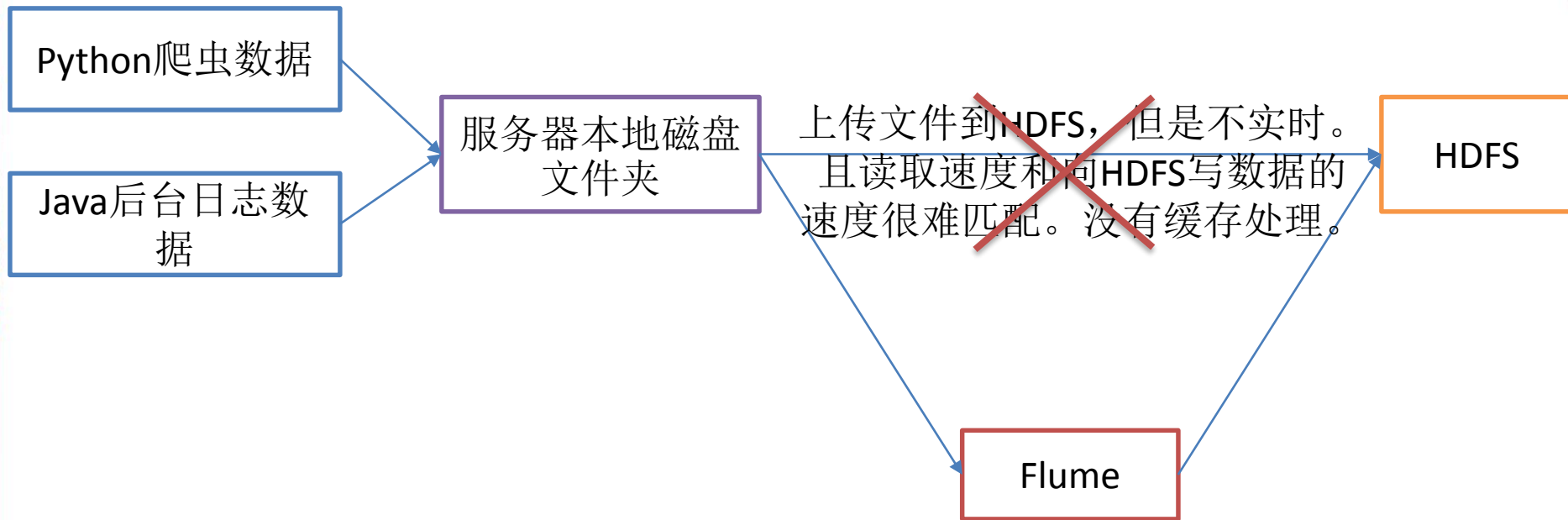




大数据技术图解

尚硅谷大数据研究院

为什么选用Flume



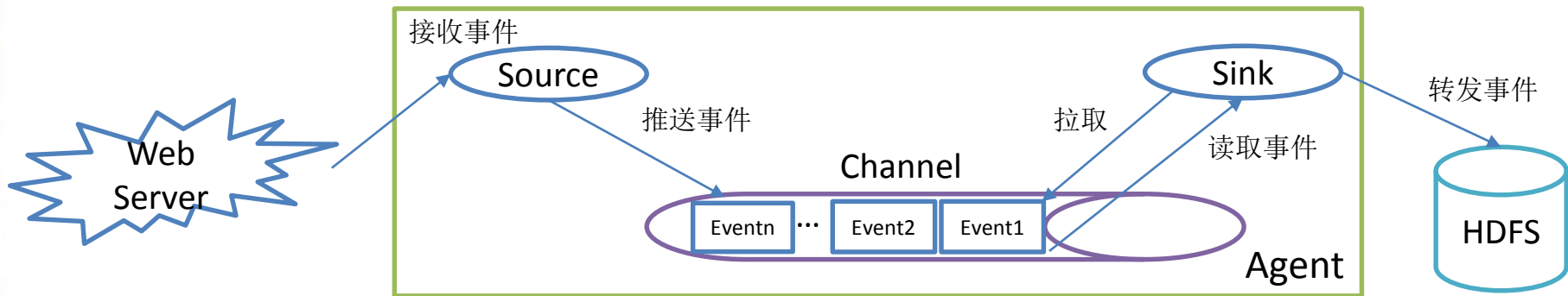
Flume最主要的作用就是，实时读取服务器本地磁盘的数据，将数据写入到HDFS。

