Introduction to Apache Kafka

Jun Rao

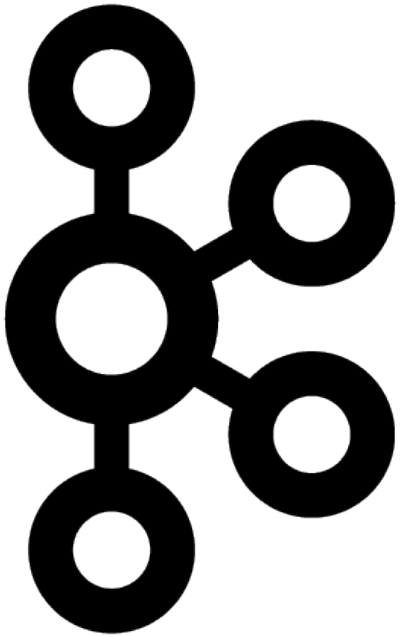
Co-founder of Confluent

Agenda

- Why people use Kafka
- Technical overview of Kafka
- What's coming

What’s Apache Kafka

Distributed, high throughput pub/sub system



Kafka Usage


airbnb


dish
NETWORK


Pinterest


Goldman
Sachs


Microsoft


coursera


CISCO


NETFLIX


UBER

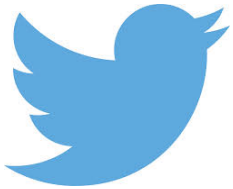

mozilla
FOUNDATION


verizon


box

