05: Q37 – Q50 Apache Flume interview questions & answers

Posted on June 6, 2016 by Arulkumaran Kumaraswamipillai

Q37. Where do use Apache Flume in the BigData world?
A37. Apache Flume is used to **ingest** big data into HDFS. BigData is generally ingested from

1) Sporadic bulk loading processes, such as database and mainframe offloads and batched data dumps from legacy systems.

2) High-throughput streams such as applications logs, GPS tacking systems, social media updates, and digital sensors.

Here is a tutorial & example of ingesting high volume transactional data from Websphere MQ and then storing it on

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

open all | close all

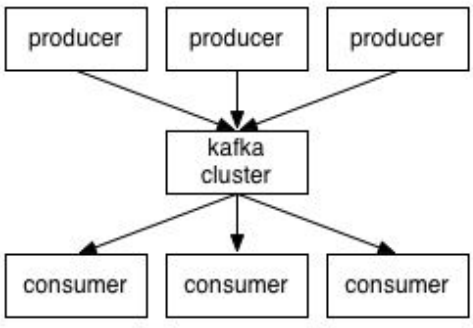
- Ice Breaker Interview
- Core Java Interview C
- JEE Interview Q&A (3
- Pressed for time? Jav
- SQL, XML, UML, JSC
- Hadoop & BigData Int
  - ♥ 01: Q1 – Q6 Had
  - 02: Q7 – Q15 Hadc
  - 03: Q16 – Q25 Hac
  - 04: Q27 – Q36 Apa
  - 05: Q37 – Q50 Apa
  - 05: Q37-Q41 – Dat
  - 06: Q51 – Q61 HBa
  - 07: Q62 – Q70 HDI
- Java Architecture Inte
- Scala Interview Q&As
- Spring, Hibernate, & M
- Testing & Profiling/Sa
- Other Interview Q&A f
- ▶Free Java Interviev

HDFS.

Q38. What are the other tools used for ingesting high volume data into HDFS?
A38. Scribe and Kafka. Kafka can be used with Flume, and known as the "**Flafka**".

Q39. How does Kafka differ from Flume?
A39. Flume and Kakfa are real-time event processing systems. Flume and Kafka are actually two quite different products.

Kafka is a general purpose publish-subscribe model messaging system, which offers strong durability, scalability and fault-tolerance support. Kafka is not specifically designed for Hadoop, but it can be used in Hadoop.



from http://kafka.apache.org

.

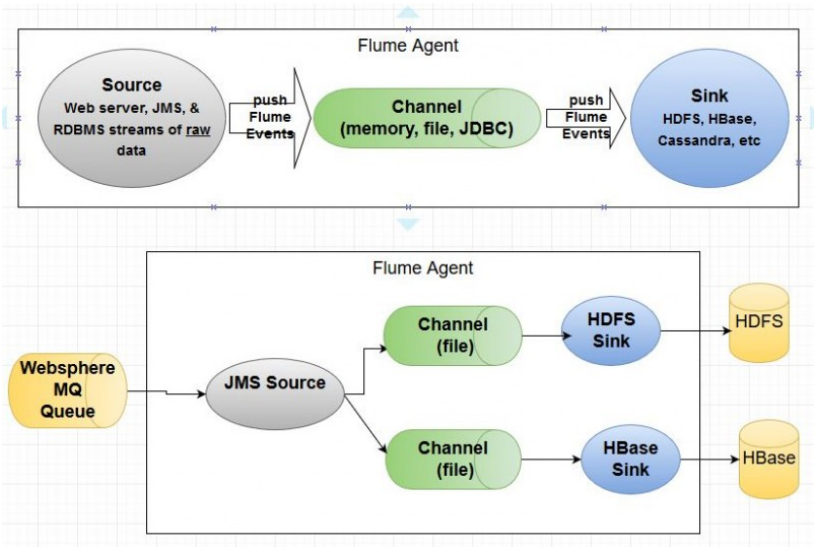
Flume is a distributed, reliable, and highly available system for efficiently collecting, aggregating, and moving large amounts of data from many different sources to a centralized data sink such as HDFS and HBase. It is more tightly integrated with Hadoop ecosystem, and works well with the HDFS security. You can chain sources, channels, and sinks to suit your requirements.

