Getting Started with Apache Kafka

Douglas Eadline

LIMULUS COMPUTING™

Presenter

Douglas Eadline

deadline@limulus-computing.com

@thedeadline

http://www.limulus-computing.com

- HPC/Hadoop Consultant/Writer/Educator
- Teach "Big Data Techniques", on-line Masters in Data Science from Juniata College *https://www.juniata.edu/academics/graduate-programs/data-science.php*

LIMULUS COMPUTING™

Outline

- **Segment 1:** Introduction and Training Goals (15 m)
- **Segment 2:** Why Do I need a Message Broker? (20 m)
- **Segment 3:** Kafka Components (25 m)
- **Segment 4:** Basic Examples (40 m)
- **Segment 5:** Using a KafkaEsque UI (30 m)
- **Segment 6:** Example 1: Streaming Weather Data (40 m)
- **Segment 7:** Example 2: Image Streaming with Kafka (40 m)
- **Segment 8:** Wrap-up, Questions, and Resources (10 m)

LIMULUS COMPUTING™

Segment 1

Introduction and Training Goals

LIMULUS COMPUTING™

Recommended Approach To Class

- Training covers a lot of material!
- Designed to get you started (“hello.c” approach)
- Sit back and watch the examples
- All examples are provided in a notes file
- I will refer to file throughout the class (cut and paste)
- The notes files are available for download along with some help on installing software from class web page.

LIMULUS COMPUTING™

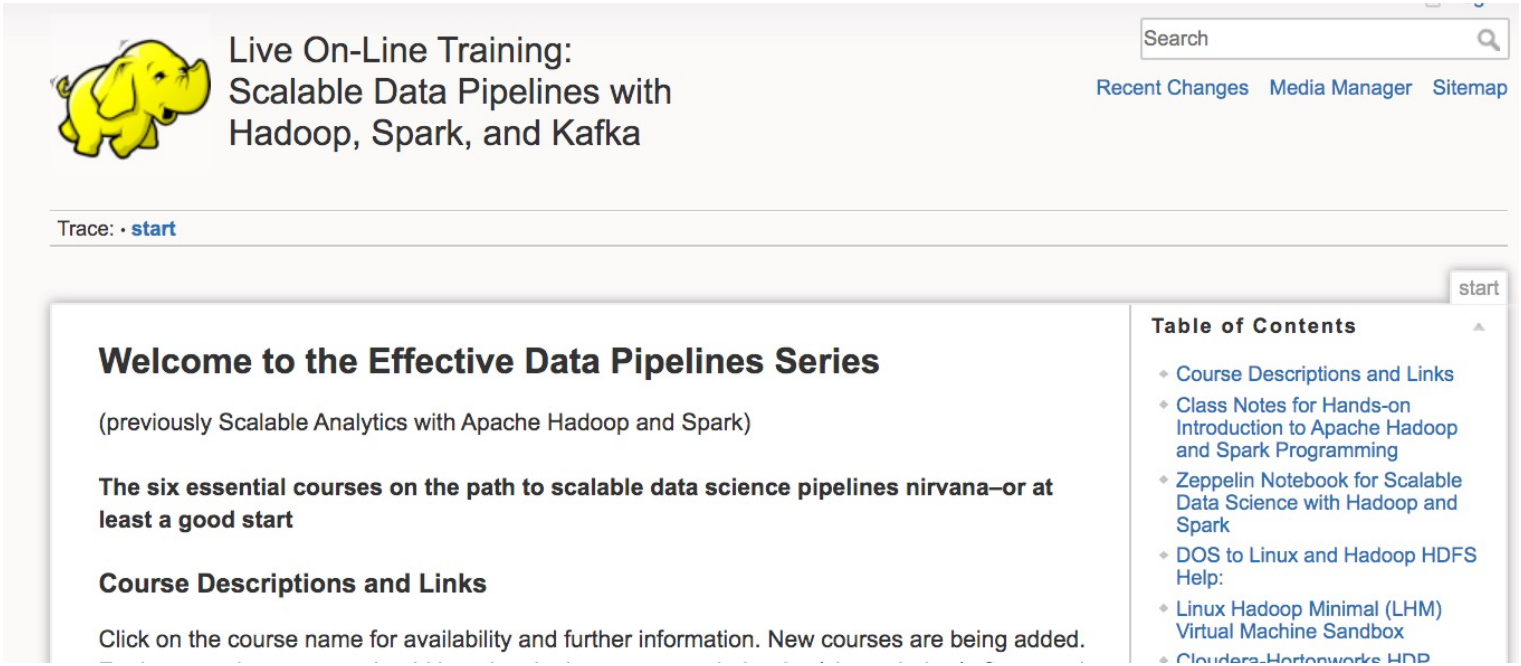
Trainings In Scalable Data Pipelines

1. **Apache Hadoop, Spark, and Kafka Foundations** (3 hours-1 day)
2. **Bash Programming Quick Start for Data Science** (3.5 hours-1 day)
3. **Hands-on Introduction to Apache Hadoop, Spark, and Kafka Programming** (6 hours-2 days)
4. **Data Engineering at Scale with Apache Hadoop and Spark** (3 hours-1 day)
5. **Scalable Analytics with Apache Hadoop, Spark, and Kafka** (3 hours-1 day)
6. **Up and Running with Kubernetes** (3 hours-1 day)

LIMULUS COMPUTING™

Training Webpage

https://www.clustermonkey.net/scalable-analytics



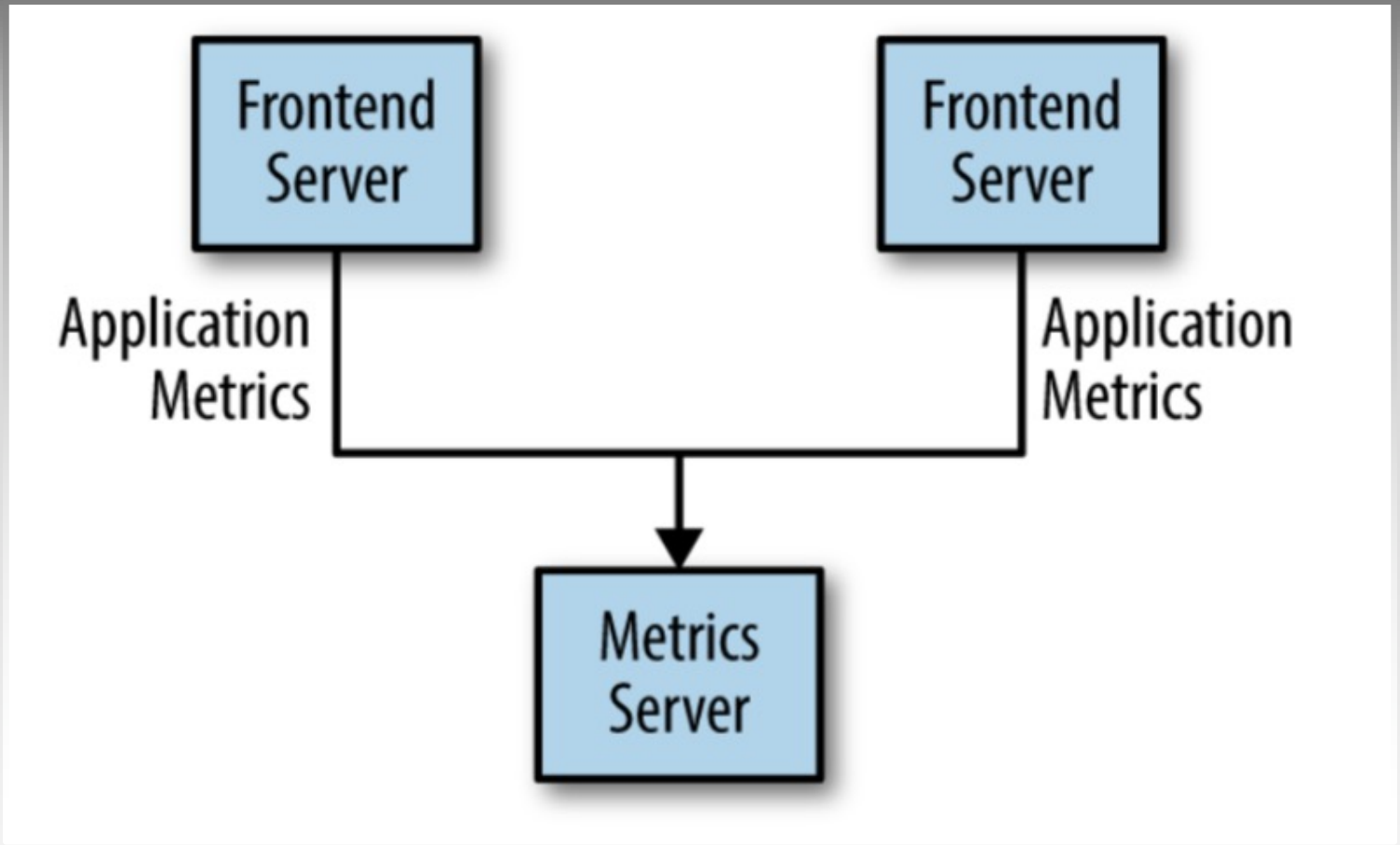
LIMULUS COMPUTING

Segment 2

Why Do I need a Message Broker?