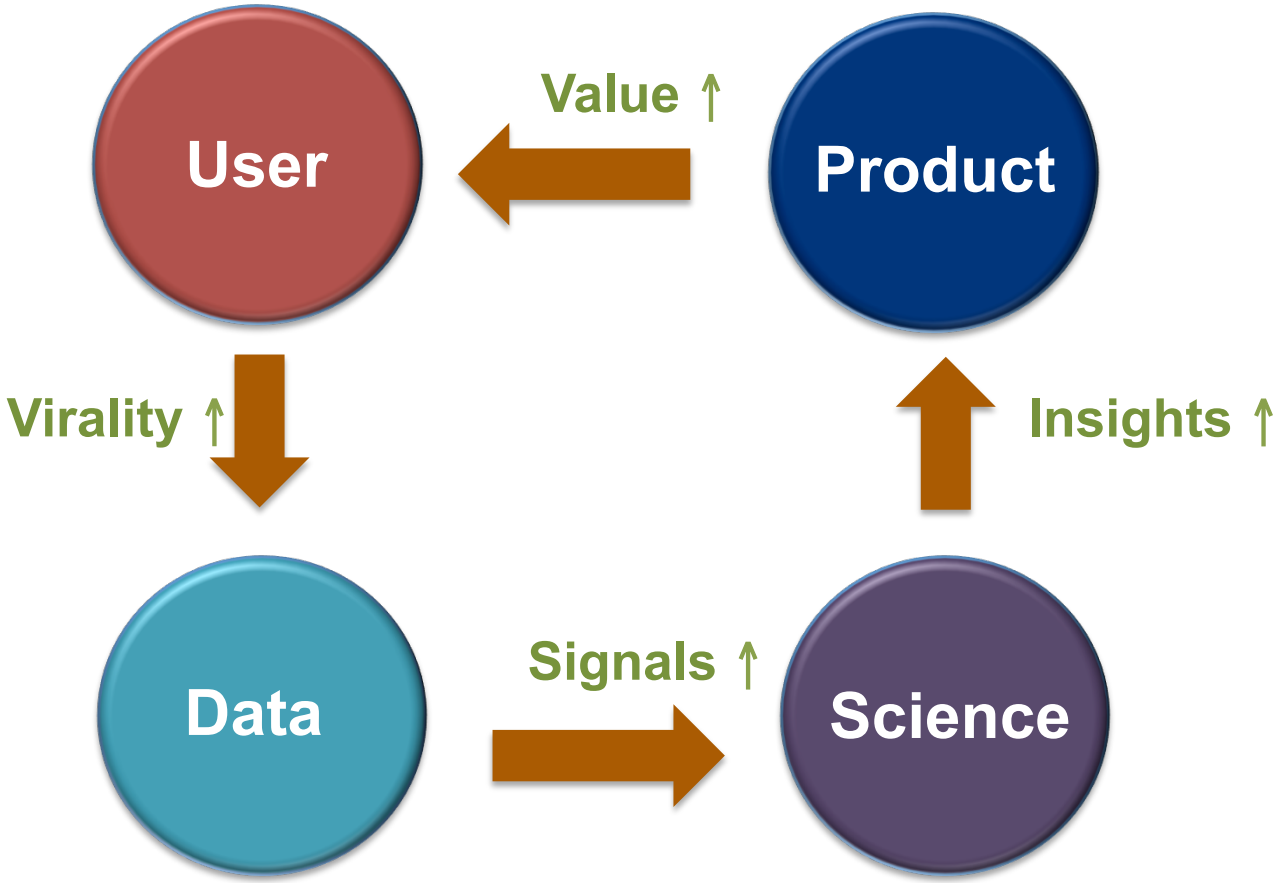

Square


LinkedIn




salesforce

Common Pattern in Data-driven Companies



Getting Data Is Hard

- Data scientists spent 70% time on getting and cleaning data
- Why?

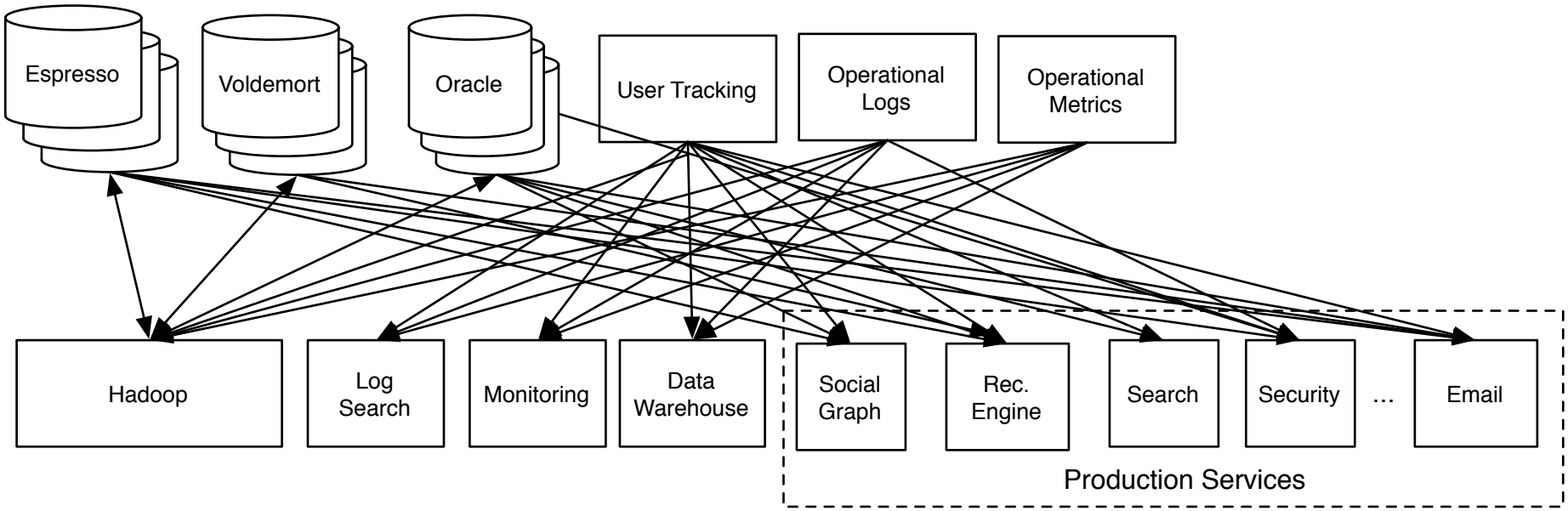
Variety in Data Sources

- Database records
  - Users, products, orders
- Business metrics
  - Clicks, impressions, pageviews
- Operational metrics
  - CPU usage, requests/sec
- Application logs
  - Service calls, errors
- IOT
- …

