Amazon EMR

What is Amazon EMR

Easy to use
Launch a cluster in minutes

Low cost
Pay per-second

Open-source variety
Latest versions of software

Managed
Spend less time monitoring

Secure
Easy to enable options

Flexible
Full customization and control

Enterprise-grade Hadoop & Spark

Deploy latest releases in Hadoop and Spark ecosystems

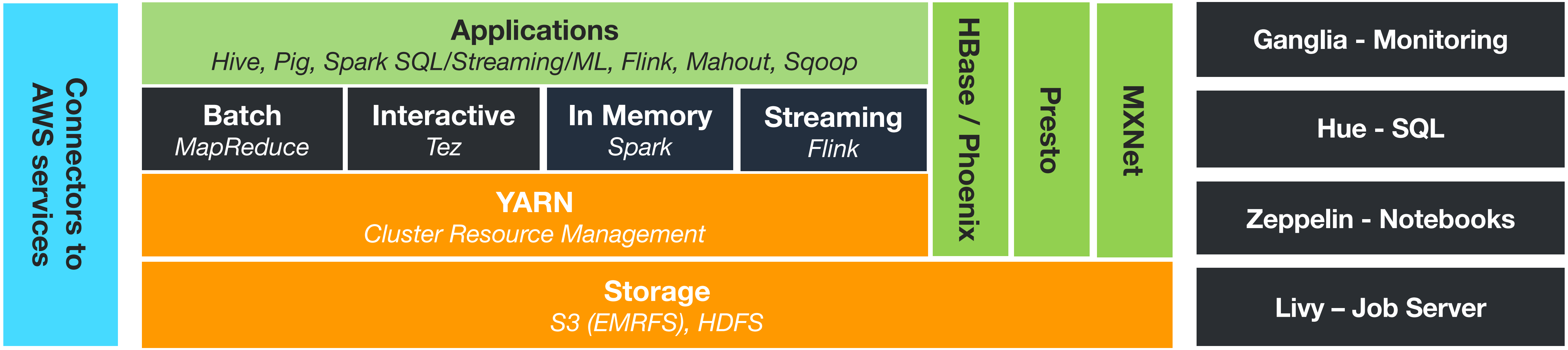
EMR Releases

	Hadoop	Ganglia	HBase	Hive & Catalog	Hue	Mahout	Oozie	Phoenix	Pig	Presto	Spark	Sqoop	Tez	Zeppelin	Zookeeper	Flink	Livy	MXNet
Emr-5.11.0 December 2015	2.7.3	3.7.2	1.3.1 + S3	2.3.2	4.0.1	0.13.0	4.3.0	4.11.0	0.17.0	.187	2.2.1	1.4.6	0.8.4	0.7.3	3.4.10	1.3.2	0.4.0	0.12.0
Emr-5.3.0 January 2017	2.7.3	3.7.2	1.2.3 + S3	2.1.1	3.11.0	0.12.2	4.3.0	4.7.0	0.16.0	0.157.1	2.1.0	1.4.6	0.8.4	0.6.2	3.4.9	1.1.4		
Emr-4.7.0 June 2016	2.7.2	3.7.2	1.2.1	1.0.0	3.7.1	0.12.0	4.2.0	4.7.0	0.14.0	.147	1.6.1	1.4.6	0.8.3	0.5.6	3.4.8			
Emr-4.0.0 July 2015	2.6.0			1.0.0		0.10.0			0.14.0		1.4.1							

- 19 open-source projects: Apache Hadoop, Spark, HBase, Presto, and more

- Updated with the latest open source frameworks within 30 days of release

Open-source applications



Connectors to AWS services

Applications
Hive, Pig, Spark SQL/Streaming/ML, Flink, Mahout, Sqoop

Batch *MapReduce*

Interactive *Tez*

In Memory *Spark*

Streaming *Flink*

YARN
Cluster Resource Management

Storage
S3 (EMRFS), HDFS

HBase / Phoenix

Presto

MXNet

Ganglia - Monitoring

Hue - SQL

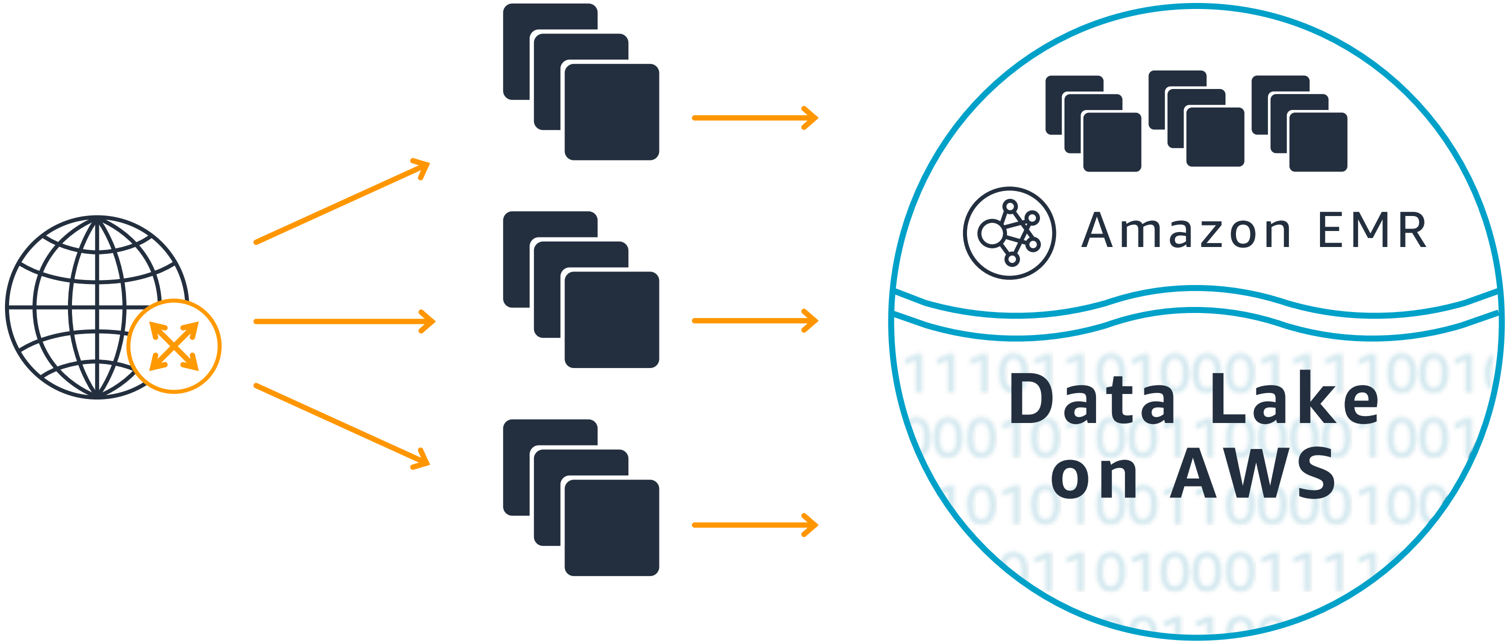
Zeppelin - Notebooks

Livy – Job Server

Amazon EMR service

Enterprise-grade Hadoop & Spark

Scale to any size



- Scale compute (EMR) & storage (S3) independently
- Store, and process any amount of data—PB to EBs
- Provision one, hundreds, or thousands of nodes
- Auto-scaling

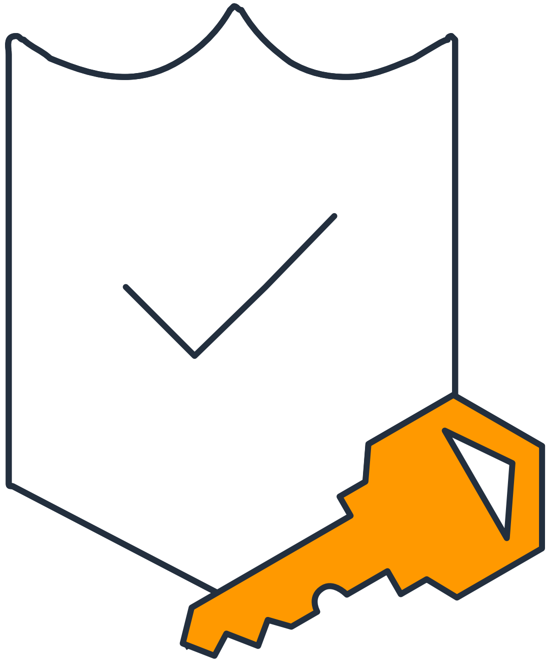
Enterprise-grade Hadoop & Spark

Highly available and durable

- S3 is designed to deliver 99.999999999% durability
- EMR monitors your cluster—replacing poorly performing & failed nodes, and restarting services
- Monitor your clusters using Amazon CloudWatch
- Built-in console to view job history & browse logs
- EMR has on-cluster HDFS for data persistence

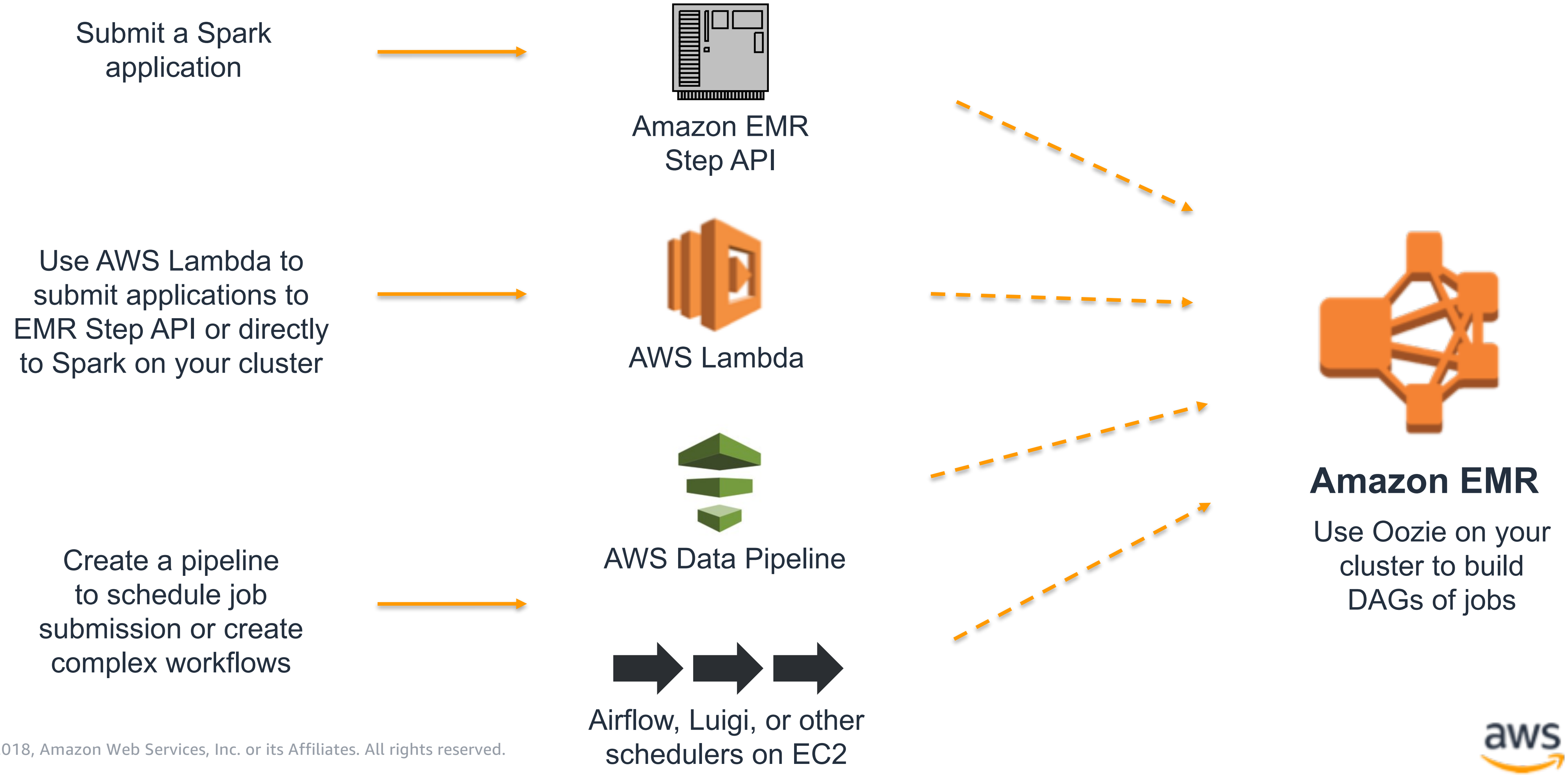
Enterprise-grade Hadoop & Spark

Highly secure



- Encryption of data at rest and in-transit
- ML-powered security with Amazon Macie
- Network isolation using Amazon VPC
- Access and permissions control with IAM policies
- Log, and audit activity with AWS CloudTrail
- Microsoft AD integration with Kerberos support

