Lecture 11 - Flume, Spark

BDAT 1002

Apache Flume

Motivation

- One of the first uses of Hadoop was to injest application log files and do some analysis on it
- Example:
  - Amazon recommendation engine
  - Very powerful and accurate
  - How is it done?

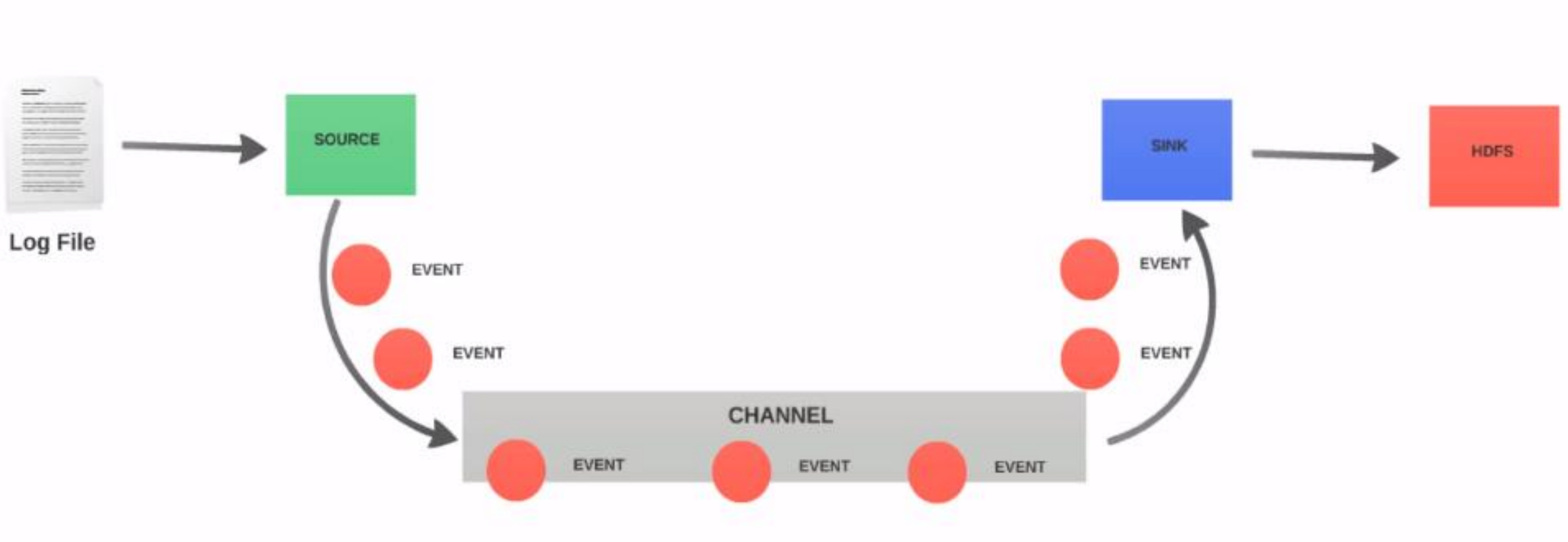
Motivation

- Amazon logs EVERY movement you do when you are on their website
  - Click
  - Order of click
  - Where you spend most of time (product, reviews etc)
  - Then they use ML to find insights
- But real question is how do you move this info to HDFS?
  - copyFromLocal?

