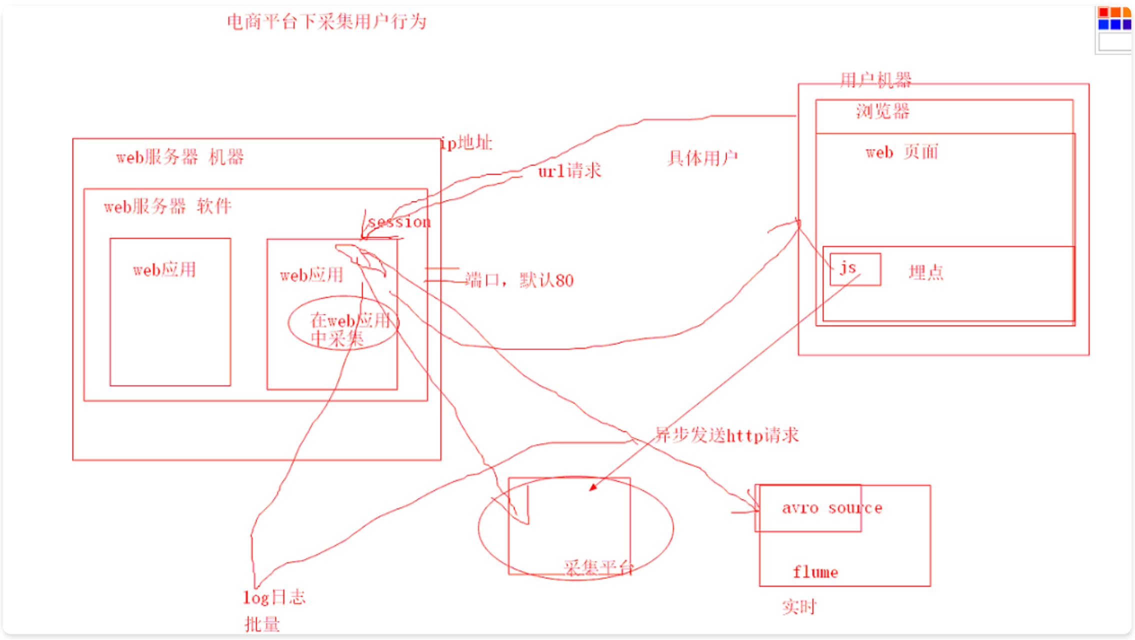
Day21 flume Interceptor

hadoop Java



一个source对应两个channel

```
a1.sources = r1
a1.channels = c1 c2
a1.sinks = s1 s2

a1.sources.r1.type=avro
a1.sources.r1.bind=master
a1.sources.r1.port=9999

a1.channels.c1.type= memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.channels.c2.type= memory
a1.channels.c2.capacity = 1000
a1.channels.c2.transactionCapacity = 100

a1.sinks.s2.type = logger

a1.sinks.s1.type = hdfs
a1.sinks.s1.hdfs.path = hdfs://master:9000/flumelog/%Y%m%d
a1.sinks.s1.hdfs.fileSuffix = .log
a1.sinks.s1.hdfs.rollInterval = 0
```

```
a1.sinks.s1.hdfs.rollSize = 0
a1.sinks.s1.hdfs.rollCount = 100
a1.sinks.s1.hdfs.fileType = DataStream
a1.sinks.s1.hdfs.writeFormat = Text
a1.sinks.s1.hdfs.useLocalTimeStamp = true

a1.sources.r1.channels=c1 c2
a1.sinks.s1.channel=c1
a1.sinks.s2.channel=c2
```

