AWS Data Lake - Data Ingestion & Central Storage!

Tame Your Big Data with AWS Kinesis Firehose, S3, Glue, Athena, Quicksight

Mohan CV, Solutions Architect, World Wide Public Sector (WWPS)
Virtual, March 25
mohancv@amazon.com

Overview

Objective

- This workshop is meant to give the audience a hands-on experience with mentioned AWS services.
- Data Lake workshop helps customers build a cloud-native and future-proof serverless data lake architecture.
- It allows hands-on time with AWS Big Data and Analytics services including:
  - Amazon Kinesis Services for streaming data ingestion and analytics
  - AWS Glue for ETL and Data Catalogue Management
  - Amazon Athena to query data lake

Target Audience

- Data analysts, IT Managers, Solutions Architects, Data Scientists.
- AWS hands-on knowlegde: AWS Console, IAM, S3, CloudWatch.
- Concept knowledge: Hadoop, NoSQL, ETL, Kafka Streams. HIVE, Presto engines. JSON, Parquet data formats.

Pre-requisites

Lab Preparation:

1. **Access and setup your AWS account**

2. **Setup Kinesis Data Generator Tool:** Follow the KDG Guide on https://awslabs.github.io/amazon-kinesis-data-generator/web/help.html to setup and configure the KDG on your AWS account. You simply need to use the CloudFormation template to create KDG within the us-west-2 (Oregon) region

3. **CloudFormation Stack Creation** for Service Role Permissions for use within the labs

Agenda

TIME	Description	Presenter
12:00 – 12:20p	Welcome and Introduction!	Andrew Marsh
12:20 – 12:40p	**Lecture Module 1:** Data Ingestion & Central Storage (with AWS Kinesis Data Generator, Kinesis Firehose, S3)	Mohan CV
12:40 – 01:00p	**Lab Preparation**	All Presenters
1:00 – 1:30p	**Proctored Lab 1.1:** Create simulated data in real-time data with KDG	All Presenters
1:30 – 2:10p	**Proctored Lab 1.2:** Create Kinesis Firehose Delivery Stream to load streaming data on S3	All Presenters
2:10 – 2:20p	BREAK	
2:20 – 2:50p	**Lecture Module 2:** Data Cataloging & ETL (with AWS Glue)	Adebimpe (Bims) Daniells
2:50 – 3:35p	**Proctored Lab 2.1:** Cataloging a Data Source with Glue	All Presenters
3:35 – 4:30p	**Proctored Lab 2.2:** Transforming a Data Source with Glue	All Presenters
4:30 – 5:00p	Kahoot & Close-Out	Andrew Marsh

A data lake is a centralized repository that allows you to store all your structured and unstructured data at any scale

Why Data Lakes?

The State of Data Today

There is **more data** than people think

Data

grows
>10x
every 5 years

Data platforms need to

live for
15
years

scale
1,000x

* IDC, Data Age 20215: The Evolution of Data to Life-Critical Don't Focus on Big Data Focus on the Data That's Big Apr 2017