16 Technical Key Areas

open all | close all
⊞ Best Practice (6)
⊞ Coding (26)
⊞ Concurrency (6)
⊞ Design Concepts (7)
⊞ Design Patterns (11)
⊞ Exception Handling (3
⊞ Java Debugging (21)
⊞ Judging Experience In
⊞ Low Latency (7)
⊞ Memory Managemen
⊞ Performance (13)
⊞ QoS (8)
⊞ Scalability (4)
⊞ SDLC (6)
⊞ Security (13)
⊞ Transaction Managen

80+ step by step Java Tutorials

open all | close all
⊞ Setting up Tutorial (6)
⊞ Tutorial - Diagnosis (2
⊞ Akka Tutorial (9)
⊞ Core Java Tutorials (2
⊞ Hadoop & Spark Tuto
⊞ JEE Tutorials (19)
⊞ Scala Tutorials (1)
⊞ Spring & HIbernate Tu
⊞ Tools Tutorials (19)
⊞ Other Tutorials (45)



Flume Agent

Kafka has better message durability and scalability than Flume, but you need to write the consumer code. Flume is mainly configuration driven. Kafka can scale by adding more consumers without any down time. Kakfa **pulls** messages as opposed to pushing, and acts as a "shock absorber" between the producers and consumers. Kafka can handle events at 80k+ per second rate streamed from producers.

Flume's durability is file based. The file-based channel does not replicate event data to a different node. It totally depends on the durability of the storage it writes upon. If message durability is crucial, it is recommended to use SAN or RAID.

Kafka supports both synchronous and asynchronous replication based on your durability requirements by using a commodity hardware.

Q40. Can you get the best of both worlds by combining Flume & Kafka?
A40. Yes. Flume is configuration driven and supports HDFS security, whereas Kafka is more scalable & durable, but you need to write custom code for the consumers.

The recent flume version has introduced "**Kafka Source**" to push data to a "**Kafka Sink**", via a more durable "**kafka channel**".

Q41. What is an anatomy of a Flume agent?
A41. Flume is deployed as one or more agents, which is its own instance of the Java Virtual Machine (JVM). Agents consist of three components defined via a configuration file with **sources**, **sinks**, and **channels**. An agent must have at least one of each in order to run.

Sources collect incoming data as events. Sinks write events out, and channels provide a queue to connect the source and sink. Flume sources listen for and consume events. Events can range from newline-terminated strings in stdout to HTTP POSTs and JMS messages sent to a messaging queue.

Channels assist with transfer of events from their sources to their sinks. Events written to the channel by a source are not removed from the channel until a sink removes that event in a transaction. In an unlikely event of the network between a Flume agent and a Hadoop cluster goes down, the channel will keep all events queued in memory or file until the sink can correctly write to the cluster and close its transactions with the channel.

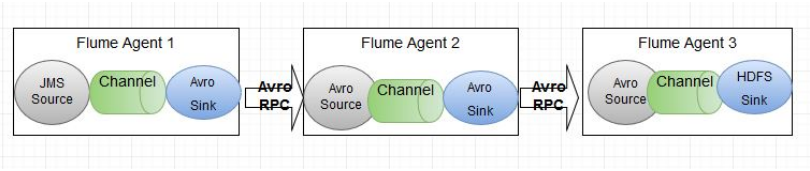
Sinks provide Flume agents plug and play output capability. You can plug different existing sinks like HDFS sink, HBase sink, etc via the configuration file or write your own sink class.

Q42. How does Flume support reliability
A42. Sinks remove events **transactionally** from the channel and write them to output. Transactions close when the event is successfully written, ensuring that all events are committed to their final destination.