Mapping for multiplexing selector

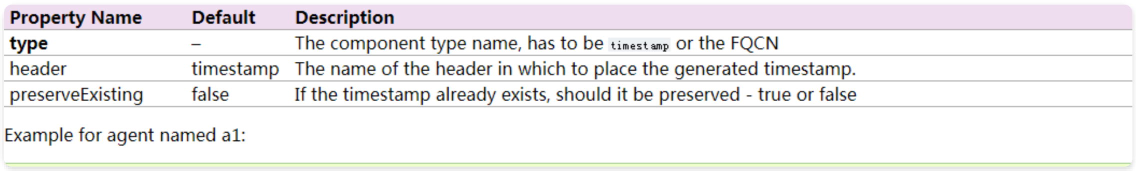
```
a1.sources = r1
a1.sinks = s1 s2 s3 s4
a1.channels = c1 c2 c3 c4

a1.sources.r1.type = avro
a1.sources.r1.bind = master
a1.sources.r1.port = 8888

a1.channels.c1.type= memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.channels.c2.type= memory
a1.channels.c2.capacity = 1000
a1.channels.c2.transactionCapacity = 100

a1.channels.c3.type= memory
a1.channels.c3.capacity = 1000
a1.channels.c3.transactionCapacity = 100

a1.channels.c4.type= memory
a1.channels.c4.capacity = 1000
a1.channels.c5.transactionCapacity = 100

a1.sinks.s1.type = hdfs
a1.sinks.s1.hdfs.path = hdfs://master:9000/flumelog/%Y%m%d/henan
a1.sinks.s1.hdfs.fileSuffix = .log
a1.sinks.s1.hdfs.rollInterval = 0
a1.sinks.s1.hdfs.rollSize = 0
a1.sinks.s1.hdfs.rollCount = 100
a1.sinks.s1.hdfs.fileType = DataStream
a1.sinks.s1.hdfs.writeFormat = Text
a1.sinks.s1.hdfs.useLocalTimeStamp = true

a1.sinks.s2.type = hdfs
a1.sinks.s2.hdfs.path = hdfs://master:9000/flumelog/%Y%m%d/hebei
a1.sinks.s2.hdfs.fileSuffix = .log
a1.sinks.s2.hdfs.rollInterval = 0
a1.sinks.s2.hdfs.rollSize = 0
a1.sinks.s2.hdfs.rollCount = 100
a1.sinks.s2.hdfs.fileType = DataStream
a1.sinks.s2.hdfs.writeFormat = Text
a1.sinks.s2.hdfs.useLocalTimeStamp = true

a1.sinks.s3.type = hdfs
a1.sinks.s3.hdfs.path = hdfs://master:9000/flumelog/%Y%m%d/shandong
a1.sinks.s3.hdfs.fileSuffix = .log
a1.sinks.s3.hdfs.rollInterval = 0
a1.sinks.s3.hdfs.rollSize = 0
a1.sinks.s3.hdfs.rollCount = 100
a1.sinks.s3.hdfs.fileType = DataStream
```

```
a1.sinks.s3.hdfs.writeFormat = Text
a1.sinks.s3.hdfs.useLocalTimeStamp = true

a1.sinks.s4.type = hdfs
a1.sinks.s4.hdfs.path = hdfs://master:9000/flumelog/%Y%m%d/qita
a1.sinks.s4.hdfs.fileSuffix = .log
a1.sinks.s4.hdfs.rollInterval = 0
a1.sinks.s4.hdfs.rollSize = 0
a1.sinks.s4.hdfs.rollCount = 100
a1.sinks.s4.hdfs.fileType = DataStream
a1.sinks.s4.hdfs.writeFormat = Text
a1.sinks.s4.hdfs.useLocalTimeStamp = true

a1.sinks.s1.channel=c1
a1.sinks.s2.channel=c2
a1.sinks.s3.channel=c3
a1.sinks.s4.channel=c4

a1.sources.r1.channels =c1 c2 c3 c4
a1.sources.r1.selector.type = multiplexing
a1.sources.r1.selector.header = province
a1.sources.r1.selector.mapping.henan = c1
a1.sources.r1.selector.mapping.hebei = c2
a1.sources.r1.selector.mapping.shandong = c3
a1.sources.r1.selector.default.qita = c4
```