Motivation

- We like to stream the data in near real time
- Flume is a distributive service for efficiently collecting, aggregating large amounts of log data into Hadoop
- Flume is very simple to use but first we need to look at the components

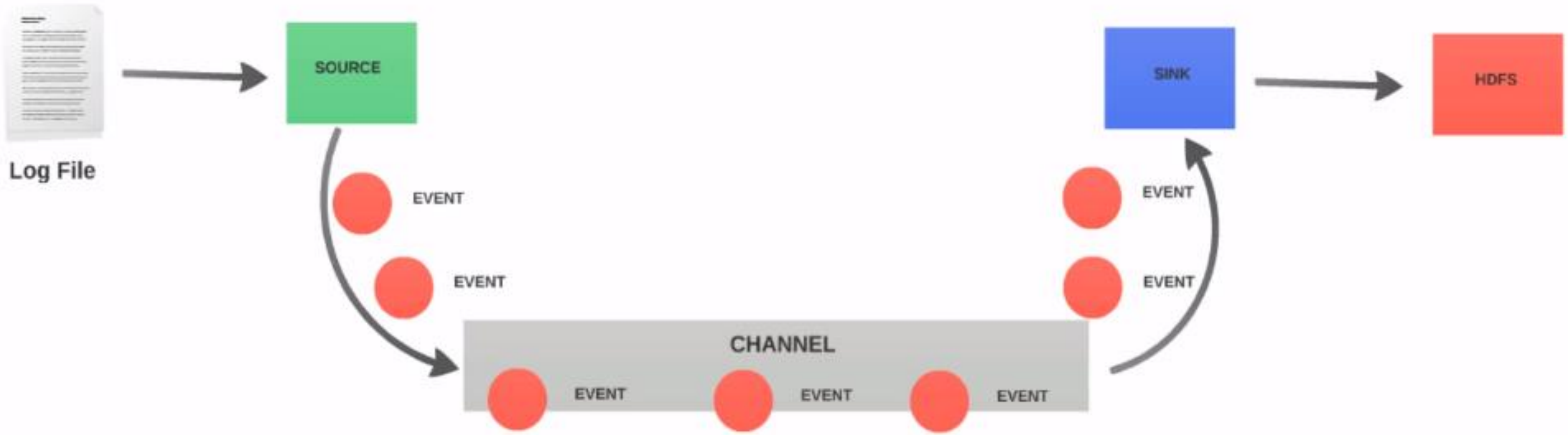
Flume Components



Flume Components

- To make flume work we need a flume agent
- There are three components required to configure a Flume agent
  - Source
  - Channel
  - Sink

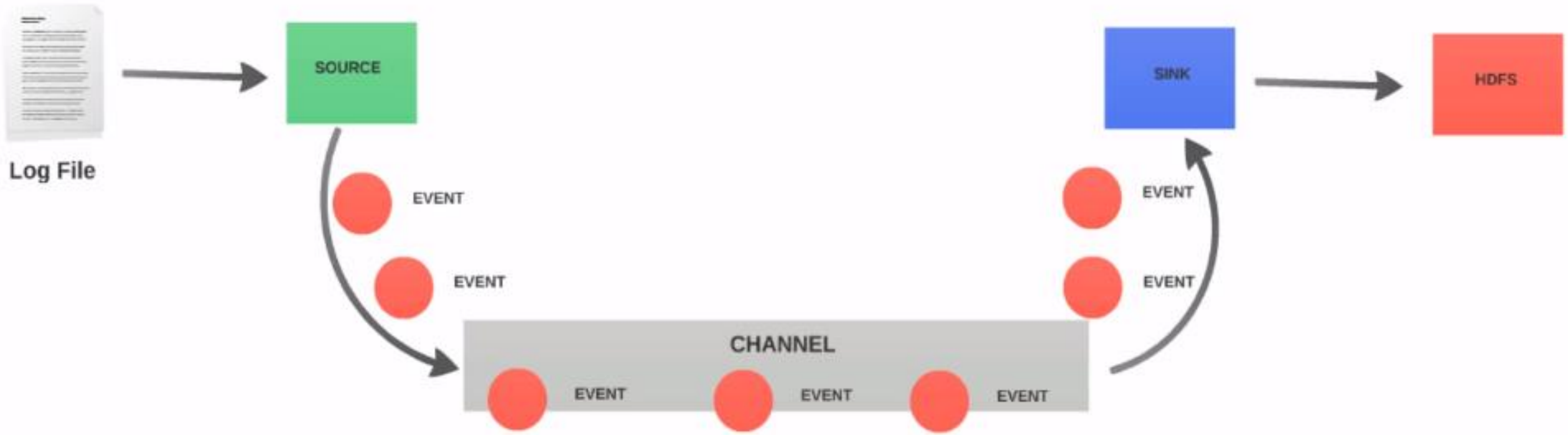
Flume Components



Source

- As the name suggests, source is where the data originates
- Source can be as simple as a logfile or HTTP (get post requests)
  - We can also have a custom source, we will do this later