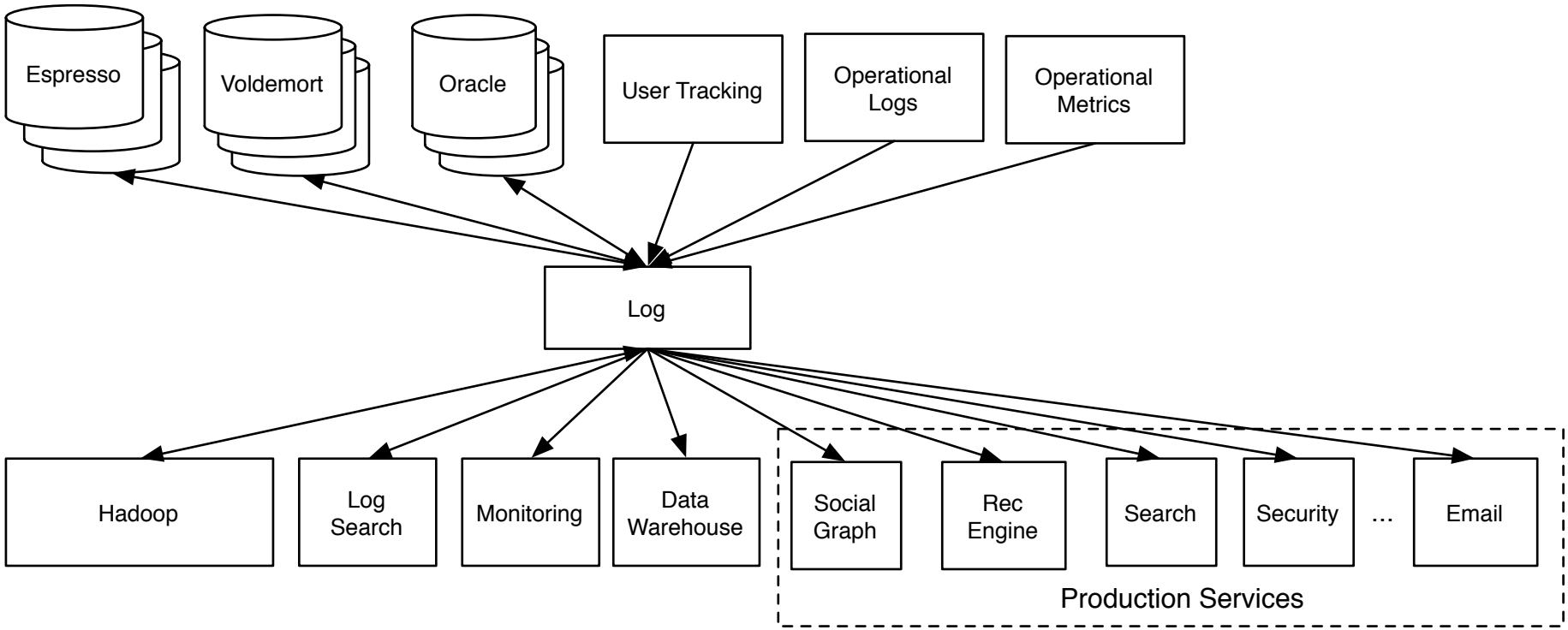
Variety in New Specialized Systems

- Batch oriented
  - Hadoop ecosystem (Pig, Hive, Spark, …)
- Real time
  - Key/value store (Cassandra, MongoDB, HBase, …)
  - Search (ElasticSearch, Solr, …)
  - Stream processing (Storm, Spark streaming, Samza)
  - Graph (GraphLab, FlockDB, …)
  - Time series DB (open TSDB, …)
  - …

Danger of Point-to-point Pipelines



Ideal Architecture: Stream Data Platform



Kafka is the center of stream data platform!

