Real-time Streaming Data on AWS

Deep Dive & Best Practices Using Amazon Kinesis, Spark Streaming, AWS Lambda, and Amazon EMR

Radhika Ravirala
Solutions Architect, Amazon Web Services

October 27th, 2016

Pop-up Loft

Agenda

Real-time streaming overview

Use cases and design patterns

Amazon Kinesis deep dive

Streaming data ingestion

Stream processing

Q&A

It’s All About the Pace

Batch Processing	Stream Processing
Hourly server logs	Real-time metrics
Weekly or monthly bills	Real-time spending alerts/caps
Daily web-site clickstream	Real-time clickstream analysis
Daily fraud reports	Real-time detection

Streaming Data Scenarios Across Verticals

Scenarios/ Verticals	Accelerated Ingest-Transform-Load	Continuous Metrics Generation	Responsive Data Analysis
Digital Ad Tech/Marketing	Publisher, bidder data aggregation	Advertising metrics like coverage, yield, and conversion	User engagement with ads, optimized bid/buy engines
IoT	Sensor, device telemetry data ingestion	Operational metrics and dashboards	Device operational intelligence and alerts
Gaming	Online data aggregation, e.g., top 10 players	Massively multiplayer online game (MMOG) live dashboard	Leader board generation, player-skill match
Consumer Online	Clickstream analytics	Metrics like impressions and page views	Recommendation engines, proactive care

Customer Use Cases

SONOS

Sonos runs near real-time streaming analytics on device data logs from their connected hi-fi audio equipment.

Glu Mobile collects billions of gaming events data points from millions of user devices in real-time every single day.

HEARST

Analyzing 30TB+ clickstream data enabling real-time insights for Publishers.

NORDSTROM

Nordstorm recommendation team built online stylist using Amazon Kinesis Streams and AWS Lambda.

Streaming Data Challenges: Variety & Velocity

- Streaming data comes in different types and formats
  - Metering records, logs and sensor data
  - JSON, CSV, TSV
- Can vary in size from a few bytes to kilobytes or megabytes
- High velocity and continuous

```
{
  "payerId": "Joe",
  "productCode": "AmazonS3",
  "clientProductCode": "AmazonS3",
  "usageType": "Bandwidth",
  "operation": "PUT",
  "value": "22490",
  "timestamp": "1216674828"
}
```

Metering Record

```
{
  127.0.0.1 user-
  identifier frank
  [10/Oct/2000:13:5
  5:36 -0700] "GET
  /apache_pb.gif
  HTTP/1.0" 200
  2326
}
```

Common Log Entry

```
{
  <165>1 2003-10-11T22:14:15.003Z
  mymachine.example.com evntslog -
  ID47 [exampleSDID@32473 iut="3"
  eventSource="Application"
  eventID="1011"][examplePriority@
  32473 class="high"]
}
```

Syslog Entry

```
{
   "SeattlePublicWa
   ter/Kinesis/123/
   Realtime" -
   412309129140
}
```

MQTT Record

Two Main Processing Patterns

Stream processing (real time)

- Real-time response to events in data streams

Examples:

- Proactively detect hardware errors in device logs
- Notify when inventory drops below a threshold
- Fraud detection

Micro-batching (near real time)

- Near real-time operations on small batches of events in data streams

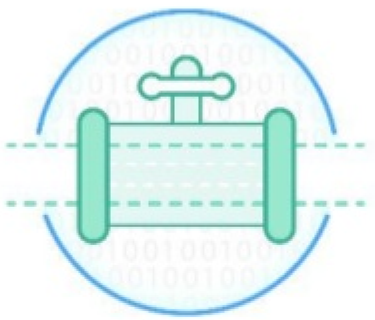
Examples:

- Aggregate and archive events
- Monitor performance SLAs

Amazon Kinesis Deep Dive

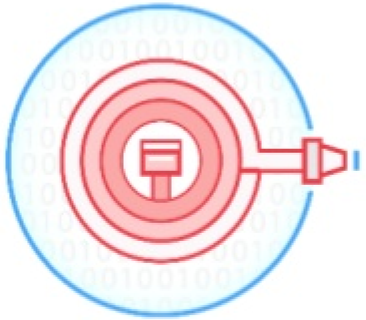
Amazon Kinesis: Streaming Data Made Easy

Services make it easy to capture, deliver and process streams on AWS



Amazon Kinesis Streams

- For Technical Developers
- Build your own custom applications that process or analyze streaming data