Data Scientists

Business Users

Analysts

Applications

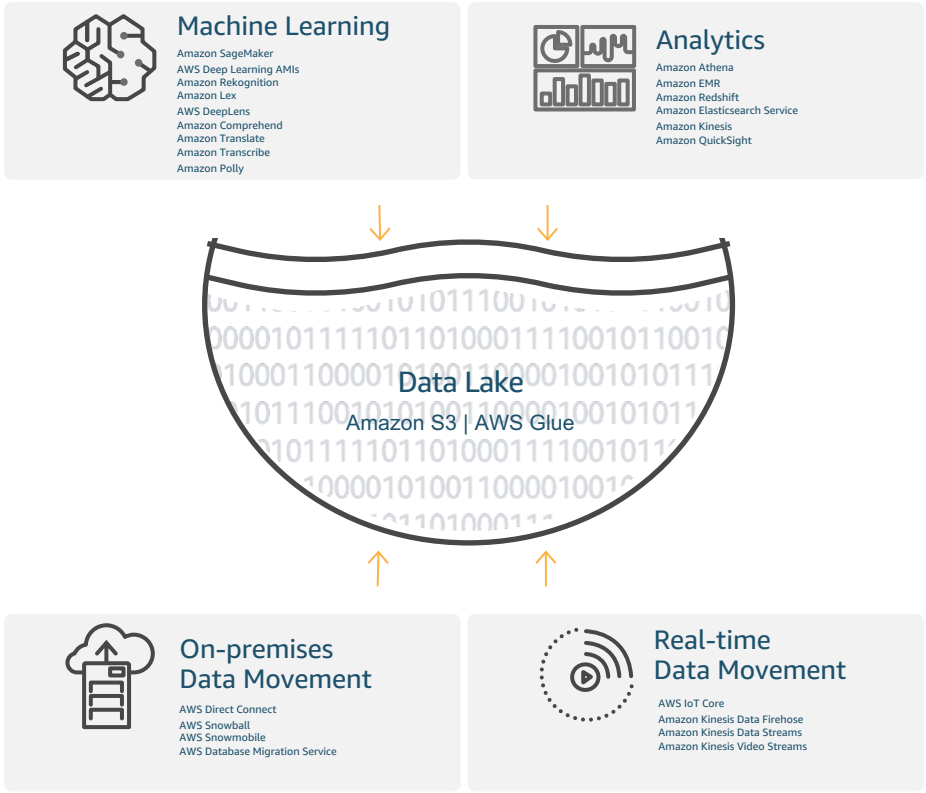
Secure **Real time**

Flexible **Scalable**

There are **more people** accessing data

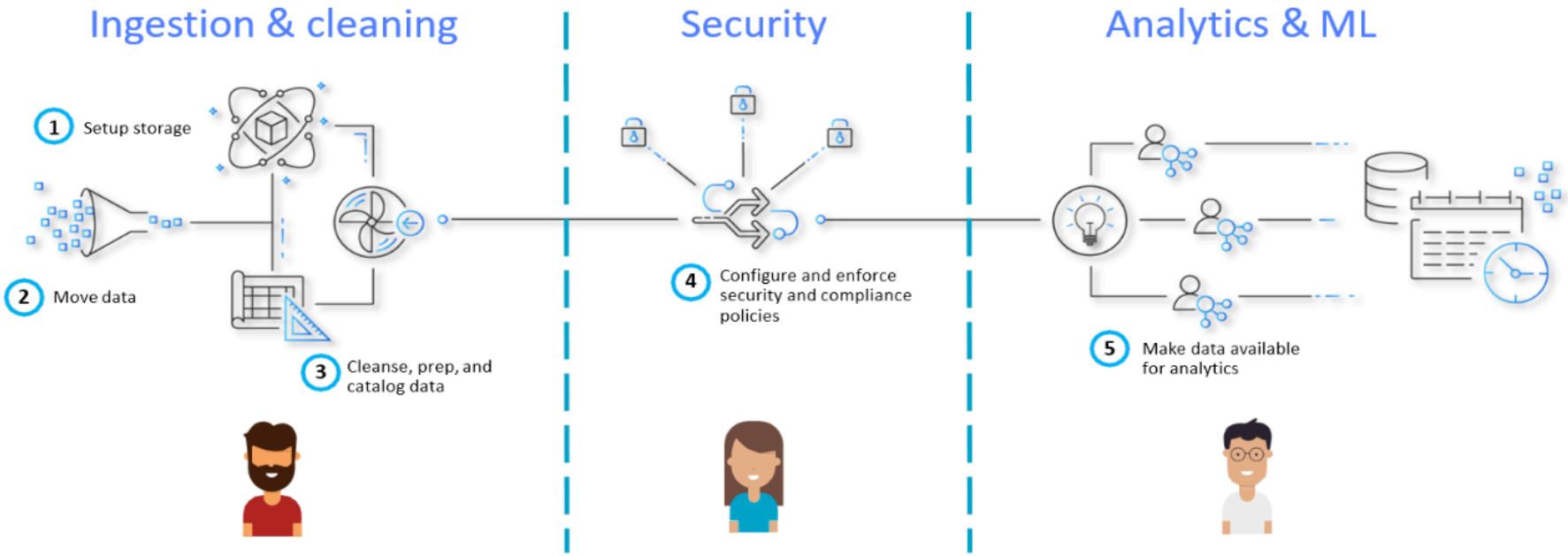
And **more requirements** for making data available