Kafka at LinkedIn

- 800 billion messages, 175 TB data per day
- 13 million messages written/sec
- 65 million messages read/sec
- Tens of thousands of producers
- Thousands of consumers

Agenda

- Why people use Kafka
- Technical overview of Kafka
  - Throughput and scalability
  - Real time and batch consumption
  - Durability and availability
- What's coming

Concepts and API

- Topic defines a message queue
- Producer (Java)

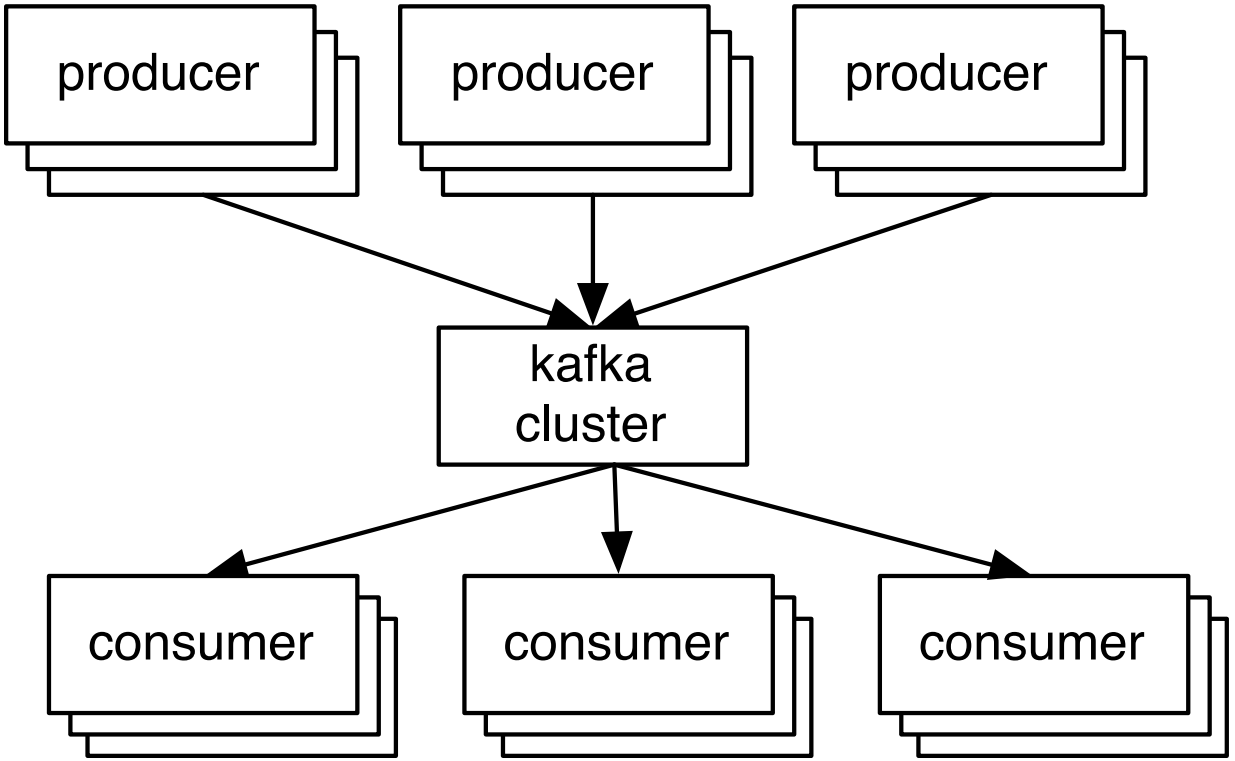
```
byte[] key = "key".getBytes();
byte[] value = "value".getBytes();
record = new ProducerRecord("my-topic", key, value);
producer.send(record);
```

- Consumer (Java)

```
streams[] = Consumer.createMessageStreams("topic1", 1);

for(message: streams[0]) {
   // do something with message
}
```