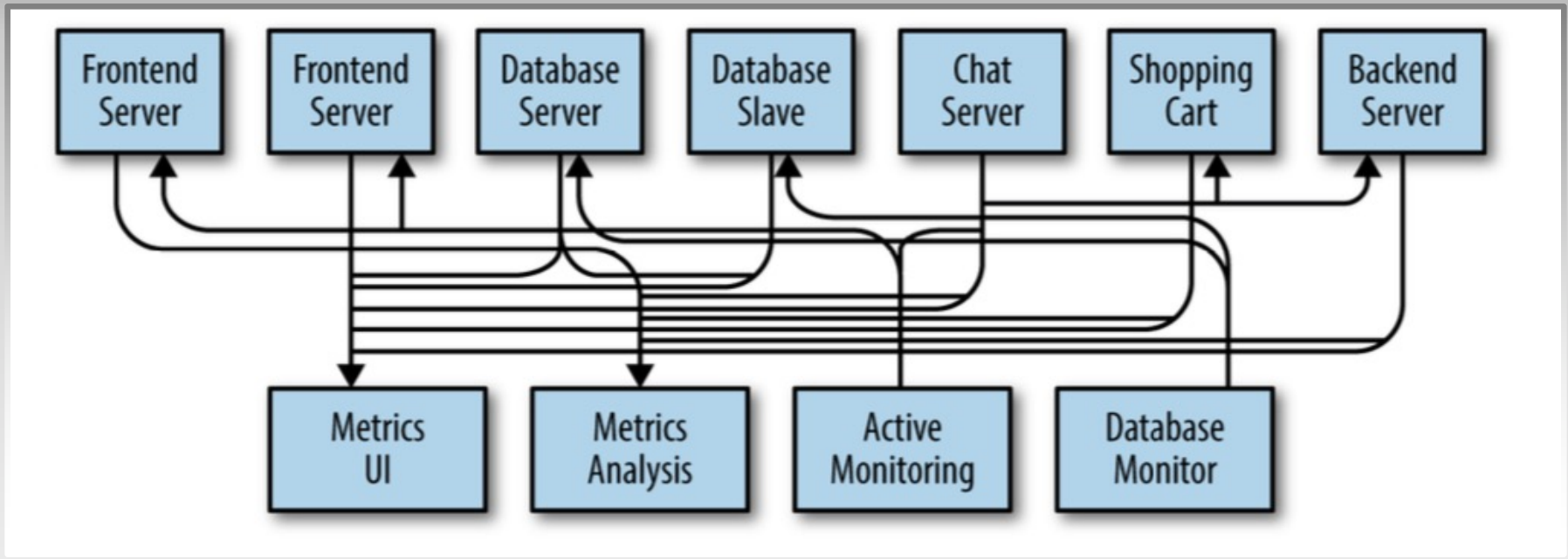
LIMULUS COMPUTING™

How It Started



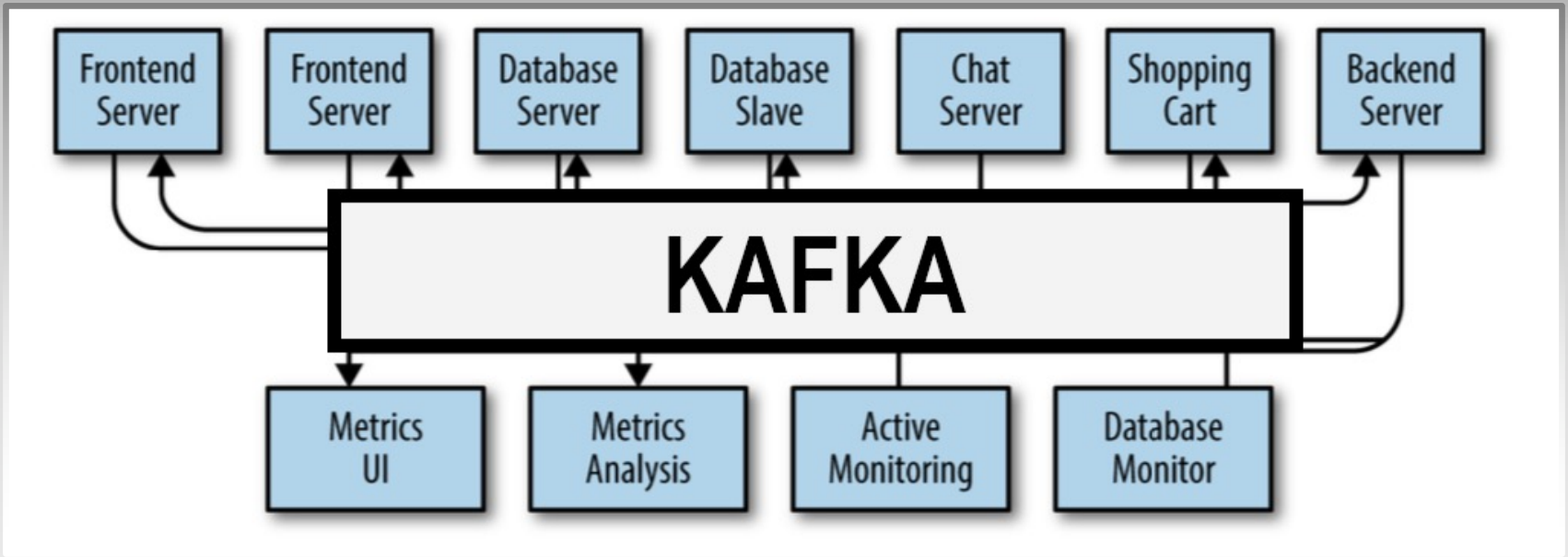
LIMULUS COMPUTING™

How It's Going



LIMULUS COMPUTING™

Message Broker



LIMULUS COMPUTING™

What is Kafka?

- Kafka is used for real-time streams of data, to collect big data, or to do real time analysis (or both) “a message broker”
- Kafka is a high throughput, scalable, and reliable (replication) service that records all kinds of data in real-time.
- Data can include: logs, web apps, messages, manufacturing, database, weather, financial streams, and anything else.
- Kafka collects data from a variety of sources (streams) and time-lines so that it can be consistently available for processing at a later time (decouples input/output)

