

# 大数据Hadoop高薪直通车课程

## 文件收集工具Flume

讲师：轩宇（北风网版权所有）

# 课程大纲

1

**Flume 设计架构**

2

**Flume 初步使用**

3

**Source/Channel/Sink**

4

**Flume 项目架构**

5

**Flume 实战案例**

# 课程大纲

1

**Flume 设计架构**

2

**Flume 初步使用**

3

**Source/Channel/Sink**

4

**Flume 项目架构**

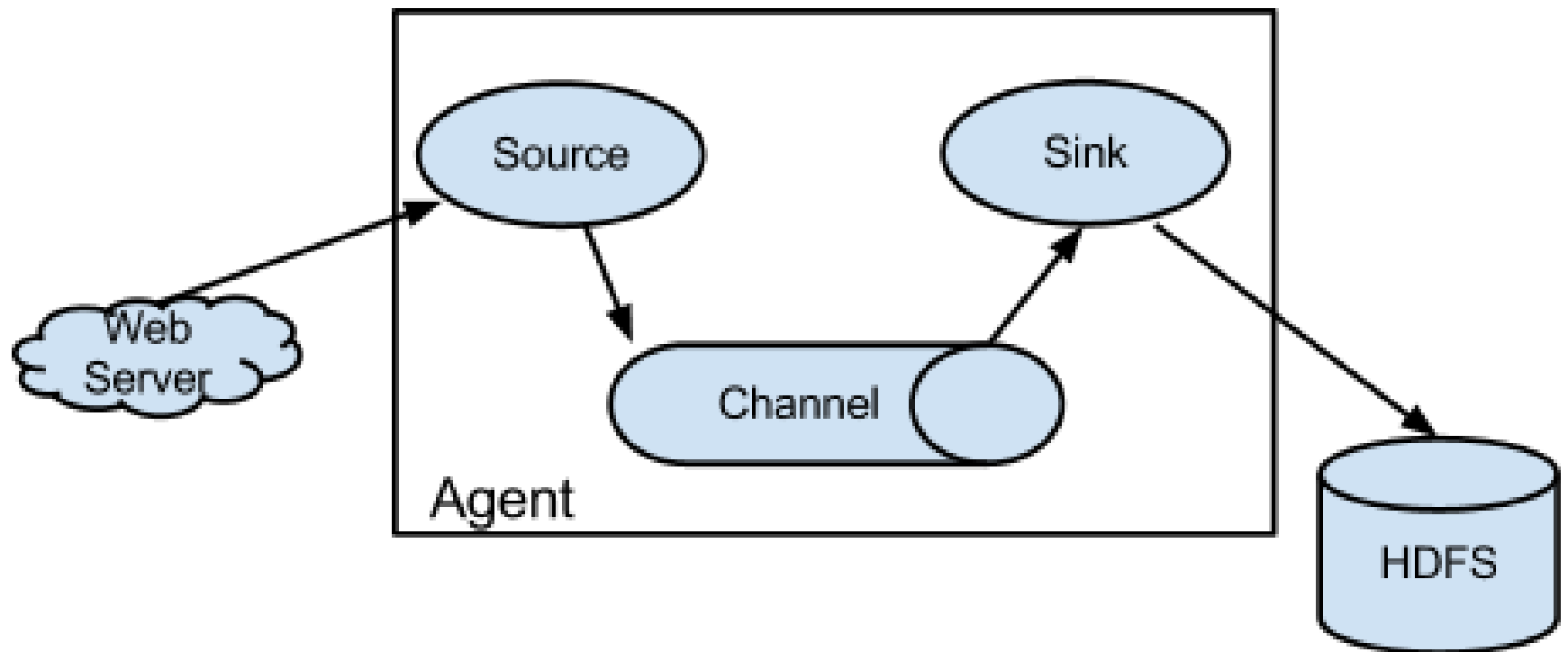
5

**Flume 实战案例**

# Apache Flume

- ◆ Flume is a **distributed, reliable, and available** service for efficiently **collecting, aggregating, and moving** large amounts of log data.
- ◆ It has a simple and flexible architecture based **on streaming data flows**. It is **robust** (健壮) **and fault tolerant** (容错) with tunable reliability mechanisms and many **failover and recovery** mechanisms.
- ◆ It uses a simple extensible data model that allows for **online analytic application**.