Distributed Architecture

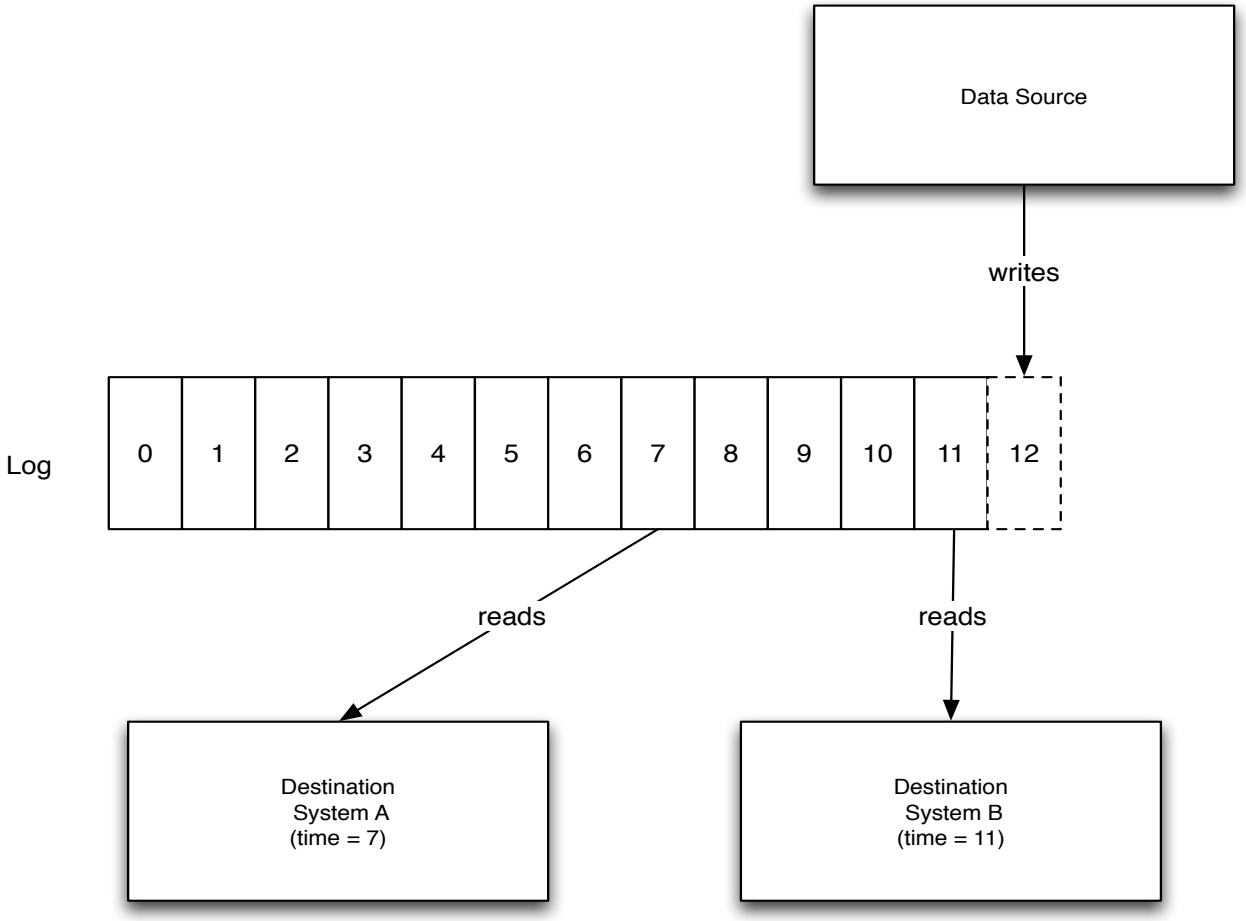


Topics are partitioned for parallelism

Built-in Cluster Management

- Automated failure detection and failover
  - Leverage Apache Zookeeper
- Online data movement

Simple Efficient Storage with a Log



Batching and Compression

- Batching
  - Producer, broker, consumer
- Compression
  - Gzip, snappy, lz4
  - End-to-end

Real Time and Batch Consumption

- Multi-subscription model
- Data persisted on disk, NOT cached in JVM
  - Rely on pagecache
- Zero-copy transfer (broker -> consumer)
- Ordered consumption per topic partition
  - Significantly less bookkeeping