Amazon Kinesis Firehose

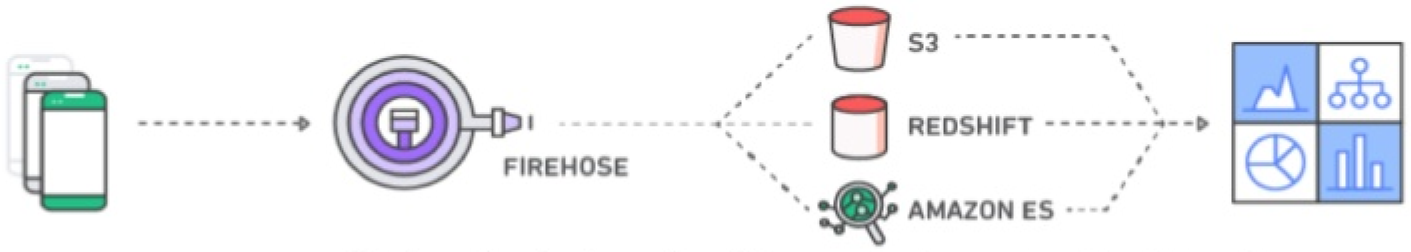
- For all developers, data scientists
- Easily load massive volumes of streaming data into S3, Amazon Redshift and Amazon ElasticSearch

Amazon Kinesis Analytics

- For all developers, data scientists
- Easily analyze data streams using standard SQL queries

Amazon Kinesis Firehose

Load massive volumes of streaming data into Amazon S3, Amazon Redshift and Amazon Elasticsearch



Capture and submit streaming data to Firehose

Firehose loads streaming data continuously into S3, Amazon Redshift and Amazon Elasticsearch

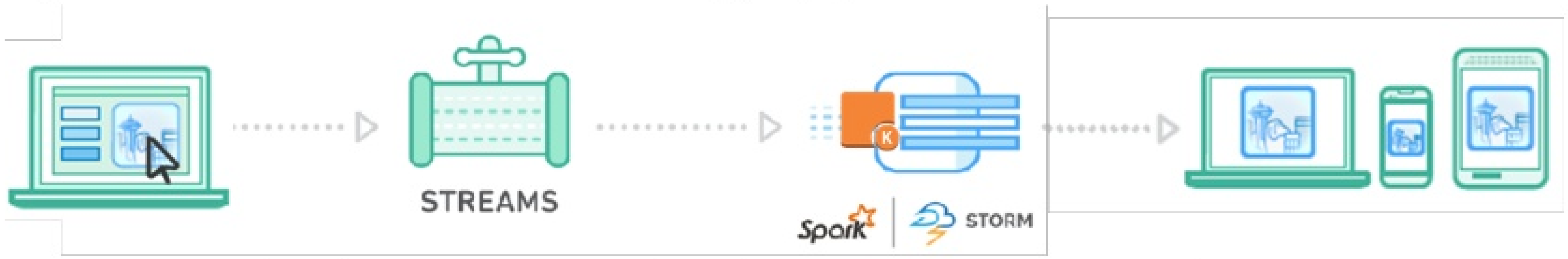
Analyze streaming data using your favorite BI tools

Zero administration: Capture and deliver streaming data into Amazon S3, Amazon Redshift, and other destinations **without writing an application or managing infrastructure**.

Direct-to-data store integration: **Batch, compress**, and **encrypt** streaming data for delivery into data destinations **in as little as 60 secs** using simple configurations.

Seamless elasticity: Seamlessly scales to match data throughput w/o intervention

Amazon Kinesis Streams
Build your own data streaming applications



Send clickstream data to Kinesis Streams

Kinesis Streams stores and exposes clickstream data for processing

Custom application built on Kinesis Client Library makes real-time content recommendations

Readers see personalized content suggestions

Easy administration: Simply **create** a new stream, and **set the desired level** of capacity with shards. Scale to match your data throughput rate and volume.

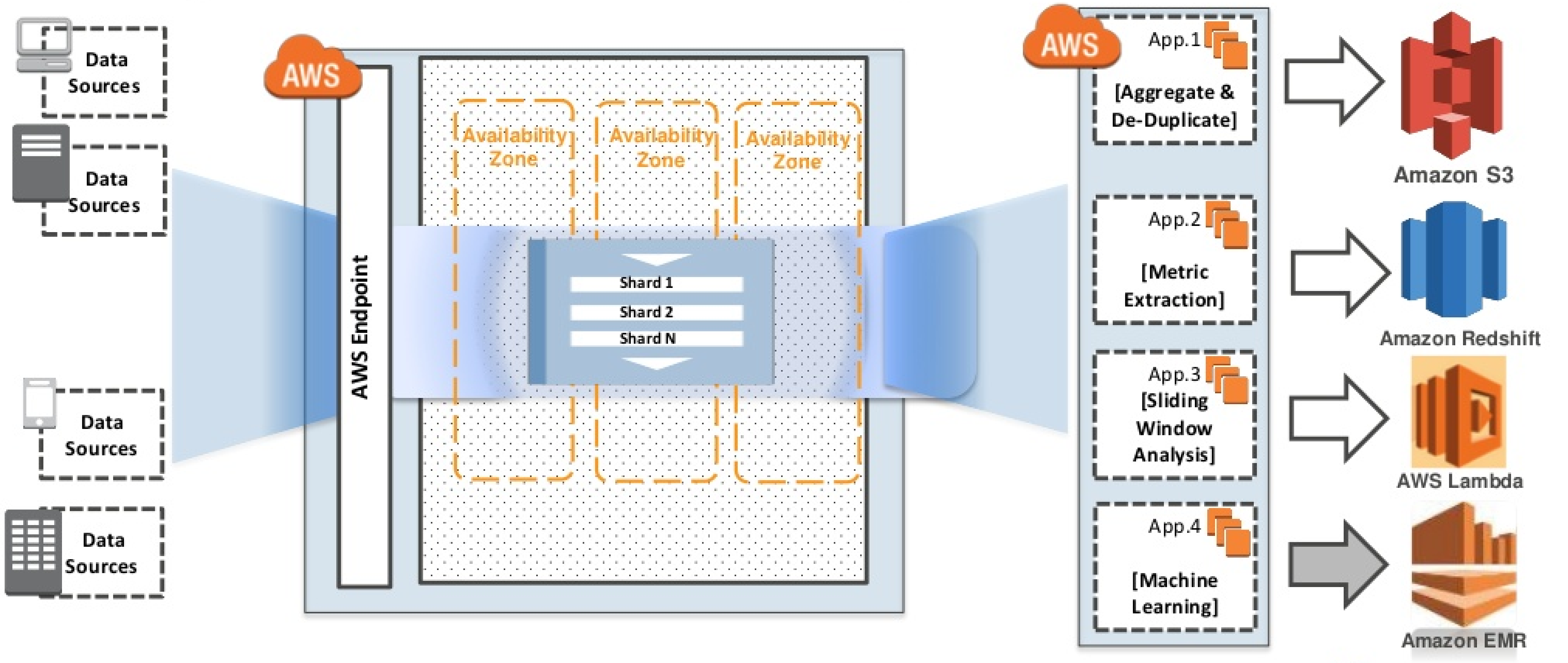
Build real-time applications: Perform **continual processing on streaming data** using Kinesis Client Library (KCL), Apache Spark/Storm, AWS Lambda, and more.