Flume Interceptors

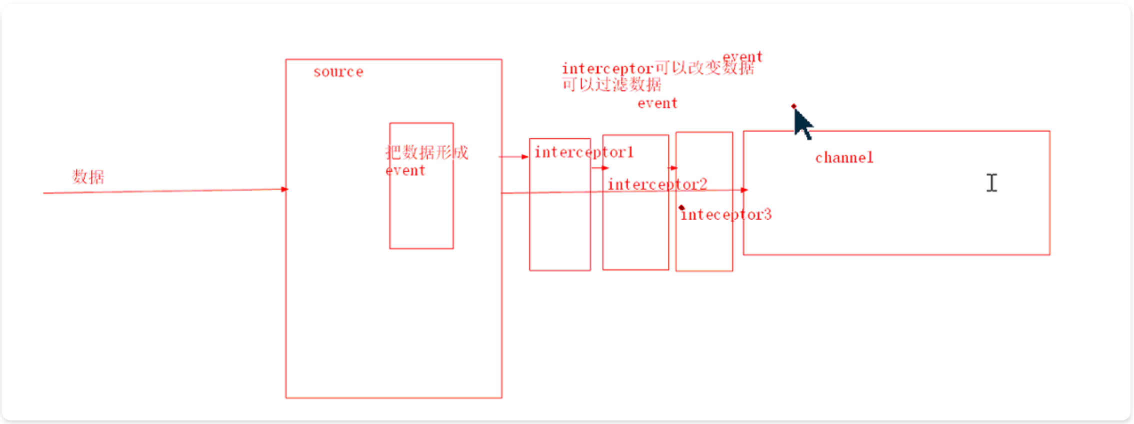
- timestrap Interceptor

This interceptor inserts into the event headers, the time in millis at which it processes the event. This interceptor inserts a header with key timestamp (or as specified by the header property) whose value is the relevant timestamp. This interceptor can preserve an existing timestamp if it is already present in the configuration.



Property Name	Default	Description
type	–	The component type name, has to be `timestamp` or the FQCN
header	timestamp	The name of the header in which to place the generated timestamp.
preserveExisting	false	If the timestamp already exists, should it be preserved - true or false

Example for agent named a1:

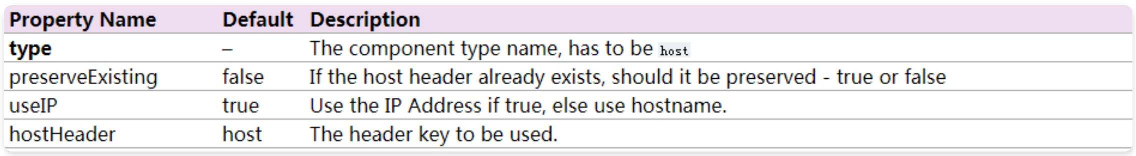


```
a1.sources = r1
a1.sinks=s1
a1.channels=c1
```

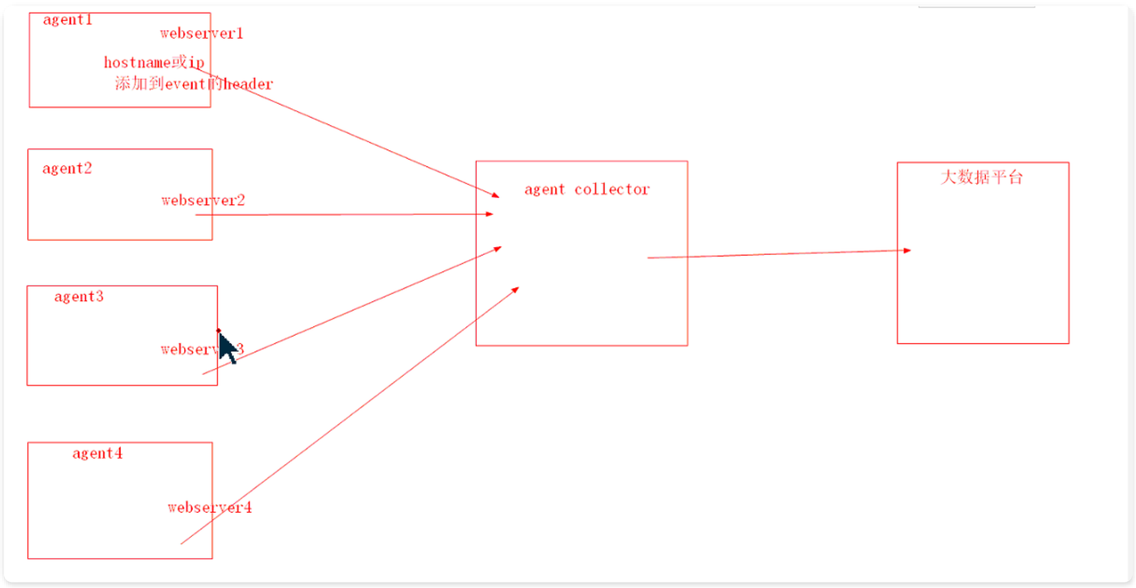
```
a1.sources.r1.type = avro
a1.sources.r1.bind = master
a1.sources.r1.port = 8888

a1.channels.c1.type= memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sinks.s1.type = logger

a1.sources.r1.channels = c1
a1.sinks.s1.channel = c1

a1.sources.r1.interceptors = i1
a1.sources.r1.interceptors.i1.type = timestamp
```