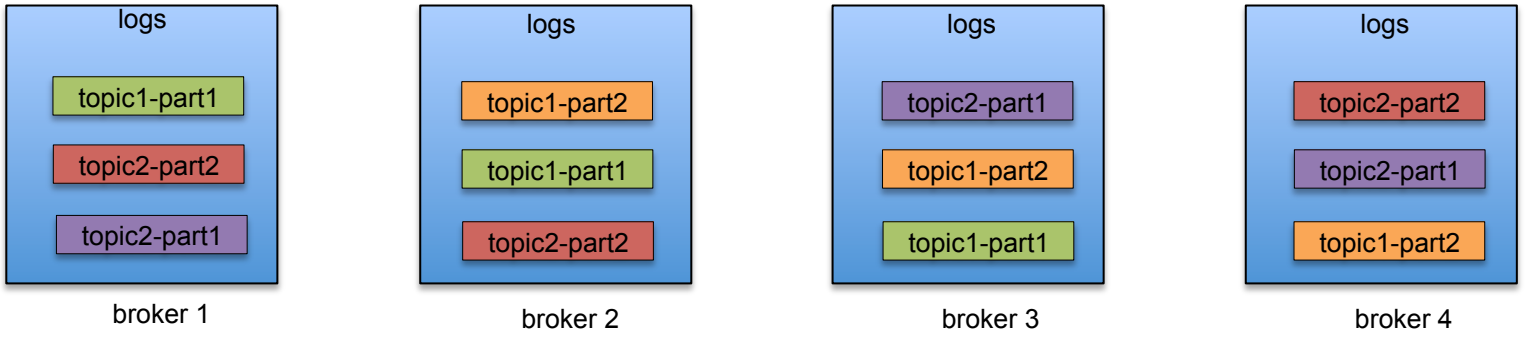
Durability and Availability

Built-in replication

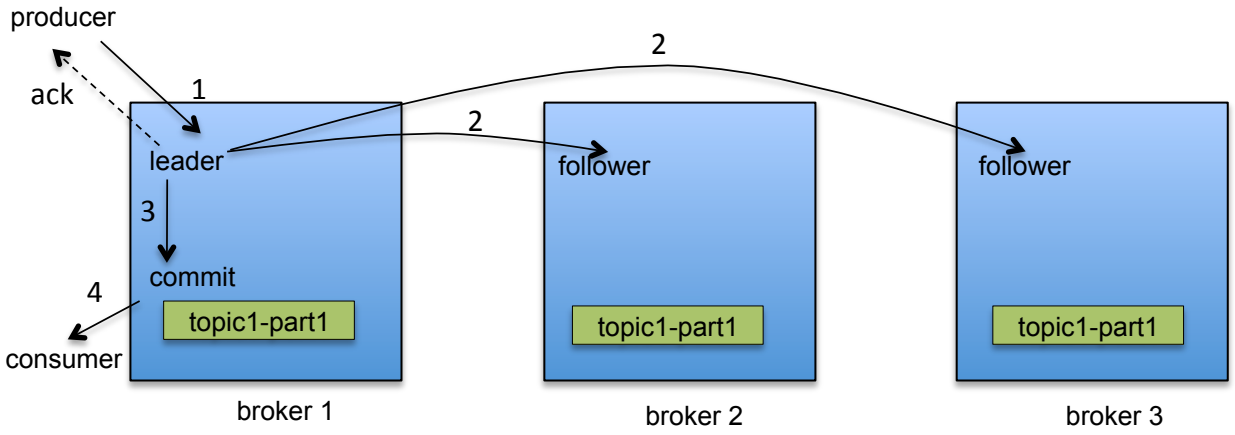
- Configurable replication factor
- Tolerating f – 1 failures with f replicas
- Automated failover