Essential Elements of a Data Lake and Analytics Solution



Any **analytic workload**, any scale, at the lowest possible cost

Typical steps of building a data lake



One platform for data and analytics

Broadest and deepest portfolio of purpose-built services

Business Intelligence & Machine Learning


Amazon QuickSight


Machine Learning

Relational Databases


Amazon Aurora


Amazon RDS

Non-Relational Databases


Amazon DynamoDB
(Key value/Document)


Amazon ElastiCache
(Redis, Memcached)

Analytics

DW | Big Data Processing | Ad hoc


Amazon Redshift


Amazon EMR


Amazon Athena

Real-time


Amazon ES


Kinesis Data Analytics

Data Lake


Amazon S3/Glacier
(Storage)

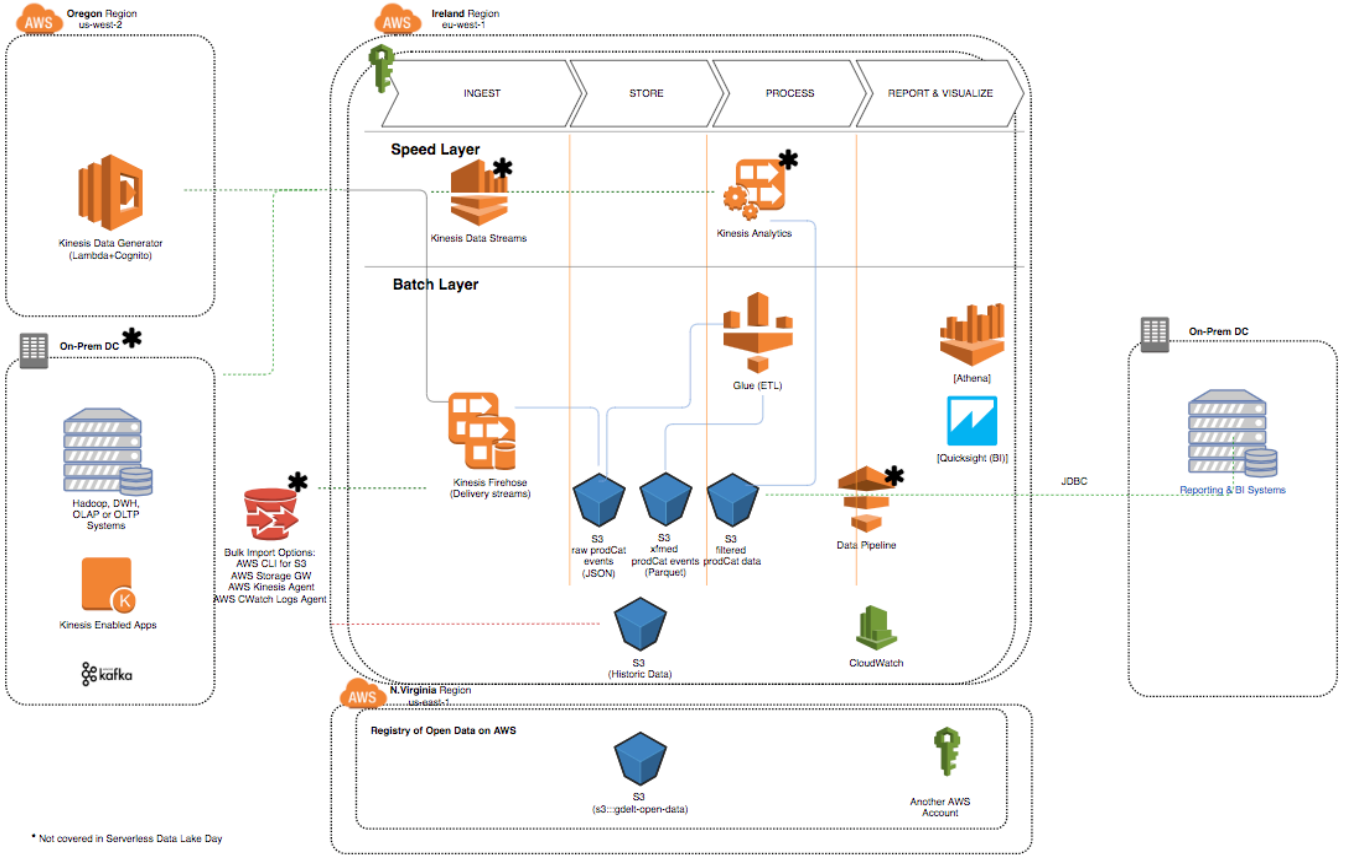

AWS Glue
(ETL & Data Catalog)


Amazon Macie
(Data Protection)

Data Movement

Database Migration Service | Snowball | Snowmobile | Kinesis Data Firehose | Kinesis Data Streams

Workshop Solution Architecture



Architecture Highlights

- Serverless
- Hybrid

PB3: Note: Draw.io source for the diagram will be shared at the Immersion Day

Before we start the labs…

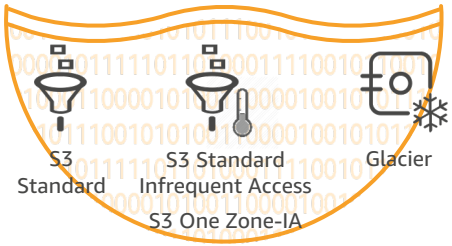
In order to perform the immersion day labs, please follow the preparation steps outlined within the lab preparation document. The document guides you with the following:

1. Access your AWS account
2. Kinesis Data Generator Tool: Setup and configure the KDG on your AWS account
3. CloudFormation Stack Creation for Service Role Permissions

LAB 1.1

Service Used in this Lab

S3: Store data at any scale with unmatched Durability & Availability



- Built to store any amount of data
- Runs on the world's largest global cloud infrastructure
- Designed to deliver 99.999999999% durability
- Geographic redundancy & automatic replication
- Seamlessly replicates data between any region
- Tiered storage to optimize price/performance: Store data at $0.023/GB/month at S3 ($0.004/GB/month at Glacier)