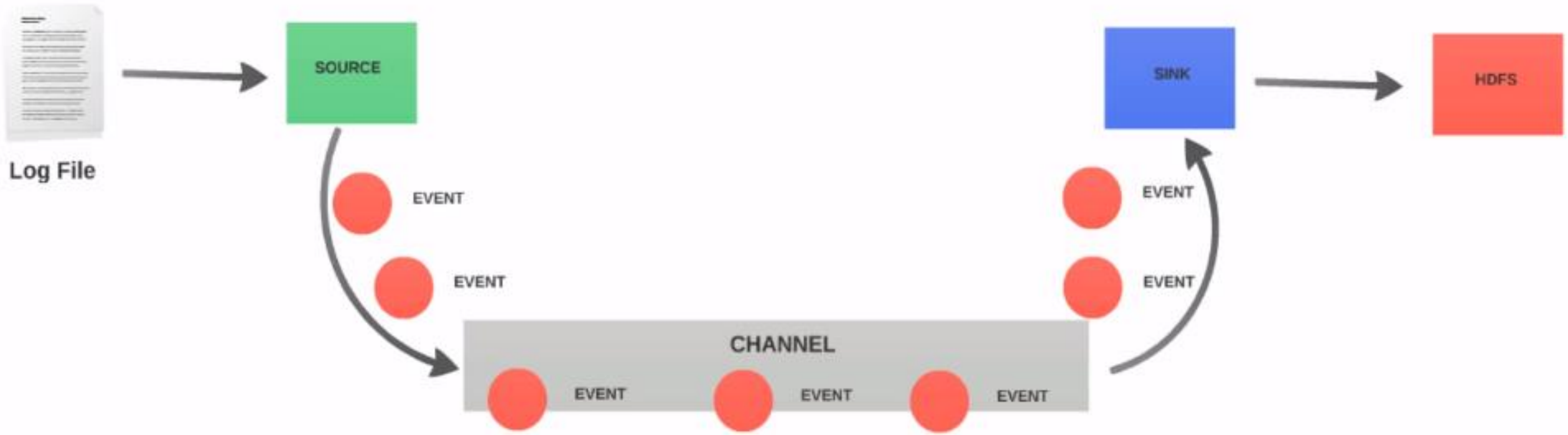
Flume Components



Sink

- Sink defines the destination
  - Where do you like flume events to go
  - It can be HDFS, file in local system or no SQL database (ex Hbase)

Flume Components



Channel

- Think of channel as pipes where Flume events flow through
- Flume guarantees that events will not be lost between source and sink
  - This guarantee is made possible with channels
- Channels are buffers that sit between sources and sinks
- Sources write data into one or more channels which are read by one or more sinks

Channel

- Channels provides transactional capability that allow Flume to provide explicit guarantee on the data that is returned
  - Every event pushed to the channel will be delivered to the sink
- Channels can be one of four types
  - File
  - Memory
  - JDBC
  - custom

Simple Flume Agent

Approach

- We will consume messages returned to a logfile by an application into HDFS
- First create an application that returns logfiles
  - Could use HDFS but not frequent enough
- Will use a shell script to create log files

Approach

- We will then configure a Flume agent
- This is very simple, need to is specify
  - Source
  - Sink
  - Channel
  - File
- Run the logfile script first, then start your agent