In multi-hop Flume topologies, reliability is ensured by this Flume's transaction model. The sink on the sending agent does not close its transaction until receipt is acknowledged by the receiver. In the same vein, the receiver does not acknowledge receipt until the incoming event has been committed to its channel.

Q43. How do you chain and aggregate events in multi-hop flume topologies?

A43. Flume provides multi-hop deployments via Apache Avro-serialized RPC calls.



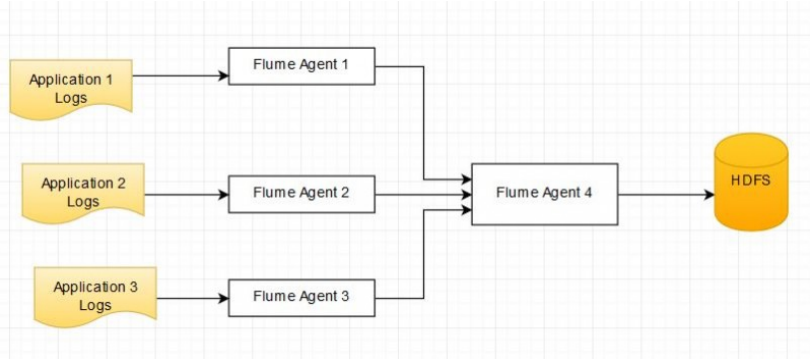
Flume multi-hop topology

Q44. Can you describe the “Fan-In” & “Fan-Out” multi-hop flume topologies?

A44.

Fan-In

This is the most common topology. The data flow in which the data will be transferred from many sources to one channel is known as fan-in flow.



Fan-In Flume Topology

Flume agents 1 to 3 write to respective Avro Sinks, and the Flume agent 4 collects to an “Avro source” and writes to a “HDFS Sink”.

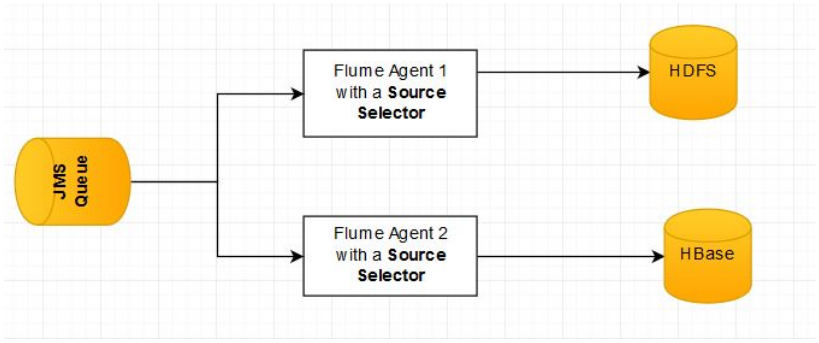
Fan-Out

The dataflow from one source to multiple channels is known as fan-out flow. There are 2 types of flows.

1) Replication flow − where the data will be replicated in all the configured channels.

2) Multiplexing flow – where the data will be sent to a selected channel which is mentioned in the header of the event.

Fan-out topologies are made possible via **Flume channel selectors**. An event can be written to all of the channels or to just selected ones based on some Flume header value. The internal mechanism for this in Flume is called a **channel selector**. Selectors will be replicating, and sending all events to multiple channels (aka **multiplexing**). Multiplexed sources can be partitioned by mappings defined on events via interceptors.



fan-out flume topology

Multiplexed selectors are useful when data is destined for partitioned Hive tables.

Q45. How does Flume handle in flight modification of data?
A45. Flume uses "**interceptors**" for the inflight modification of flume events. **Interceptors sit between a source & a channel**. Interceptors can transform or decorate incoming data. Interceptors can decorate the incoming events with metadata to be multiplexed or to be processed via a **processor** after written to a sink. The common decorating interceptors are timestamps, host names, and other static data.

The **Regex Filtering Interceptor** allows events to be dropped if when they match the provided regular expression. The regex is useful when the static data is inflexible.