Options to submit jobs



Submit a Spark application

Use AWS Lambda to submit applications to EMR Step API or directly to Spark on your cluster

Create a pipeline to schedule job submission or create complex workflows

Amazon EMR Step API

AWS Lambda

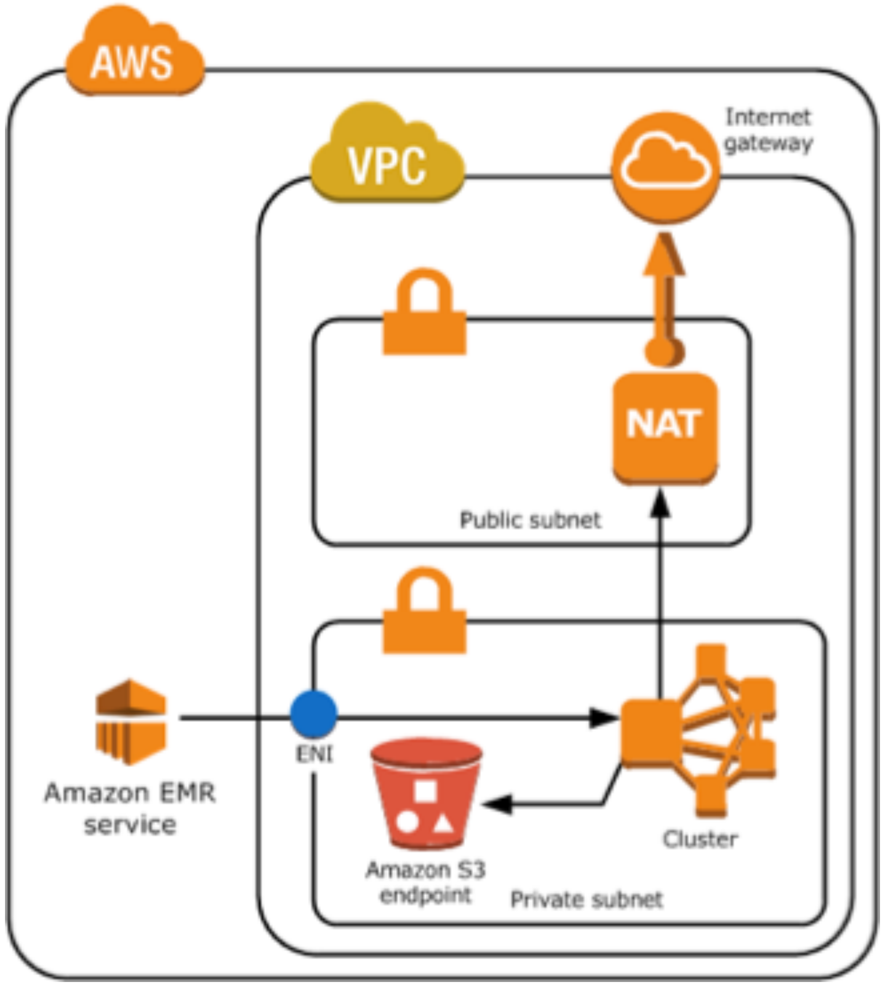
AWS Data Pipeline

Airflow, Luigi, or other schedulers on EC2

Amazon EMR

Use Oozie on your cluster to build DAGs of jobs

Networking: VPC private subnets



- Use Amazon S3 Endpoints for connectivity to S3
- Use Managed NAT for connectivity to other services or the Internet
- Control the traffic using Security Groups
  - ElasticMapReduce-Master-Private
  - ElasticMapReduce-Slave-Private
  - ElasticMapReduce-ServiceAccess

Access Control: IAM Users and Roles

- IAM Policies for access to Amazon EMR service (IAM users or federated users)
  - **AmazonElasticMapReduceFullAccess**
  - **AmazonElasticMapReduceReadOnlyAccess**

- IAM Policies for Amazon EMR cluster
  - Service role (**AmazonElasticMapReduceRole)** - Allowable actions for Amazon EMR service, like creating EC2 instances.
  - Instance profile (**AmazonElasticMapReduceforEC2Role)** - Applications that run on Amazon EMR, like access to Amazon S3 for EMRFS on your cluster.

Easy to Configure End-to-End Security

- Encryption for data at rest
  - Process encrypted data from Amazon S3 with support for all Amazon S3 encryption features
  - Configurable Local disk and HDFS encryption

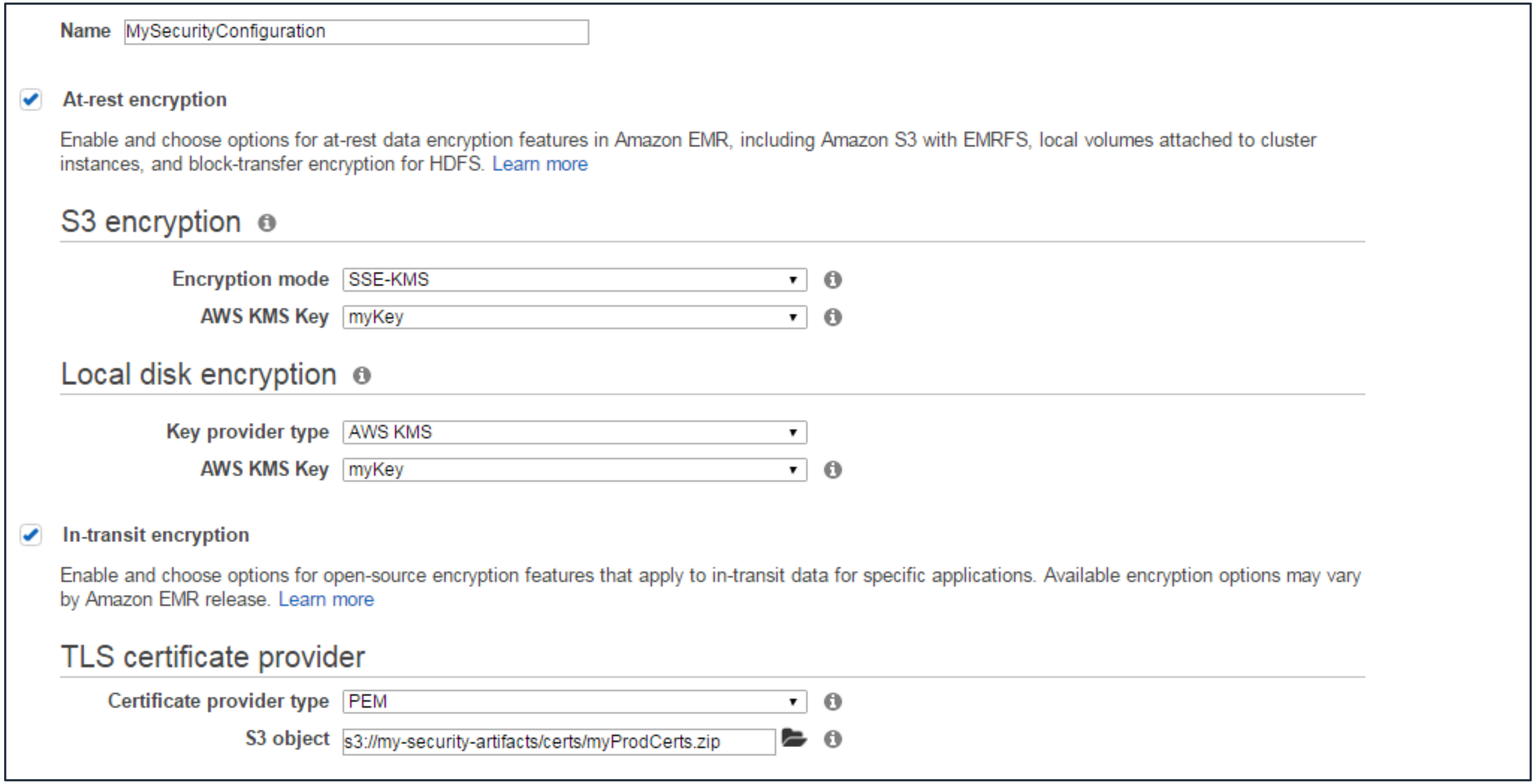
- Encryption for data in transit
  - Run EMR clusters in VPC private subnets
  - Encrypted inter-node communication for Hadoop, MapReduce, Spark
  - Data transfer to other services over SSL

- Integrated with AWS IAM
  - Support for IAM roles, Bucket policies & ACLs, Tag-based permissions

- Authentication and authorization with native Hadoop ecosystem feature set

- Compliance and Auditing
  - SOC 1/2/3, PCI-DSS, FedRAMP, HIPAA-eligible
  - All API calls logged in CloudTrail
  - Object access logging for S3 data

Encryption – use security configurations



Name	MySecurityConfiguration

☑ **At-rest encryption**

Enable and choose options for at-rest data encryption features in Amazon EMR, including Amazon S3 with EMRFS, local volumes attached to cluster instances, and block-transfer encryption for HDFS. Learn more