Low cost: **Cost-efficient** for workloads of any scale.

amazon web services | Pop-up Loft

Amazon Kinesis Streams

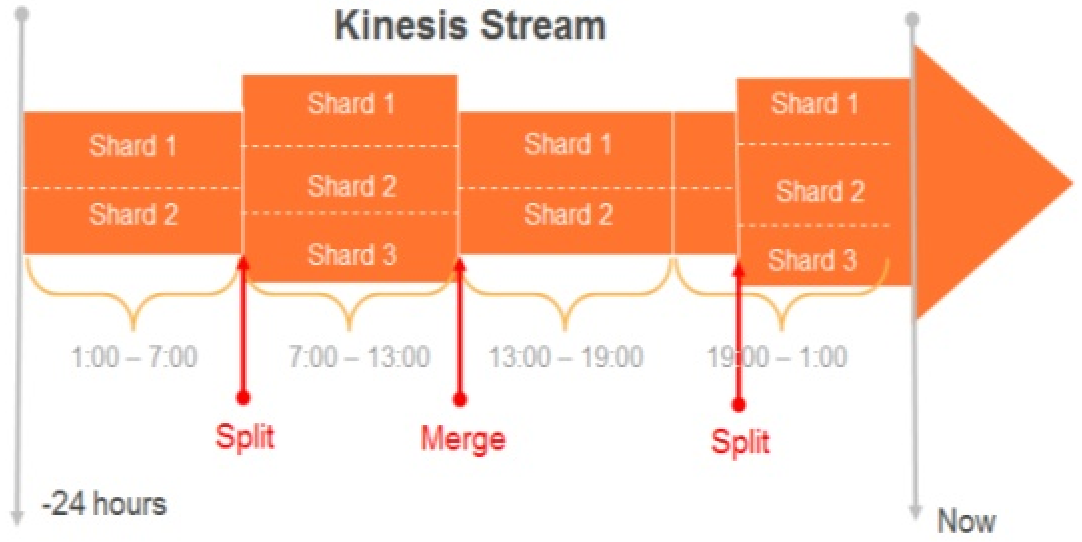
Managed service for real-time streaming



amazon web services | Pop-up Loft

Amazon Kinesis Streams

Managed ability to capture and store data



- Streams are made of **shards**
- Each shard ingests up to 1MB/sec, and 1000 records/sec
- Each shard emits up to 2 MB/sec
- All data is **stored for 24 hours by default**; storage can **be extended for up to 7 days**
- **Scale** Kinesis streams using scaling util
- **Replay** data inside of 24-hour window

amazon web services | Pop-up Loft

Amazon Kinesis Firehose vs. Amazon Kinesis Streams

Amazon Kinesis Streams

Amazon Kinesis Streams is for use cases that require **custom processing**, per incoming record, with sub-1 second processing latency, and a choice of stream processing frameworks.

Amazon Kinesis Firehose

Amazon Kinesis Firehose is for use cases that require zero administration, ability to **use existing analytics tools based on Amazon S3, Amazon Redshift and Amazon Elasticsearch**, and a data latency of 60 seconds or higher.