LIMULUS COMPUTING™

Kafka History

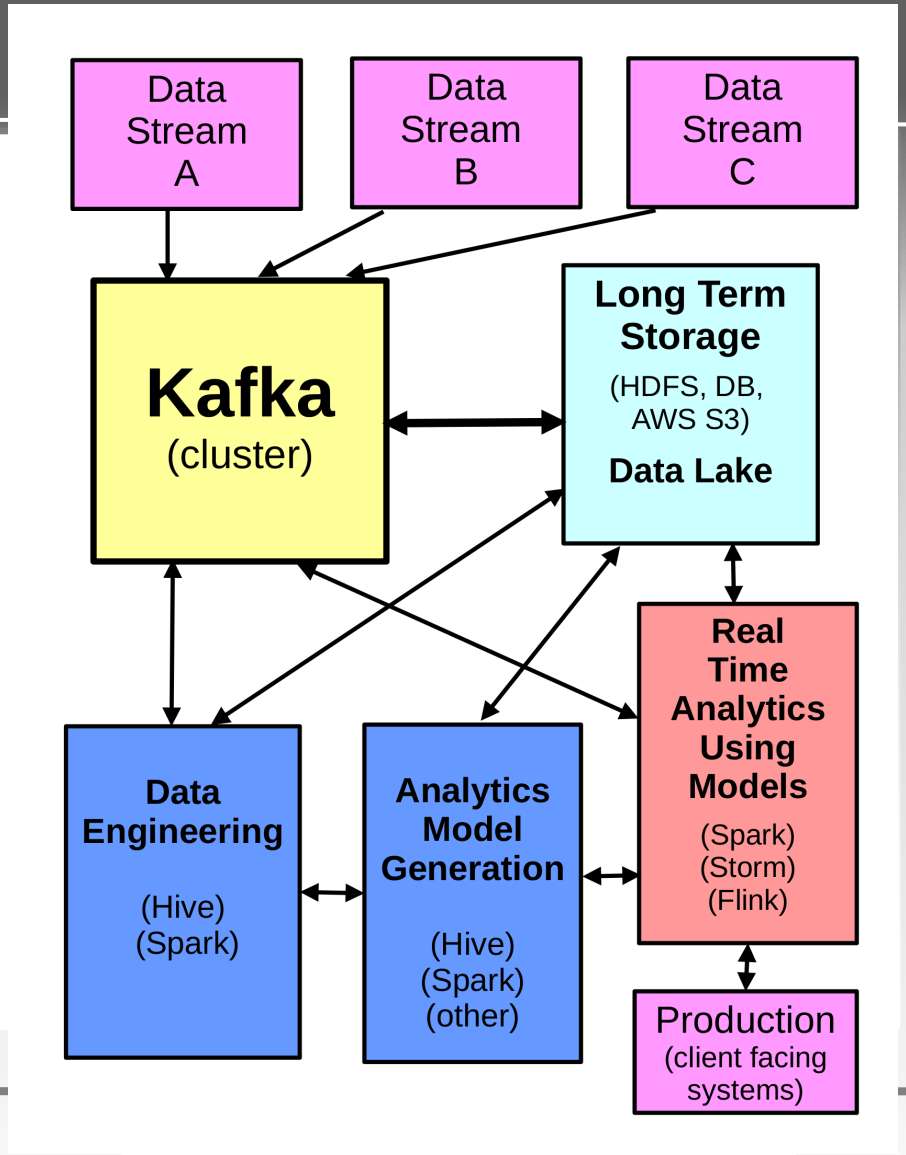
- Kafka was created at LinkedIn by Jay Kreps as a way to address the numerous data pipelines developing in their applications. It was designed handle many types of data and provide clean, structured data about user activity and system metrics in real time. (Kreps is now CEO of Confuent a Kafka management company.)
- Kafka became an Apache project in 2011 (open source).
- Why the name "Kafka"? Jay Kreps:

  "*I thought that since Kafka was a system optimized for writing, using a writer's name would make sense. I had taken a lot of lit classes in college and liked Franz Kafka. Plus the name sounded cool for an open source project.*"

LIMULUS COMPUTING™

Example Data Science Kafka Dataflow

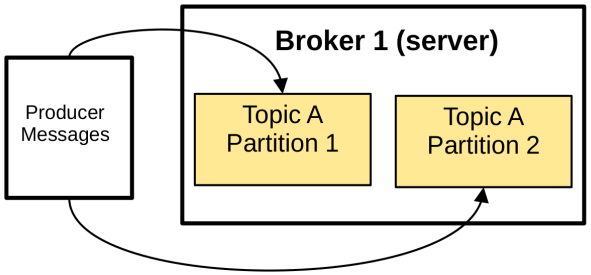
Kafka acts as a data broker (or buffer) keeping track of incoming data and allowing various applications (or groups) to access data in different ways at different times decoupled from the data source.



LIMULUS COMPUTING

Segment 3

Kafka Components



LIMULUS COMPUTING™

Kafka Concepts/Components

- A Kafka **Producer** is an independent application that sends data to the Kafka cluster. A producer publishes messages to one or more Kafka **Topics**.
- A Kafka **Consumer** is a client or independent program that consumes messages held by Kafka. The rate of consumption is independent of the rate that the producer sends messages.
- A Kafka **Topic** is a category/feed name to which messages are stored and published. Producer applications write data to topics and consumer applications read from topics.

LIMULUS COMPUTING™

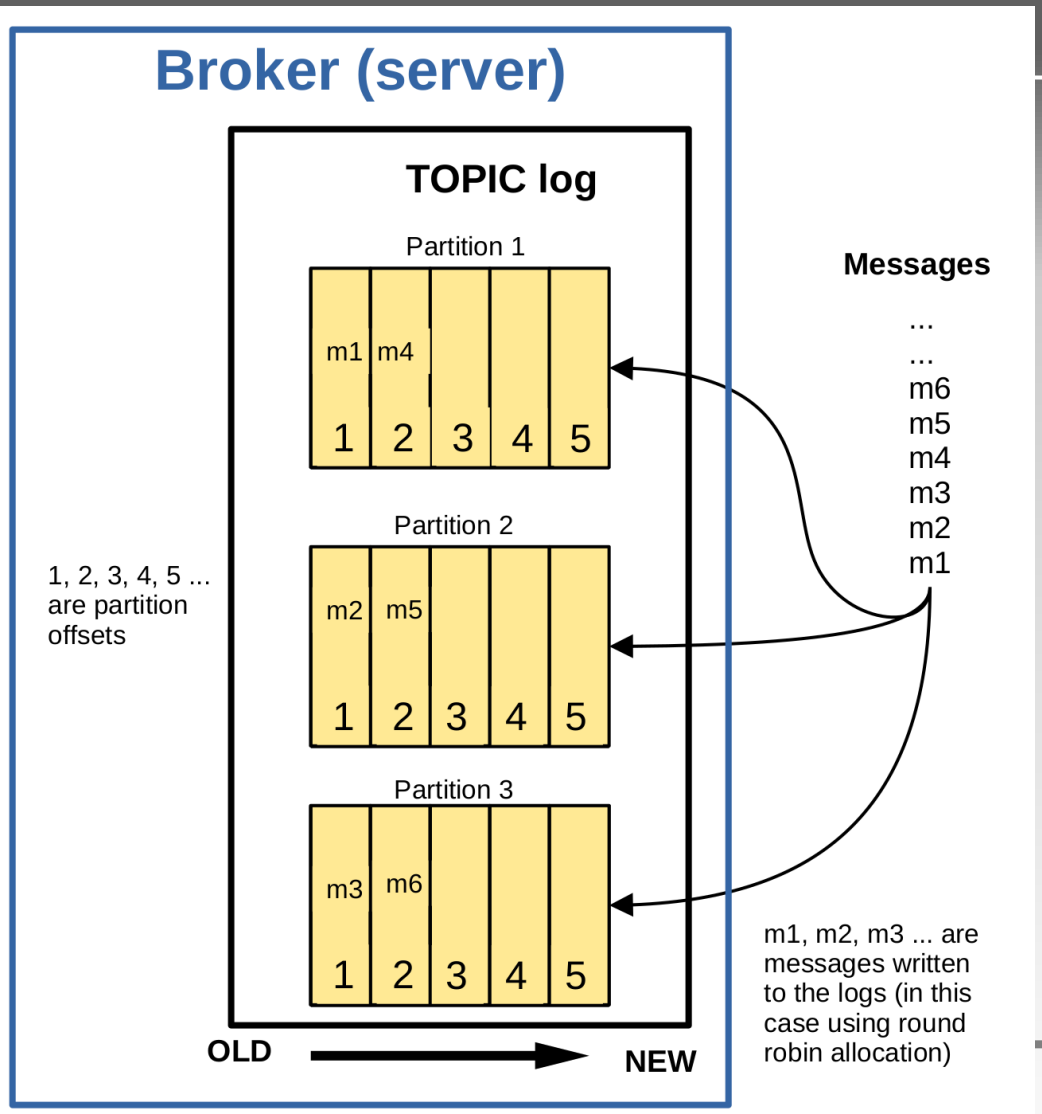
Kafka Brokers

- Sitting between Producer and Consumer applications is a Kafka **Broker**.
- A Broker is minimally one machine or machine instance.
- Brokers can manage many different **Topics.**
- For scalability (message throughput), multiple Brokers can used.
- Topics can be divided into a multiple partitions on one or more Brokers
- When using multiple Brokers, each Broker writes to its local Topic partition(s) providing scalable performance.
- For reliability, partitions are replicated on all Brokers.