S3 encryption

Encryption mode	SSE-KMS
AWS KMS Key	myKey

Local disk encryption

Key provider type	AWS KMS
AWS KMS Key	myKey

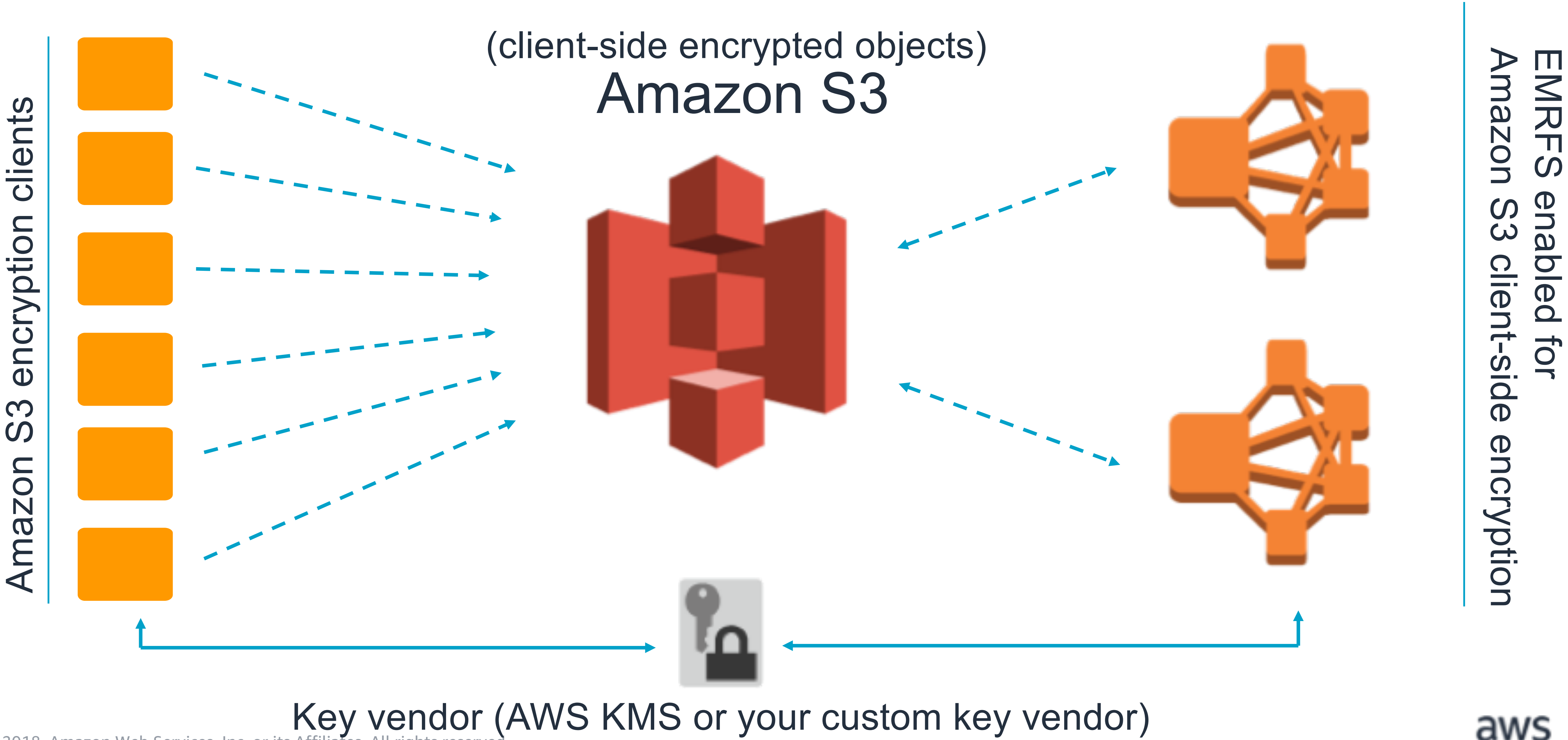
☑ **In-transit encryption**

Enable and choose options for open-source encryption features that apply to in-transit data for specific applications. Available encryption options may vary by Amazon EMR release. Learn more

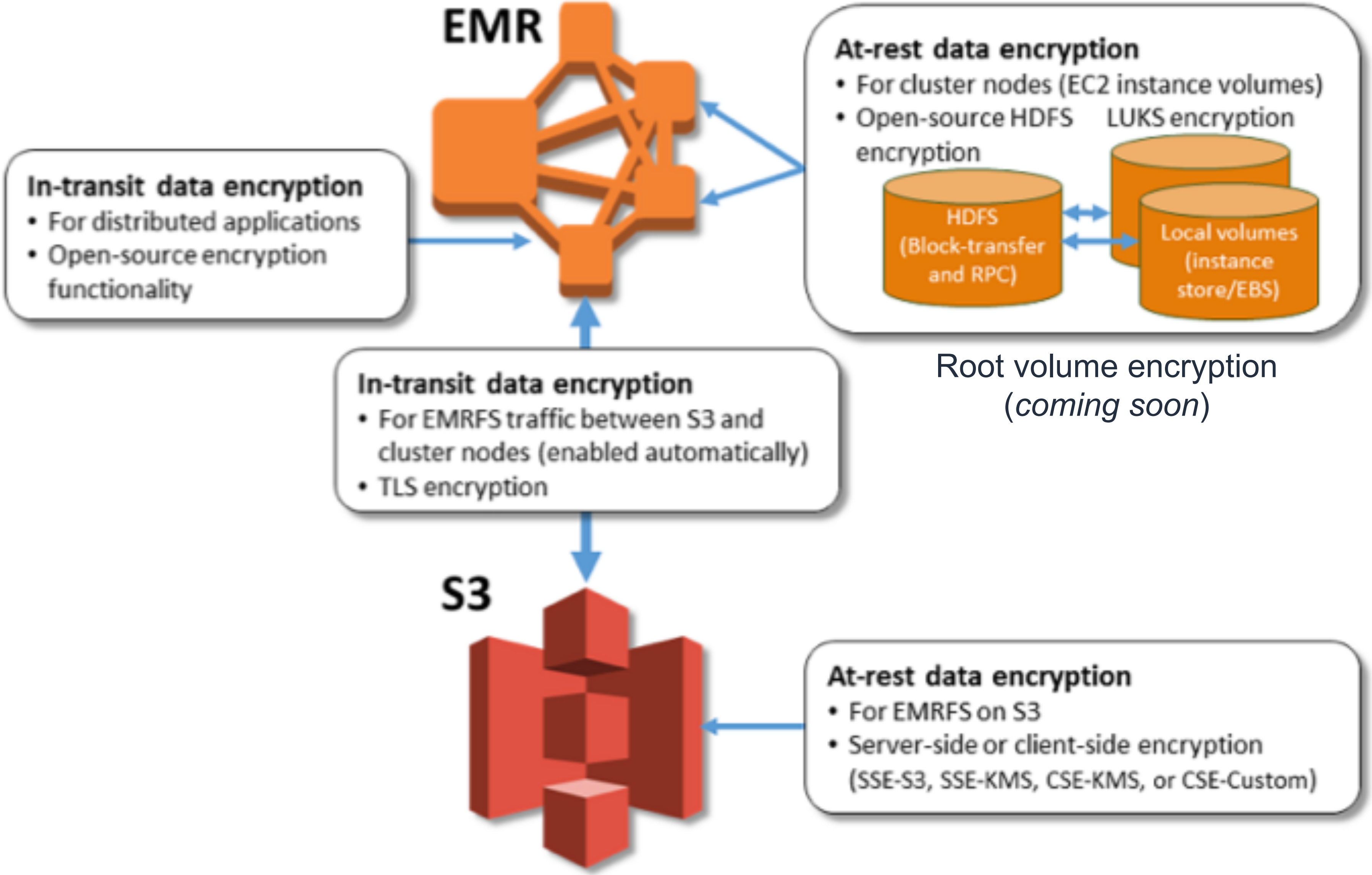
TLS certificate provider

Certificate provider type	PEM
S3 object	s3://my-security-artifacts/certs/myProdCerts.zip

Data at Rest: S3 client-side encryption


Amazon S3 encryption clients
(client-side encrypted objects)
Amazon S3
EMRFS enabled for
Amazon S3 client-side encryption
Key vendor (AWS KMS or your custom key vendor)

Security - Encryption


EMR
S3
HDFS (Block-transfer and RPC)
Local volumes (instance store/EBS)

In-transit data encryption
- For distributed applications
- Open-source encryption functionality

At-rest data encryption
- For cluster nodes (EC2 instance volumes)
- Open-source HDFS encryption

LUKS encryption

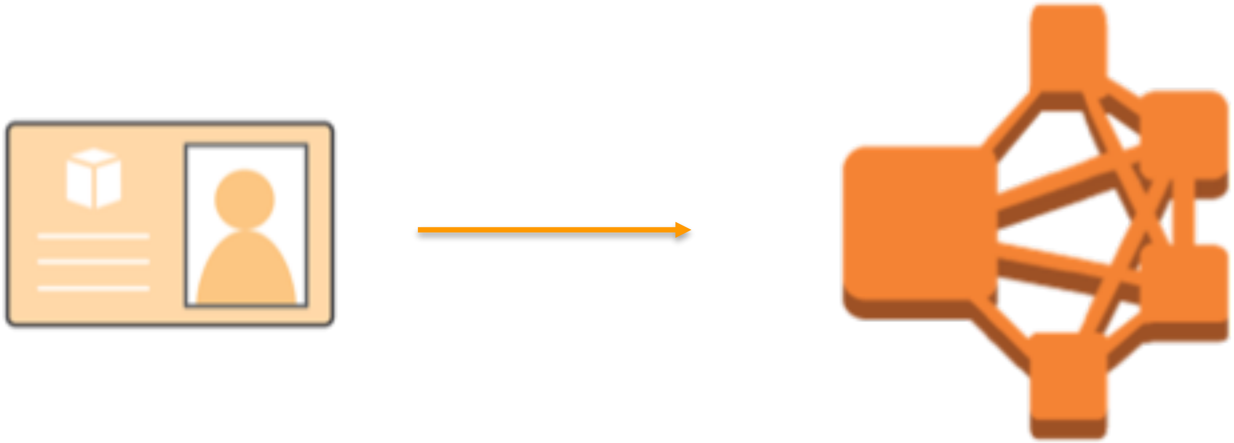
In-transit data encryption
- For EMRFS traffic between S3 and cluster nodes (enabled automatically)
- TLS encryption

At-rest data encryption
- For EMRFS on S3
- Server-side or client-side encryption (SSE-S3, SSE-KMS, CSE-KMS, or CSE-Custom)

Root volume encryption
(*coming soon*)

Supported
- Spark
- Tez
- MapReduce

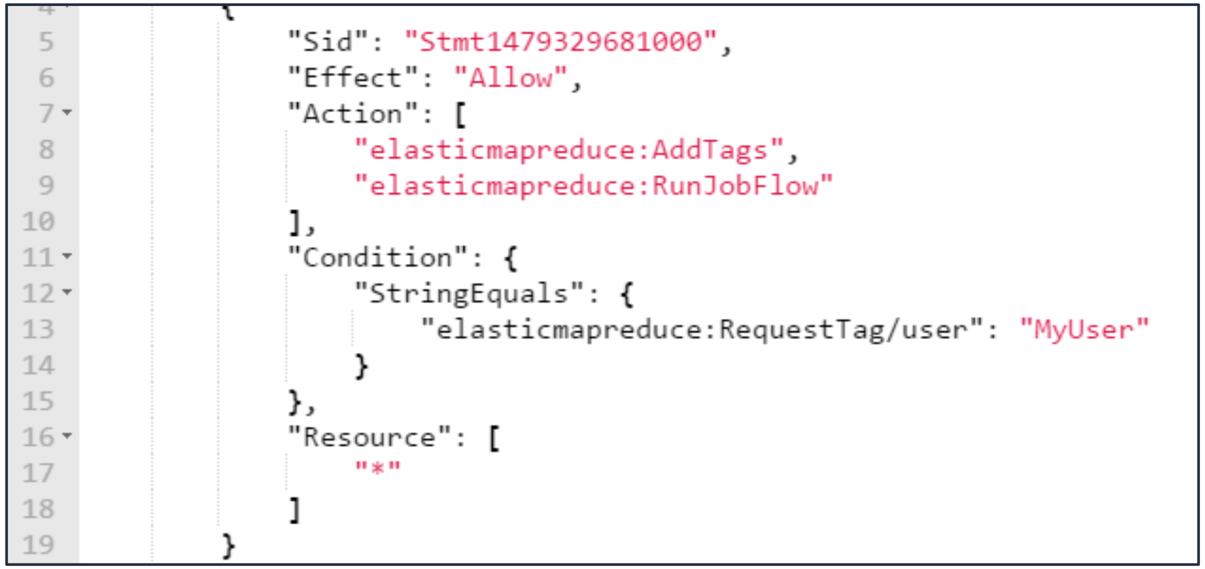
Security – Authentication and Authorization



