Tos

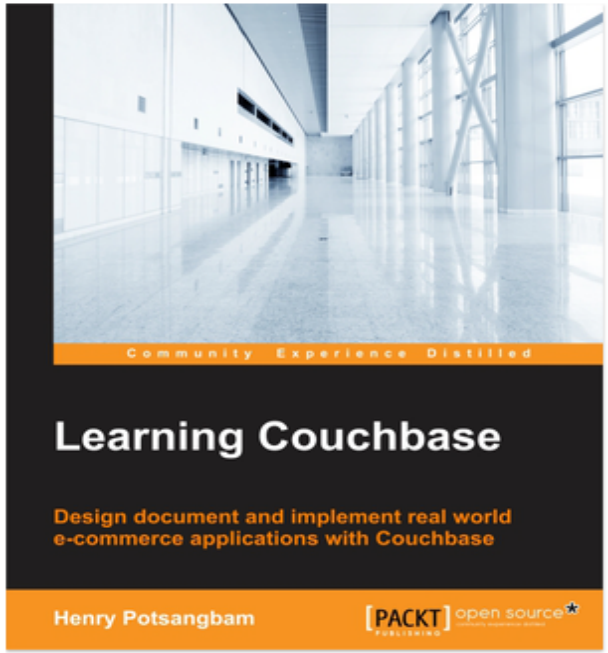
Kafka – Admins & Operations

Henry R.P

- **Certified Cassandra Admin**
- **Mapr Certified – Hadoop Administrator**
- **IBM Certified Application Developer**
- **IBM Certified Solution Designer**
- **SAP Certified ABAP & Portal Consultant .**
- **CIPM – Certificate in Project Management.**
- **TOGAF – Enterprise Architect**

IT Architect & Corporate trainer
20 +Year of IT Experience

NOSQL, Streaming Platform & Bigdata



Author

Clientele

Tos

Outline

Tos

Introduction
Real-Time Processing is Becoming Prevalent
Kafka: A Stream Data Platform
Kafka Fundamentals
An Overview of Kafka
Kafka Producers

Kafka CLI
Zookeeper

Managing a Kafka Cluster
Installing and Running Kafka
Monitoring Kafka
Basic Cluster Management
Log Retention and Compaction

Providing Durability
Basic Replication Concepts
Durability Through Intra-Cluster Replication
Writing Data to Kafka Reliably
Broker Shutdown and Failures
Controllers in the Cluster
The Kafka Log Files
Offset Management

Producer Architecture

Java API - Kafka Consumer.
Failover and Consumer Failover

Consumer
Push vs. pull
Message Delivery
Reprocess of the Failed Message
Message Offset management .

Designing for High Availability
Kafka Reference Architecture
Brokers

Optimizing Kafka Performance
Producer Performance
Broker Performance
Load Balancing Consumption
Consumption Performance
Performance Testing

Kafka Security
SSL for Encryption and Authentication
SASL for Authentication
Data at Rest Encryption

Kafka – DR – (Kafka mirroring (MirrorMaker))

Integrating Systems with Kafka Connect
The Motivation for Kafka Connect
Types of Connectors
Kafka Connect Implementation

KAFKA STREAMS
Understands Streams Application
Core Concepts
Architecture

KSQL

Monitoring Kafka

Introduce Yourself.

Tos

Name
Year of Experience.
Skills Level
- Java / Linux
- Messaging System / Kafka

Expectation, if any.

Note: Basic knowledge of Java & Linux are required.

Learning Objectives : Operations of Kafka & API to connect to it.

Schedule

Tos

Time	
9.30 – 11.00 AM	Session I
11.00 AM to 11.15 AM	Tea Break
11.15 AM to 12.45 PM	Session II
12.45 PM to 1.45 PM	Lunch Break
1.45 PM to 3.15 PM	Session III
3.15 PM to 3.30 PM	Tea Break
3.30 PM to 5.30 PM	Session IV

30 January 2022

Kafka – An Overview

Why Kafka is Needed?