Process Data in Place…

Amazon Athena

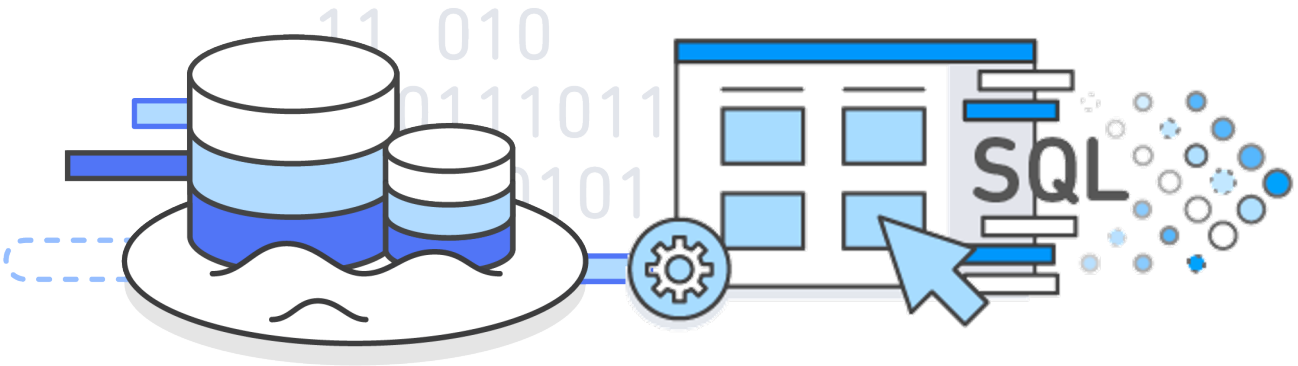
Amazon Redshift Spectrum

Amazon SageMaker

AWS Glue

Amazon S3

Amazon S3 Select and Amazon Glacier Select



Select subset of data from an object based on a SQL expression

Motivation Behind Amazon S3 Select

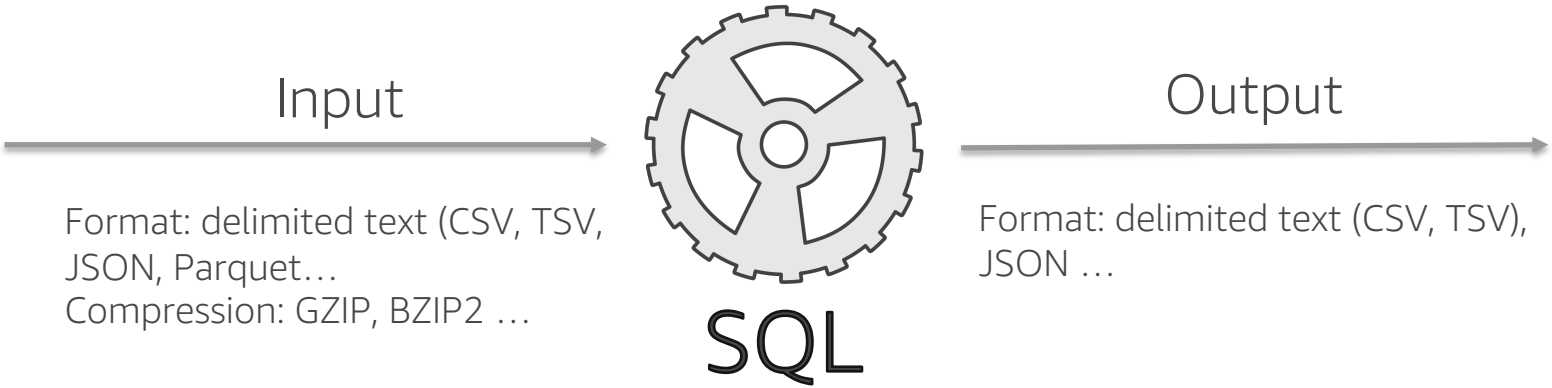
GET all the data from S3 objects, and my application will filter the data that I need

Redshift Spectrum Example:

- Customer: Run 50,000 queries
- Amount of data fetched from S3: 6 PBs
- Amount of data used in Amazon Redshift: 650 TB

Data needed from S3: 10%

Amazon S3 Select



Clauses	Data types	Operators	Functions
Select	String	Conditional	String
From	Integer, Float, Decimal	Math	Cast
Where	Timestamp	Logical	Math
	Boolean	String (Like, ||)	Aggregate

Choosing the Right Data Formats

There is no such thing as the “best” data format

All involve tradeoffs, depending on workload & tools

- CSV, TSV, JSON are easy, but not efficient
  - Compress & store/archive as raw input
- Columnar compressed are generally preferred
  - Parquet or ORC
  - Smaller storage footprint = lower cost
  - More efficient scan & query
- Row oriented (AVRO) good for full data scans

Key considerations are cost, performance & support

Choosing the Right Data Formats (con't.)