Streaming Data Ingestion

Putting Data into Amazon Kinesis Streams

Determine your partition key strategy

- Managed buffer or streaming MapReduce job
- Ensure high cardinality for your shards

Provision adequate shards

- For ingress needs
- Egress needs for all consuming applications: if more than two simultaneous applications
- Include headroom for catching up with data in stream

Putting Data into Amazon Kinesis

Amazon Kinesis Agent – (supports pre-processing)

- http://docs.aws.amazon.com/firehose/latest/dev/writing-with-agents.html

Pre-batch before Puts for better efficiency

- Consider Flume, Fluentd as collectors/agents
- See https://github.com/awslabs/aws-fluent-plugin-kinesis

Make a tweak to your existing logging

- log4j appender option
- See https://github.com/awslabs/kinesis-log4j-appender

Amazon Kinesis Producer Library

- Writes to one or more Amazon Kinesis streams with automatic, configurable retry mechanism
- Collects records and uses PutRecords to write multiple records to multiple shards per request
- Aggregates user records to increase payload size and improve throughput
- Integrates seamlessly with KCL to de-aggregate batched records
- Use Amazon Kinesis Producer Library with AWS Lambda (**New!**)
- Submits Amazon CloudWatch metrics on your behalf to provide visibility into producer performance

Record Order and Multiple Shards

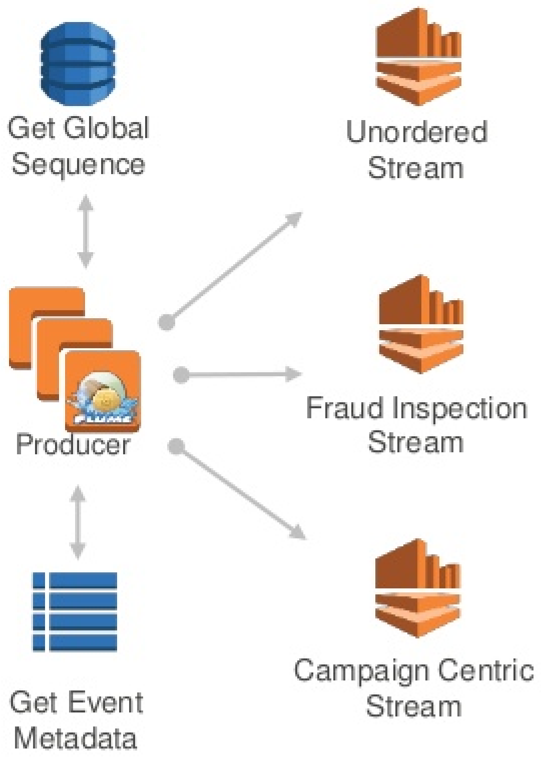
Unordered processing

- Randomize partition key to distribute events over many shards and use multiple workers

Exact order processing

- Control partition key to ensure events are grouped into the same shard and read by the same worker

Need both? Use global sequence number



Sample Code for Scaling Shards

```
java -cp
KinesisScalingUtils.jar-complete.jar
-Dstream-name=MyStream
-Dscaling-action=scaleUp
-Dcount=10
-Dregion=eu-west-1 ScalingClient
```

Options:

- **stream-name - The name of the stream to be scaled**
- **scaling-action - The action to be taken to scale. Must be one of "scaleUp", "scaleDown" or "resize"**
- **count - Number of shards by which to absolutely scale up or down, or resize**