Tos

- Real time streaming data processed for real time analytics
- Service calls, track every call, IOT sensors
- Apache Kafka is a fast, scalable, durable, and fault- tolerant publish-subscribe messaging system
- Kafka is often used instead of JMS, RabbitMQ and AMQP
- higher throughput, reliability and replication

Why is Kafka needed? 2

Tos

- Kafka can works in combination with
  - Flume/Flafka, Spark Streaming, Storm, HBase and Spark for real-time analysis and processing of streaming data
  - Feed your data lakes with data streams
- Kafka brokers support massive message streams for follow- up analysis in Hadoop or Spark
- Kafka Streaming/KSQL (subproject) can be used for real-time analytics

Why is Kafka Popular?

Tos

- ***Great performance***
- Operational Simplicity, easy to setup and use, easy to reason
- Stable, Reliable Durability,
- Flexible Publish-subscribe/queue (scales with N-number of consumer groups),
- Robust Replication,
- Producer Tunable Consistency Guarantees,
- Ordering Preserved at shard level (Topic Partition)
- Works well with systems that have data streams to process, aggregate, transform & load into other stores

Kafka growth exploding

Tos

- 1/3 of all Fortune 500 companies
- Top ten travel companies, 7 of ten top banks, 8 of ten top insurance companies, 9 of ten top telecom companies
- LinkedIn, Microsoft and Netflix process 4 comma message a day with Kafka (1,000,000,000,000)
- Real-time streams of data, used to collect big data or to do real time analysis (or both)

Who uses Kafka?

Tos

- ***LinkedIn***: Activity data and operational metrics
- ***Twitter***: Uses it as part of Storm – stream processing infrastructure
- ***Square***: Kafka as bus to move all system events to various Square data centers (logs, custom events, metrics, an so on). Outputs to Splunk, Graphite, Esper-like alerting systems
- Spotify, Uber, Tumbler, Goldman Sachs, PayPal, Box, Cisco, CloudFlare, DataDog, LucidWorks, MailChimp, NetFlix, etc.

Why is Kafka so fast?