LIMULUS COMPUTING™

Brokers - Topics - Partitions

- Broker runs on single server (or cloud instance)
- This Topic (log) has three Partitions
- Messages are written to each partition in a "round-robin" fashion
- Messages can be written to a specific Partition using a "key"



LIMULUS COMPUTING

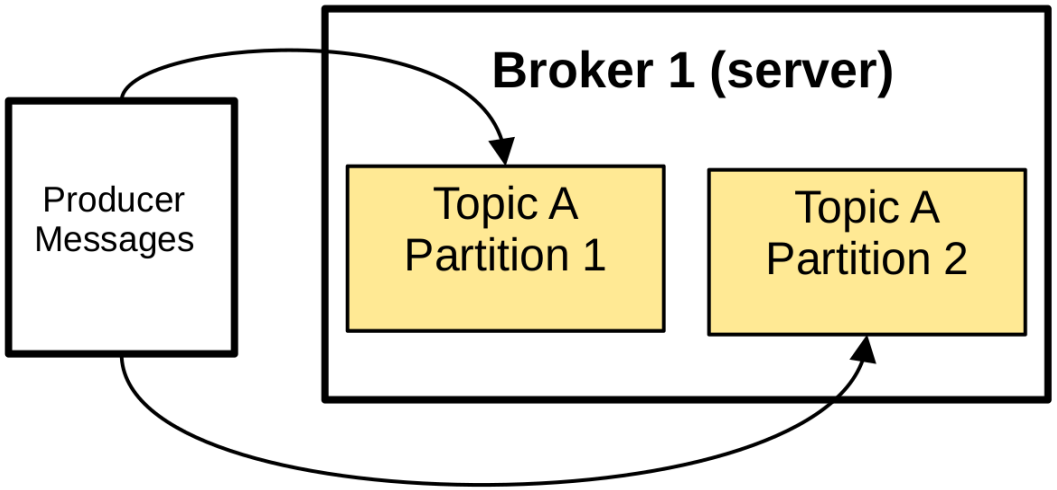
Kafka Topic Logs

- A Kafka Topic Log is an append-only log of events (messages). Similar to a web server log.
- A Topic Log can be spread across one or N partitions.
- Kafka logs can only be read by seeking an offset in the log and then moving forward in a sequential fashion.
- Topic logs are immutable--data is fixed until it expires.
- Topics can be configured to expire data after it has reached a certain age, or size. Ages can range from seconds to “forever.”

LIMULUS COMPUTING™

Partitioning and Redundancy

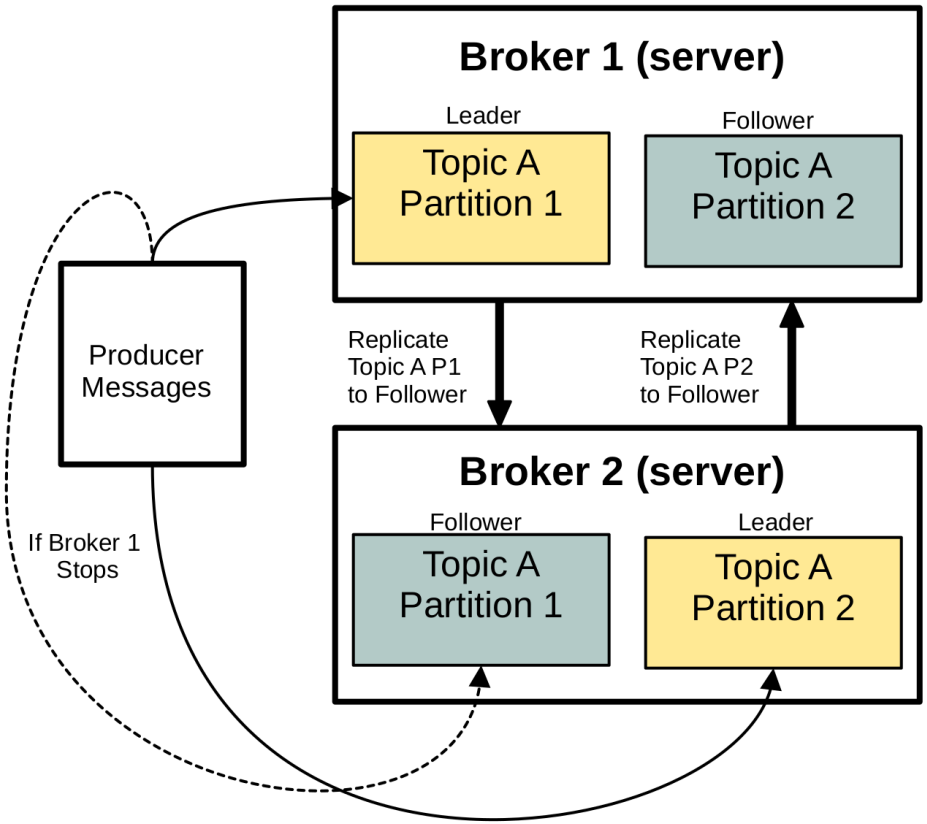
- A single Broker is vulnerable
- If the Broker server goes down, everything stops
- New data are lost and log data can't be retrieved



LIMULUS COMPUTING

Partition Replication for Producer

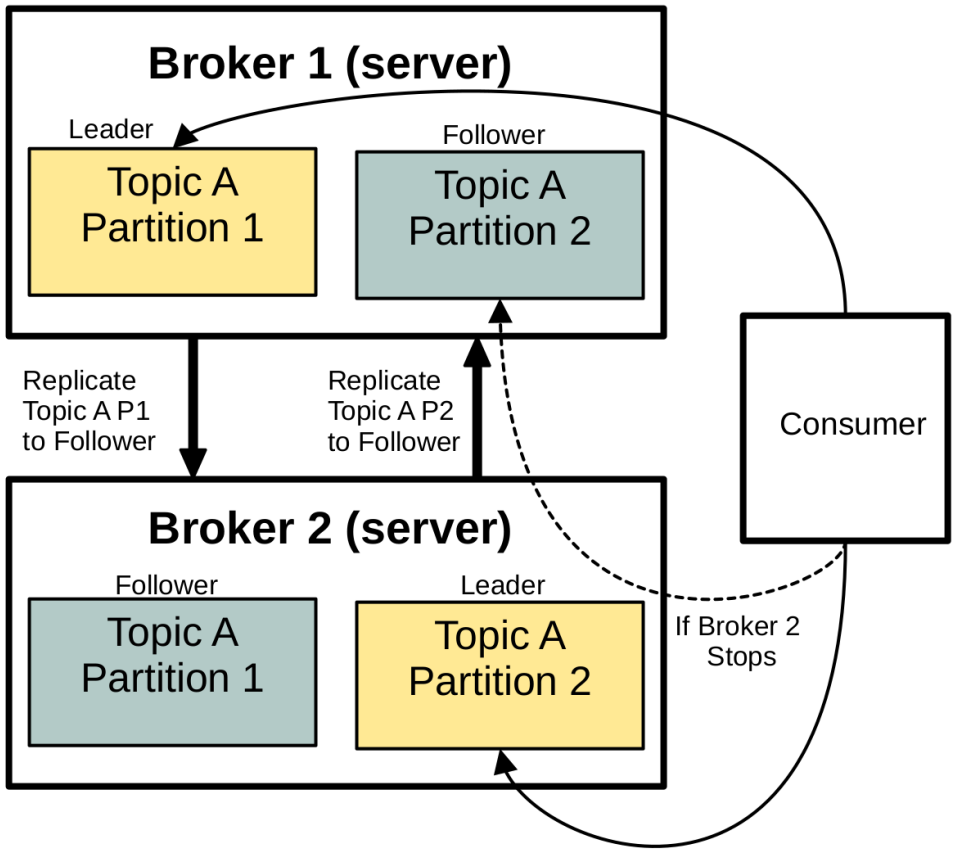
- Employ two or more Brokers (servers/instance)
- Each Partition has is replicated on other server
- One Partition is always the **Leader** and is active on different Broker
- Replicated **Follower** Partitions are kept in sync with "Leader" (happens in background)
- If a Broker 1 stops, **Follower** takes over (happens in background)