See https://github.com/awslabs/amazon-kinesis-scaling-utils

amazon web services | Pop-up Loft

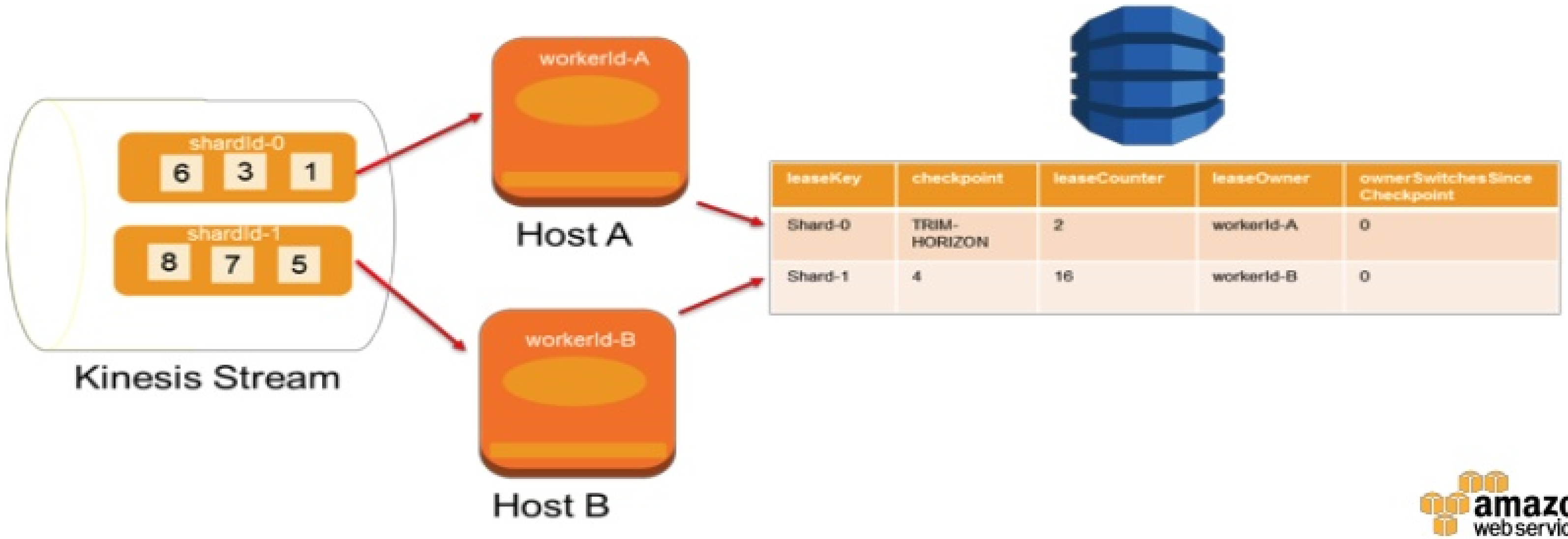
Amazon Kinesis Stream Processing

Amazon Kinesis Client Library

- Build Kinesis Applications with Kinesis Client Library (KCL)
- Open source client library available for Java, Ruby, Python, Node.JS dev
- Deploy on your EC2 instances
- KCL Application includes three components:
  1. **Record Processor Factory** – Creates the record processor
  2. **Record Processor** – Processor unit that processes data from a shard in Amazon Kinesis Streams
  3. **Worker** – Processing unit that maps to each application instance

State Management with Kinesis Client Library

- One record processor maps to one shard and processes data records from that shard
- One worker maps to one or more record processors
- Balances shard-worker associations when worker / instance counts change
- Balances shard-worker associations when shards split or merge



leaseKey	checkpoint	leaseCounter	leaseOwner	ownerSwitchesSince Checkpoint
Shard-0	TRIM-HORIZON	2	workerId-A	0
Shard-1	4	16	workerId-B	0

Other Options

- Third-party connectors(for example, Splunk)
- AWS IoT platform
- **AWS Lambda**
- **Amazon EMR with Apache Spark, Pig or Hive**

Apache Spark and Amazon Kinesis Streams

Apache Spark is an **in-memory analytics** cluster using RDD for fast processing

Spark Streaming can read **directly** from an Amazon Kinesis stream

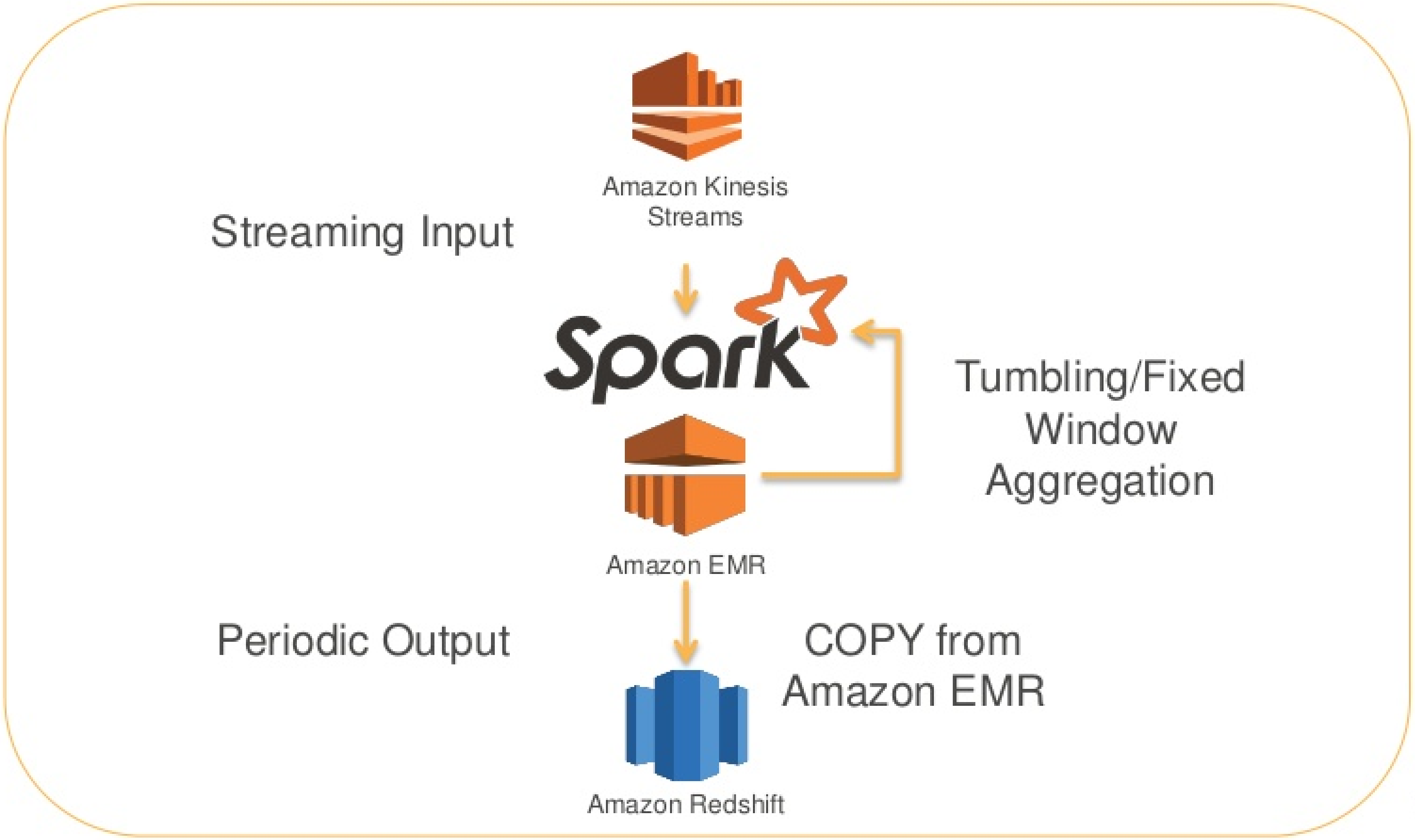
Amazon software license linking – Add ASL dependency to SBT/MAVEN project, `artifactId = spark-streaming-kinesis-asl_2.10`