Pay by the amount of data scanned per query

Use Compressed Columnar Formats

- Parquet
- ORC

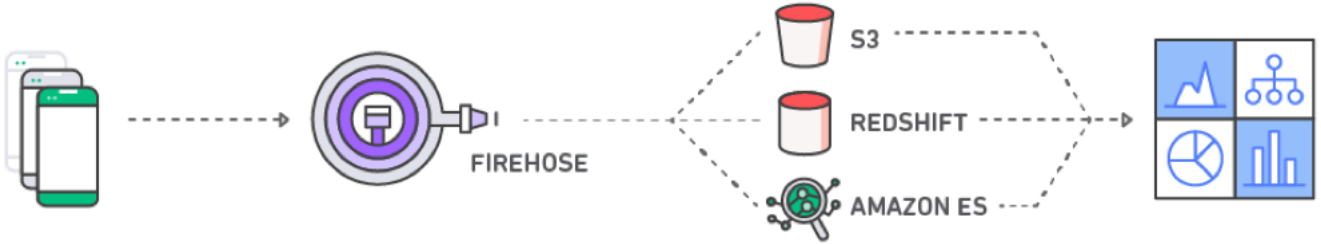
Easy to integrate with wide variety of tools

```
SELECT elb_name,
       uptime,
       downtime,
       cast(downtime as DOUBLE)/cast(uptime as DOUBLE) uptime_downtime_ratio
FROM
    (SELECT elb_name,
        sum(case elb_response_code
        WHEN '200' THEN
        1
        ELSE 0 end) AS uptime, sum(case elb_response_code
        WHEN '404' THEN
        1
        ELSE 0 end) AS downtime
    FROM elb_logs_raw_native
    GROUP BY  elb_name)
```

Dataset	Size on Amazon S3	Query	Data Scanned	Cost
Logs stored as Text files	1 TB	237 seconds	**1.15TB**	$5.75
Logs stored in Apache Parquet format*	130 GB	5.13 seconds	**2.69 GB**	$0.013
Savings	**87% less with Parquet**	**34x faster**	**99% less data scanned**	**99.7% cheaper**

Service Used in this Lab

Amazon Kinesis Firehose



Capture and submit streaming data to Firehose

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

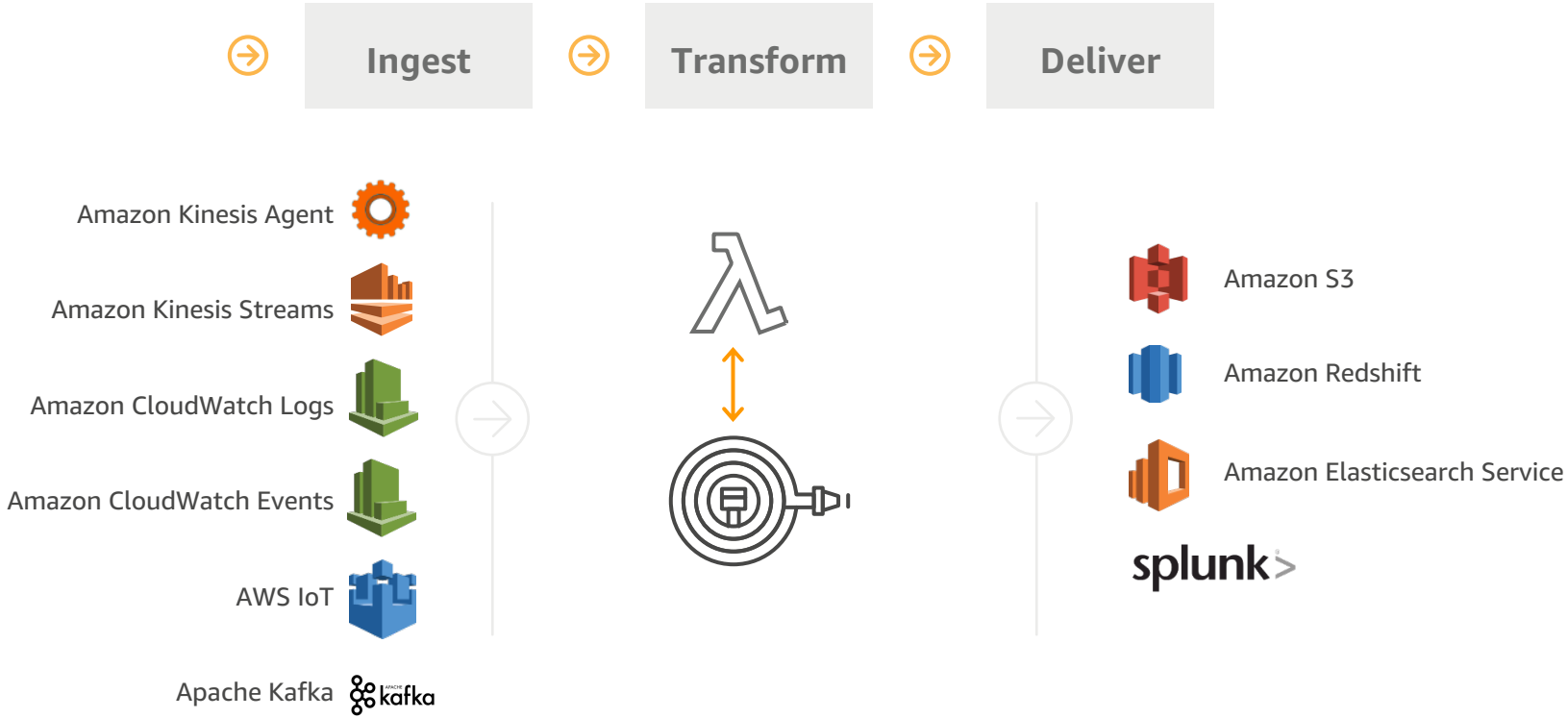
Analyze streaming data using your favorite BI tools