LIMULUS COMPUTING

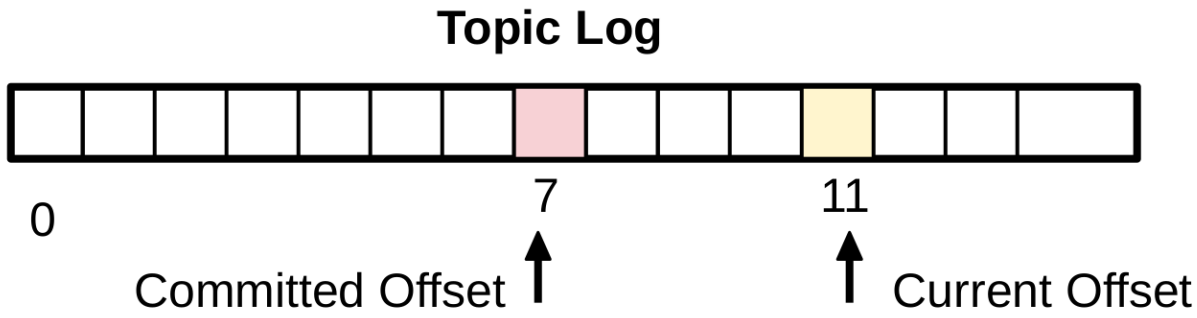
Partition Replication for Producer

- Like Producer failover, Consumers read from **Leader** partition
- If Broker 2 fails, **Follower** Partition becomes active



LIMULUS COMPUTING

Topic Log Offsets



- **Current offset** – starts at 0, each time consumer asks for messages the current offset becomes the next available offset. Used to avoid resending same records again to the same consumer.
- **Committed offset** - the position that a consumer has confirmed it has processed received messages (i.e. like a DB commit). Used to avoid resending same records to a new consumer in the event of partition rebalance or due to a consumer change.

LIMULUS COMPUTING™

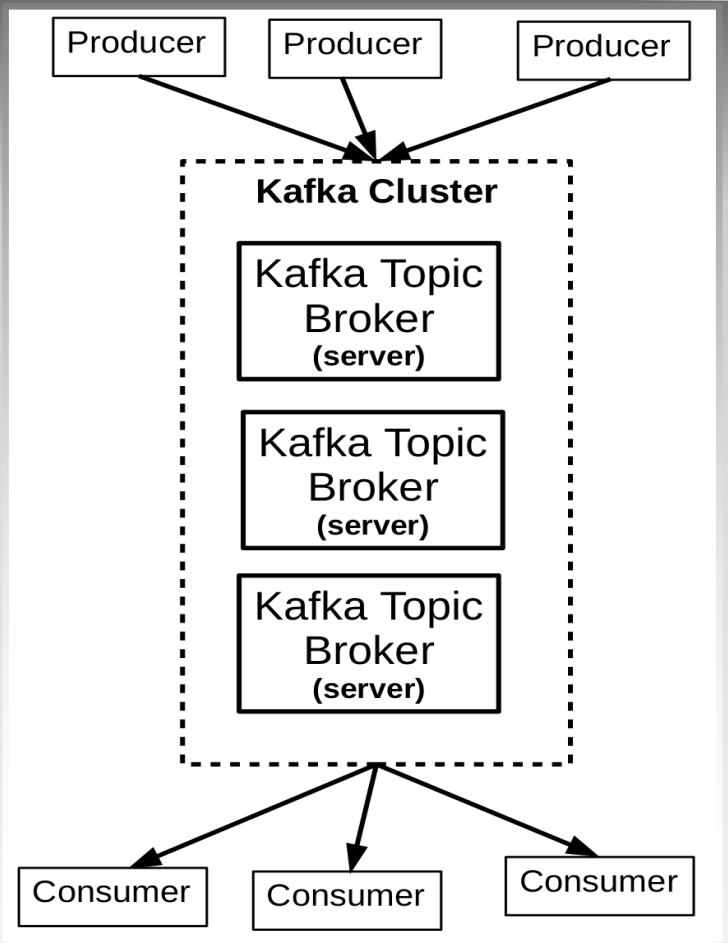
Auto and Manual Log Commits

- **Auto-commit** (default) – uses a commit interval (default is 5 seconds) so that Kafka will commit your current offset every five seconds. Careful, possible messages can be read twice if second consumer takes over before commit period completes.
- **Manual-commit** – offers both a **synchronous** (slower, but sure) and an **asynchronous** (faster, but risky) commit option. When using synchronous commits it will block until the commit is confirmed and will also retry if there are recoverable errors.

  If using an asynchronous commit the request will be sent and continue. There are no retries on error. To preserve ordering, if it fails, data are lost.

LIMULUS COMPUTING™

User View of Kafka Cluster



LIMULUS COMPUTING™

Kafka with Hadoop and Spark

- **Connecting with HDFS:** HDFS V2/3 Sink connector is available that allows export of data from Kafka topics to HDFS V2/3 files in a variety of formats. Integration with Hive make data immediately available for Hive queries.
- **Connecting with Spark:** The Spark Streaming API enables scalable, high-throughput, fault-tolerant stream processing of live data streams. Data can be ingested from Kafka and can be processed using map, reduce, join and window. Finally, processed data can be pushed out to HDFS, local file systems, databases, and live dash-boards.

LIMULUS COMPUTING™

Questions ?

LIMULUS COMPUTING™

Break

LIMULUS COMPUTING™

Segment 4

Basic Examples

```
Message: 1 Monkeys jumping on the bed,
Message: 2 Monkeys jumping on the bed,
Message: 3 Monkeys jumping on the bed,
Message: 4 Monkeys jumping on the bed,
Message: 5 Monkeys jumping on the bed,
Message: 6 Monkeys jumping on the bed,
Message: 7 Monkeys jumping on the bed,
```