host interceptor

This interceptor inserts the hostname or IP address of the host that this agent is running on. It inserts a header with key host or a configured key whose value is the hostname or IP address of the host, based on configuration.



Property Name	Default	Description
type	–	The component type name, has to be `host`
preserveExisting	false	If the host header already exists, should it be preserved - true or false
useIP	true	Use the IP Address if true, else use hostname.
hostHeader	host	The header key to be used.



```
a1.sources = r1
a1.sinks=s1
a1.channels=c1

a1.sources.r1.type = avro
a1.sources.r1.bind = master
a1.sources.r1.port = 8888

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

```
a1.sinks.s1.type = logger

a1.sources.r1.channels = c1
a1.sinks.s1.channel = c1

a1.sources.r1.interceptors = i1
a1.sources.r1.interceptors.i1.type = host
```

Static Interceptor

Static interceptor allows user to append a static header with static value to all events.

The current implementation does not allow specifying multiple headers at one time. Instead user might chain multiple static interceptors each defining one static header.

Property Name	Default	Description
type	–	The component type name, has to be `static`
preserveExisting	true	If configured header already exists, should it be preserved - true or false
key	key	Name of header that should be created
value	value	Static value that should be created

```
a1.sources = r1
a1.sinks=s1
a1.channels=c1

a1.sources.r1.type = avro
a1.sources.r1.bind = master
a1.sources.r1.port = 8888

a1.channels.c1.type= memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sinks.s1.type = logger

a1.sources.r1.channels = c1
a1.sinks.s1.channel = c1

a1.sources.r1.interceptors = i1
a1.sources.r1.interceptors.i1.type = static

a1.sources.r1.interceptors.i1.key  =  数据中心
a1.sources.r1.interceptors.i1.value  =  NEW_YORK
```

Search and Replace Interceptor

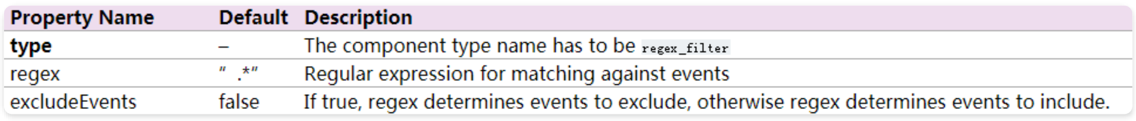
This interceptor provides simple string-based search-and-replace functionality based on Java regular expressions. Backtracking / group capture is also available. This interceptor uses the same rules as in the Java Matcher.replaceAll() method.

Property Name	Default	Description
type	–	The component type name has to be `search_replace`
searchPattern	–	The pattern to search for and replace.
replaceString	–	The replacement string.
charset	UTF-8	The charset of the event body. Assumed by default to be UTF-8.

```
a1.sources = r1
a1.sinks=s1
a1.channels=c1

a1.sources.r1.type = avro
a1.sources.r1.bind = master
a1.sources.r1.port = 8888

a1.channels.c1.type= memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sinks.s1.type = logger

a1.sources.r1.channels = c1
a1.sinks.s1.channel = c1

a1.sources.r1.interceptors = i1
a1.sources.r1.interceptors.i1.type = search_replace