Approach

- Run the logfile script first, then start your agent
- Script creates a file and writes some text to the file
- Flume agent will read this and push it to the channel
- And finally these messages will end up in HDFS

Interview Question

- Any difference between source and sink channel property?
- Why?
  - Source can send to multiple channels
  - Sink can receive from only one channel

Summary

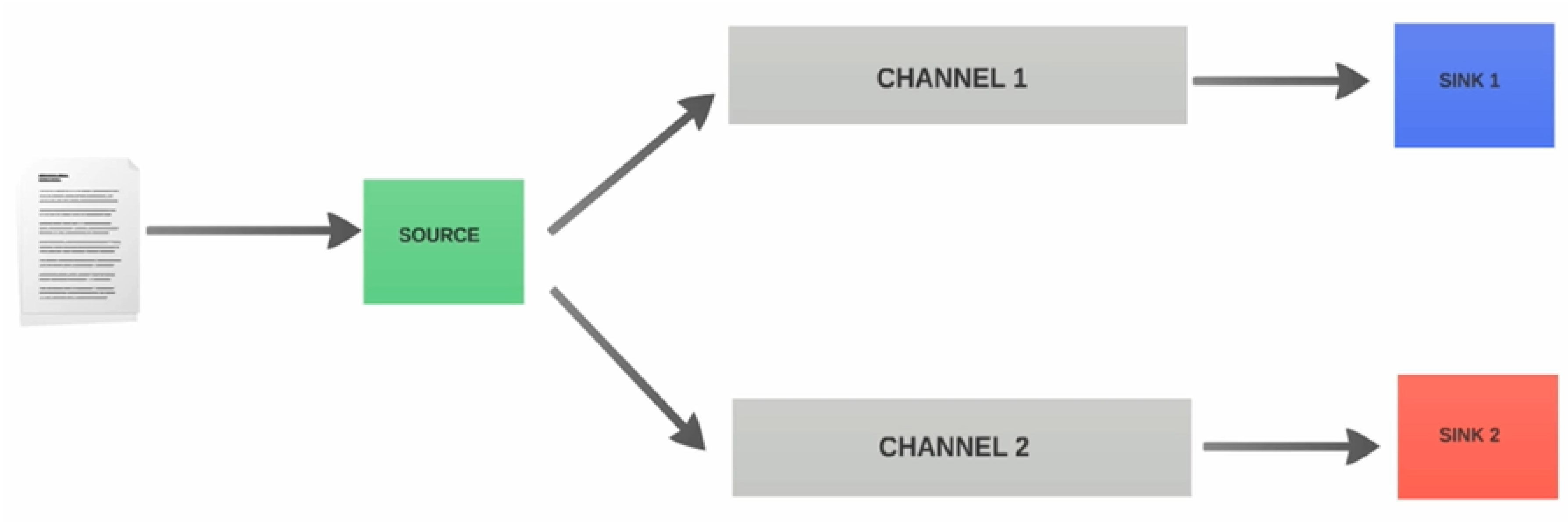
- Flume is made up of three component
  - Source
  - Channel
  - Sink
- We created a simple flume agent

Replication Setup

Experiment

- We want to configure a Flume job that sends files to more than one sink or more than one destination

Scenario



Scenario

- We are going to get messages from an application
- Each Flume event will be posted simultaneously to two different channels
- Each channel will be consumed by a separate sink
- First sink will post the messages to a file in the local file system
- The second sink will post the log message to a file in HDFS