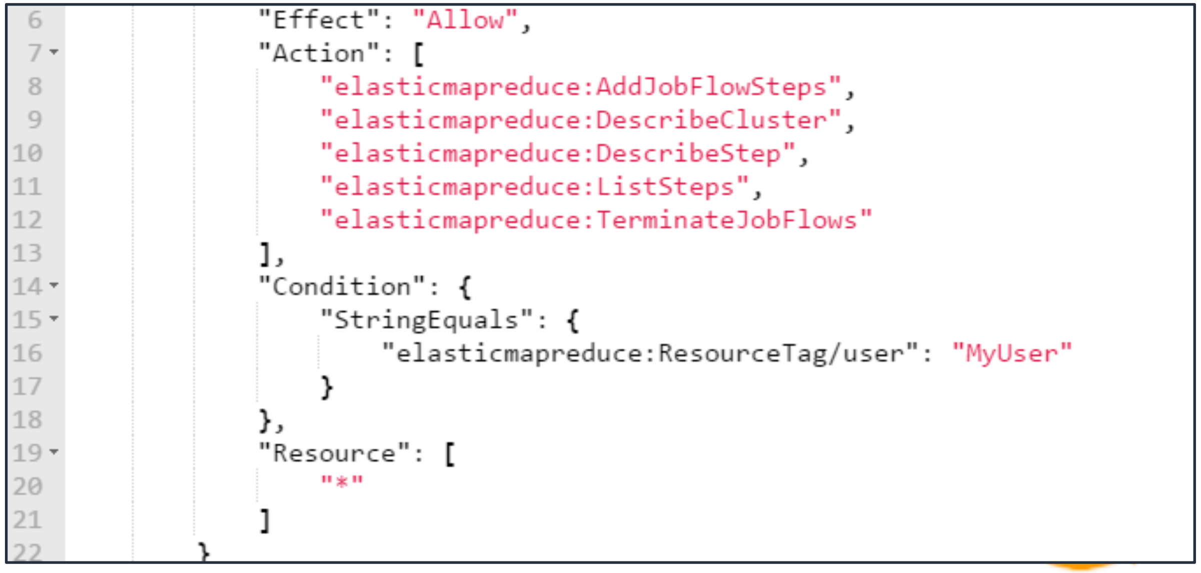
EMR role
EC2 role
SSH key

IAM user: `MyUser`

Tag: `user = MyUser`

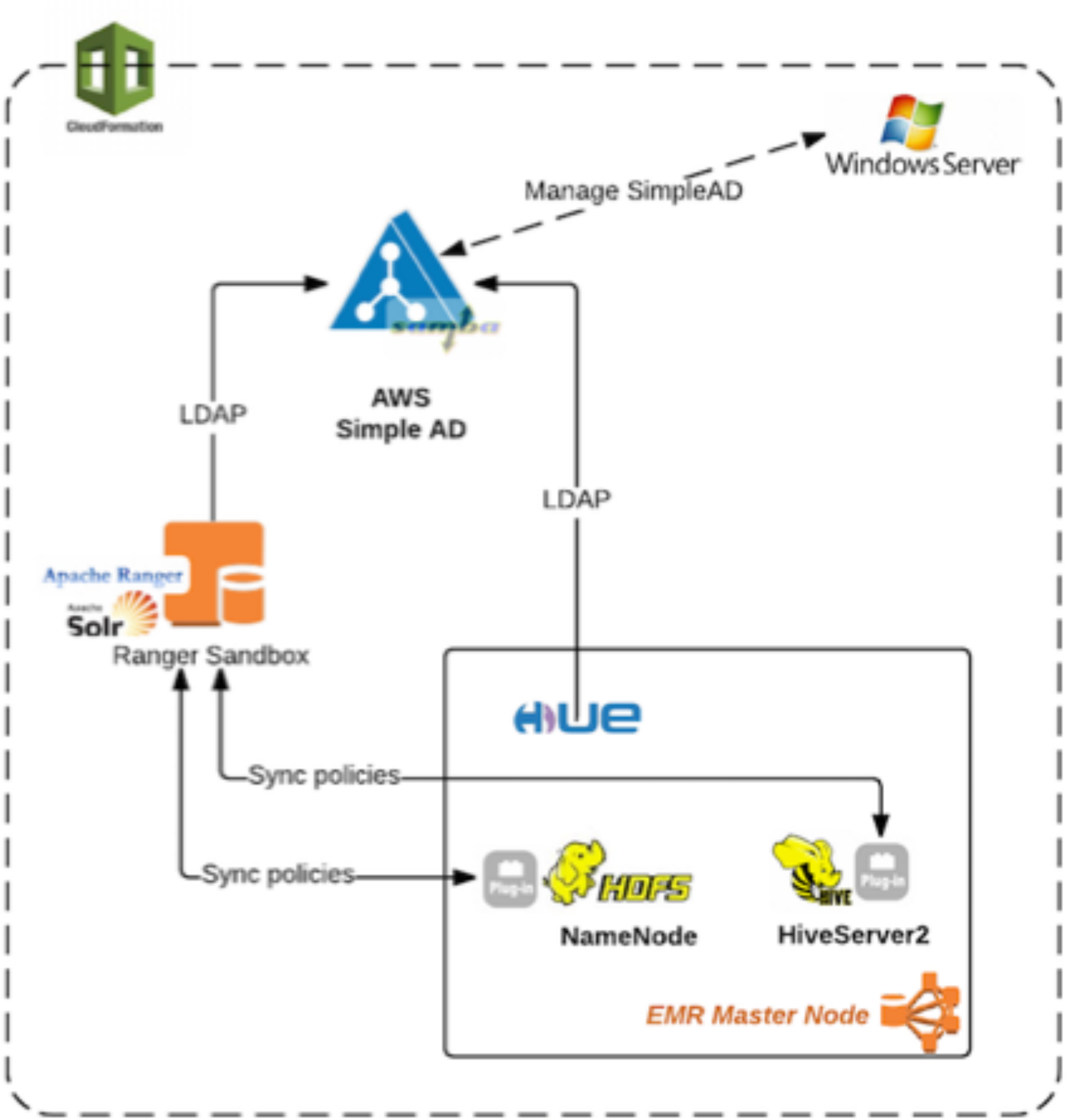


```
    {
        "Sid": "Stmt1479329681000",
        "Effect": "Allow",
        "Action": [
            "elasticmapreduce:AddTags",
            "elasticmapreduce:RunJobFlow"
        ],
        "Condition": {
            "StringEquals": {
                "elasticmapreduce:RequestTag/user": "MyUser"
            }
        },
        "Resource": [
            "*"
        ]
    }
```



```
        "Effect": "Allow",
        "Action": [
            "elasticmapreduce:AddJobFlowSteps",
            "elasticmapreduce:DescribeCluster",
            "elasticmapreduce:DescribeStep",
            "elasticmapreduce:ListSteps",
            "elasticmapreduce:TerminateJobFlows"
        ],
        "Condition": {
            "StringEquals": {
                "elasticmapreduce:ResourceTag/user": "MyUser"
            }
        },
        "Resource": [
            "*"
        ]
    }
```

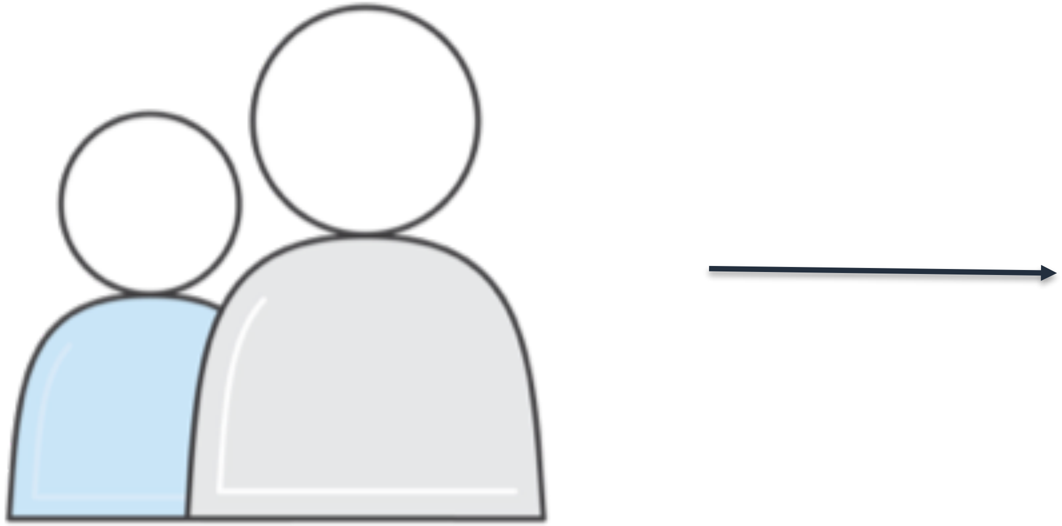
Security - Authentication and Authorization



Apache Ranger

- Plug-ins for Hive, HBase, YARN, and HDFS
- Row-level authorization for Hive (with data-masking)
- Full auditing capabilities with embedded search
- Run Ranger on an edge node – visit the AWS Big Data Blog

Authentication

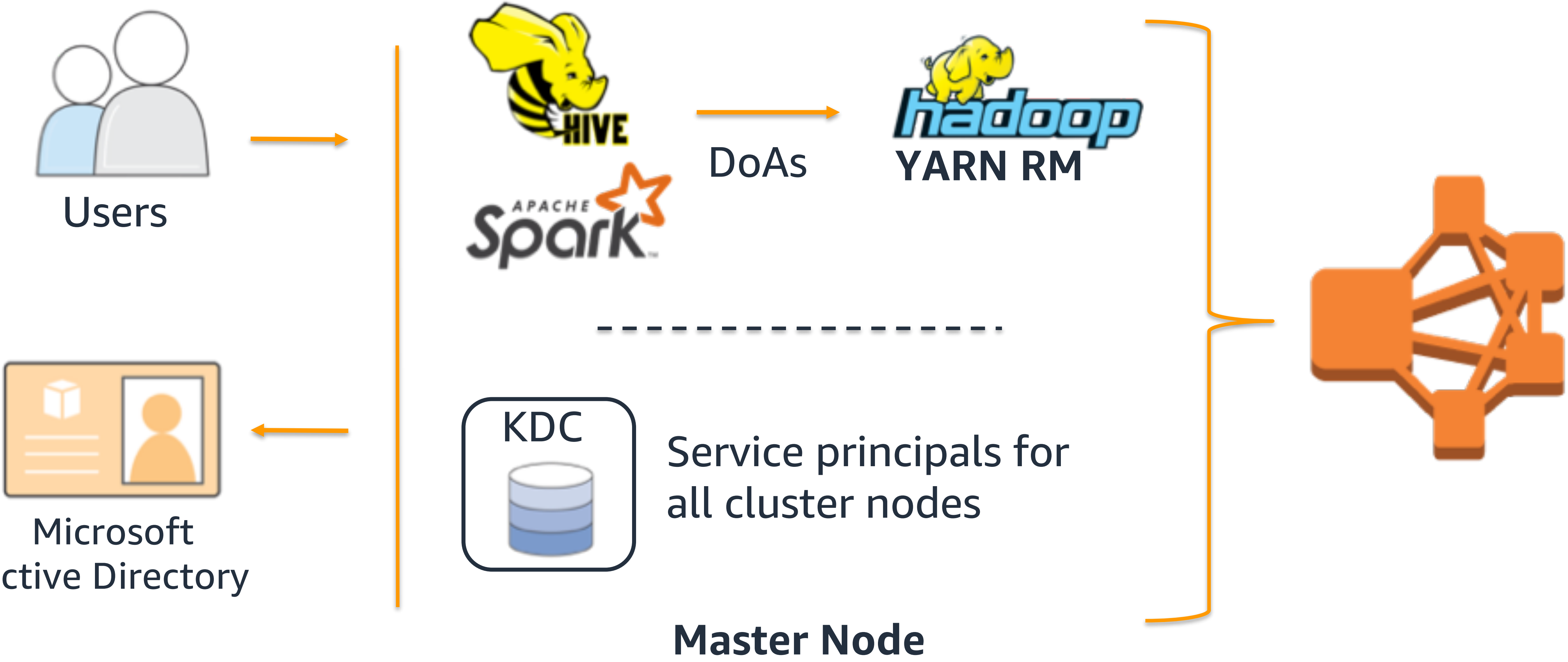


LDAP
HiveServer2
Presto Coordinator
Spark Thrift Server
Hue Server
Zeppelin Server

EC2 key pair
SSH as “hadoop”

AWS credentials
EMR Step (EMR API)

