.channel=c1

# config sources
a1.sources.s1.type=netcat
a1.sources.s1.bind=localhost
a1.sources.s1.port=44444

# config sinks
a1.sinks.k1.type=logger

# config channel
a1.channels.c1.type=memory
a1.channels.c1.capacity=1000
a1.channels.c1.transactionCapacity=100
```

启动Flume

```
flume-ng agent --conf conf \
--conf-file /apps/flume/conf/case_netcat.conf \
--name a1 \
-Dflume.root.logger=INFO,console
```



下面使用 telnet 向端口发送数据 ,先安装

```
sudo apt install openbsd-inetd
sudo apt install telnetd
```

查看运行状态

```
chen@ubuntu:~/flume/spool/logs$ sudo systemctl status openbsd-inetd
● inetd.service - Internet superserver
   Loaded: loaded (/lib/systemd/system/inetd.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2020-11-17 19:04:31 PST; 1min 2s ago
     Docs: man:inetd(8)
 Main PID: 10456 (inetd)
    Tasks: 1 (limit: 4633)
   CGroup: /system.slice/inetd.service
           └─10456 /usr/sbin/inetd

Nov 17 19:04:31 ubuntu systemd[1]: Starting Internet superserver...
Nov 17 19:04:31 ubuntu systemd[1]: Started Internet superserver.
chen@ubuntu:~/flume/spool/logs$
```

查看 telnet 服务是否开启

```
sudo apt install net-tools
sudo netstat -a | grep telnet
```

```
chen@ubuntu:~/flume/spool/logs$ sudo netstat -a | grep telnet
tcp        0      0 0.0.0.0:telnet          0.0.0.0:*               LISTEN
chen@ubuntu:~/flume/spool/logs$
```

发送数据

```
telnet localhost 44444
```

```
chen@ubuntu:~/flume/spool/logs$ telnet localhost 44444
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
^]
OK
hello flume!
OK
are you OK?
OK
```

另一边会接收到数据

```
2020-11-17 19:08:51,031 INFO sink.LoggerSink: Event: { headers:{} body: 5E 5D 0D
                                ^]. }
2020-11-17 19:08:57,153 INFO sink.LoggerSink: Event: { headers:{} body: 68 65 6C 6C 6F
20 66 6C 75 6D 65 21 0D          hello flume!. }
2020-11-17 19:09:19,161 INFO sink.LoggerSink: Event: { headers:{} body: 61 72 65 20 79
6F 75 20 4F 4B 3F 0D             are you OK?. }
```

- 2.5 Syslogtcp Source

Syslogtcp source 接收 tcp 协议发过来的数据

将下面的内容保存为/apps/flume/conf/case_syslogtcp.conf

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

```
# bind source and sink to channel
a1.sources.s1.channels=c1
a1.sinks.k1.channel=c1

# config sources
a1.sources.s1.type=syslogtcp
a1.sources.s1.port=9999
a1.sources.s1.host=localhost

# config sinks
a1.sinks.k1.type=logger

# config channel
a1.channels.c1.type=memory
```

启动 Flume

```
flume-ng agent --conf conf \
--conf-file /apps/flume/conf/case_syslogtcp.conf \
--name a1 \
-Dflume.root.logger=INFO,console
```

```
2020-11-17 19:13:34,151 INFO node.Application: Starting Channel c1
2020-11-17 19:13:34,203 INFO instrumentation.MonitoredCounterGroup: Monitored counter g
roup for type: CHANNEL, name: c1: Successfully registered new MBean.
2020-11-17 19:13:34,203 INFO instrumentation.MonitoredCounterGroup: Component type: CHA
NNEL, name: c1 started
2020-11-17 19:13:34,203 INFO node.Application: Starting Sink k1
2020-11-17 19:13:34,204 INFO node.Application: Starting Source s1
2020-11-17 19:13:34,277 INFO source.SyslogTcpSource: Syslog TCP Source starting...
2020-11-17 19:13:34,295 INFO instrumentation.MonitoredCounterGroup: Monitored counter g
roup for type: SOURCE, name: s1: Successfully registered new MBean.
2020-11-17 19:13:34,295 INFO instrumentation.MonitoredCounterGroup: Component type: SOU
RCE, name: s1 started
```

另外打开一个终端，向端口发送内容