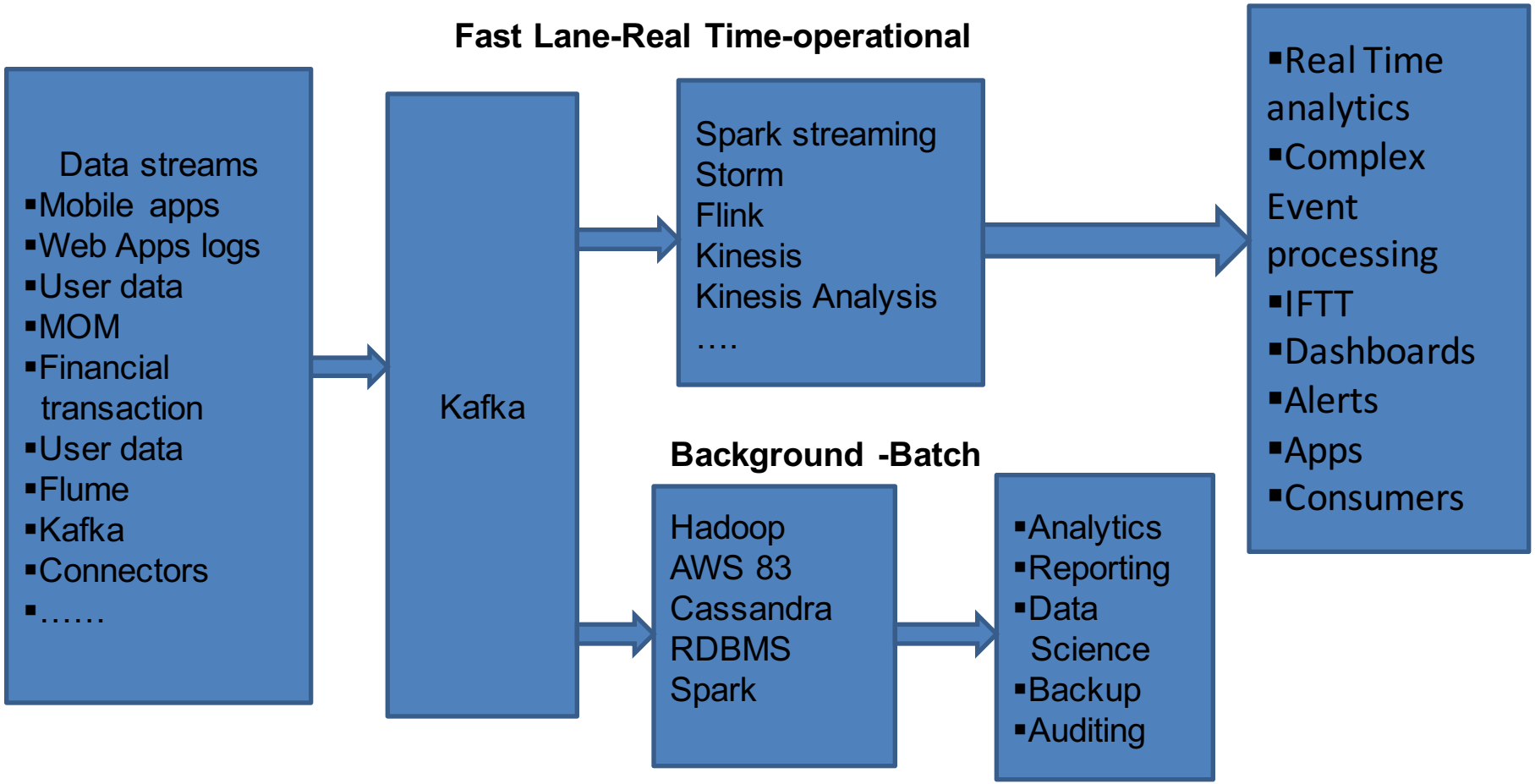
Tos

- Zero Copy - calls the OS kernel direct rather to move data fast
- **Batch Data in Chunks** - Batches data into chunks
  - end to end from Producer to file system to Consumer
  - Provides More efficient data compression. Reduces I/O latency
- **Sequential Disk Writes** - Avoids Random Disk Access
  - writes to immutable commit log. No slow disk seeking. No random I/O operations. Disk accessed in sequential manner
- **Horizontal Scale** - uses 100s to thousands of partitions for a single topic
  - spread out to thousands of servers
  - handle massive load