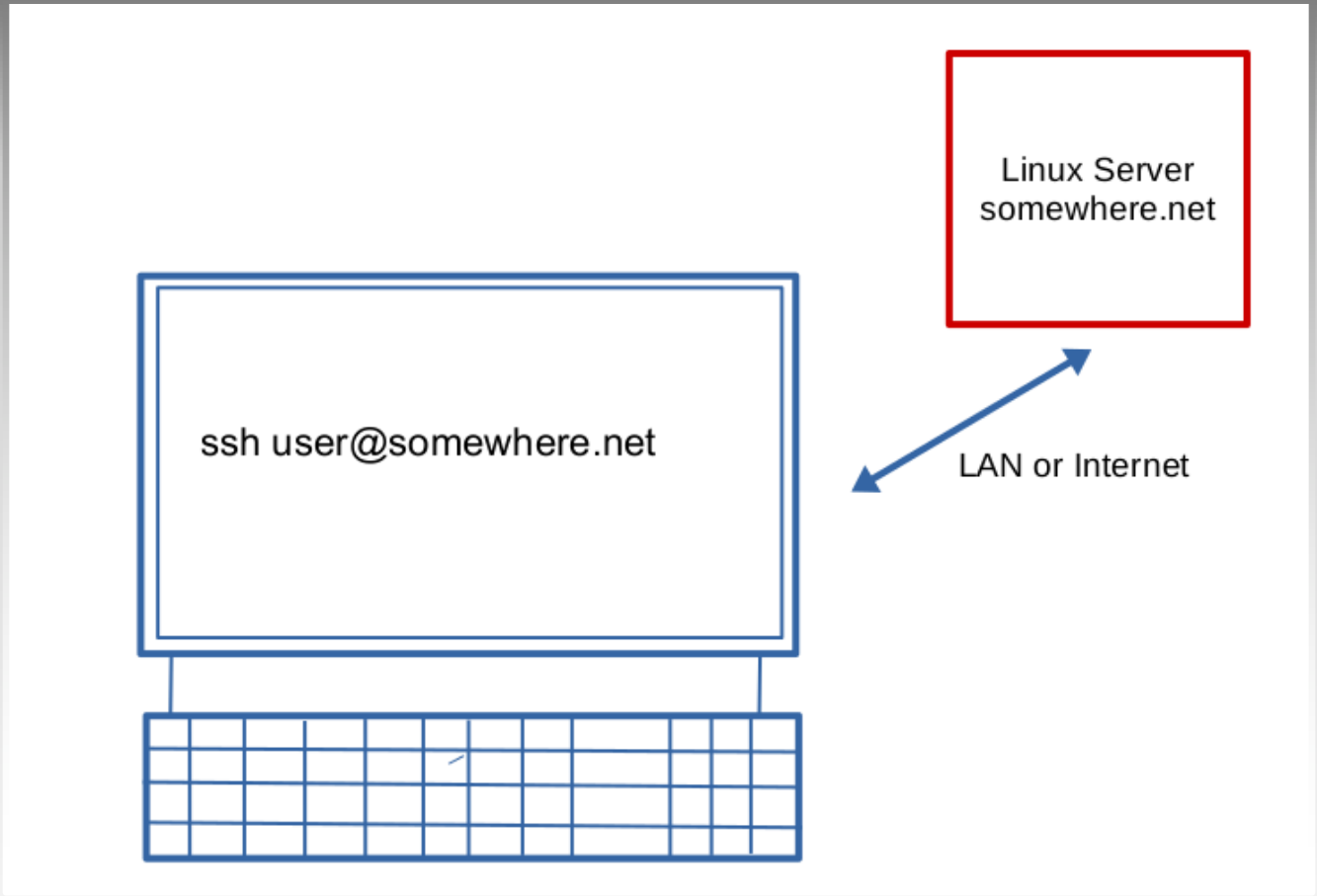
LIMULUS COMPUTING™

LHM: Your Personal Linux Server

- Use a virtual machine (VM) to create a real Linux environment. (See resource page for access.)
- Run on Laptop or Desktop with Virtual Box
- The Linux Hadoop Minimal VM is designed to be a small, single server Hadoop/Spark system (used in other trainings).
- An x86_64 based Notebook or Laptop with at least 4 cores/threads, 4+ GB of Memory, 70 GB of disk space.
- Linux CentOS 7 (Red Hat rebuild) image.
- Walks and talks like a separate computer.

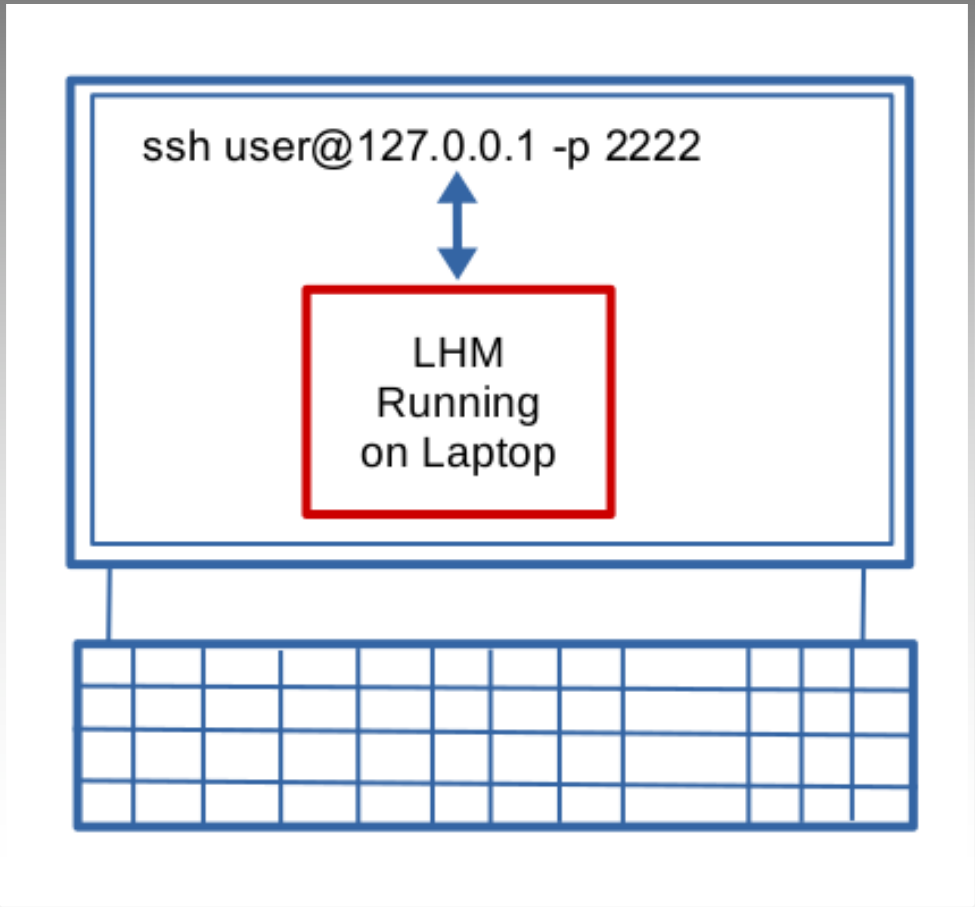
LIMULUS COMPUTING™

ssh Out to LAN/Internet



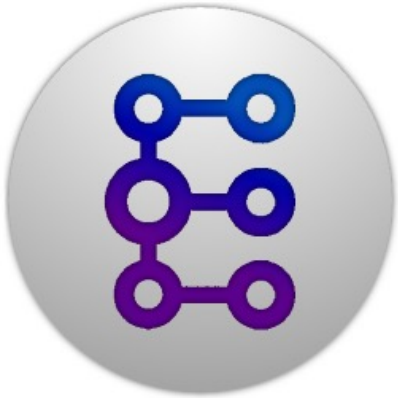
LIMULUS COMPUTING™

Connect to the LHM



LIMULUS COMPUTING™

Segment 5



KafkaEsque

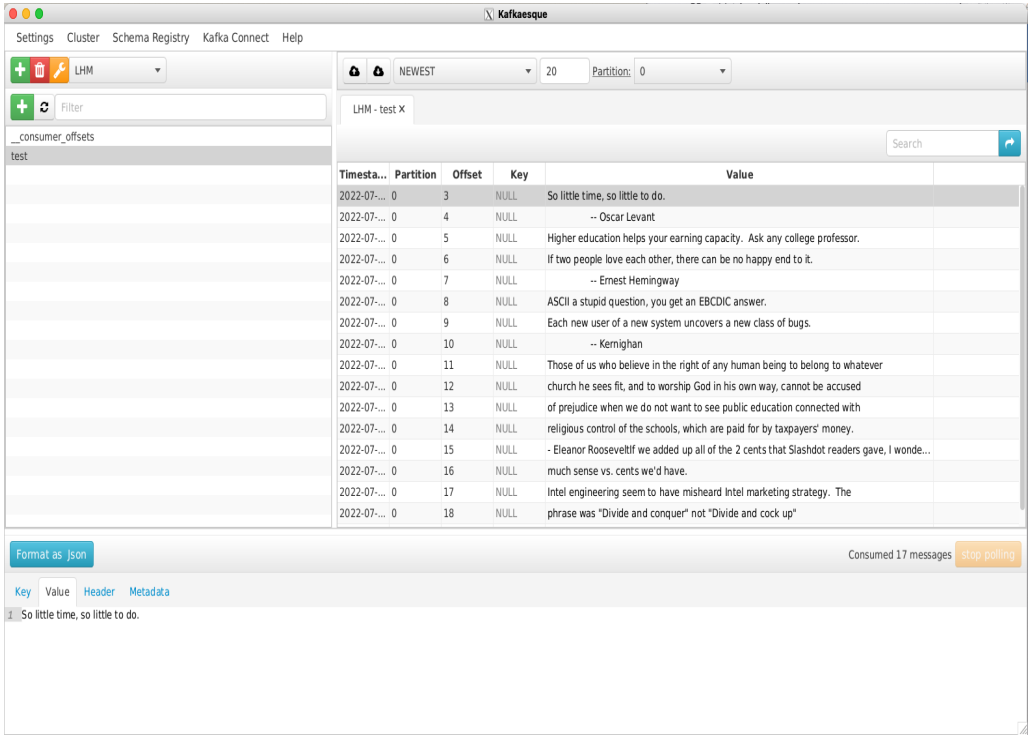
A GUI for Kafka

https://kafka.esque.at

LIMULUS COMPUTING™

KafkaEsque

- Basic GUI for Kafka (open source)
- Available for Windows, Mac, Linux
- Provides replaying, tracing, and configuration of Kafka logs and topics



LIMULUS COMPUTING™

Segment 6

Example One: Streaming Weather Data

```
>>> from noaa_sdk import NOAA
>>> n = NOAA()
>>>
>>> res = n.get_forecasts('180
>>> for i in res:
...     print(i)
...
{'number': 1, 'name': 'Today',
4-22T18:00:00-04:00', 'isDayti
atureTrend': None, 'windSpeed'
```