```
echo "hello flume" | nc localhost 9999
```

在 flume 的启动界面就会输出接收到的数据

```
2020-11-17 19:14:31,919 WARN source.SyslogUtils: Event created from Invalid Syslog data
.
2020-11-17 19:14:36,225 INFO sink.LoggerSink: Event: { headers:{Severity=0, Facility=0,
 flume.syslog.status=Invalid} body: 68 65 6C 6C 6F 20 66 6C 75 6D 65             hel
lo flume }
```

- 2.6 Hdfs Sink

将下面的内容保存为/apps/flume/conf/case_syslogtcp_hdfs.conf

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

# bind source and sink to channel
a1.sources.s1.channels=c1
a1.sinks.k1.channel=c1
```

```
# config sources
a1.sources.s1.type=syslogtcp
a1.sources.s1.port=9999
a1.sources.s1.host=localhost

# config sinks
a1.sinks.k1.type=hdfs
a1.sinks.k1.hdfs.path=hdfs://localhost:9000/flume
a1.sinks.k1.hdfs.fileType=DataStream

# config channel
a1.channels.c1.type=memory
```

启动Hadoop

```
start-all.sh
```

启动 Flume

```
flume-ng agent --conf conf \
--conf-file /apps/flume/conf/case_syslogtcp_hdfs.conf \
--name a1 \
-Dflume.root.logger=INFO,console
```

另外打开一个终端，向端口发送内容

```
echo "hello flume" | nc localhost 9999
```

在 flume 的启动界面可以看到在 HDFS 上创建了一个文件

```
2020-11-17 19:21:09,233 WARN source.SyslogUtils: Event created from Invalid Syslog data
.
2020-11-17 19:21:09,360 INFO hdfs.HDFSDataStream: Serializer = TEXT, UseRawLocalFileSys
tem = false
2020-11-17 19:21:09,471 INFO hdfs.BucketWriter: Creating hdfs://localhost:9000/flume/Fl
umeData.1605669669361.tmp
```

查看其中的内容，确认就是我们发送的字符串。

```
chen@ubuntu:~/flume/spool/logs$ hadoop fs -ls /flume/*
-rw-r--r--   1 chen supergroup         12 2020-11-17 19:21 /flume/FlumeData.160566966931
61
chen@ubuntu:~/flume/spool/logs$
```

- 2.7配置多代理流程

设置一个多层的流程，第一个 agent 需要有一个 Avro sink 指向第二个 agent 的 Avro 源。 第一个 agent 将转发 Event 到下一个 agent。 创建文件/apps/flume/conf/case_avro_sink.conf

```
# Name the components on this agent
a2.sources = s1
a2.sinks = k1
a2.channels = c1
# Bind the source and sink to the channel
a2.sources.s1.channels = c1
a2.sinks.k1.channel = c1
# Describe/configure the source
```

```
a2.sources.s1.type = syslogtcp
a2.sources.s1.host = localhost
a2.sources.s1.port = 33333
# Describe the channel
a2.channels.c1.type = memory
a2.channels.c1.capacity = 1000
a2.channels.c1.transactionCapacity = 100
# Describe the sink
a2.sinks.k1.type = avro
a2.sinks.k1.hostname = localhost
a2.sinks.k1.port = 22222
```

创建文件 /apps/flume/conf/case_avro.conf

```
# Name the components on this agent
a1.sources = s1
a1.sinks = k1
a1.channels = c1
# Bind the source and sink to the channel
a1.sources.s1.channels = c1
a1.sinks.k1.channel = c1
# Describe/configure the source
a1.sources.s1.type = avro
a1.sources.s1.bind = localhost
a1.sources.s1.port = 22222
# Describe the channel
a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100
# Describe the sink
a1.sinks.k1.type = logger
```

注意 case_avro_sink.conf 是前面的 Agent，case_avro.conf 是后面的 Agent.

先在一个终端中启动 Avro 的 Source,监听端口

```
flume-ng agent --conf conf --conf-file \
/apps/flume/conf/case_avro.conf --name a1 \
-Dflume.root.logger=DEBUG,console \
-Dorg.apache.flume.log.printconfig=true \
-Dorg.apache.flume.log.rawdata=true
```

再在另一个终端中启动 Avro 的 Sink

```
flume-ng agent --conf conf --conf-file \
/apps/flume/conf/case_avro_sink.conf --name a2 \
-Dflume.root.logger=DEBUG,console \
-Dorg.apache.flume.log.printconfig=true \
-Dorg.apache.flume.log.rawdata=true
```