Summary

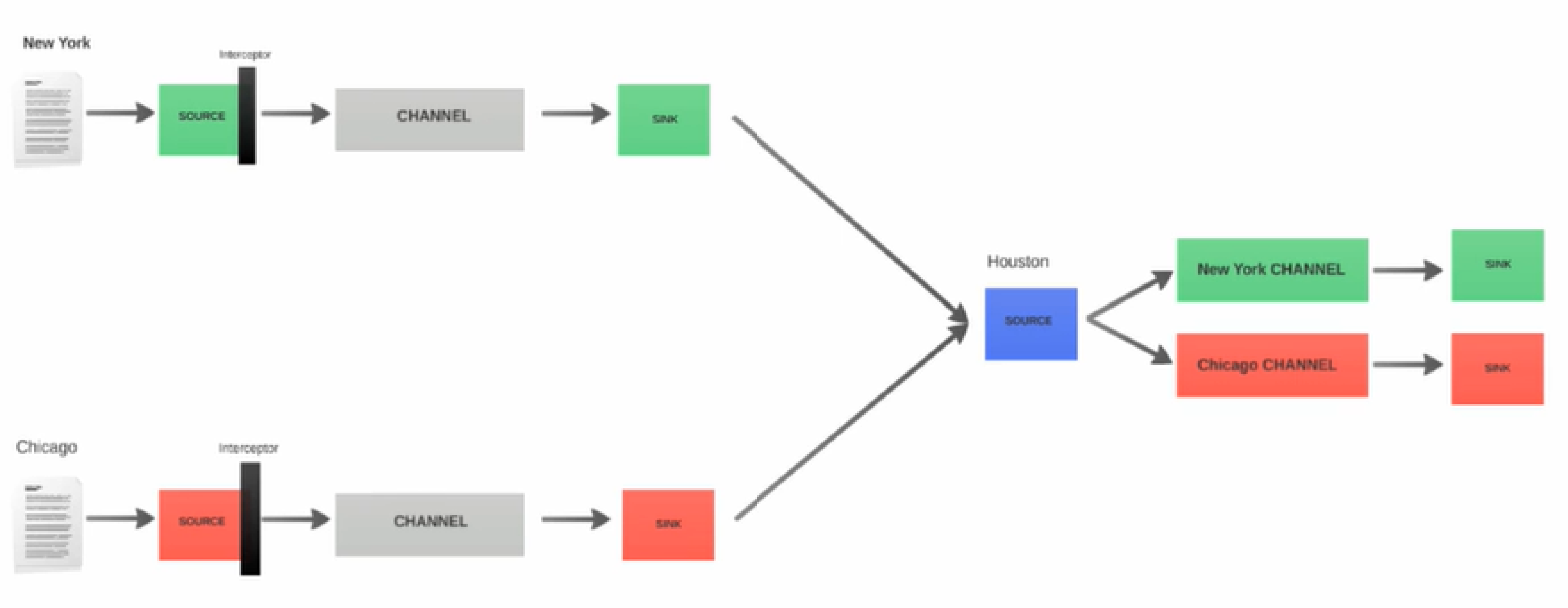
- We saw how to replicate Flume events from a single source to multiple sinks with multiple channels

Multiplexing

Experiment

- We are going to see a commonly used strategy called multiplexing
- We want to chain multiple flume events and consolidate multiple flume events from different Flume agents

Scenario


New York
Interceptor
SOURCE
CHANNEL
SINK
Chicago
Interceptor
SOURCE
CHANNEL
SINK
Houston
SOURCE
New York CHANNEL
SINK
Chicago CHANNEL
SINK

Scenario

- You have an application that is running on two data centre
  - One in New York and one in Chicago
  - We like to consolidate the log message coming from New York and Chicago into a third data centre in Houston
- However we like to identify the messages coming from New York and Chicago and store them separately

