**ASSIGNMENT 12.1**

1. **Explain the need of Flume:**

Apache Flume is a distributed reliable and available service for efficiently collecting, aggregating and moving large amounts of streaming data into the Hadoop Distributed File System (HDFS). It has a simple and flexible architecture based on streaming data flows and is robust and fault tolerant with tunable reliability mechanisms for failover and recover.

Need for FLUME:

1. Often it is assumed that the data is already in HDFS, or can be copied there in bulk.
2. However, there are many systems that don’t meet this assumption.
3. They produce streams of data that we would like to aggregate, store, and analyze using Hadoop — and these are the systems that Apache Flume is an ideal fit for.
4. Flume is designed for high-volume ingestion into Hadoop of event-based data.
5. The canonical example is using Flume to collect log files from a bank of web servers, then moving the log events from those files into new aggregated files in HDFS for processing.
6. The usual destination (or sink in Flume parlance) is HDFS. However, Flume is flexible enough to write to other systems, like HBase or Solr.
7. To use Flume, we need to run a Flume agent, which is a long-lived Java process that runs sources and sinks, connected by channels. A source in Flume produces events and delivers them to the channel, which stores the events until they are forwarded to the sink.
8. You can think of the source-channel-sink combination as a basic Flume building block.
9. Flume installation is made up of a collection of connected agents running in a distributed topology.
10. Agents on the edge of the system (co-located on web server machines, for example) collect data and forward it to agents that are responsible for aggregating and then storing the data in its final destination.
11. Agents are configured to run a collection of particular sources and sinks, so using Flume is mainly a configuration exercise in wiring the pieces together.
12. YARN co-ordinates data from APACHE Flume and other services that deliver raw data into an Enterprise Hadoop cluster.
13. **Explain the working of Flume and its components in brief.**

Flume lets Hadoop users ingest high-volume streaming data into HDFS for storage. The architecture of Flume is based on a few concepts that together help achieve this objective. Some of these concepts have existed in the past implementation, but have changed drastically. Here is a summary of concepts that Flume introduces, reuses or redefines from earlier implementation:

1. **EVENT:**

A byte payload with optional string headers that represent the unit of data that Flume can transport from its point of origination to its final destination.

1. **FLOW:**

Movement of events from point of origin to their final destination is considered a data flow, or simply flow.

1. **CLIENT:**

An interface implementation that operates at the point of origin of events and delivers them to a Flume agent. Clients typically operate in the process space of the application they are consuming data from.

1. **AGENT:**

An independent process that hosts flume components such as sources, channels and sinks, and thus has the ability to receive, store and forward events to their next destination.

1. **SOURCE:**

An interface implementation that can consume events delivered to it via a specific mechanism. When a source receives an event, it hands it over to one or more channels.

1. **CHANNEL:**

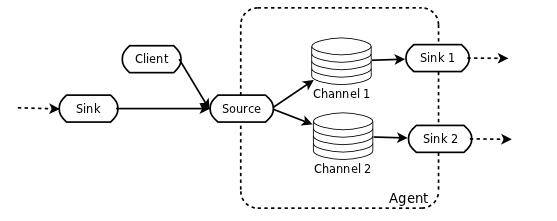
A transient store for events, where events are delivered to channel through sources operating within the agent. An event put in a channel stays in that channel until a sink removes it for further transport. Channel plays an important role in ensuring durability of the flows.

1. **SINK:**

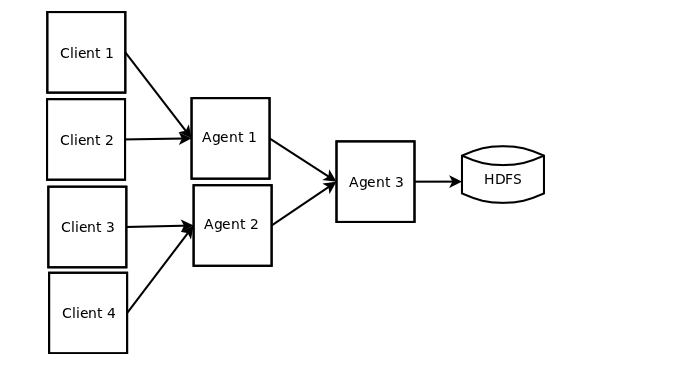
An interface implementation that can remove events from a channel and transmit them to the next agent in the flow, or to the event’s final destination. Sinks that transmit the event to its final destination are also known as terminal sinks. The Flume HDFS sink is an example of a terminal sink. Whereas the FLUME HDFS sink is an example of a terminal sink. Whereas Flume AVRO sink is an example of a regular sink that can transmit messages to other agents that are running an AVRO sources.

**Working:**

The flow in the FLUME starts from the client. The client transmits the event to its next destination. This destination is an agent i.e (the destination is a source operating within the agent). The source receiving this event will then deliver it to one or more channels.

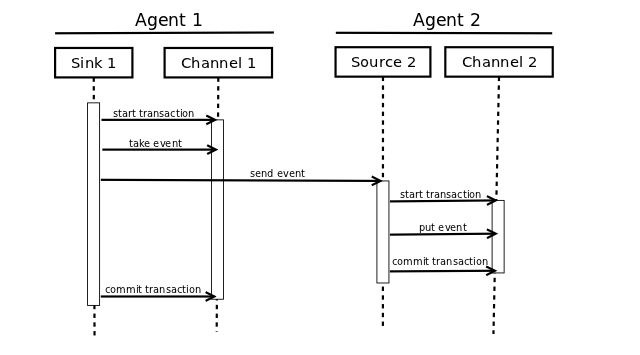


The channels that receive the event are drained by one or more sinks operating within the same agent. If the sink is a regular sink, it will forward the event to its next destination which will be another agent. If instead it is a terminal sink, it will forward the event to its final destination. Channels allow the decoupling of sources from sinks using the familiar producer-consumer model of data exchange. This allows sources and sinks to have different performance and runtime characteristics and yet be able to effectively use the physical resources available to the system.



By configuring a source to deliver the event to more than one channel flows can fan-out to more than one destination.

**RELIABILITY AND FAILURE HANDLING:**

Flume uses channel based transactions to guarantee reliable message deliver. When a message moves from one agent to another, two transactions are started, one on the agent that delivers the event and the other on the agent that receives the event.  In order for the sending agent it commit its transaction, it must receive success indication from the receiving agent. The receiving agent only returns a success indication if its own transaction commits properly first.