Kafka: A Stream Data Platform

Kafka Streaming Architecture

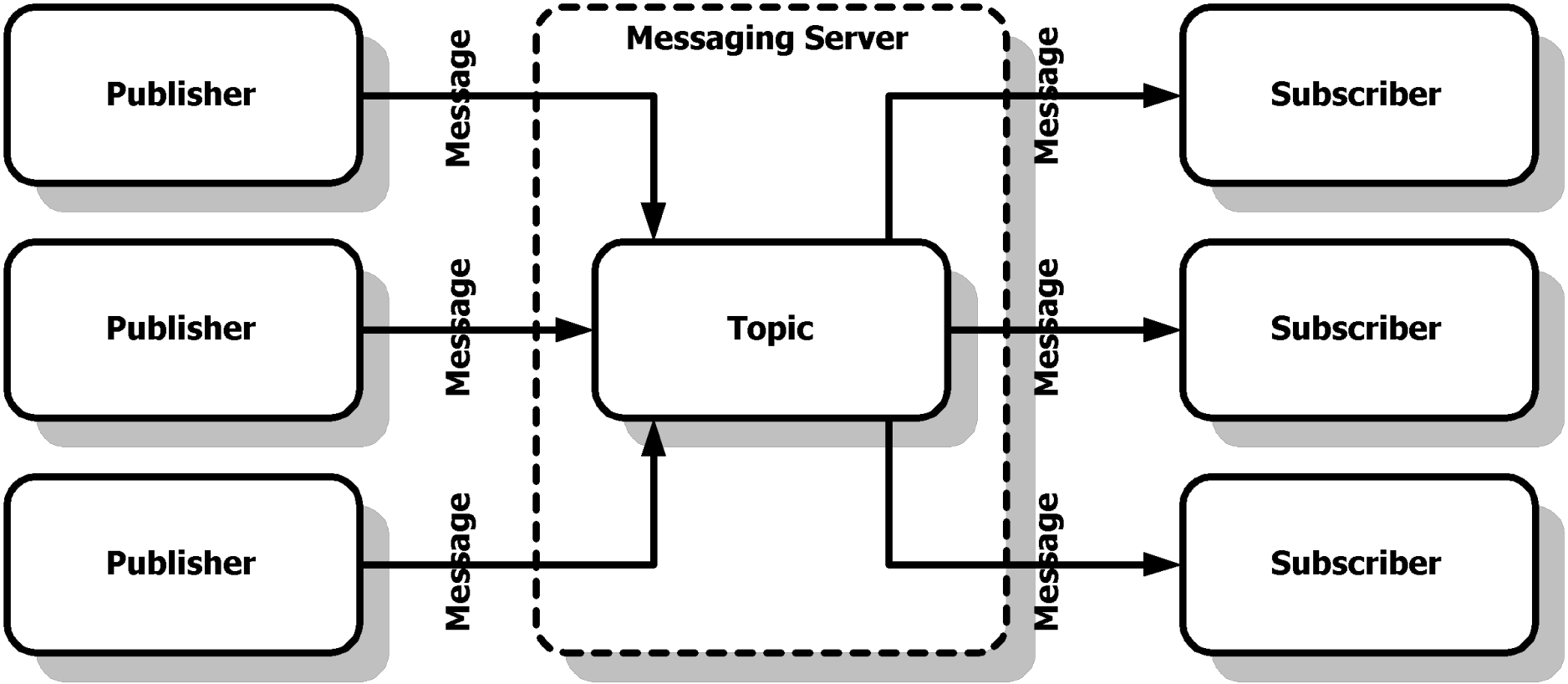
Tos



What is Kafka?

What is Kafka?

Tos



What is Kafka?

Tos

- Distributed Streaming Platform
  - Publish and Subscribe to streams of records
  - Fault tolerant storage
    - Replicates Topic Log Partitions to multiple servers
  - Process records as they occur
  - Fast, efficient IO, batching, compression, and more
- Used to decouple data streams

Kafka helps decouple data streams

Tos

- Kafka decouple data streams
- producers don’t know about consumers
- Flexible message consumption
  - Kafka broker delegates log partition offset (location) to Consumers (clients)