# Apache Flume



# Apache Flume

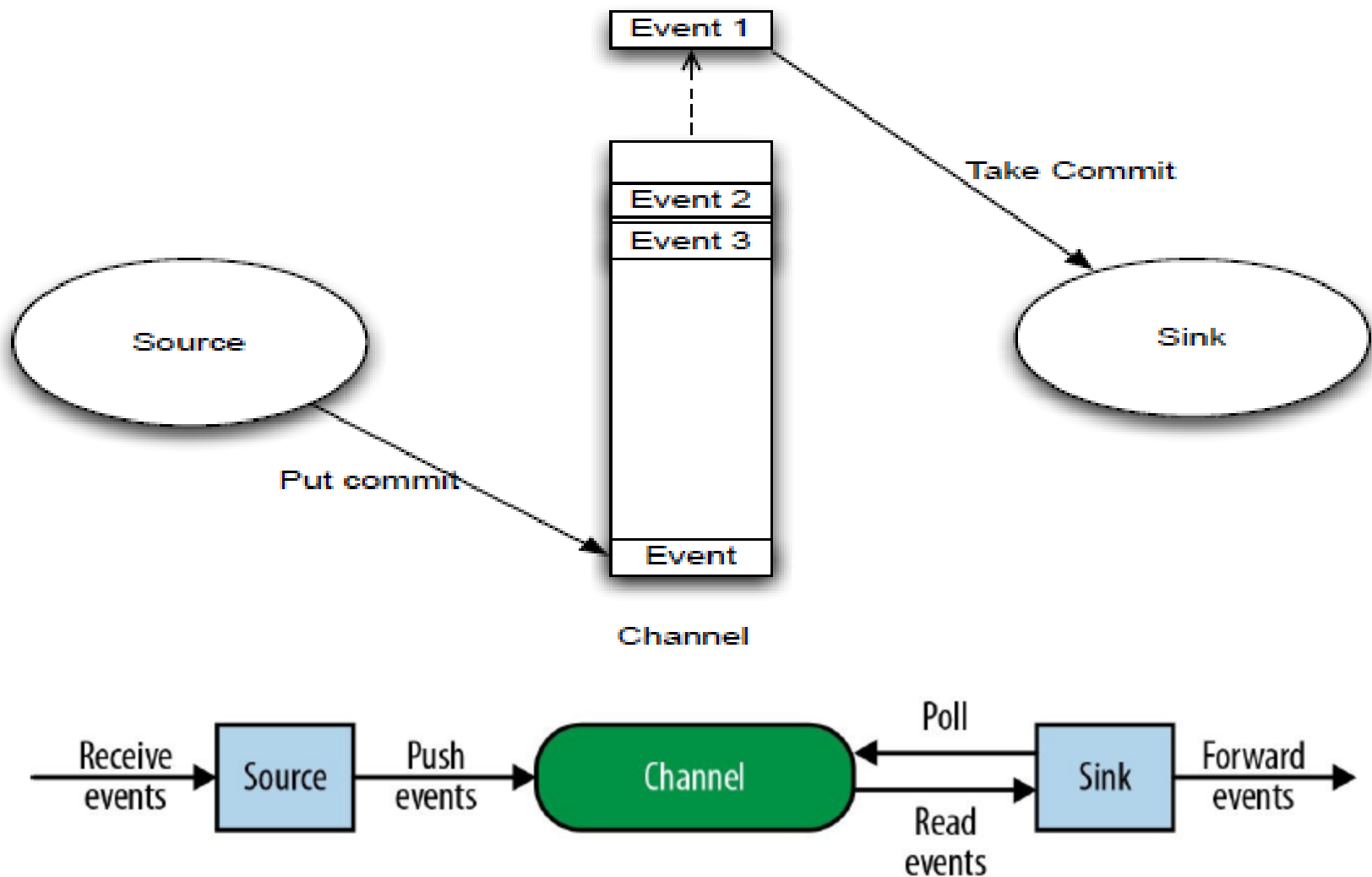
◆Flume-ng只有一个角色的节点：agent的角色，agent有source、channel、sink组成。

角色	简介
Source	Source用于采集数据，Source是产生数据流的地方，同时Source会将产生的数据流传输到Channel
Channel	连接 sources 和 sinks ，这个有点像一个队列。
Sink	从Channel收集数据，将数据写到目标源，可以是下一个Source也可以是HDFS或者HBase。

# Events

- ◆ Event是Flume数据传输的基本单元
- ◆ Flume以事件的形式将数据从源头传送到最终的目的
- ◆ Event由可选的header和载有数据的一个byte array构成
  - 载有的数据对flume是不透明的
  - Header是容纳了key-value字符串对的无序集合，key在集合内是唯一的。
  - Header可以在上下文路由中使用扩展

# Channel/Event/Sink



**A Flume agent with one flow**

**Figure 2-1. A simple Flume agent with one source, channel, and sink**



# 课程大纲

1

**Flume 设计架构**

2

**Flume 初步使用**

3

**Source/Channel/Sink**

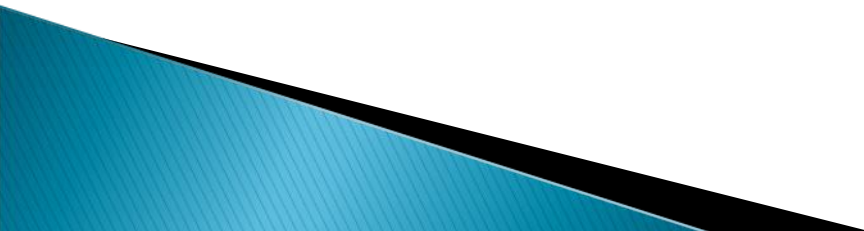
4

**Flume 项目架构**

5

**Flume 实战案例**

# System Requirements

- ◆ **Java Runtime Environment** - Java 1.6 or later (Java 1.7 Recommended)
  - ◆ **Memory** - Sufficient memory for configurations used by sources, channels or sinks
  - ◆ **Disk Space** - Sufficient disk space for configurations used by channels or sinks
  - ◆ **Directory Permissions** - Read/Write permissions for directories used by agent
- 