Example: Counting tweets on a sliding window

```
KinesisUtils.createStream('twitter-stream')
    .filter(_.getText.contains("Open-Source"))
    .countByWindow(Seconds(5))
```

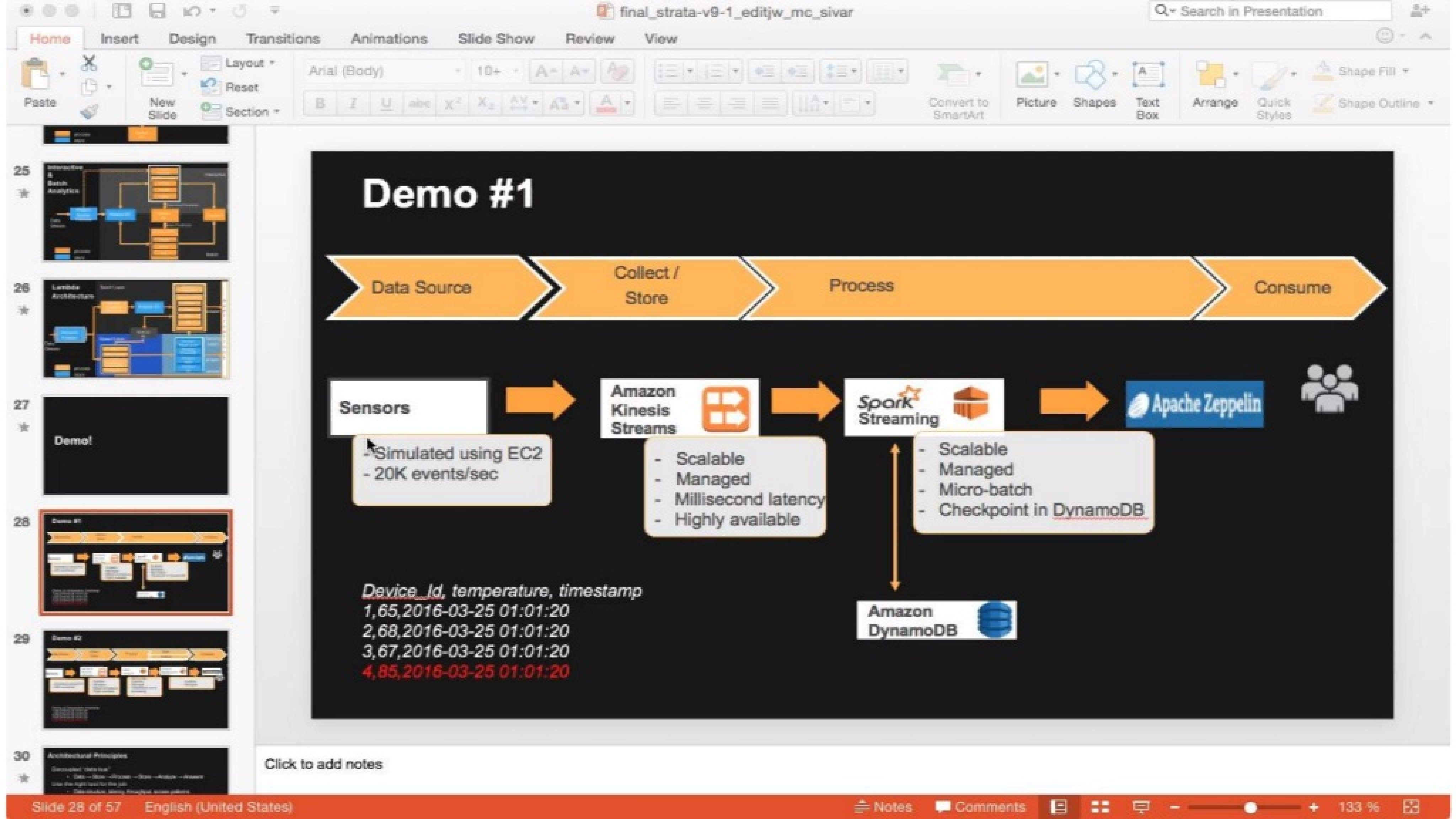
Common Integration Pattern with Amazon EMR