Flume组成架构

数据输入端

Flume流式处理

数据输出端



Source数据输入端的类型有：avro、thrift、exec、jms、spooling directory、netcat、sequence generator、syslog、http、legacy等。但是目前在企业中使用最广泛的就是日志文件。

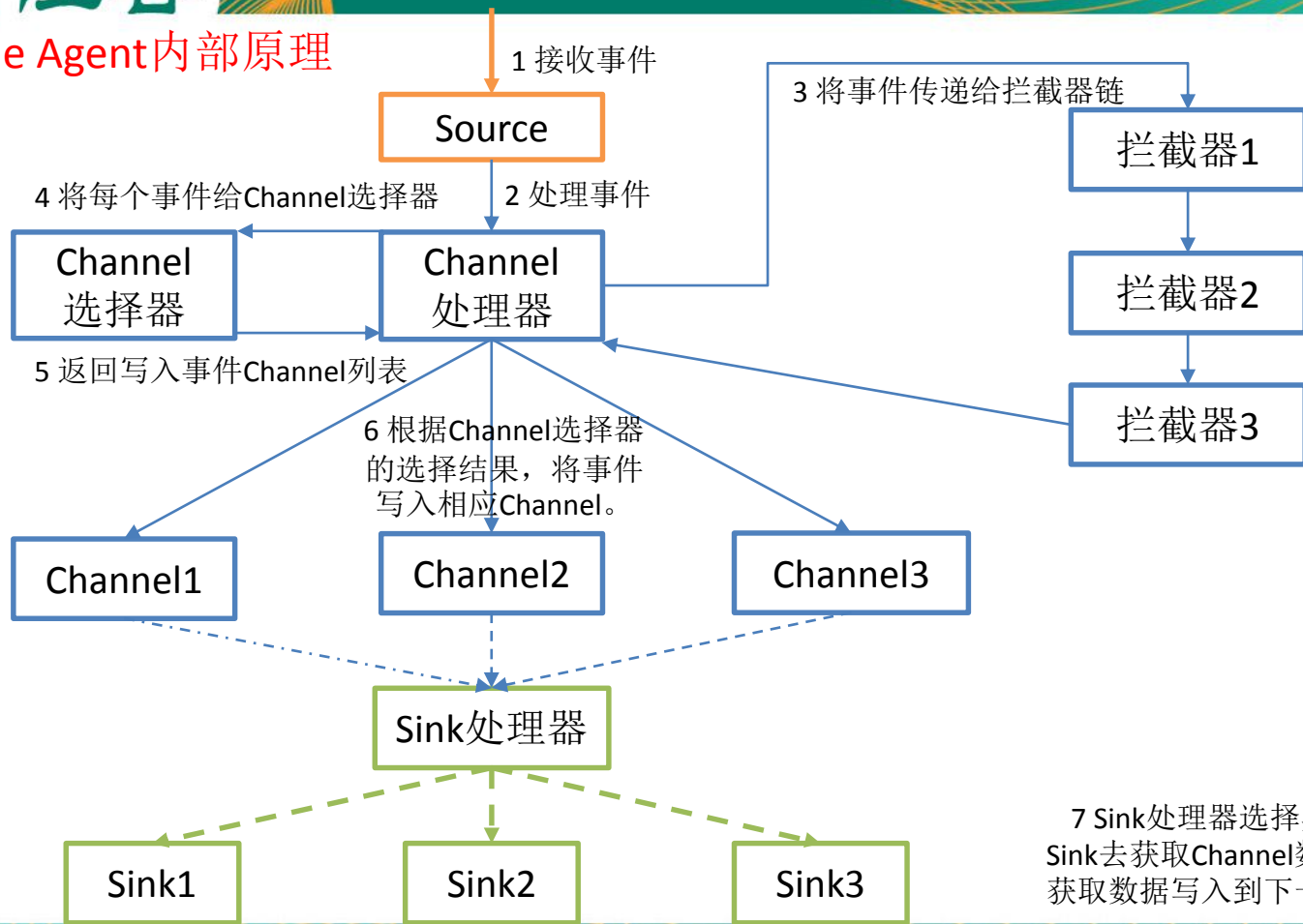
Channel是位于**Source**和**Sink**之间的缓冲区。

Flume自带两种Channel：Memory Channel和File Channel。

Memory Channel是基于内存缓存，在不需关心数据丢失的情景下适用。

File Channel是Flume的持久化Channel。系统宕机不会丢失数据。

Sink组件目的地包括hdfs、kafka、logger、avro、thrift、ipc、file、null、HBase、solr、自定义。但是目前在企业中使用最广泛的是HDFS和Kafka。