# Starting an agent

First, explode the .tar archive of the binary distribution we downloaded earlier:

```
$ tar -zxf apache-flume-1.5.2-bin.tar.gz  
$ cd apache-flume-1.5.2-bin
```

Next, let's briefly look at the help. Run the flume-ng command with the help command:

```
$ ./bin/flume-ng help  
Usage: ./bin/flume-ng <command> [options]...
```

commands:

help	display this help text
agent	run a Flume agent

# Starting an agent

An agent is started using a shell script called `flume-ng` which is located in the `bin` directory of the Flume distribution. You need to specify the agent name, the config directory, and the config file on the command line:

```
$ bin/flume-ng agent -n $agent_name -c conf -f conf/flume-conf.properties.template
```

Now the agent will start running source and sinks configured in the given properties file.

# Starting an agent

```
# example.conf: A single-node Flume configuration

# Name the components on this agent
al.sources = r1
al.sinks = k1
al.channels = c1

# Describe/configure the source
al.sources.r1.type = netcat
al.sources.r1.bind = localhost
al.sources.r1.port = 44444

# Describe the sink
al.sinks.k1.type = logger

# Use a channel which buffers events in memory
al.channels.c1.type = memory
al.channels.c1.capacity = 1000
al.channels.c1.transactionCapacity = 100

# Bind the source and sink to the channel
al.sources.r1.channels = c1
al.sinks.k1.channel = c1
```

# Starting an agent

This configuration defines a single agent named `a1`. `a1` has a source that listens for data on port 44444, a channel that buffers event data in memory, and a sink that logs event data to the console. The configuration file names the various components, then describes their types and configuration parameters. A given configuration file might define several named agents; when a given Flume process is launched a flag is passed telling it which named agent to manifest.

Given this configuration file, we can start Flume as follows:

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

Note that in a full deployment we would typically include one more option: `--conf=<conf-dir>`. The `<conf-dir>` directory would include a shell script `flume-env.sh` and potentially a `log4j` properties file. In this example, we pass a Java option to force Flume to log to the console and we go without a custom environment script.

From a separate terminal, we can then telnet port 44444 and send Flume an event:

```
$ telnet localhost 44444
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
Hello world! <ENTER>
OK
```

# Starting an agent

## ◆ 进入 **【FLUME\_HOME/conf】** 目录下执行启动命令

```
bin/flume-ng agent \  
-c /usr/local/cdh-5.2.0/flume-1.5.0/conf \  
-f /usr/local/cdh-5.2.0/flume-1.5.0/conf/flume_hdfs.conf \  
-n agent1 \  
-Dflume.root.logger=INFO,console
```

## ◆ 参数说明：

- -c或者--conf 后跟配置目录
- -f或者--conf-file 后跟具体的配置文件
- -n或者--name 指定agent的名称

# 课程大纲

1

Flume 设计架构

2

Flume 初步使用

3

Source/Channel/Sink

4

Flume 项目架构

5

Flume 实战案例



# Flume—Sources

<http://flume.apache.org/FlumeUserGuide.html#flume-sources>

- Avro Source
- Thrift Source
- Exec Source
- JMS Source
  - Converter
- Spooling Directory Source
  - Event Deserializers
    - LINE
    - AVRO
    - BlobDeserialize

- Twitter 1% firehose Source (experimental)
- Kafka Source
- NetCat Source
- Sequence Generator Source
- Syslog Sources
  - Syslog TCP Source
  - Multiport Syslog TCP Source
  - Syslog UDP Source

- HTTP Source
  - JSONHandler
  - BlobHandler
- Stress Source
- Legacy Sources
  - Avro Legacy Source
  - Thrift Legacy Source
- Custom Source
- Scribe Source

# Exec Source

Property Name	Default	Description
channels	-	
type	-	The component type name, needs to be <code>exec</code>
command	-	The command to execute
shell	-	A shell invocation used to run the command. e.g. <code>/bin/sh -c</code> . Required only for commands relying on shell features like wildcards, back ticks, pipes etc.
restartThrottle	10000	Amount of time (in millis) to wait before attempting a restart
restart	false	Whether the executed cmd should be restarted if it dies
logStdErr	false	Whether the command's stderr should be logged
batchSize	20	The max number of lines to read and send to the channel at a time
selector.type	replicating	replicating or multiplexing
selector.*		Depends on the selector.type value
interceptors	-	Space-separated list of interceptors
interceptors.*		

# Spooling Directory Source

Property Name	Default	Description
<code>channels</code>	<code>-</code>	
<code>type</code>	<code>-</code>	The component type name, needs to be <code>spoolDir</code> .
<code>spoolDir</code>	<code>-</code>	The directory from which to read files from.
<code>fileSuffix</code>	<code>.COMPLETED</code>	Suffix to append to completely ingested files
<code>deletePolicy</code>	<code>never</code>	When to delete completed files: <code>never</code> or <code>immediate</code>
<code>fileHeader</code>	<code>false</code>	Whether to add a header storing the absolute path filename.
<code>fileHeaderKey</code>	<code>file</code>	Header key to use when appending absolute path filename to event header.
<code>basenameHeader</code>	<code>false</code>	Whether to add a header storing the basename of the file.
<code>basenameHeaderKey</code>	<code>basename</code>	Header Key to use when appending basename of file to event header.
<code>ignorePattern</code>	<code>^\$</code>	Regular expression specifying which files to ignore (skip)
<code>trackerDir</code>	<code>.flumespool</code>	Directory to store metadata related to processing of files. If this path is not an absolute path, then it is interpreted as relative to the <code>spoolDir</code> .