Authentication with Kerberos



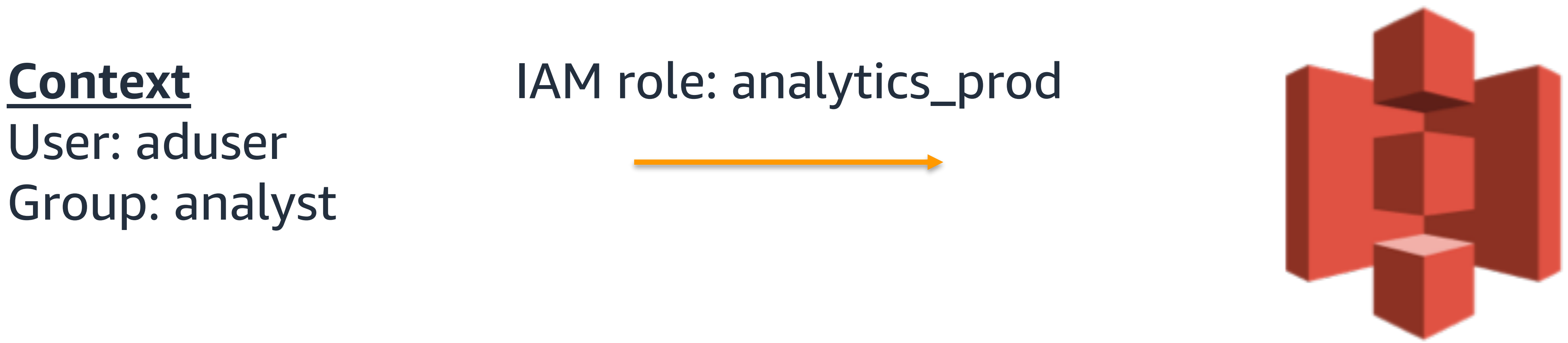
Service principals for all cluster nodes

Master Node

Authorization

- Storage-based
  - EMRFS/S3
  - HDFS
- HiveServer2 and Presto (SQL-based)
- HBase
- YARN queues
- Fine-grained access control by cluster tag (IAM)
- Apache Ranger on edge node (using CloudFormation)

EMRFS fine-grained authorization



Context
User: aduser
Group: analyst

IAM role: analytics_prod

Can map IAM roles to user, group, or S3 prefix

Security – Governance and Auditing

- AWS CloudTrail for EMR APIs
- S3 access logs for cluster S3 access
- YARN and application logs
- Ranger for UI for application level auditing

Monitoring

Full **visibility** of your AWS environment

- AWS CloudTrail will record access to API calls and save logs in your S3 buckets, no matter how those API calls were made

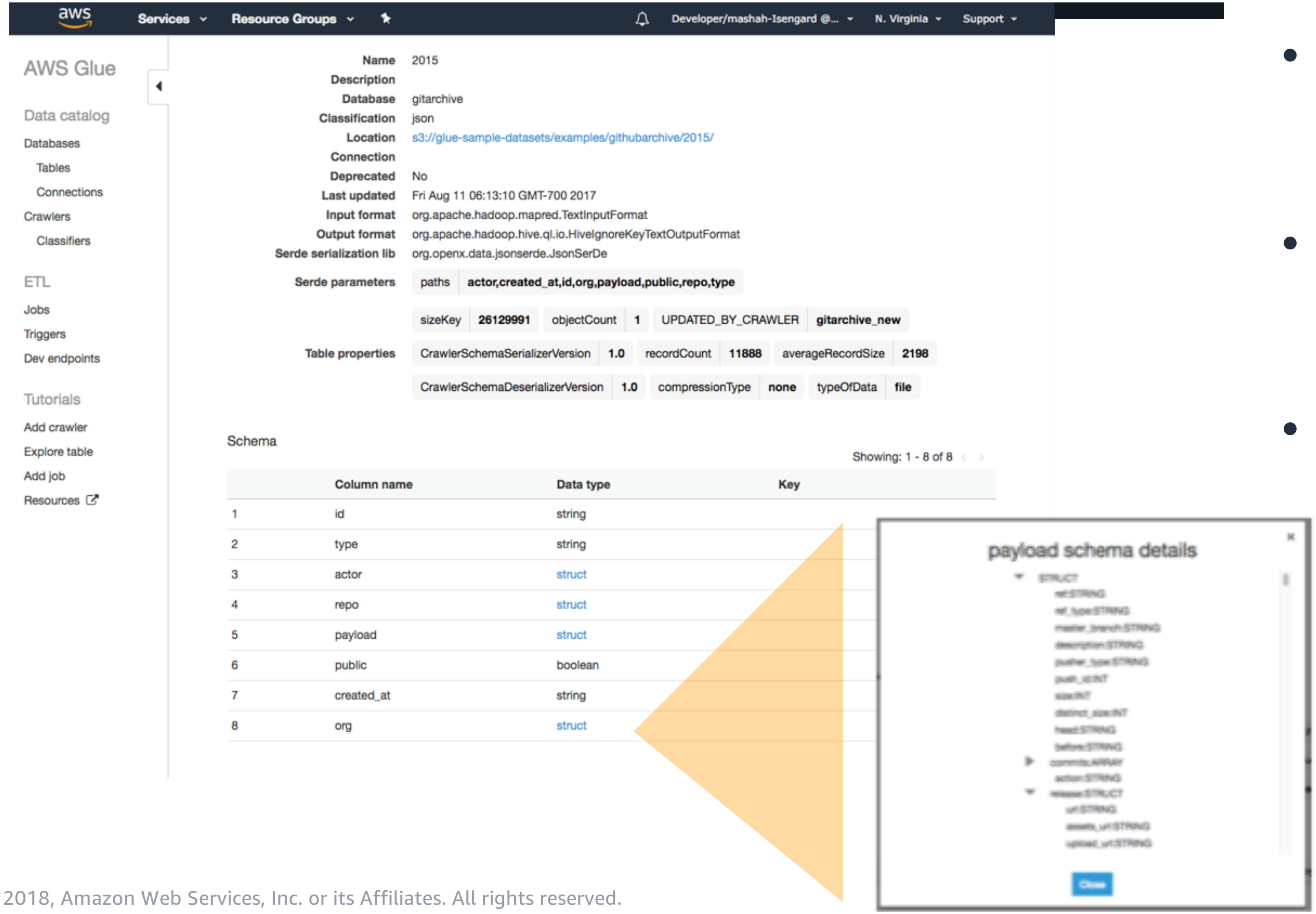
Who did **what** and **when** and from **where** (IP address)

- AWS CloudTrail support for many AWS services and growing - including **Amazon EMR**

Monitoring

- Amazon S3
  - Bucket access logs
- Amazon EMR
  - Archives various log files to Amazon S3 at 5 minute intervals.
  - Log files are available after the cluster terminates
- CloudWatch Metrics
  - Updated every five minutes and archived for two weeks
- Ganglia

Use the AWS Glue Data Catalog



Name 2015
Description
Database gitarchive
Classification json
Location s3://glue-sample-datasets/examples/githubarchive/2015/
Connection
Deprecated No
Last updated Fri Aug 11 06:13:10 GMT-700 2017
Input format org.apache.hadoop.mapred.TextInputFormat
Output format org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Serde serialization lib org.openx.data.jsonserde.JsonSerDe
Serde parameters paths **actor,created_at,id,org,payload,public,repo,type**
Table properties sizeKey **26129991** objectCount **1** UPDATED_BY_CRAWLER **gitarchive_new** CrawlerSchemaSerializerVersion **1.0** recordCount **11888** averageRecordSize **2198** CrawlerSchemaDeserializerVersion **1.0** compressionType **none** typeOfData **file**

Schema

Showing: 1 - 8 of 8

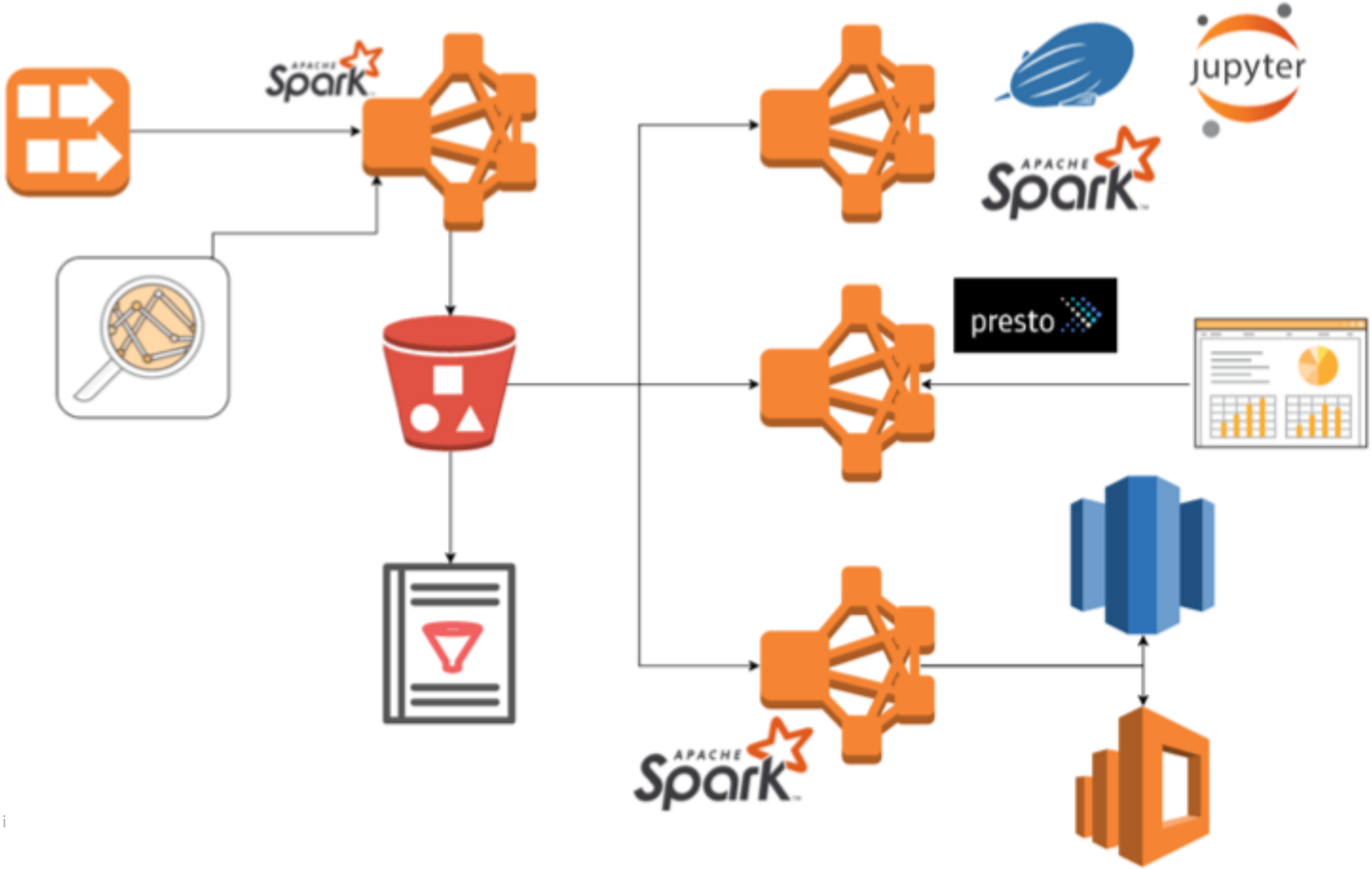
	Column name	Data type	Key
1	id	string	
2	type	string	
3	actor	struct	
4	repo	struct	
5	payload	struct	
6	public	boolean	
7	created_at	string	
8	org	struct	