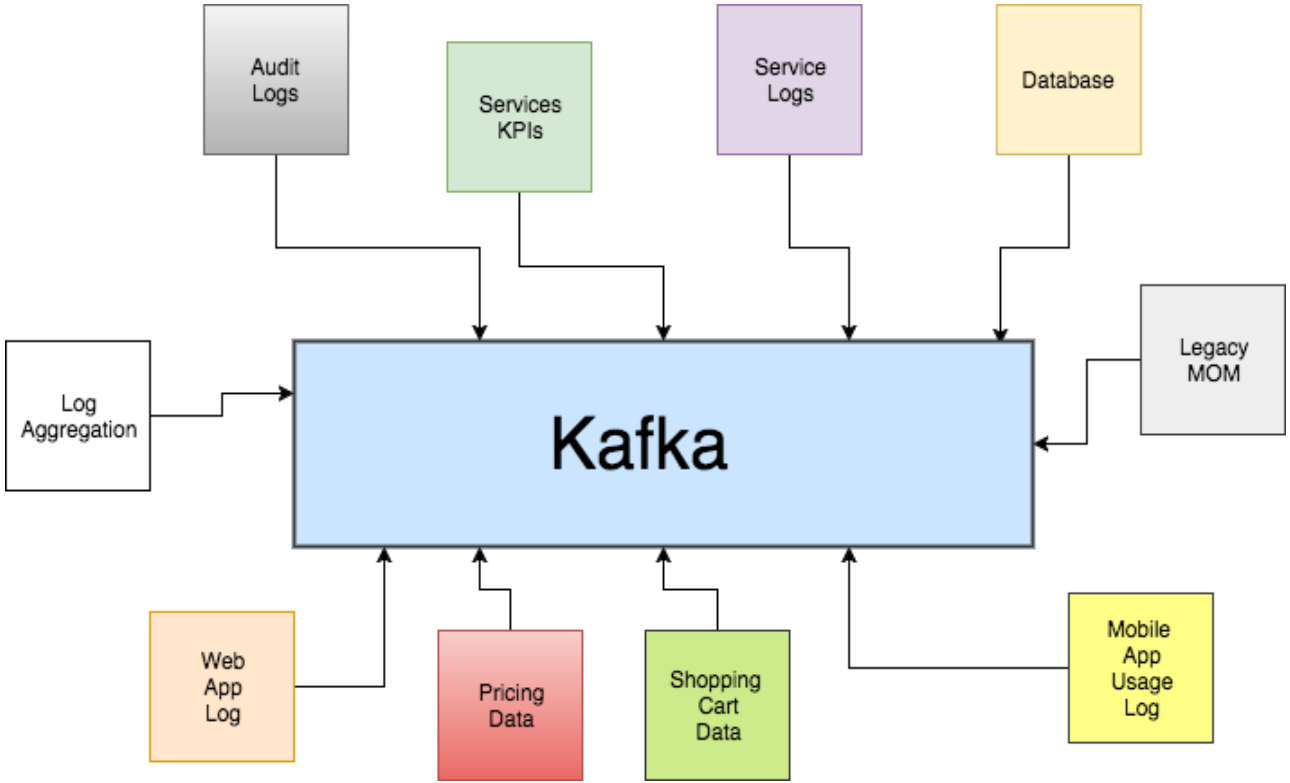
Kafka messaging allows

Tos

- Feeding of high-latency daily or hourly data analysis into Spark, Hadoop, etc.
- Feeding micro services real-time messages
- Sending events to CEP system
- Feeding data to do real-time analytic systems
- Up to date dashboards and summaries
- At same time

Kafka Decoupling Data Streams

Tos



Don’t couple the stream

Kafka Polyglot clients / Wire protocol

Tos

- Kafka communication from clients and servers wire protocol over TCP protocol
- Protocol versioned
- Maintains backwards compatibility
- Many languages supported
- Kafka REST proxy allows easy integration (not part of core)
- Also provides Avro/Schema registry support via Kafka ecosystem (not part of core)

Kafka Usage

Tos

- Build real-time streaming applications that react to streams
  - Real-time data analytics
  - Transform, react, aggregate, join real-time data flows
  - Feed events to CEP for complex event processing
  - Feed data lakes
- Build real-time streaming data pipe-lines
  - Enable in-memory micro services (actors, Akka, Vert.x, Qbit, RxJava)

Kafka Use Cases

Tos

- Stream Processing
- Website Activity Tracking
- Metrics Collection and Monitoring
- Log Aggregation
- Real time analytics
- Capture and ingest data into Spark / Hadoop
- CRQS, replay, error recovery
- Guaranteed distributed commit log for in-memory computing