# Flume—Channels

<http://flume.apache.org/FlumeUserGuide.html#flume-channels>

- Memory Channel
- JDBC Channel
- Kafka Channel
- File Channel
- Spillable  
Memory Channel
- Pseudo  
Transaction  
Channel
- Custom Channel

# Memory Channel

Property Name	Default	Description
<b>type</b>	-	The component type name, needs to be <code>memory</code>
<code>capacity</code>	100	The maximum number of events stored in the channel
<code>transactionCapacity</code>	100	The maximum number of events the channel will take from a source or give to a sink per transaction
<code>keep-alive</code>	3	Timeout in seconds for adding or removing an event
<code>byteCapacityBufferPercentage</code>	20	Defines the percent of buffer between <code>byteCapacity</code> and the estimated total size of all events in the channel, to account for data in headers. See below.
<code>byteCapacity</code>	see description	Maximum total <b>bytes</b> of memory allowed as a sum of all events in this channel. The implementation only counts the Event body, which is the reason for providing the <code>byteCapacityBufferPercentage</code> configuration parameter as well. Defaults to a computed value equal to 80% of the maximum memory available to the JVM (i.e. 80% of the <code>-Xmx</code> value passed on the command line). Note that if you have multiple memory channels on a single JVM, and they happen to hold the same physical events (i.e. if you are using a replicating channel selector from a single source) then those event sizes may be double-counted for channel <code>byteCapacity</code> purposes. Setting this value to 0 will cause this value to fall back to a hard internal limit of about 200 GB.

Example for agent named `a1`:

```
a1.channels = c1
a1.channels.c1.type = memory
a1.channels.c1.capacity = 10000
a1.channels.c1.transactionCapacity = 10000
a1.channels.c1.byteCapacityBufferPercentage = 20
a1.channels.c1.byteCapacity = 800000
```

# File Channel

Property Name	Default	Description
type	-	The component type name, needs to be <code>file</code> .
checkpointDir	~/.flume/file-channel/checkpoint	The directory where checkpoint file will be stored
useDualCheckpoints	false	Backup the checkpoint. If this is set to <code>true</code> , <code>backupCheckpointDir</code> <b>must</b> be set
backupCheckpointDir	-	The directory where the checkpoint is backed up to. This directory <b>must not</b> be the same as the data directories or the checkpoint directory
dataDirs	~/.flume/file-channel/data	Comma separated list of directories for storing log files. Using multiple directories on separate disks can improve file channel performance
transactionCapacity	10000	The maximum size of transaction supported by the channel
checkpointInterval	30000	Amount of time (in millis) between checkpoints
maxFileSize	2146435071	Max size (in bytes) of a single log file
minimumRequiredSpace	524288000	Minimum Required free space (in bytes). To avoid data corruption, File Channel stops accepting take/put requests when free space drops below this value
capacity	1000000	Maximum capacity of the channel

Example for agent named `a1`:

```
a1.channels = c1
a1.channels.c1.type = file
a1.channels.c1.checkpointDir = /mnt/flume/checkpoint
a1.channels.c1.dataDirs = /mnt/flume/data
```

# Flume—Sinks

<http://flume.apache.org/FlumeUserGuide.html#flume-sinks>

- HDFS Sink
- Hive Sink
- Logger Sink
- Avro Sink
- Thrift Sink
- IRC Sink
- File Roll Sink
- Null Sink
- HBaseSinks
  - HBaseSink
  - AsyncHBaseSink

- MorphlineSolrSink
- ElasticSearchSink
- Kite Dataset Sink
- Kafka Sink
- Custom Sink

# HDFS Sink

hdfs.filePrefix	FlumeData	Name prefixed to files created by Flume in hdfs directory
hdfs.fileSuffix	-	Suffix to append to file (eg .avro - <i>NOTE: period is not automatically added</i> )
hdfs.inUsePrefix	-	Prefix that is used for temporal files that flume actively writes into
hdfs.inUseSuffix	.tmp	Suffix that is used for temporal files that flume actively writes into
hdfs.rollInterval	30	Number of seconds to wait before rolling current file (0 = never roll based on time interval)
hdfs.rollSize	1024	File size to trigger roll, in bytes (0: never roll based on file size)
hdfs.rollCount	10	Number of events written to file before it rolled (0 = never roll based on number of events)

Example for agent named a1:

```
a1.channels = c1
a1.sinks = k1
a1.sinks.k1.type = hdfs
a1.sinks.k1.channel = c1
a1.sinks.k1.hdfs.path = /flume/events/%y-%m-%d/%H%M/%S
a1.sinks.k1.hdfs.filePrefix = events-
a1.sinks.k1.hdfs.round = true
a1.sinks.k1.hdfs.roundValue = 10
a1.sinks.k1.hdfs.roundUnit = minute
```

The above configuration will round down the timestamp to the last 10th minute. For example, an event with timestamp 11:54:34 AM, June 12, 2012 will cause the hdfs path to become `/flume/events/2012-06-12/1150/00`.