payload schema details

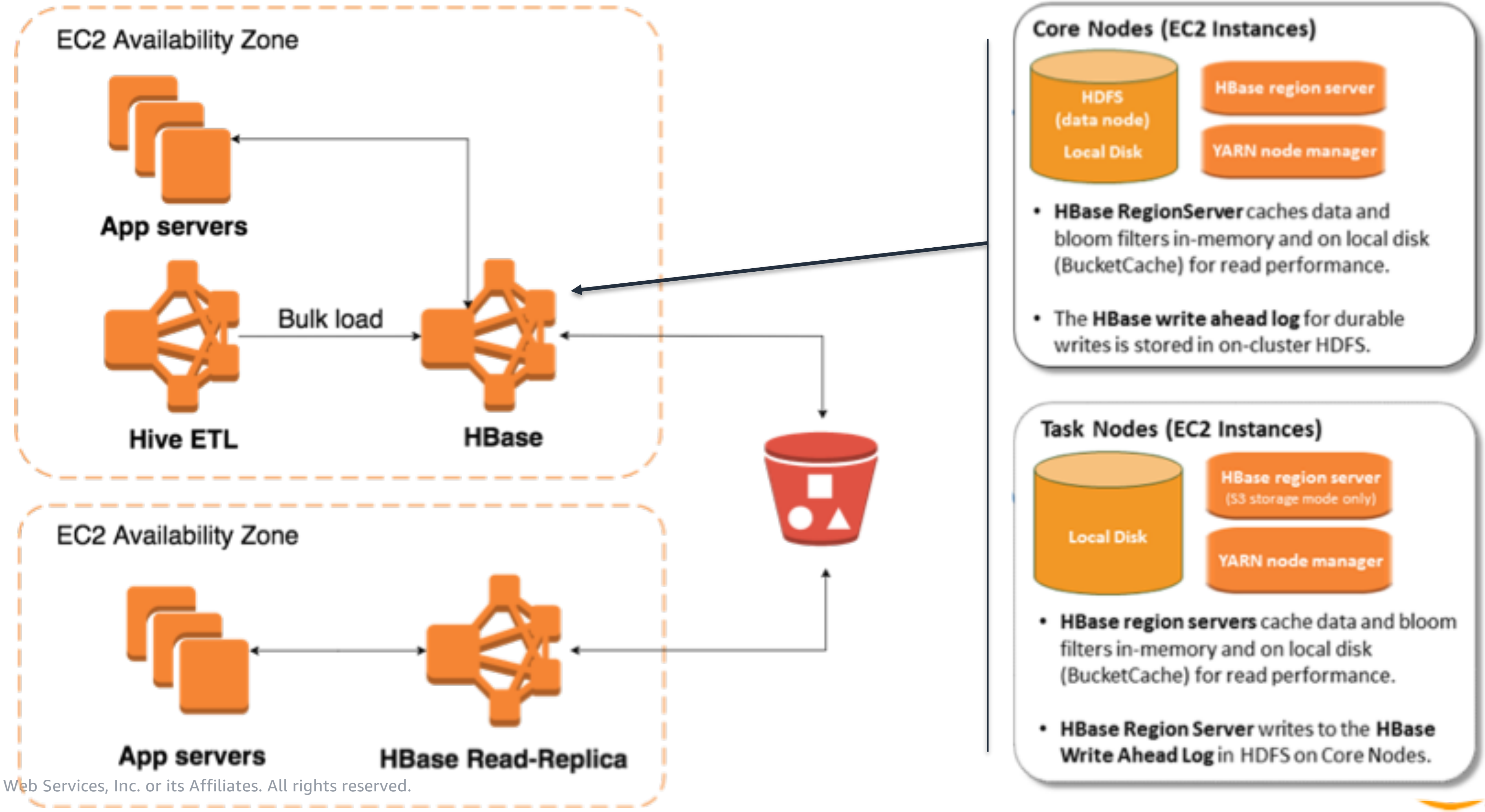
```
▼ STRUCT
    ref:STRING
    ref_type:STRING
    master_branch:STRING
    description:STRING
    pusher_type:STRING
    push_id:INT
    size:INT
    distinct_size:INT
    head:STRING
    before:STRING
  ▶ commits:ARRAY
    action:STRING
  ▼ release:STRUCT
      url:STRING
      assets_url:STRING
      upload_url:STRING
```

- Support for Spark, Hive and Presto
- Auto-generate schema and partitions
- Managed table updates

Real-time and batch processing



HBase for random access at massive scale



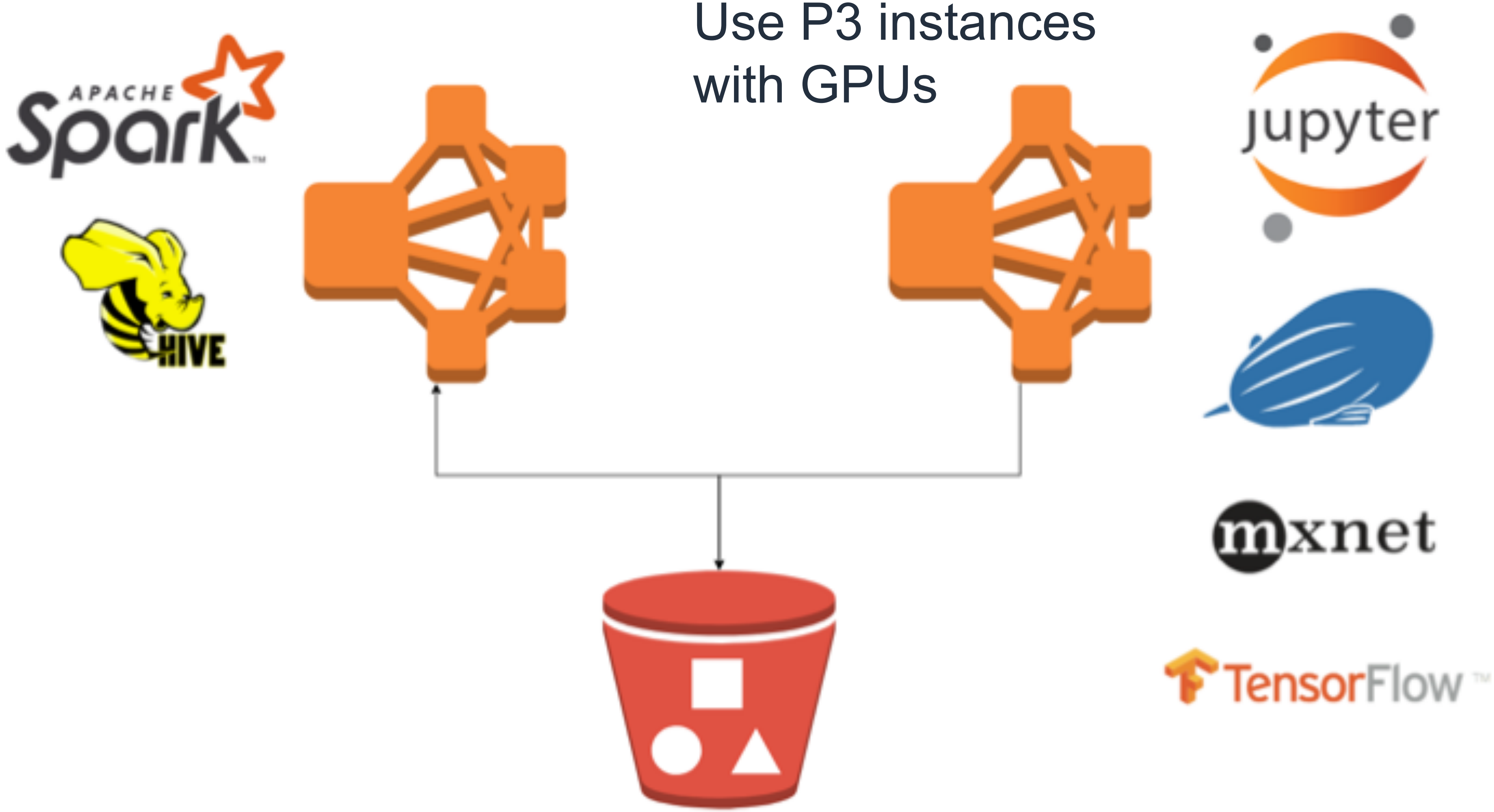
Core Nodes (EC2 Instances)

- **HBase RegionServer** caches data and bloom filters in-memory and on local disk (BucketCache) for read performance.
- The **HBase write ahead log** for durable writes is stored in on-cluster HDFS.

Task Nodes (EC2 Instances)

- **HBase region servers** cache data and bloom filters in-memory and on local disk (BucketCache) for read performance.
- **HBase Region Server** writes to the **HBase Write Ahead Log** in HDFS on Core Nodes.

Deep learning with GPU instances

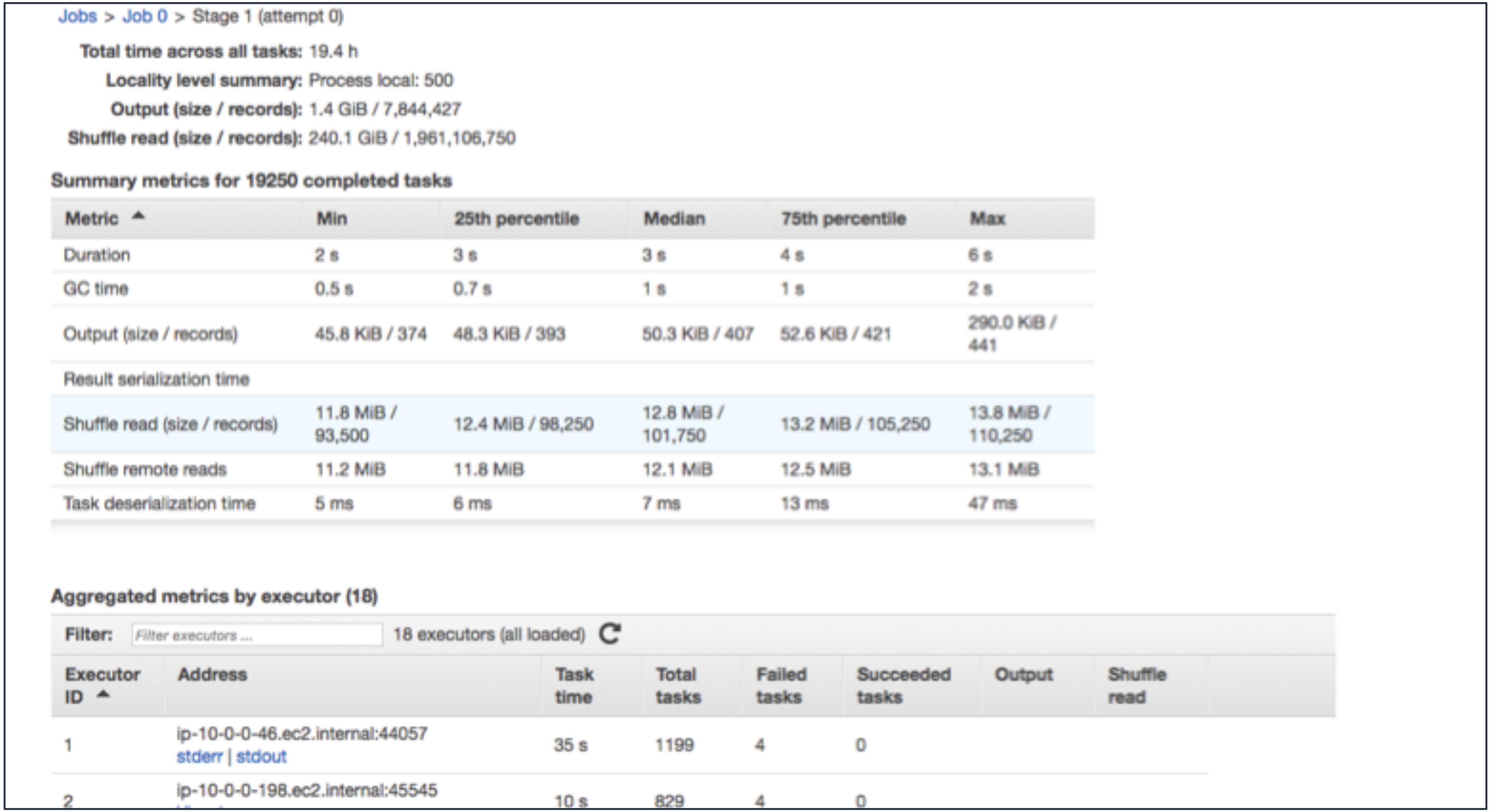


Use P3 instances with GPUs

Use a custom Amazon Linux AMI

- Launch clusters with your Amazon Linux AMI
- Preinstall custom software for faster start times
- Encrypt the root volume with an AWS KMS key
- Adjust root volume size for custom applications

Application History – EMR console



Jobs > Job 0 > Stage 1 (attempt 0)

Total time across all tasks: 19.4 h
Locality level summary: Process local: 500
Output (size / records): 1.4 GiB / 7,844,427
Shuffle read (size / records): 240.1 GiB / 1,961,106,750