Q46. What are uses of a Sink processor in Apache Flume?
A46. Sink Processor is the mechanism through which we can create fail-over paths. For example, when there are a lot of events, we can configure the load balancing using the **sink processor** so that the load can be distributed to multiple sinks.

A **sink group** is used to create a logical grouping of sinks. The behavior of this grouping is dictated by the sink processor, which determine how events are routed.

Q47. When would you need to write a custom Flume **deserializer**?
A47. The default deserializer of Flume's Spooling Directory Source is **LineDeserializer**, which simply parses each line as an Flume event. A custom deserializer is required in scenarios where you want to parse the whole XML file and extract events based on the XML structure.

Creating a custom deserializer involves implementing the **EventDeserializer** interface. A custom deserializer reads data from an InputStream and output List<Event> through the readEvents() function.

Q48. When would you need to write a custom Flume **serializer**?
A48. Custom **serializers** are useful for writing events into HDFS or HBase in a format of the user's choice.

For example, Apache Flume with Custom classes for JMS Source & HDFS Sink shows how a custom serializer includes "JMSCorrelationID" as a **composite key** along with the timestamp when writing to a sequence file. This tutorial also demonstrates a custom JMS **message converter** to capture the "JMSCorrelationID" from the JMS message properties to the Flume event headers.

Q49. What is a Flume event?

A49. A unit of data with some header values is known as a Flume event. For example, look at the "org.apache.flume.event.SimpleEvent" API, which has getBody(), getHeaders(), setBody(byte[] body), and setHeaders(Map<String,String> headers) methods.

Q50. What are the important steps in a Flume configuration?

A50. Each source must have **at least** one channel, every sink must have **only one** channel, and every component must have a specific type (e.g. jms, memory, hdfs, etc).

JMS Source to HDFS Sink example

```
1
2  eai_agent.sources = jms
3  eai_agent.channels = memory
4  eai_agent.sinks = hadoop
5
6  eai_agent.sources.jms.channels=memory
7  eai_agent.sinks.hadoop.channel=memory
8
9  # Websphere MQ Source
10
11 eai_agent.sources.jms.type = jms
12 eai_agent.sources.jms.providerURL = file:///home
13 eai_agent.sources.jms.initialContextFactory = co
14 eai_agent.sources.jms.destinationType=QUEUE
15 eai_agent.sources.jms.destinationName=MyQueue
16 eai_agent.sources.jms.connectionFactory=ABCQUEUE
17 eai_agent.sources.jms.batchSize=10
18
19 # Channels
20 eai_agent.channels.memory.type = memory
21 eai_agent.channels.memory.capacity = 10000
22 eai_agent.channels.memory.transactionCapacity =
23
24
25 # HDFS Sink
26 eai_agent.sinks.hadoop.type = hdfs
27 eai_agent.sinks.hadoop.hdfs.fileType = DataStrea
28 eai_agent.sinks.hadoop.serializer = header_and_t
29 eai_agent.sinks.hadoop.hdfs.useLocalTimeStamp =
30 eai_agent.sinks.hadoop.hdfs.path=/user/GG/Respon
31 eai_agent.sinks.hadoop.hdfs.filePrefix=GG_MYFILE
32 eai_agent.sinks.hadoop.hdfs.rollInterval = 30
33 eai_agent.sinks.hadoop.hdfs.rollSize = 104857600
34
35
```

Popular Posts

- 11 Spring boot interview questions & answers

828 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

768 views

18 Java scenarios based interview Questions and Answers

401 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

389 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

296 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

286 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

280 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

240 views

001B: ♦ Java architecture & design concepts interview questions & answers

202 views

Bio | Latest Posts

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription

based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 02: Apache Flume with Custom classes for JMS Source & HDFS Sink

03: Spark submit – reading a Sequence File from HDFS ›

Posted in Hadoop & BigData Interview Q&A, member-paid

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from “Blah blah” to “Wow”? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category

© Disclaimer

 ↑