a1.sources.r1.interceptors.i1.searchPattern = (\\d{3})\\d{4}(\\d{4})
a1.sources.r1.interceptors.i1.replaceString =$1xxxx$2
```

Regex Filtering Interceptor

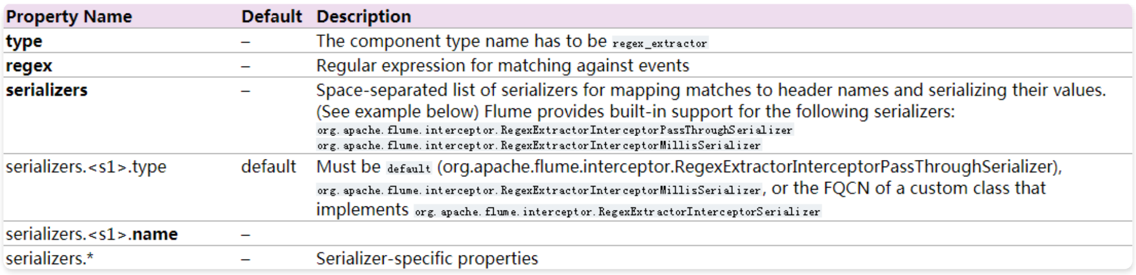
This interceptor filters events selectively by interpreting the event body as text and matching the text against a configured regular expression. The supplied regular expression can be used to include events or exclude events.



Property Name	Default	Description
type	–	The component type name has to be `regex_filter`
regex	" .*"	Regular expression for matching against events
excludeEvents	false	If true, regex determines events to exclude, otherwise regex determines events to include.

```
a1.sources = r1
a1.sinks=s1
a1.channels=c1

a1.sources.r1.type = spooldir
a1.sources.r1.spoolDir = /root/tmp
a1.sources.r1.fileHeader = true

a1.channels.c1.type= memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sinks.s1.type = logger

a1.sources.r1.channels = c1
a1.sinks.s1.channel = c1

a1.sources.r1.interceptors = i1
a1.sources.r1.interceptors.i1.type =  regex_filter

a1.sources.r1.interceptors.i1.regex =.*[\\w|\\d]+\\@[\\w|\\d]+.*
a1.sources.r1.interceptors.i1.excludeEvents=false
```

Regex Extractor Interceptor

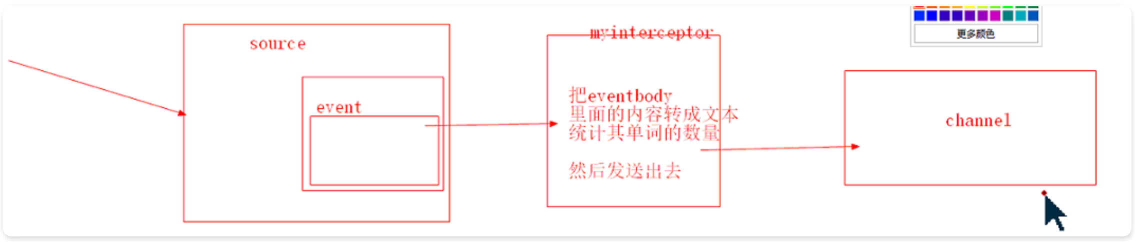


Property Name	Default	Description
type	–	The component type name has to be `regex_extractor`
regex	–	Regular expression for matching against events
serializers	–	Space-separated list of serializers for mapping matches to header names and serializing their values. (See example below) Flume provides built-in support for the following serializers: `org.apache.flume.interceptor.RegexExtractorInterceptorPassThroughSerializer` `org.apache.flume.interceptor.RegexExtractorInterceptorMillisSerializer`
serializers.<s1>.type	default	Must be `default` (org.apache.flume.interceptor.RegexExtractorInterceptorPassThroughSerializer), `org.apache.flume.interceptor.RegexExtractorInterceptorMillisSerializer`, or the FQCN of a custom class that implements `org.apache.flume.interceptor.RegexExtractorInterceptorSerializer`
serializers.<s1>.**name**	–	
serializers.*	–	Serializer-specific properties

```
a1.sources = r1
a1.sinks=s1
a1.channels=c1

a1.sources.r1.type = spooldir
a1.sources.r1.spoolDir = /root/tmp
a1.sources.r1.fileHeader = true

a1.channels.c1.type= memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sinks.s1.type = logger

a1.sources.r1.channels = c1
a1.sinks.s1.channel = c1

a1.sources.r1.interceptors = i1
a1.sources.r1.interceptors.i1.type = regex_extractor

a1.sources.r1.interceptors.i1.regex =.*[\\w|\\d]+\\@[\\w|\\d]+.*
a1.sources.r1.interceptors.i1.serializers = e1
a1.sources.r1.interceptors.i1.serializers.e1.name = one
```

Customer Interceptor

需求：自定义Interceptor 把接收到的数据转成文本，把event里面的内容替换成文本的单词个数
定义一个execlude参数，排除某些单词不计算在内