Summary metrics for 19250 completed tasks

Metric ▲	Min	25th percentile	Median	75th percentile	Max
Duration	2 s	3 s	3 s	4 s	6 s
GC time	0.5 s	0.7 s	1 s	1 s	2 s
Output (size / records)	45.8 KiB / 374	48.3 KiB / 393	50.3 KiB / 407	52.6 KiB / 421	290.0 KiB / 441
Result serialization time					
Shuffle read (size / records)	11.8 MiB / 93,500	12.4 MiB / 98,250	12.8 MiB / 101,750	13.2 MiB / 105,250	13.8 MiB / 110,250
Shuffle remote reads	11.2 MiB	11.8 MiB	12.1 MiB	12.5 MiB	13.1 MiB
Task deserialization time	5 ms	6 ms	7 ms	13 ms	47 ms

Aggregated metrics by executor (18)

Filter: Filter executors ... 18 executors (all loaded)

Executor ID ▲	Address	Task time	Total tasks	Failed tasks	Succeeded tasks	Output	Shuffle read
1	ip-10-0-0-46.ec2.internal:44057 stderr | stdout	35 s	1199	4	0		
2	ip-10-0-0-198.ec2.internal:45545	10 s	829	4	0		

Lowest cost

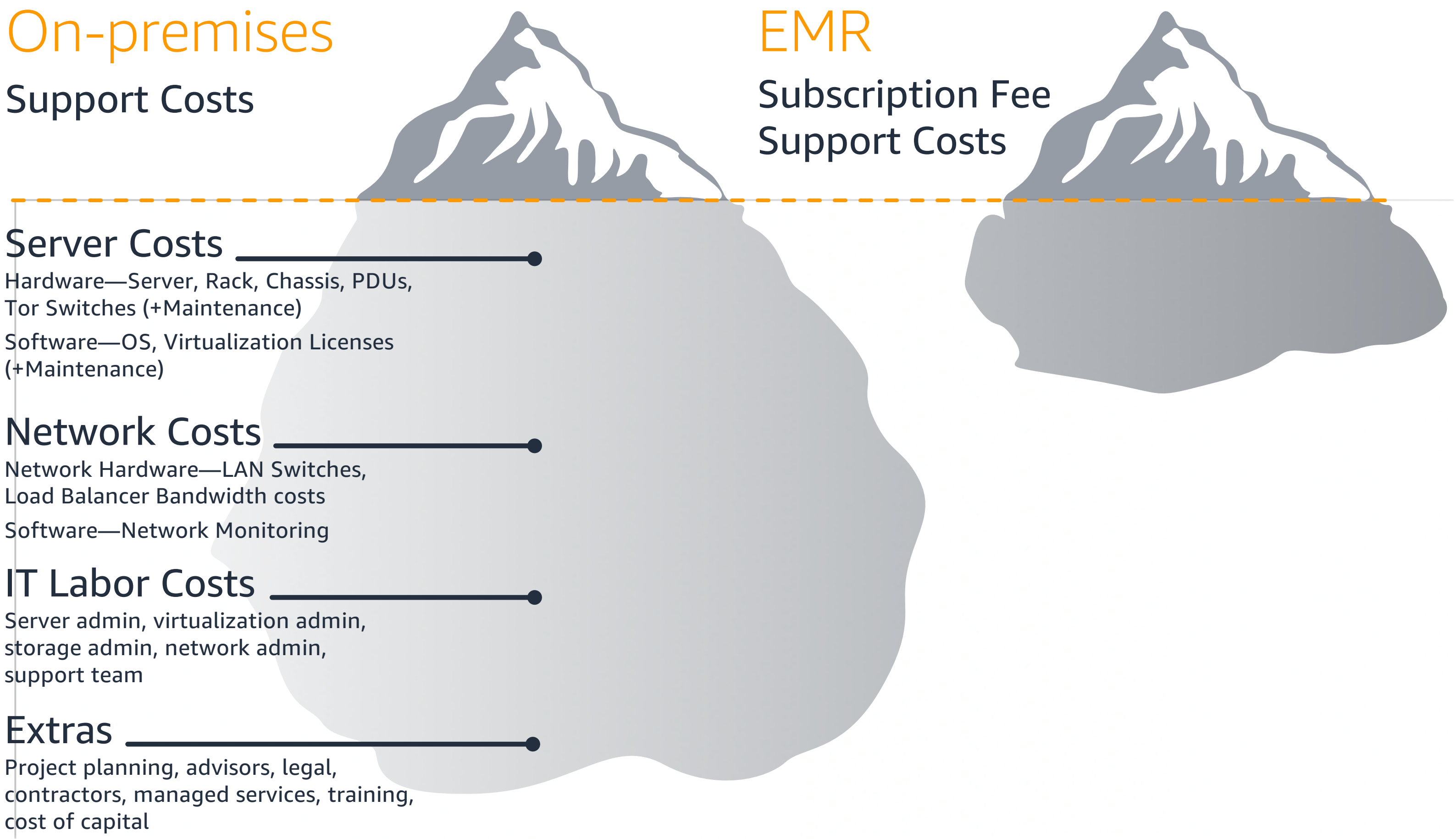
Save 75–90% using Reserved Instances and Spot

On-demand
EC2 nodes
EC2 Spot
up to
90% savings

- Commit to a set term and save up to 75% with Reserved Instance
- Run on spare compute capacity and save up to 90% with Spot
  - Pay a fraction of on-demand price
  - Name your bid price. If it exceeds the market, you get the resource
  - Provision from a list of instance types with Spot and on-demand
  - Launch in the most optimal AZ based on capacity/price
  - Spot Block support

Lowest cost

Lowest TCO



On-premises

Support Costs

Server Costs

Hardware—Server, Rack, Chassis, PDUs, Tor Switches (+Maintenance)

Software—OS, Virtualization Licenses (+Maintenance)

Network Costs

Network Hardware—LAN Switches, Load Balancer Bandwidth costs

Software—Network Monitoring

IT Labor Costs

Server admin, virtualization admin, storage admin, network admin, support team

Extras

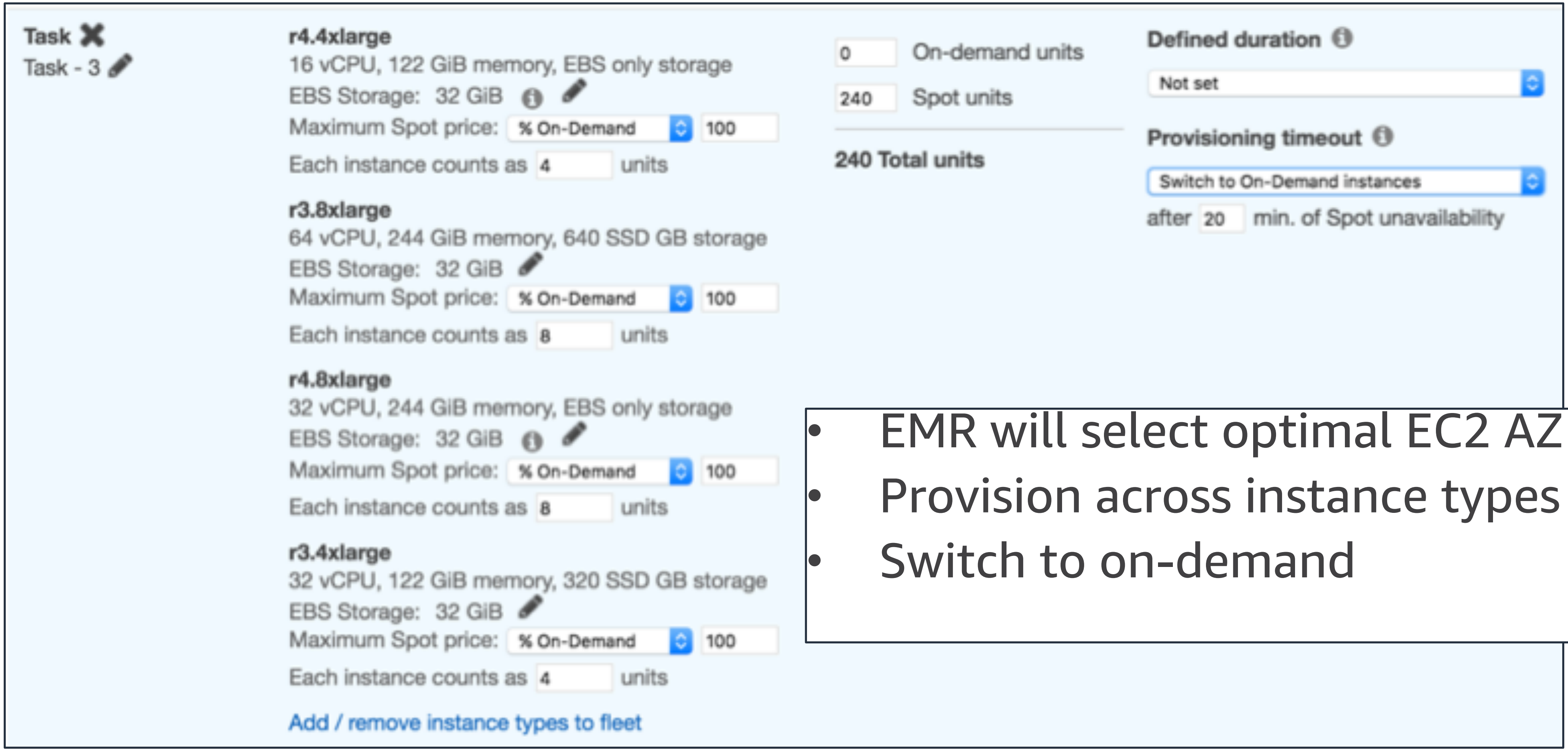
Project planning, advisors, legal, contractors, managed services, training, cost of capital

EMR

Subscription Fee
Support Costs

- Less admin time to manage, and support Hadoop clusters
- No up-front costs—hardware acquisition, installation
- Save on operating costs—data center space, power, cooling
- Business Value: Cost of delays, Risk Premium, Competitive abilities, governance, etc.