LIMULUS COMPUTING™

Break

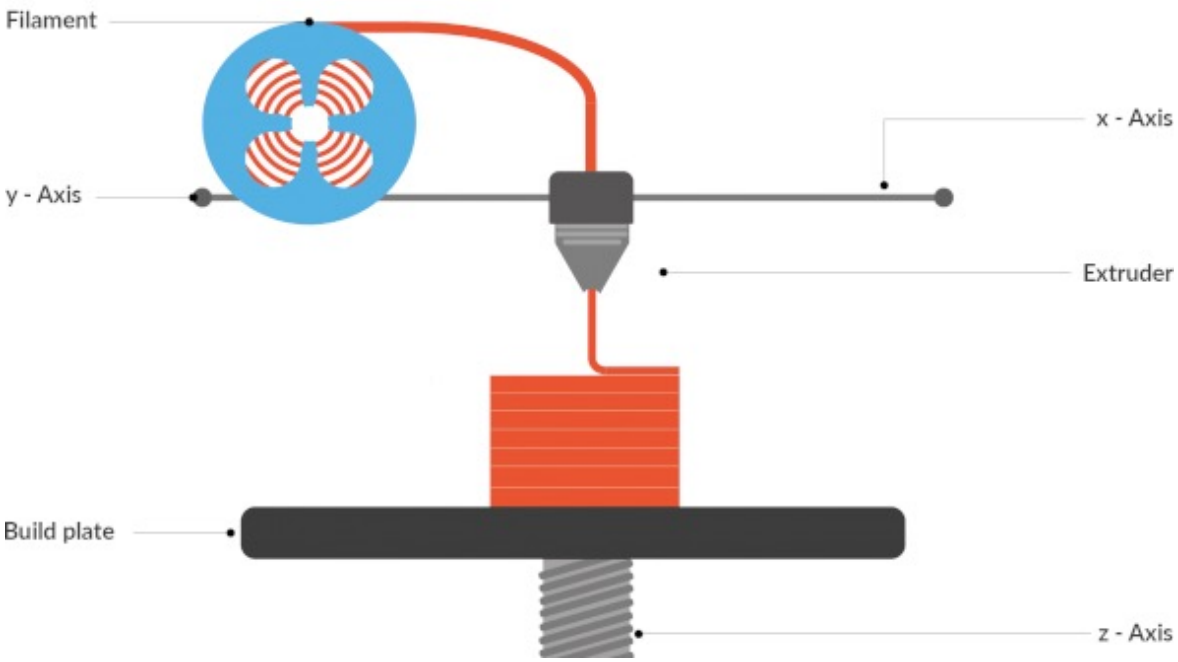
LIMULUS COMPUTING™

Segment 7

Example Two:
Image Streaming with Kafka

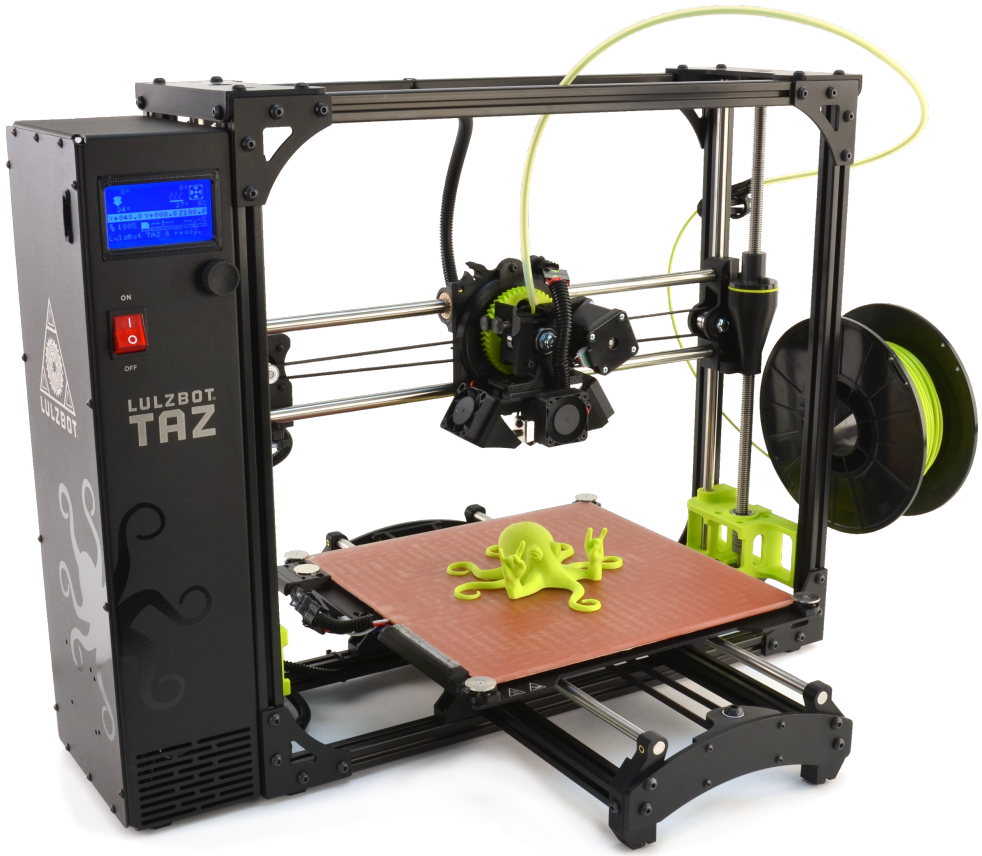
LIMULUS COMPUTING™

Local Example: FDM 3D Printing



LIMULUS COMPUTING™

Printer (Lulzbot Taz 6)

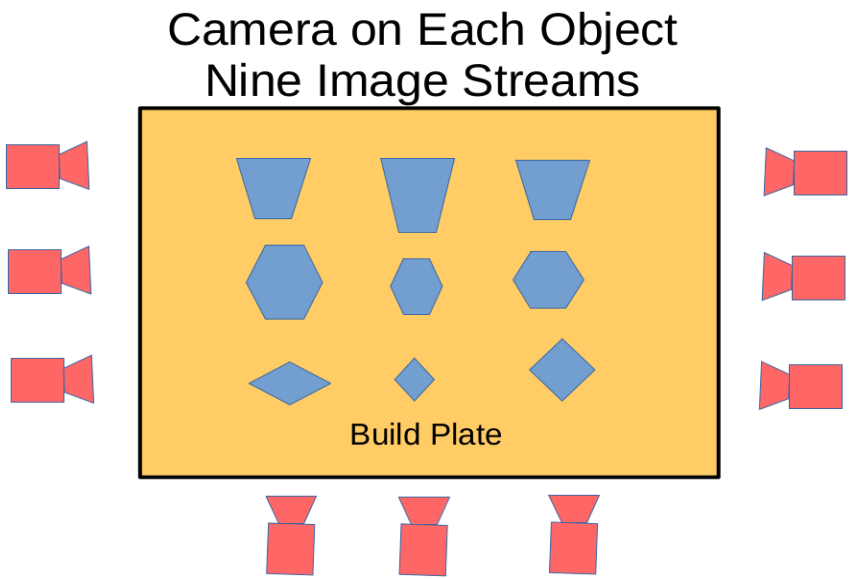


LIMULUS COMPUTING™

Why Kafka?

3D Printers can print multiple (different) objects at the same time.

- Sometimes many same objects (using "mirrored" print heads).
- Sometimes all objects are custom shapes.
- Ideal to watch each object (e.g. nine image streams)

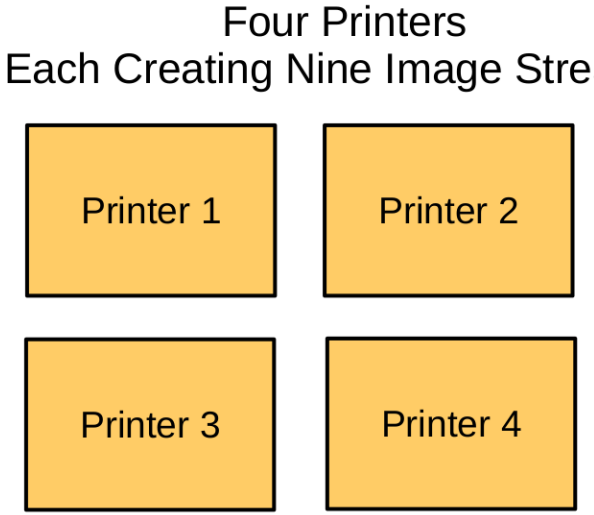


LIMULUS COMPUTING

Why Kafka?