```
package top.xiesen.interceptor;

import java.util.Arrays;
import java.util.List;

import org.apache.flume.Context;
import org.apache.flume.Event;
import org.apache.flume.interceptor.Interceptor;

public class WordCountInterceptor implements Interceptor {
 // 参数excludeWords中可以填写多个单词，多个单词之间逗号分隔
 private String excludeWords;
 private String[] excludeWordArray;
 private int eventCount;

 public WordCountInterceptor(String excludeWords) {
  this.excludeWords = excludeWords;
  if (excludeWords != null && !excludeWords.equals("")) {
   excludeWordArray = this.excludeWords.split(",");
  }
 }

 public WordCountInterceptor() {
  super();
 }

 @Override
 public void initialize() {
 }

 // 拦截器过程数据处理逻辑
 @Override
 public Event intercept(Event event) {
  eventCount = 0;
  String[] words = new String(event.getBody()).split("\\s");
  if (excludeWordArray == null || excludeWordArray.length < 1) {
   eventCount = words.length;
  } else {
   List<String> execludeList = Arrays.asList(excludeWordArray);
   for (String word : words) {
    if (!execludeList.contains(word)) {
     eventCount += 1;
    }
   }
  }
  event.setBody(String.valueOf(eventCount).getBytes());

  return event;
 }

 // 使用单个event拦截处理过程逻辑来实现list列表event的处理过程
 @Override
 public List<Event> intercept(List<Event> events) {
  for (Event event : events) {
   intercept(event);
  }
  return events;
 }

 @Override
 public void close() {
 }

 // 定义Interceptor.Builder接口的实现类,并且是Interceptor的内部类
 public static class Builder implements Interceptor.Builder {
  private String excludeWords;

  @Override
  public void configure(Context context) {
   excludeWords = context.getString("excludeWords");
  }

  @Override
  public Interceptor build() {
   return new WordCountInterceptor(excludeWords);
  }

 }
}
```

将工程打包，上传到flume的lib文件夹下
编写脚本

```
a1.sources = r1
a1.sinks = s1
a1.channels = c1

a1.sources.r1.type = netcat
a1.sources.r1.bind = localhost
a1.sources.r1.port = 44444

a1.sinks.s1.type = logger

a1.channels.c1.type= memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sources.r1.channels = c1
a1.sinks.s1.channel = c1

a1.sources.r1.interceptors=i1
a1.sources.r1.interceptors.i1.type=top.xiesen.interceptor.WordCountInterceptor$Builder
a1.sources.r1.interceptors.i1.excludeWords=abc
```

hive sink

- 启动metastore `hive --service metastore`
- 查看metastore是否启动 `netstat -alnp | grep 9083`
- 将 `/opt/software/hive/apache-hive-2.3.0-bin/hcatalog/share/hcatalog` 目录下的jar
- 创建表

```sql
create table flume_user(
user_id int,
user_name string,
age int
)
clustered by(user_id) into 2 buckets
stored as orc
tblproperties("transactional"='true');
```

- 编写脚本

```
a1.sources = r1
a1.sinks = s1
a1.channels = c1

a1.sources.r1.type = netcat
a1.sources.r1.bind = localhost
a1.sources.r1.port = 44444

a1.sinks.s1.type = hive
a1.sinks.s1.channel = c1
a1.sinks.s1.hive.metastore = thrift://master:9083
a1.sinks.s1.hive.database = db14
a1.sinks.s1.hive.table = flume_user

a1.sinks.s1.serializer = DELIMITED
a1.sinks.s1.serializer.delimiter = "\t"
a1.sinks.s1.serializer.serdeSeparator = '\t'
a1.sinks.s1.serializer.fieldnames =user_id,user_name,age
```

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

a1.sources.r1.channels = c1
a1.sinks.s1.channel = c1
```