Kafka Use Cases

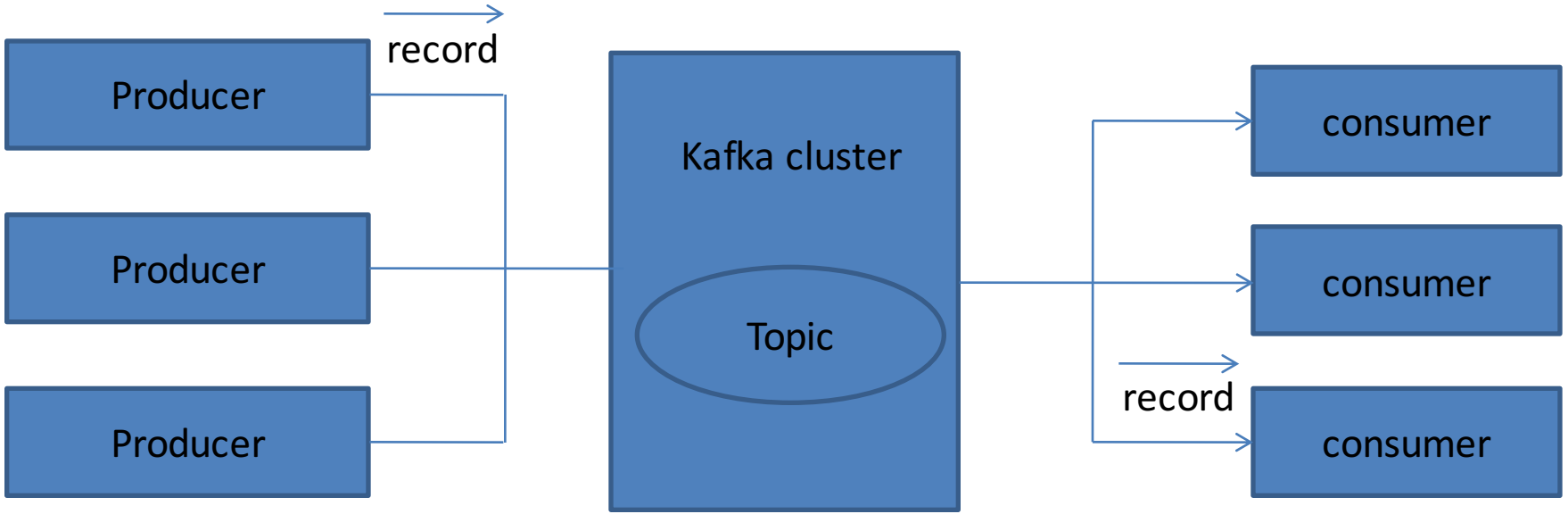
Tos

- Metrics / KPIs gathering
  - Aggregate statistics from many sources
- Event Sourcing
  - Used with micro services (in-memory) and actor systems
- Commit Log
  - External commit log for distributed systems. Replicated data between nodes, re-sync for nodes to restore state
- Real-time data analytics, Stream Processing, Log Aggregation, Messaging, Click-stream tracking, Audit trail, etc.

Kafka Architecture

Kafka: Topics, Producers, and Consumers

Tos



Kafka Fundamentals

Tos

- ***Records*** have a ***key (optional)***, ***value*** and ***timestamp; Immutable***
- ***Topic*** a stream of records (“/orders”, “/user-signups”), feed name
  - ***Log*** topic storage on disk
  - ***Partition*** / Segments (parts of Topic Log)
- ***Producer*** API to produce a streams or records
- ***Consumer*** API to consume a stream of records
- ***Broker***: Kafka server that runs in a Kafka Cluster. Brokers form a cluster. Cluster consists on many Kafka Brokers on many servers.
- ***Zoo Keeper***: Does coordination of brokers/cluster topology. Consistent file system for configuration information and leadership election for Broker Topic Partition Leaders