监听数据端口案例分析

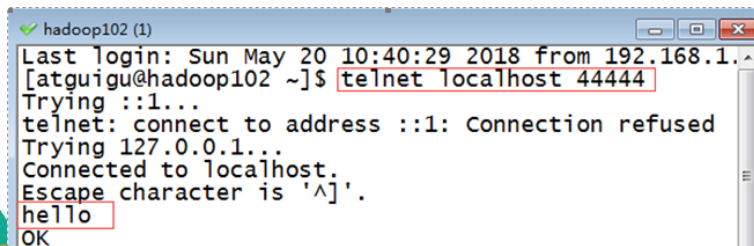
1 通过telnet工具向
localhost主机的44444端
口发送数据

```
$ telnet localhost  
  
44444
```

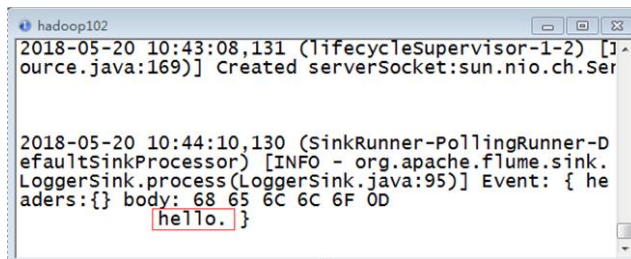
2 Flume监控localhost
主机的44444端口数据。
通过Flume的source端
读取数据。

```
$ bin/flume-ng agent --conf conf/  
--name a1  
--conf-file job/flume-telnet.conf  
-Dflume.root.logger==INFO,console
```

3 Flume将获取的数
据通过Sink端写出
到控制台



```
hadoop102 (1)  
Last login: Sun May 20 10:40:29 2018 from 192.168.1.1  
[atguigu@hadoop102 ~]$ telnet localhost 44444  
Trying ::1...  
telnet: connect to address ::1: Connection refused  
Trying 127.0.0.1...  
Connected to localhost.  
Escape character is '^['.  
hello  
OK
```



```
hadoop102  
2018-05-20 10:43:08,131 (LifecycleSupervisor-1-2) [J  
ource.java:169]] Created serverSocket:sun.nio.ch.Ser  
  
2018-05-20 10:44:10,130 (SinkRunner-PollingRunner-D  
efaultSinkProcessor) [INFO - org.apache.flume.sink.  
LoggerSink.process(LoggerSink.java:95)] Event: { he  
aders:{} body: 68 65 6C 6C 6F 0D  
hello. }
```


配置文件解析

```
# Name the components on this agent
```

a1:表示agent的名称

```
a1.sources = r1
```

r1:表示a1的输入源

```
a1.sinks = k1
```

k1:表示a1的输出目的地

```
a1.channels = c1
```

c1:表示a1的缓冲区

```
# Describe/configure the source
```

```
a1.sources.r1.type = netcat
```

表示a1的输入源类型为netcat类型

```
a1.sources.r1.bind = localhost
```

表示a1的监听的主机

```
a1.sources.r1.port = 44444
```

表示a1的监听的端口号

```
# Describe the sink
```

```
a1.sinks.k1.type = logger
```

表示a1的输出目的地是logger类型

```
# Use a channel which buffers events in memory
```

```
a1.channels.c1.type = memory
```