Zero administration: Capture and deliver streaming data to Amazon S3, Amazon Redshift, or Amazon Elasticsearch Service **without writing an app or managing infrastructure.**

Direct-to-data-store integration: Batch, compress, and **encrypt** streaming data for delivery **in as little as 60 seconds.**

Seamless elasticity: Seamlessly scales to match data throughput without intervention.

Kinesis Data Firehose—How it Works



Amazon Kinesis Streams vs Firehose

	Kinesis Streams	Kinesis Firehose
Use case	Capture and expose data streams to build arbitrary stream processing applications	Fully-managed service for automatic loading of data streams into S3, Redshift, ElasticSearch etc.
Provisioning and Resource Management	Provisioned model via Streams	No provisioning of the underlying resource called a DeliveryStream
	Customer-controllable construct of 'Shards'	No Shards (not a customer visible construct)
Scaling	Customer owns explicit scaling of stream/ shards via Split/Merge APIs	Firehose is completely elastic – no explicit scaling actions needed.
Ingestion (Put Data)	Customer owns "partition key' as part of PUT* API call	Different and simplified Put* API that doesn't need a partition key
Retrieval (Get Data)	Get API to retrieve data directly from stream	No Get* API – Data appears in destination (like S3/ Redshift) after meeting configuration criteria
	Full-flexibility to write/ manage own application with Kinesis Client Library and connector libraries	No application to be written or managed. Customers specify simple set of pre-defined configurations on the console or API

Firehose Highlights

- **Records**: The maximum size of a record (before Base64-encoding) is 1000 KB.
- **Compression**: Amazon Kinesis Data Firehose allows you to compress your data before delivering it to Amazon S3.
- **Transformation**: Firehose can invoke an AWS Lambda function to transform incoming data before delivering it to destinations.
- **Buffering**
    - Buffer size is in MBs and ranges from 1MB to 128MB.
    - Buffer interval is in seconds and ranges from 60 seconds to 900 seconds.

Source: Firehose FAQ

LAB 1.2

Summary of Lab 1

Congratulations! You have successfully created an ingestion pipeline without servers, which is ready to process huge amounts of data.

In the next lab, you will create the processing pipeline to convert, transform the data from the ingestion layer.