Tumbling Window Reporting



Using Spark Streaming with Amazon Kinesis Streams

1. **Use Spark 1.6+ with EMRFS consistent view option** – if you use Amazon S3 as storage for Spark checkpoint
2. **Amazon DynamoDB table name** – make sure there is only one instance of the application running with Spark Streaming
3. **Enable Spark-based checkpoints**
4. Number of Amazon Kinesis receivers is multiple of executors so they are load-balanced
5. Total processing time is less than the batch interval
6. Number of executors is the same as number of cores per executor
7. Spark Streaming uses default of 1 sec with KCL



Demo #1

Data Source → Collect / Store → Process → Consume

Sensors → Amazon Kinesis Streams → Spark Streaming → Apache Zeppelin

Spark Streaming ↔ Amazon DynamoDB

- Simulated using EC2
- 20K events/sec

- Scalable
- Managed
- Millisecond latency
- Highly available

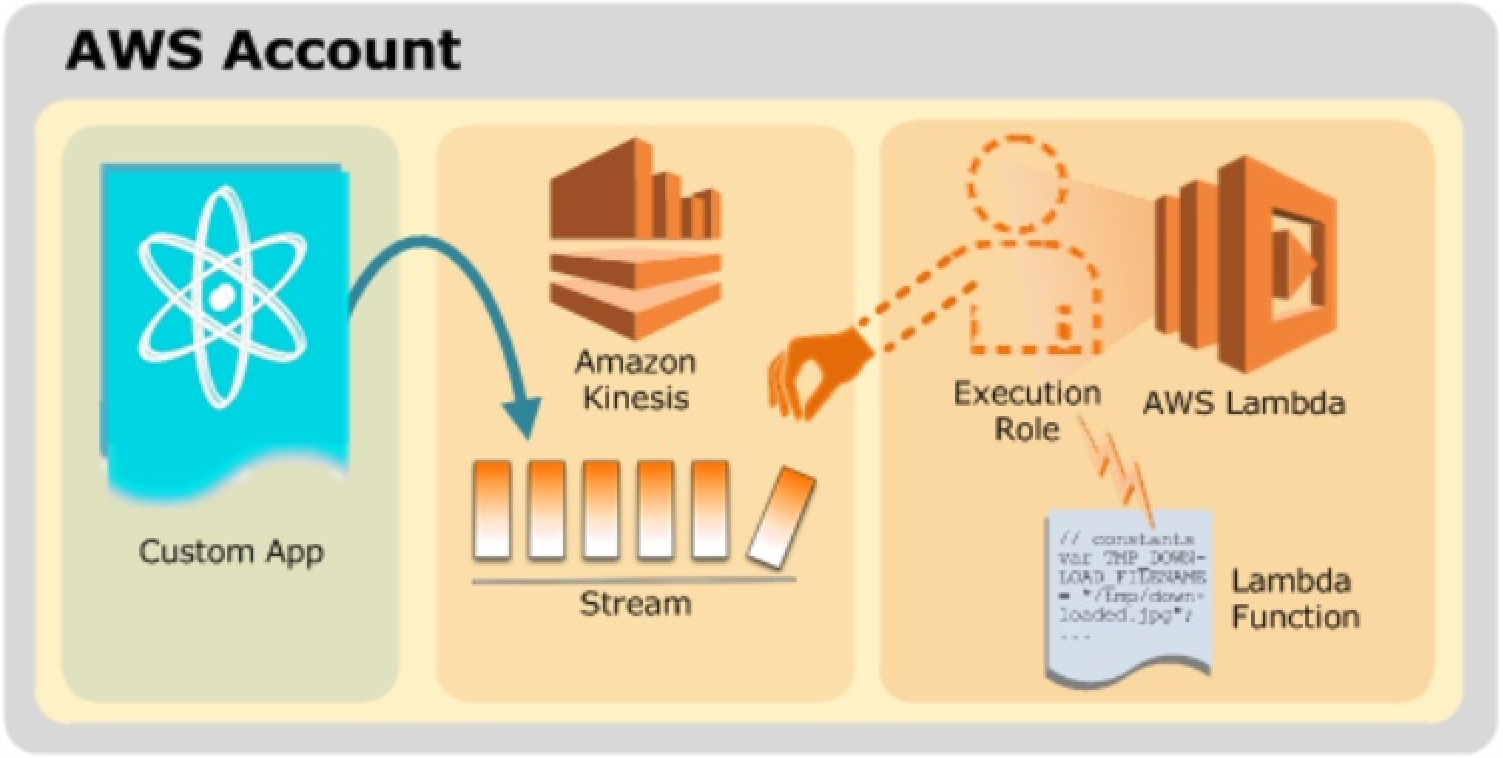
- Scalable
- Managed
- Micro-batch
- Checkpoint in DynamoDB