Replicas and Layout

- Topic partition has replicas
- Replicas spread evenly among brokers

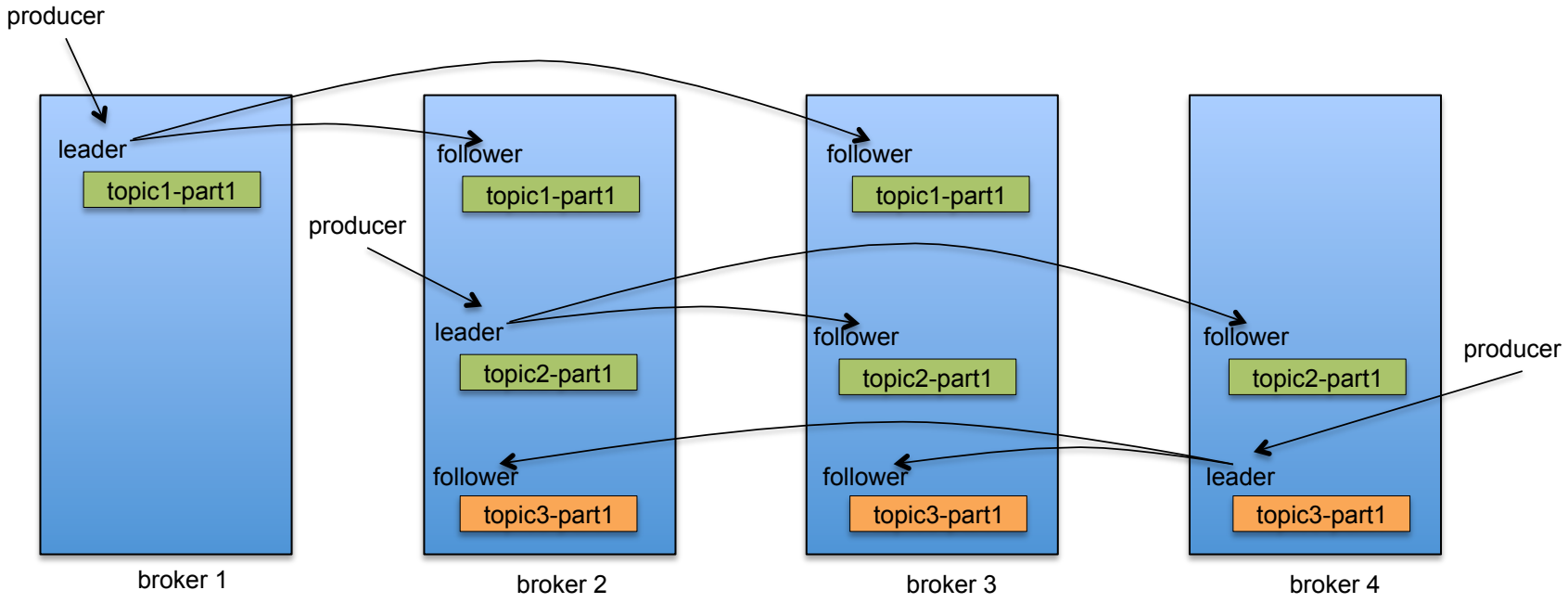


Data Flow in Replication



When producer receives ack	Latency	Durability on failures
no ack	no network delay	some data loss
wait for leader	1 network roundtrip	a few data loss
wait for committed	2 network roundtrips	no data loss