表示a1的channel类型是memory内存型

```
a1.channels.c1.capacity = 1000
```

表示a1的channel总容量1000

```
a1.channels.c1.transactionCapacity = 100
```

表示a1的channel传输总容量100

```
# Bind the source and sink to the channel
```

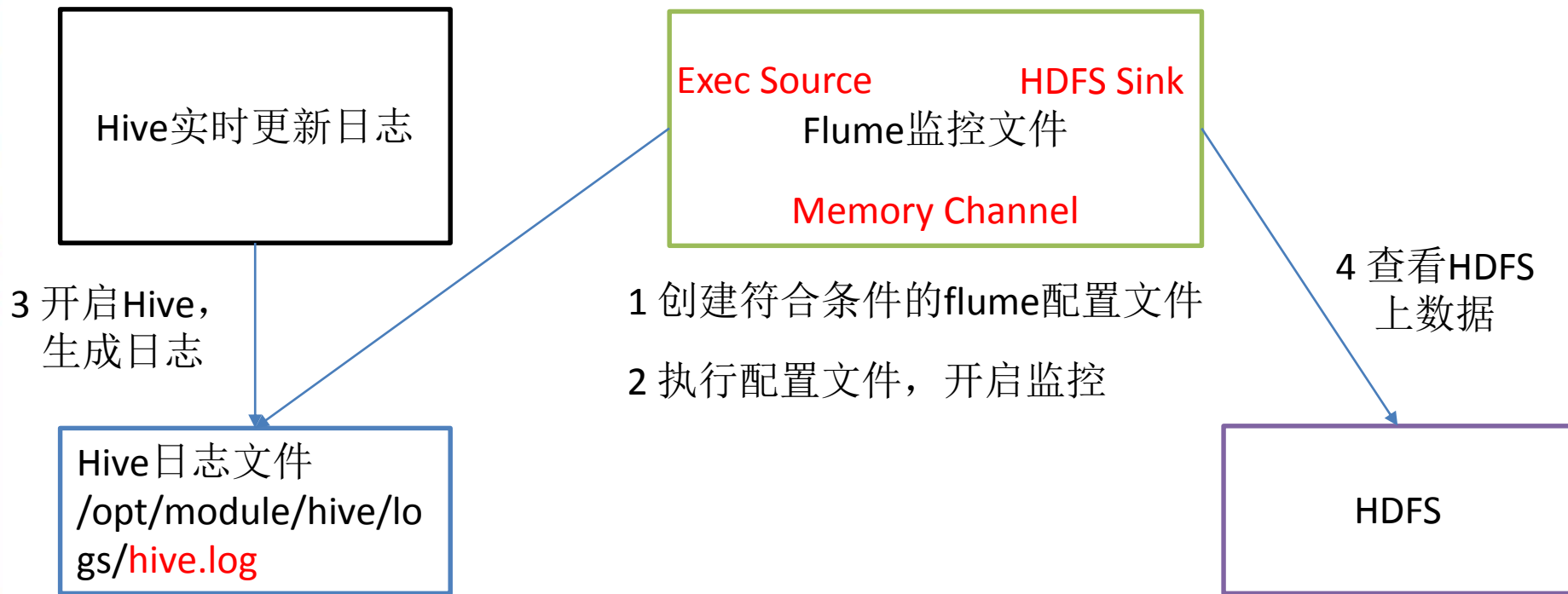
```
a1.sources.r1.channels = c1
```

表示将r1和c1连接起来

```
a1.sinks.k1.channel = c1
```

表示将k1和c1连接起来

实时读取本地文件到HDFS案例





实时读取本地文件到HDFS案例

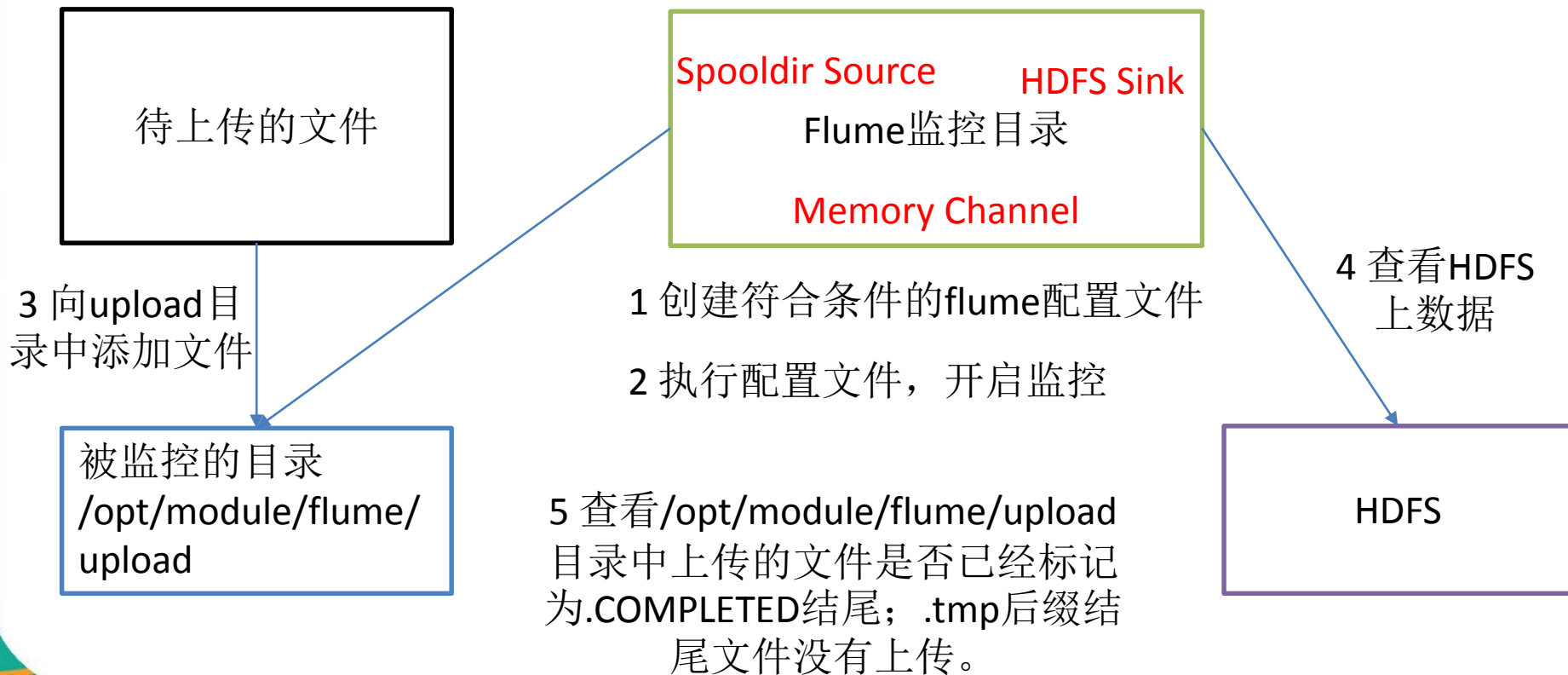
```
# Name the components on this agent
a2.sources = r2           #定义source
a2.sinks = k2             #定义sink
a2.channels = c2          #定义channel
# Describe/configure the source
a2.sources.r2.type = exec  #定义source类型为exec可执行命令的
a2.sources.r2.command = tail -F /opt/module/hive/logs/hive.log
a2.sources.r2.shell = /bin/bash -c #执行shell脚本的绝对路径
```

```
# Describe the sink
a2.sinks.k2.type = hdfs
a2.sinks.k2.hdfs.path = hdfs://hadoop102:9000/flume/%Y%m%d/%H