# 课程大纲

1

**Flume 设计架构**

2

**Flume 初步使用**

3

**Source/Channel/Sink**

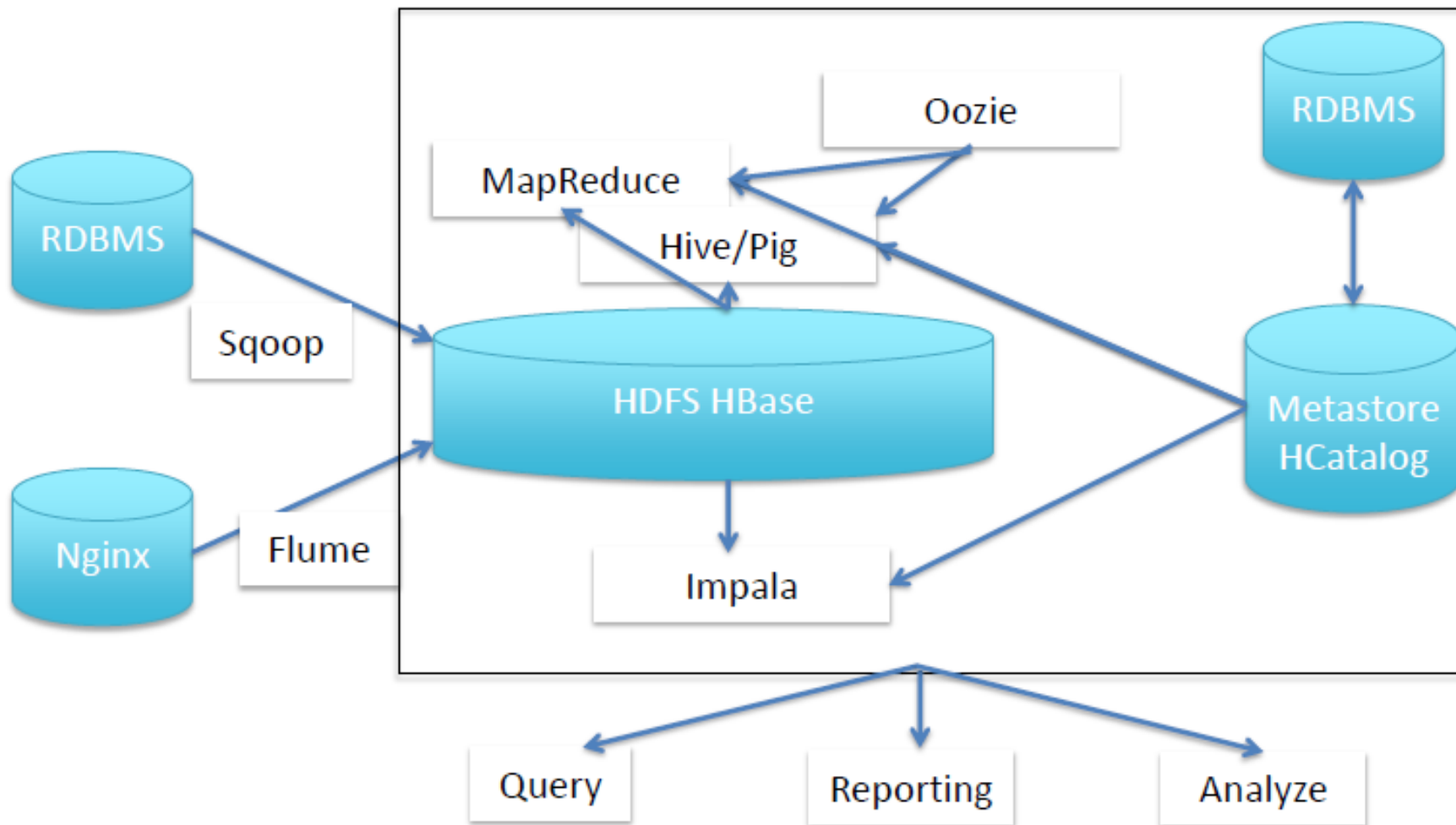
4

**Flume 项目架构**

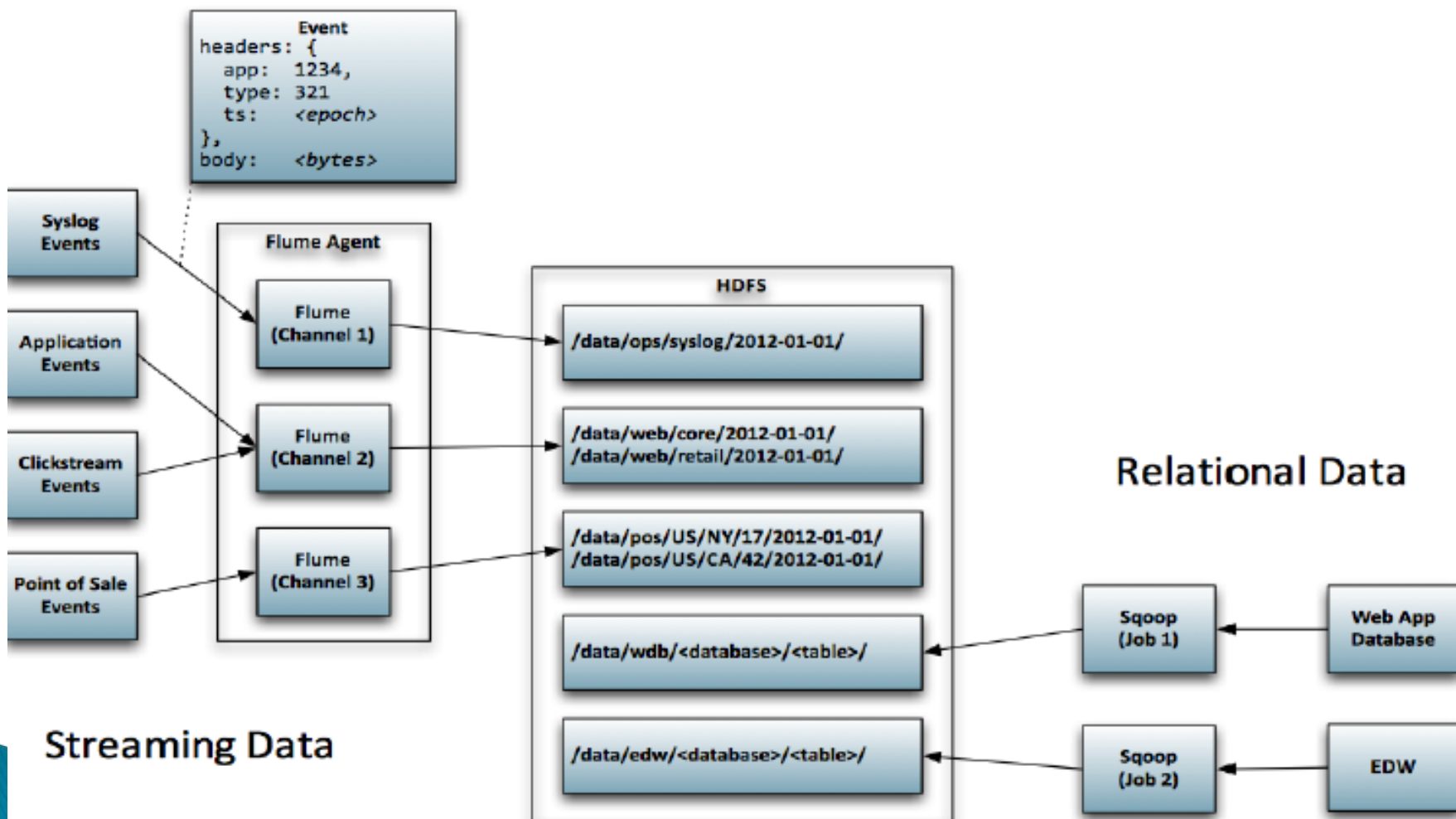
5

**Flume 实战案例**

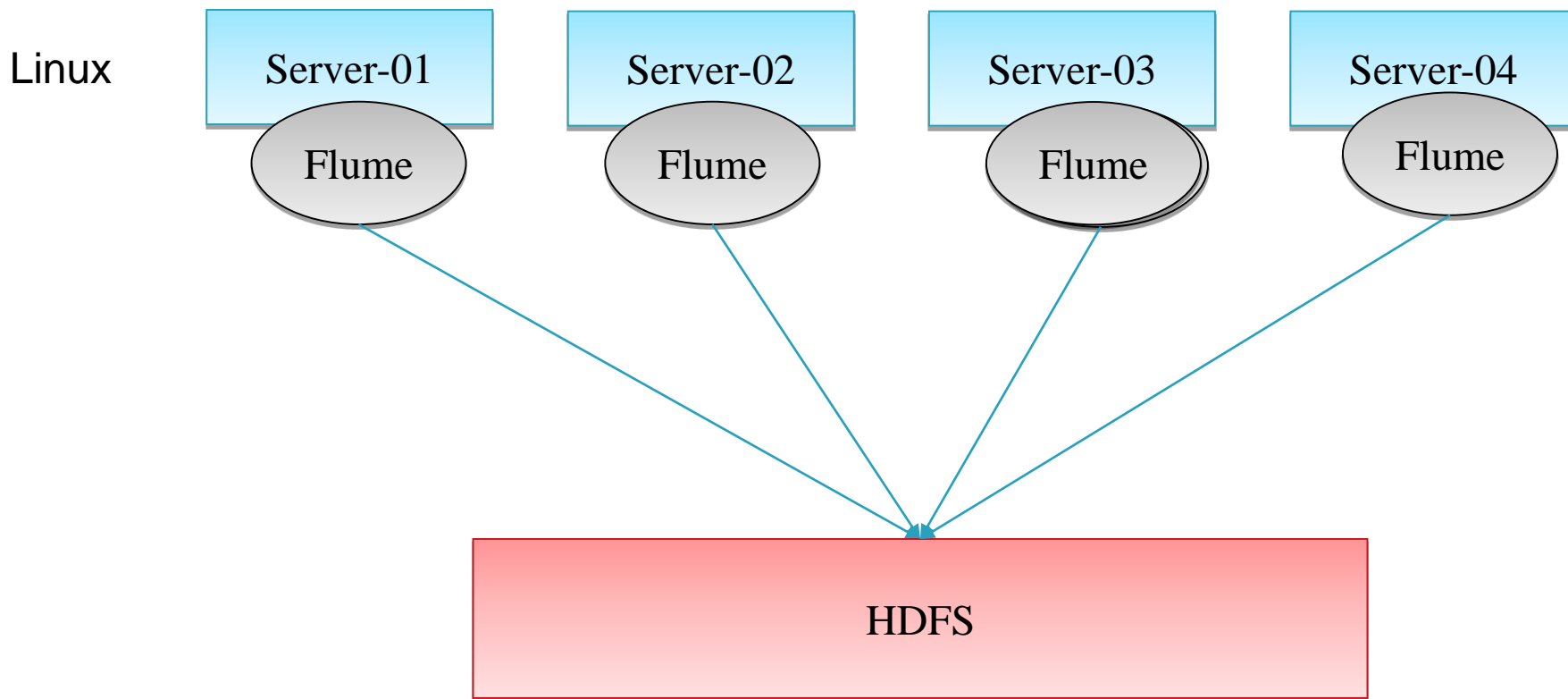
# 数据仓库架构



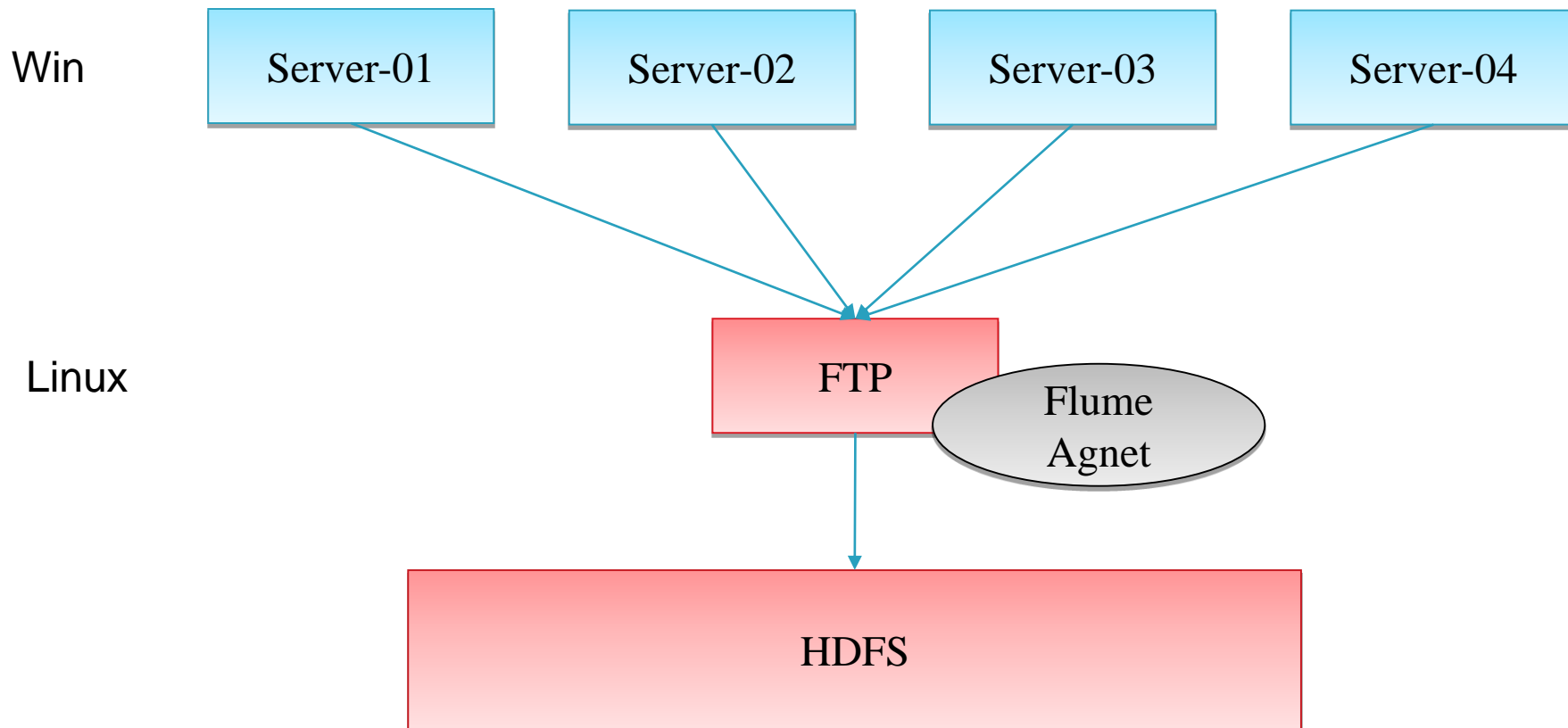