Device_Id, temperature, timestamp
1,65,2016-03-25 01:01:20
2,68,2016-03-25 01:01:20
3,67,2016-03-25 01:01:20
4,85,2016-03-25 01:01:20

Demo #2 – Kinesismon Go

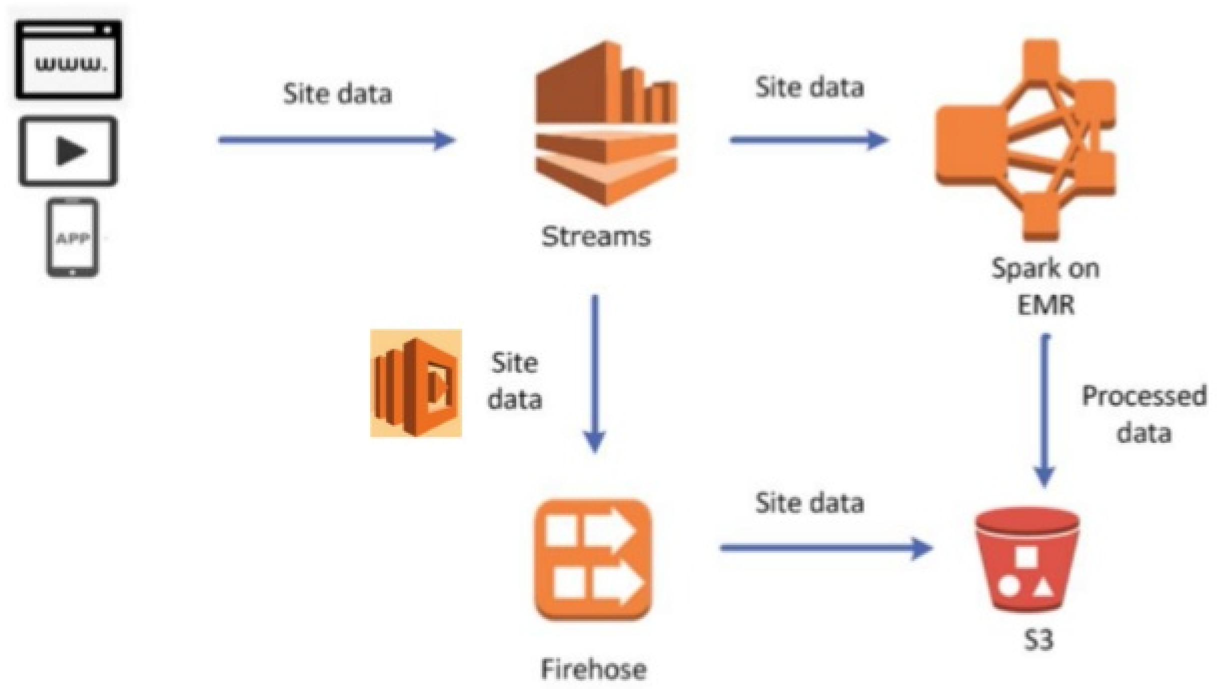
Go to this link on your mobile phone: http://tiny.cc/kademo

Amazon Kinesis Streams with AWS Lambda



```
// constants
var TMP_DOWN-
LOAD_FILENAME
= "/tmp/down-
loaded.jpg";
...
```

Build your Streaming Pipeline with Amazon Kinesis and AWS Lambda



Conclusion

- Amazon Kinesis offers: managed service to build applications, streaming data ingestion, and continuous processing
- Ingest aggregate data using Amazon Producer Library
- Process data using Amazon Connector Library and open source connectors
- Determine your partition key strategy
- Try out Amazon Kinesis at http://aws.amazon.com/kinesis/

Reference 1

- **Technical documentation**
  - Amazon Kinesis Agent
  - Amazon Kinesis Streams and Spark Streaming
  - Amazon Kinesis Producer Library Best Practice
  - Amazon Kinesis Firehose and AWS Lambda
  - Building Near Real-Time Discovery Platform with Amazon Kinesis
- **Public case studies**
  - Glu mobile – Real-Time Analytics
  - Hearst Publishing – Clickstream Analytics
  - How Sonos Leverages Amazon Kinesis
  - Nordstorm Online Stylist

Reference 2

We have many AWS Big Data Blogs which cover more examples. Full list here. Some good ones:

1. Kinesis Streams
   1. Implement Efficient and Reliable Producers with the Amazon Kinesis Producer Library
   2. Presto and Amazon Kinesis
   3. Querying Amazon Kinesis Streams Directly with SQL and Sparking Streaming
   4. Optimize Spark-Streaming to Efficiently Process Amazon Kinesis Streams
2. Kinesis Firehose
   1. Persist Streaming Data to Amazon S3 using Amazon Kinesis Firehose and AWS Lambda
   2. Building a Near Real-Time Discovery Platform with AWS
3. Kinesis Analytics
   1. Writing SQL on Streaming Data With Amazon Kinesis Analytics Part 1 | Part 2
   2. Real-time Clickstream Anomaly Detection with Amazon Kinesis Analytics

Thank You.