a2.sinks.k2.hdfs.filePrefix = logs-      #上传文件的前缀
a2.sinks.k2.hdfs.round = true            #是否按照时间滚动文件夹
a2.sinks.k2.hdfs.roundValue = 1          #多少时间单位创建一个新的文件夹
a2.sinks.k2.hdfs.roundUnit = hour        #重新定义时间单位
a2.sinks.k2.hdfs.useLocalTimeStamp = true #是否使用本地时间戳
a2.sinks.k2.hdfs.batchSize = 1000        #积攒多少个Event才flush到HDFS一次
a2.sinks.k2.hdfs.fileType = DataStream  #设置文件类型，可支持压缩
a2.sinks.k2.hdfs.rollInterval = 600      #多久生成一个新的文件
a2.sinks.k2.hdfs.rollSize = 134217700   #设置每个文件的滚动大小
a2.sinks.k2.hdfs.rollCount = 0           #文件的滚动与Event数量无关
a2.sinks.k2.hdfs.minBlockReplicas = 1    #最小冗余数
```

```
# Use a channel which buffers events in memory
a2.channels.c2.type = memory
a2.channels.c2.capacity = 1000
a2.channels.c2.transactionCapacity = 100

# Bind the source and sink to the channel
a2.sources.r2.channels = c2
a2.sinks.k2.channel = c2
```