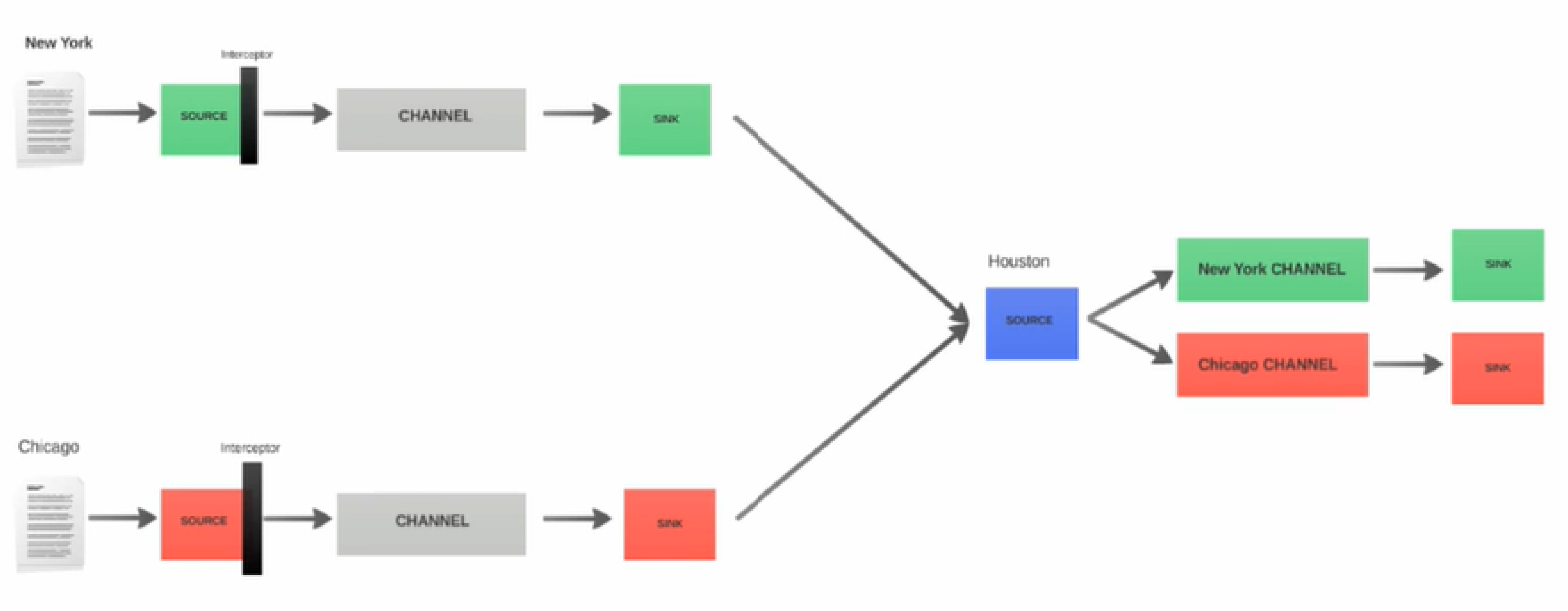
Scenario

- This requires multiple Flume agents
- Each source will have an interceptor
- The job of the interceptor is to intercept each flume event and append additional information that can be used later to identify the source of an event

Sink in this case

- Previously we saved the Flume event in a file in HDFS
- Here, we are trying to send messages to another Flume agent which is running
- We will use Avro sink type
  - Avro sink relays the Flume event to a Flume agent running in Houston

Scenario


New York
Interceptor
SOURCE
CHANNEL
SINK
Chicago
Interceptor
SOURCE
CHANNEL
SINK
Houston
SOURCE
New York CHANNEL
SINK
Chicago CHANNEL
SINK

Flume Agent in Houston

- Source of Flume agent in Houston is configured as Avro Source and it is receiving events from New York and Chicago
- We will use a concept called multiplexing to send the information to two different channels
  - One for New York
  - One for Chicago

Review

- We have one Flume agent in New York and one in Chicago
- Both these agents have interceptors to tag the flume event to indicate the location by adding a key-value pair to the event header
- Flume events from these two agents will be pushed to the third agent Houston using Avro sink

Sink in this case

- The third agent Houston will have a selector of type multiplexing to map the event to the appropriate channels
- Events from New York and Chicago will end up in different locations in HDFS

Summary

- We saw a commonly used strategy called Multiplexing
- We saw how to chain multiple flume agents and consolidate multiple flume events coming from these agents