# 数据仓库架构



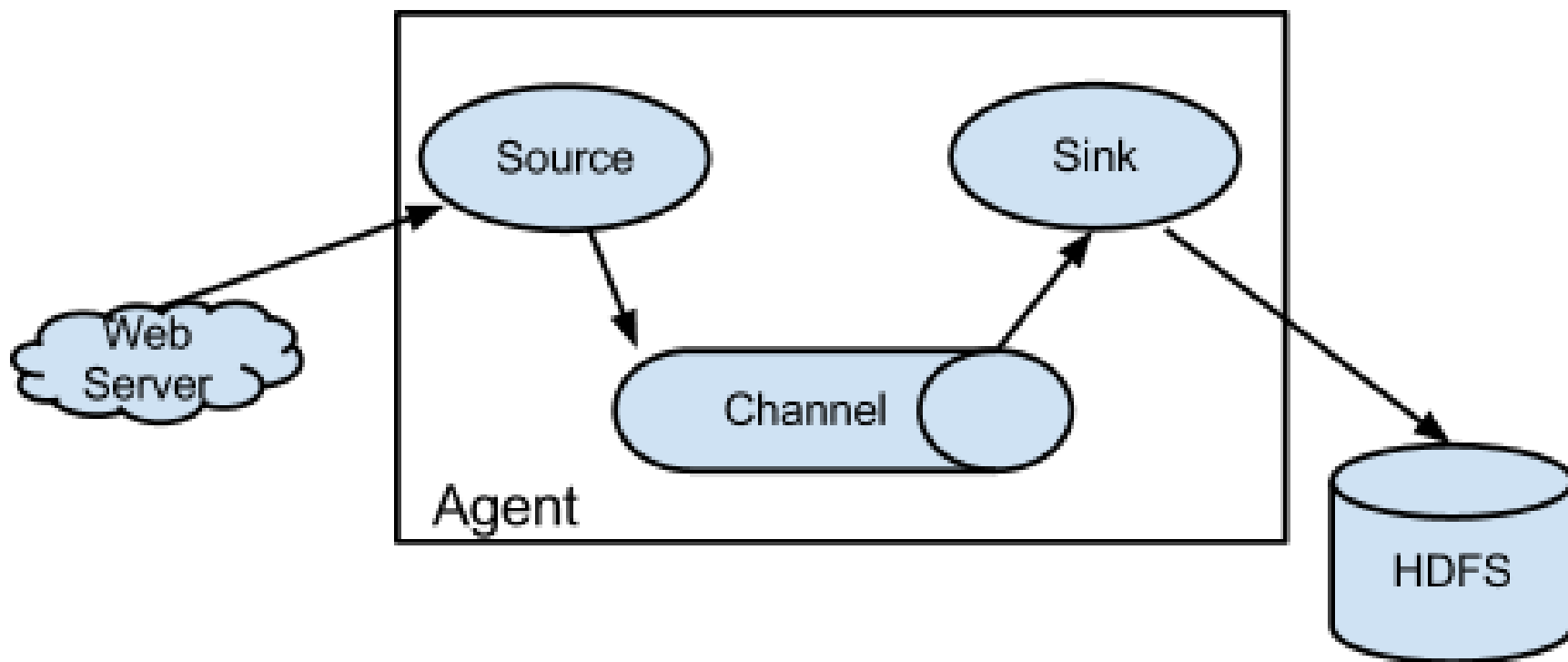
# 企业大数据仓库之数据收集架构



# 企业大数据仓库之数据收集架构



# 实时收集日志数据至HDFS



## ◆ 源端

Hive运行日志，目录：cdh-5.3.6/hive-0.13.1-cdh5.3.6/logs/hive.log

## ◆ 目标端

数据存储HDFS，目录：/user/beifeng/flume/hive-log/

# 课程大纲

1

**Flume 设计架构**

2

**Flume 初步使用**

3

**Source/Channel/Sink**

4

**Flume 项目架构**

5

**Flume 实战案例**

## Spooling Directory Source

- ◆ 在使用exec来监听数据源虽然实时性较高，但是可靠性较差，当source程序运行异常或者Linux命令中断都会造成数据丢失，在恢复正常运行之前数据的完整性无法得到保障。
- ◆ Spooling Directory Paths通过监听某个目录下的新增文件，并将文件的内容读取出来，实现日志信息的收集。实际生产中会结合log4j来使用。被传输结束的文件会修改后缀名，添加.COMPLETED后缀（可修改）。



# 监控目录实时抽取数据

## ◆ 监控目录

- 日志目录，抽取完整的日志文件，写的日志文件不抽取

## ◆ 使用FileChannel

- 本地文件系统缓冲，比内存安全性更高

## ◆ 数据存储HDFS

- 存储对应hive表的目录或者hdfs目录

## ◆ 思考？？？？？

每天抽取文件到一个目录中，自动生成目录，如何做？

本课程版权归北风网所有

欢迎访问我们的官方网站  
[www.ibeifeng.com](http://www.ibeifeng.com)