实时读取目录文件到HDFS案例





实时读取目录文件到HDFS案例

```
a3.sources = r3          #定义source
a3.sinks = k3            #定义sink
a3.channels = c3         #定义channel
```

```
# Describe/configure the source
```

```
a3.sources.r3.type = spooldir    #定义source类型为目录
a3.sources.r3.spoolDir = /opt/module/flume/upload #定义监控目录
a3.sources.r3.fileSuffix = .COMPLETED #定义文件上传完, 后缀
a3.sources.r3.fileHeader = true   #是否有文件头
a3.sources.r3.ignorePattern = ([^]*\\.tmp)
```

```
#忽略所有以.tmp结尾的文件, 不上传
```

```
# Describe the sink
```

```
a3.sinks.k3.type = hdfs    #sink类型为hdfs
```

```
a3.sinks.k3.hdfs.path =
```

```
hdfs://hadoop102:9000/flume/upload/%Y%m%d/%H #文件上传到hdfs的路径
```

```
a3.sinks.k3.hdfs.filePrefix = upload-    #上传文件到hdfs的前缀
```

```
a3.sinks.k3.hdfs.round = true            #是否按时间滚动文件
```

```
a3.sinks.k3.hdfs.roundValue = 1          #多少时间单位创建一个新的文件夹
```

```
a3.sinks.k3.hdfs.roundUnit = hour        #重新定义时间单位
```

```
a3.sinks.k3.hdfs.useLocalTimeStamp = true #是否使用本地时间戳
```

```
a3.sinks.k3.hdfs.batchSize = 100         #积攒多少个Event才flush到HDFS一次
```

```
a3.sinks.k3.hdfs.fileType = DataStream   #设置文件类型, 可支持压缩
```

```
a3.sinks.k3.hdfs.rollInterval = 600 #多久生成新文件
```

```
a3.sinks.k3.hdfs.rollSize = 134217700 #多大生成新文件
```

```
a3.sinks.k3.hdfs.rollCount = 0 #多少event生成新文件
```

```
a3.sinks.k3.hdfs.minBlockReplicas = 1 #多少副本数
```

```
# Use a channel which buffers events in memory
```

```
a3.channels.c3.type = memory
```

```
a3.channels.c3.capacity = 1000
```

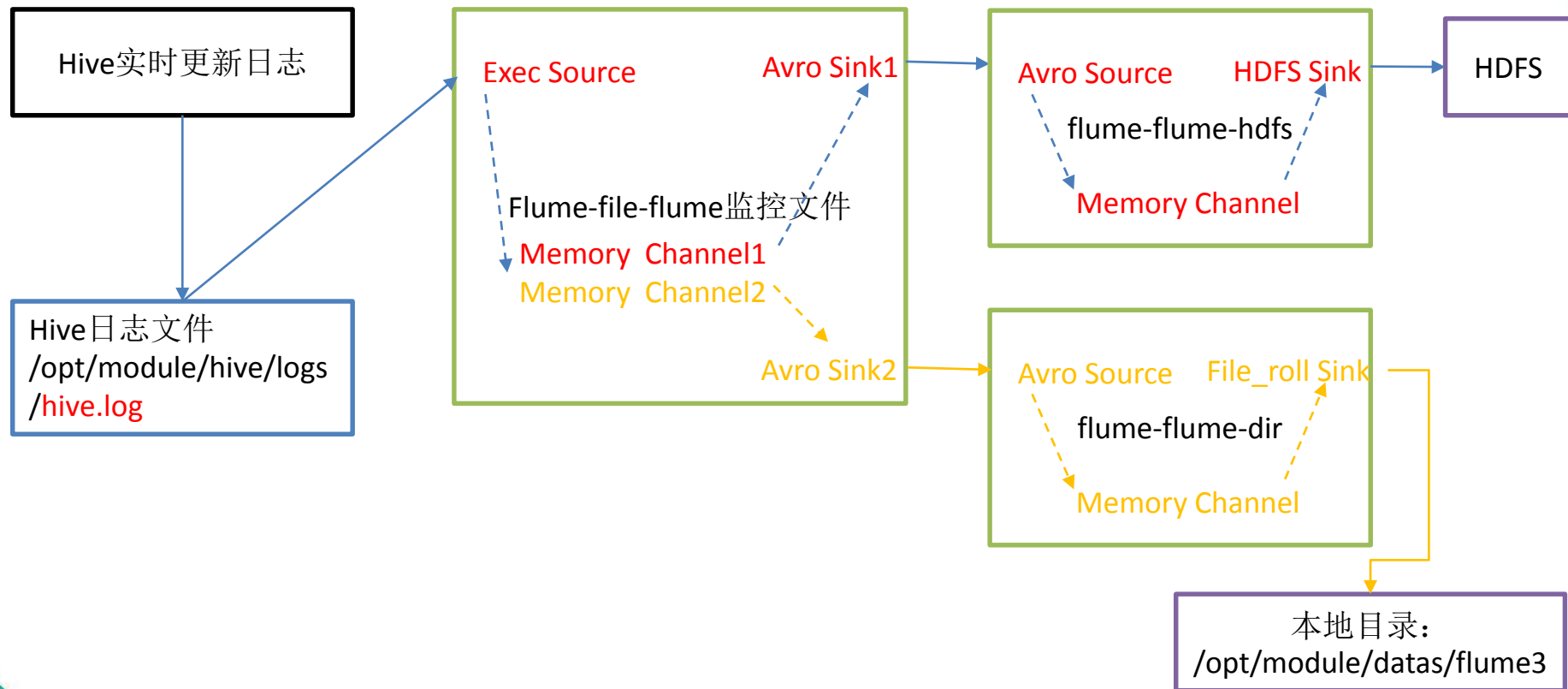
```
a3.channels.c3.transactionCapacity = 100
```

```
# Bind the source and sink to the channel
```