Multiple printers may each have their unique number of streams

e.g. Four printers may be sending nine separate image streams resulting in 36 total streams



LIMULUS COMPUTING™

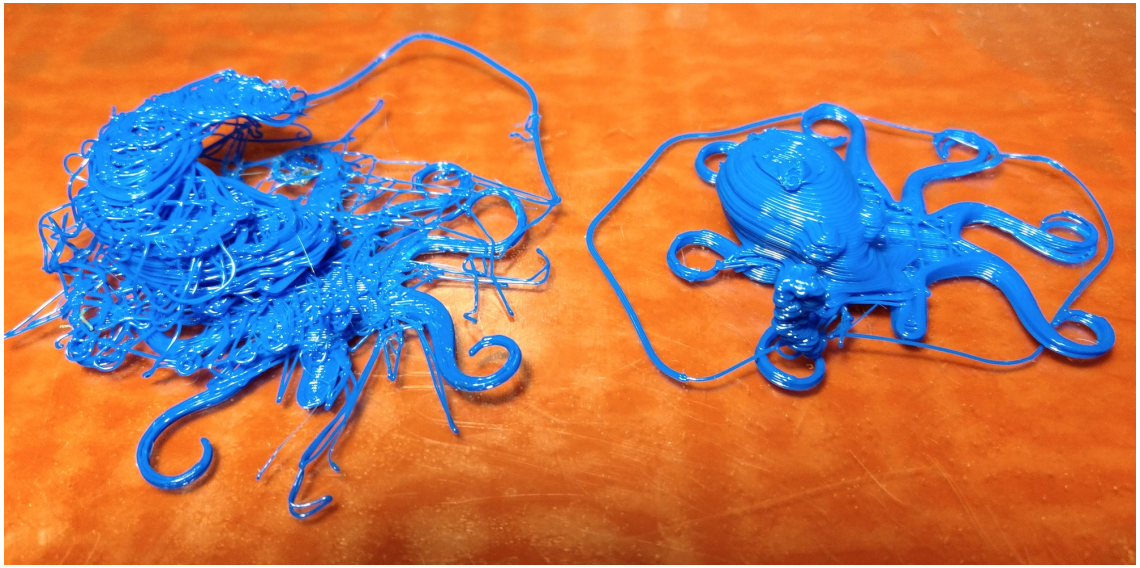
Our Example

- Lulzbot "Rocktopus"
- High speed print
  - 61 visible layers (High detail uses 129 layers)
- Height ~ 30mm
- Width ~ 80 mm
- Takes about 45 min to print

LIMULUS COMPUTING™

What Can go Wrong?

- Bed de-adhesion
  - Results in “Spaghetti”
  - Results in misaligned print



LIMULUS COMPUTING™

What Can Go Wrong ?

- Object on build plate
- Can jam print head
- Move though collision with print head

- Also, bad first layer creates bad print

LIMULUS COMPUTING™

How to “Keep and Eye” on Print

1. Place a camera and take periodic pictures of printing
2. Collect data in real time
3. Use a trained model to detected failure
4. Stop printing process and send alert if potential failure is detected

LIMULUS COMPUTING™

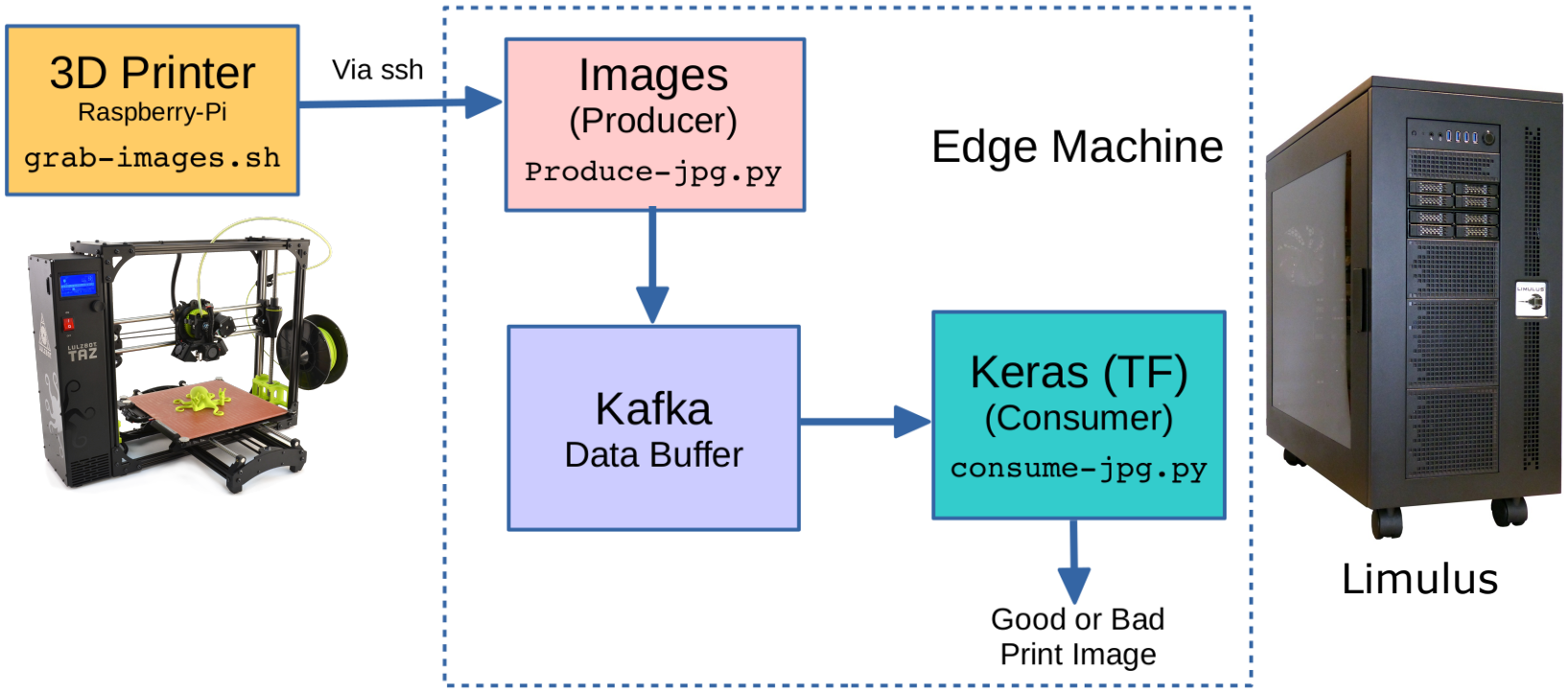
Our Resource List

- **3D Printer** – Lulzbot Taz 6 (Cura for Slicing)
- **Raspberry Pi** – Used as Printer controller (USB to Printer, wireless to local LAN)
- **Octaprint Software** – Raspberry Pi based software to control printer and record pictures pictures (uses **Octolapse** for clean images and video – snaps image at completion of each layer, no print head in picture)
- **Kafka Cluster** – Collects (buffers/brokers) images coming from printers
- **Keras-Tensor Flow** – Keras-TF used to build model and used for "inference"

LIMULUS COMPUTING™

Machine Set Up

Complete Local Data Flow for the Printing Process



LIMULUS COMPUTING™

Software Environment

Three programs running at the same time:

1. `grab-images.sh` – bash program running on 3D-Printer. Watches for new snapshots of layers, when they appear, send to Kafka Server (vis scp)
2. `produce-jpg.py` - running on Limulus reads input directory, any new files committed to Kafka, move file to archive directory
3. `consume-jpg.py` – running on Limulus pulls images out of Kafka does inference with pre-computed model, reports GOOD or FAIL
4. Simulation available on Linux Hadoop Minimal

LIMULUS COMPUTING™

Questions ?

LIMULUS COMPUTING™

Segment 8

Training Wrap-up

LIMULUS COMPUTING™

Kafka Takeaways

- Kafka is a tool to manage information flows.
- Input messages are decoupled from output processing.
- New data flows (Topics) can be easily added.
- Kafka is scalable and offers redundant failover.
- Kafka can work with databases, Hadoop, Spark, Python, and many other tools.
- Kafka message logs are permanent and immutable until data expires.

LIMULUS COMPUTING™

Thank You

Want More?

Coming XXX

Kafka Methods and Administration in Practice

LIMULUS COMPUTING™