再开一个终端中发送数据

```
echo "hello flume avro sink" | nc localhost 33333
```

可以看到已经建立连接，第二次 agent 成功接收到数据

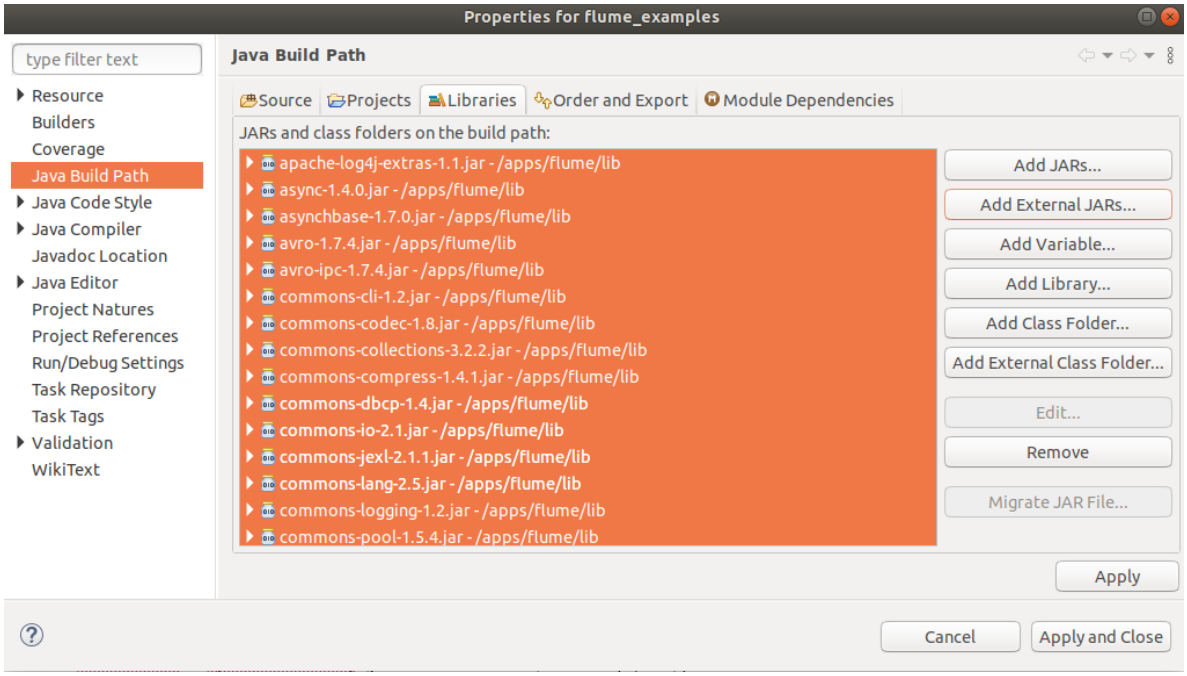
```
2020-11-17 19:29:00,435 INFO ipc.NettyServer: [id: 0xee79dc70, /127.0.0.1:53906 => /127
.0.0.1:22222] CONNECTED: /127.0.0.1:53906
2020-11-17 19:29:51,446 INFO sink.LoggerSink: Event: { headers:{Severity=0, Facility=0,
 flume.syslog.status=Invalid} body: 68 65 6C 6C 6F 20 66 6C 75 6D 65 20 61 76 72 6F hel
lo flume avro }
```

三、Flume AVRO Client开发

创建工程【flume_examples】,包【sds.flume】,类【AVRO_Client】，将下面的代码复 制其中。

```
```

将/apps/flume/lib 中的 jar 包全部导入工程中。



在/apps/flume/conf 中创建文件 avro_client.conf

```
# Name the components on this agent
a1.sources = s1
a1.sinks = k1
a1.channels = c1
# Describe/configure the source
a1.sources.s1.type = avro
a1.sources.s1.port = 42424
a1.sources.s1.bind = localhost
a1.sources.s1.channels = c1
# Describe the sink
a1.sinks.k1.channel = c1
a1.sinks.k1.type = logger
# Use a channel which buffers events inmemory
a1.channels.c1.type = memory
#a1.channels.c1.capacity = 1000
#a1.channels.c1.transactionCapacity = 100
```

在终端中启动 Flume

```
flume-ng agent -conf conf --conf-file \
/apps/flume/conf/avro_client.conf --name a1 \
-Dflume.root.logger=INFO,console
```

启动成功后，在 eclipse 中执行 AVRO_client 类，在终端 Flume 启动界面可以看到输出了

```
2020-11-17 19:54:06,603 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,605 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,608 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,611 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,615 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,620 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,621 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,621 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,621 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
2020-11-17 19:54:06,621 INFO sink.LoggerSink: Event: { headers:{} body: 48 65 6C 6C 6F
20 46 6C 75 6D 65 21                   Hello Flume! }
```