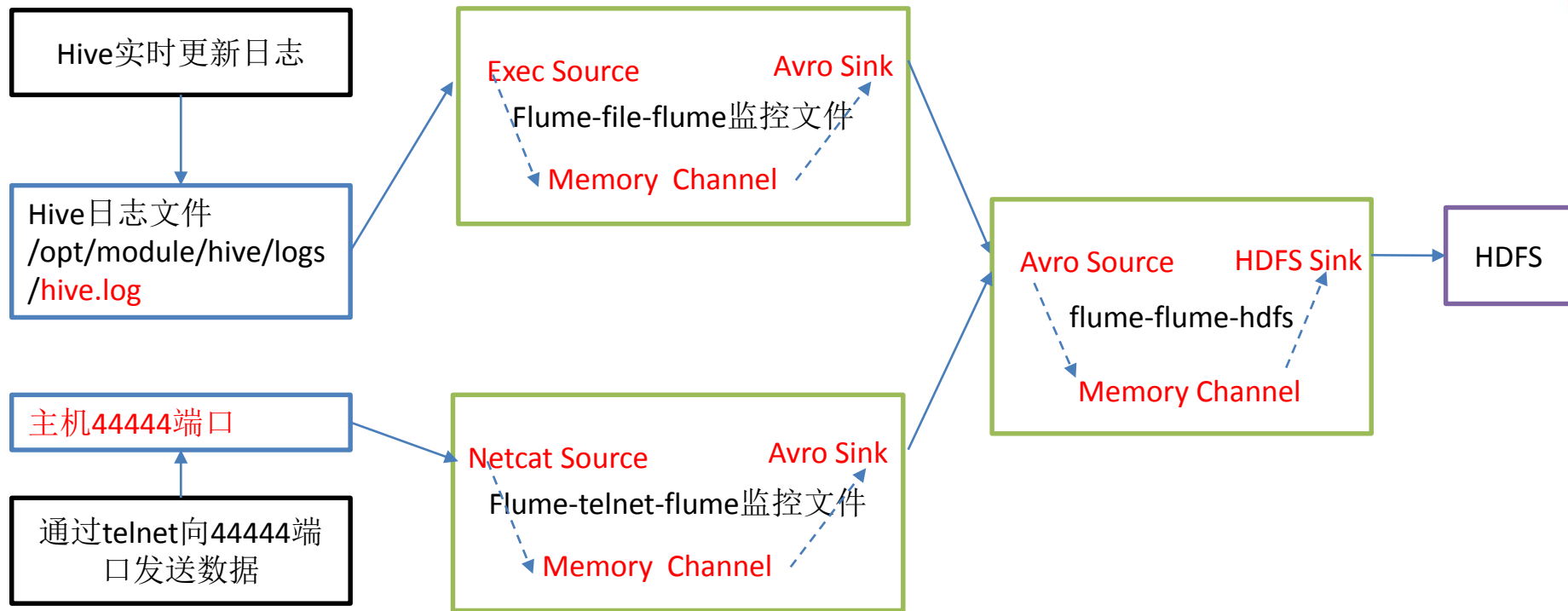
```
a3.sources.r3.channels = c3
```

```
a3.sinks.k3.channel = c3
```

单数据源多出口案例



多数据源汇总案例





谢谢！ 欢迎收看！