Extend to Multiple Partitions

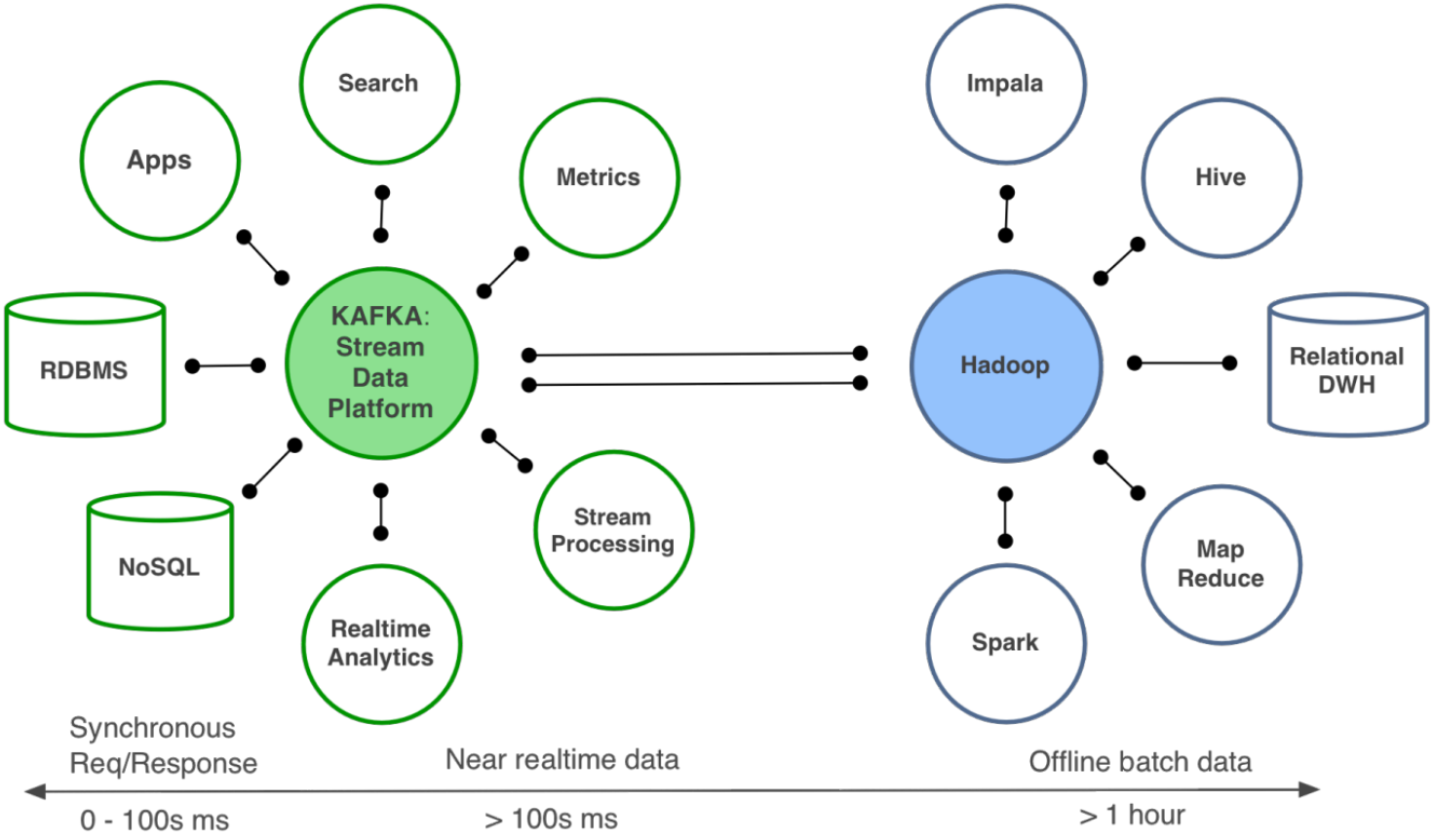


Leaders are evenly spread among brokers

Agenda

- Why people use Kafka
- Technical overview of Kafka
- What's coming

Stream Data Platform



Future Releases of Apache Kafka

- New java consumer client
  - Better performance
  - Easier protocol for non-java client
- Security
  - Authentication: SSL and Kerberos
  - Authorization
- Better cluster management
  - More automated tools
  - Quotas
- Transactional support
  - Exactly once delivery

Confluent

- Mission: Make stream data platform a reality
- Kafka development and support
- Product:
  - Metadata management (released)
  - Rest endpoint (released)
  - Connectors for common systems
  - Monitor data flow end-to-end
  - Stream processing integration

Q&A

- More info on Apache Kafka
  - http://kafka.apache.org/
- Confluent
  - http://confluent.io
  - http://confluent.io/careers
- Kafka meetup tonight at 6:30 (Texas V)