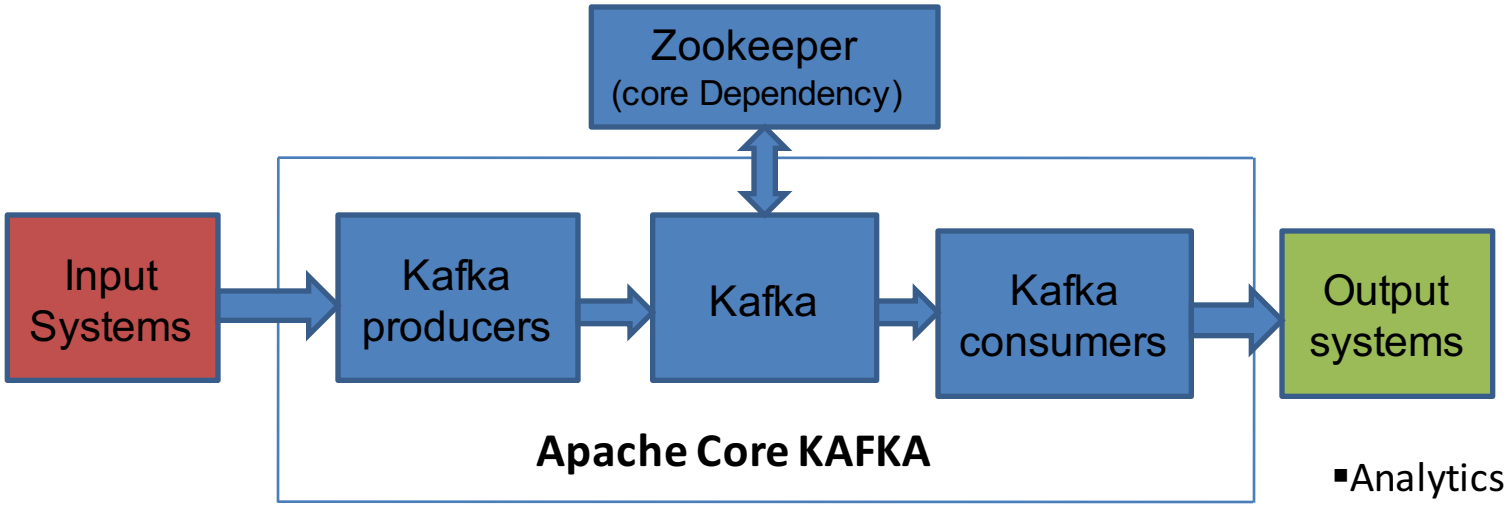
Apache Kafka - Core Kafka

Tos

- Kafka gets conflated with Kafka ecosystem
- Apache Core Kafka consists of Kafka Broker, start up scripts for Zoo Keeper, and client APIs for Kafka
- Apache Core Kafka does ***not*** include
  - Confluent Schema Registry (not an Apache project)
  - Kafka REST Proxy (not an Apache project)
  - Kafka Connect
  - Kafka Streams

Apache Kafka

Tos



- Log aggregation
- Metrics
- KPIs
- Batch imports
- Audit trial
- User activity logs
- Web logs

Not part of core

- Scheme Registry
- Avro
- Kafka REST Proxy
- Kafka Connect
- Kafka Streams

Apache Kafka Core

- Server Broker
- Scripts to start libs
- Script to start up Zookeeper
- Utils to create topics
- Utils to monitor stats

- Analytics
- Databases
- Machine learning
- Dashboards
- Indexed for search
- Business Intelligene

Kafka needs Zookeeper

Tos

- Zookeeper helps with leadership election of Kafka Broker and Topic Partition pairs
- Zookeeper manages service discovery for Kafka Brokers that form the cluster
- Zookeeper sends changes to Kafka
  - New Broker join, Broker died, etc.
  - Topic removed, Topic added, etc.
- Zookeeper provides in-sync view of Kafka Cluster configure

Installation steps

Tos

Installing Java

#tar -xvf jdk-8u45-linux-x64.tar.gz -C /opt

Set in the path variable and JAVA_HOME

Include in the profile a follow

```
[root@tos opt]# more ~/.bashrc
# .bashrc

# User specific aliases and functions

alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'

export JAVA_HOME=/opt/jdk1.8.0_45

export PATH=$PATH:$JAVA_HOME/bin
# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
[root@tos opt]#
```

Installation steps…

Tos

Installing Zookeeper

The following example installs Zookeeper with a basic configuration in `/opt/zookeeper`, storing its data in `/opt/data/zookeeper`:

```
# tar -xvf zookeeper-3.4.12.tar.gz -C /opt


# mv zookeeper-3.4.12 /opt/zookeeper
# mkdir -p /opt/data/zookeeper
# vi /opt/zookeeper/conf/zoo.cfg
  tickTime=2000
  dataDir=/opt/data/zookeeper
  clientPort=2181
# /opt/zookeeper/bin/zkServer.sh start
```

```
[root@tos opt]# /opt/zookeeper/bin/zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /opt/zookeeper/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[root@tos opt]#
```

Tos

Installing a Kafka Broker

The following example installs Kafka in `/opt/kafka`, configured to use the

Zookeeper server started previously and to store the message log segments stored in /tmp/kafka-logs:

```
# tar -zxf kafka_2.12-1.1.0.tgz –C /opt
# mv kafka_2.12-1.1.0 /opt/kafka
# mkdir /opt/data/kafka-logs
# /opt/kafka/bin/kafka-server-start.sh -daemon /opt/kafka/config/server.properties
```

```
[root@tos opt]# /opt/kafka/bin/kafka-server-start.sh -daemon /opt/kafka/config/s
erver.properties
[root@tos opt]# jps
3476 Kafka
3499 Jps
2895 QuorumPeerMain
[root@tos opt]#
```

Lab : **Installation of Kafka**