EC2 Spot and instance fleets



Task ✖

Task - 3

r4.4xlarge

16 vCPU, 122 GiB memory, EBS only storage

EBS Storage: 32 GiB

Maximum Spot price: % On-Demand 100

Each instance counts as 4 units

r3.8xlarge

64 vCPU, 244 GiB memory, 640 SSD GB storage

EBS Storage: 32 GiB

Maximum Spot price: % On-Demand 100

Each instance counts as 8 units

r4.8xlarge

32 vCPU, 244 GiB memory, EBS only storage

EBS Storage: 32 GiB

Maximum Spot price: % On-Demand 100

Each instance counts as 8 units

r3.4xlarge

32 vCPU, 122 GiB memory, 320 SSD GB storage

EBS Storage: 32 GiB

Maximum Spot price: % On-Demand 100

Each instance counts as 4 units

Add / remove instance types to fleet

0 On-demand units

240 Spot units

240 Total units

Defined duration

Not set

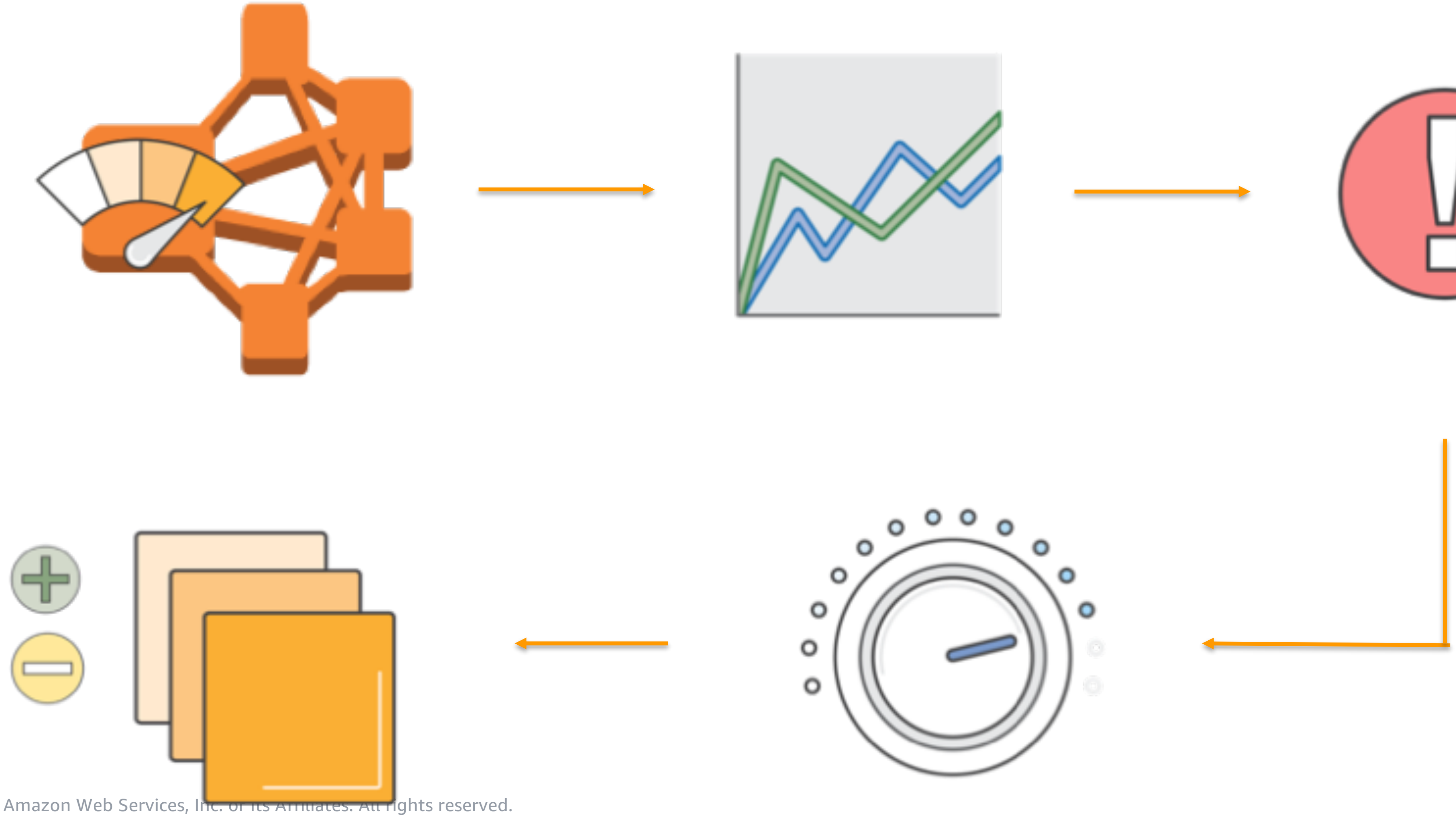
Provisioning timeout

Switch to On-Demand instances

after 20 min. of Spot unavailability

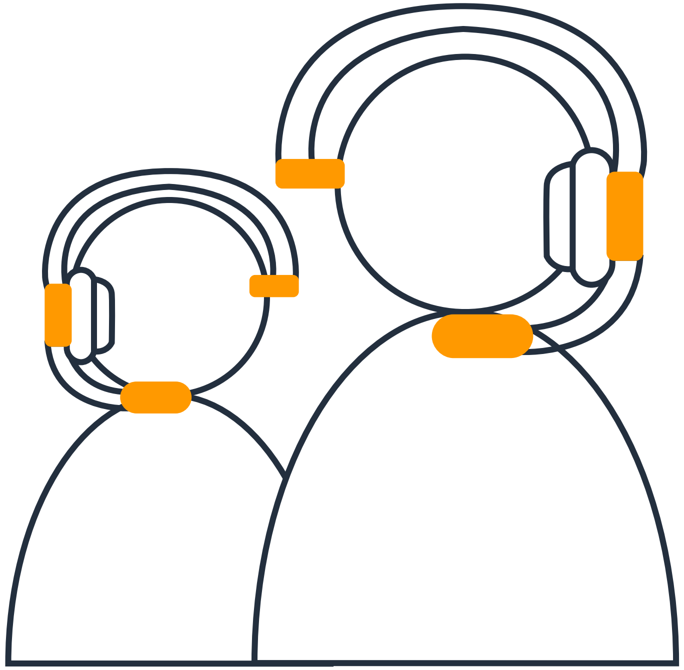
- EMR will select optimal EC2 AZ
- Provision across instance types
- Switch to on-demand

Auto Scaling



Lowest cost

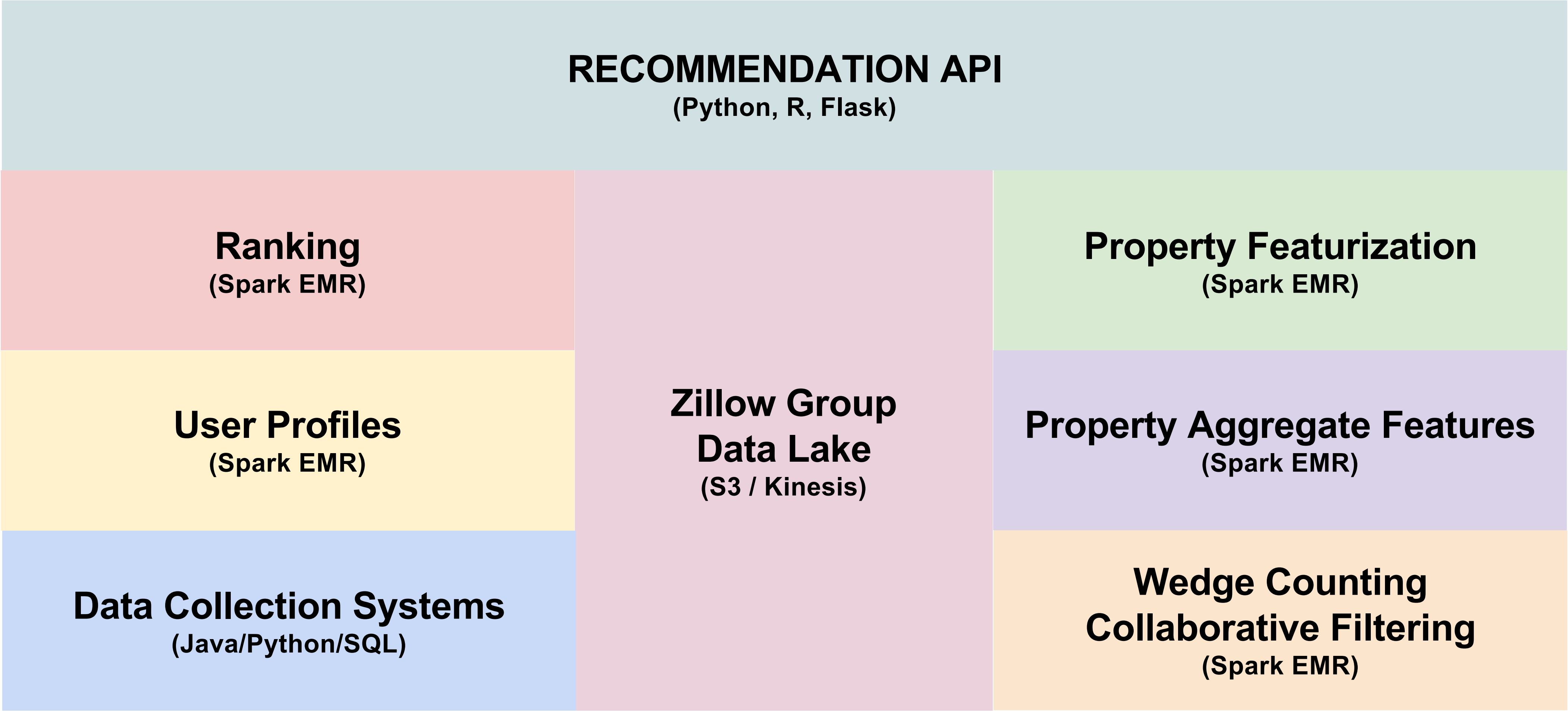
Hadoop & Spark support included



- Hadoop & Spark support included in AWS support
- Support for a 10-node cluster is $4.5K/year
- Up to 1/8th the cost of commercial Hadoop offerings

Customer Story – Zillow Group

Recommendations



RECOMMENDATION API
(Python, R, Flask)

Ranking
(Spark EMR)

User Profiles
(Spark EMR)

Data Collection Systems
(Java/Python/SQL)

Zillow Group Data Lake
(S3 / Kinesis)

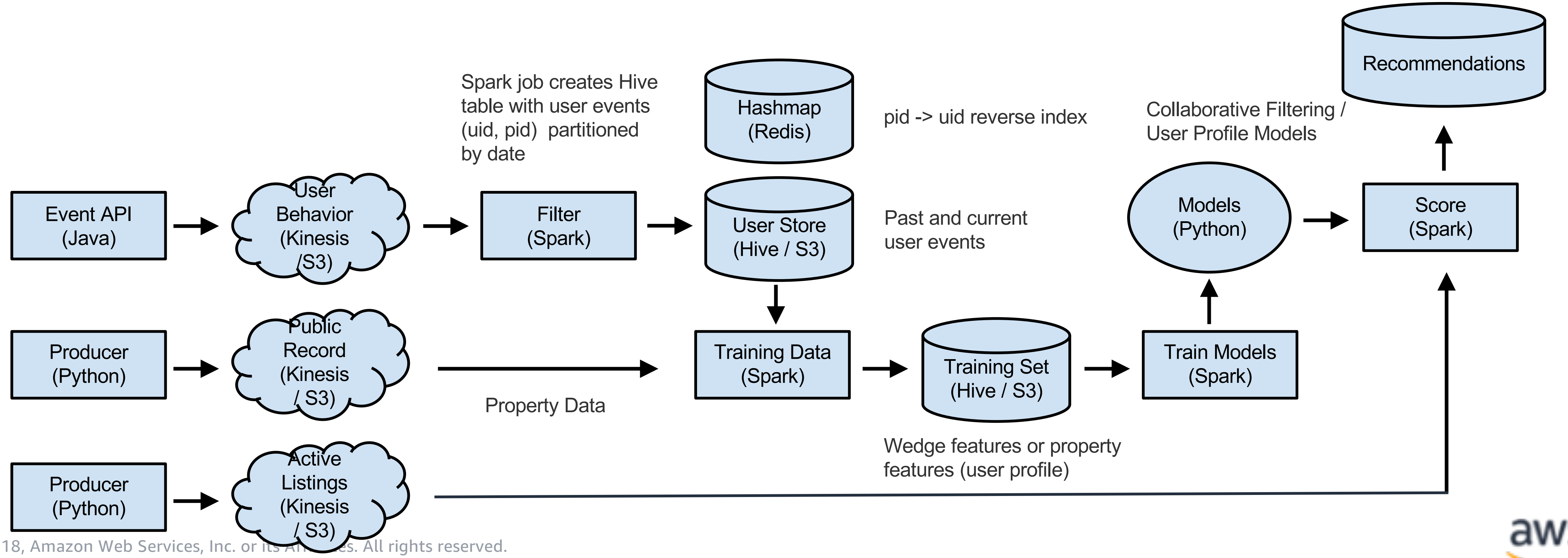
Property Featurization
(Spark EMR)

Property Aggregate Features
(Spark EMR)

Wedge Counting Collaborative Filtering
(Spark EMR)

Training & scoring

Collect user behavior and real-estate data, train the various models, generate the candidate set, and make predictions.



Event API (Java)

User Behavior (Kinesis /S3)

Spark job creates Hive table with user events (uid, pid) partitioned by date

Filter (Spark)

Hashmap (Redis)

pid -> uid reverse index

User Store (Hive / S3)

Past and current user events

Producer (Python)

Public Record (Kinesis / S3)

Property Data

Training Data (Spark)

Training Set (Hive / S3)

Wedge features or property features (user profile)

Train Models (Spark)

Models (Python)

Collaborative Filtering / User Profile Models

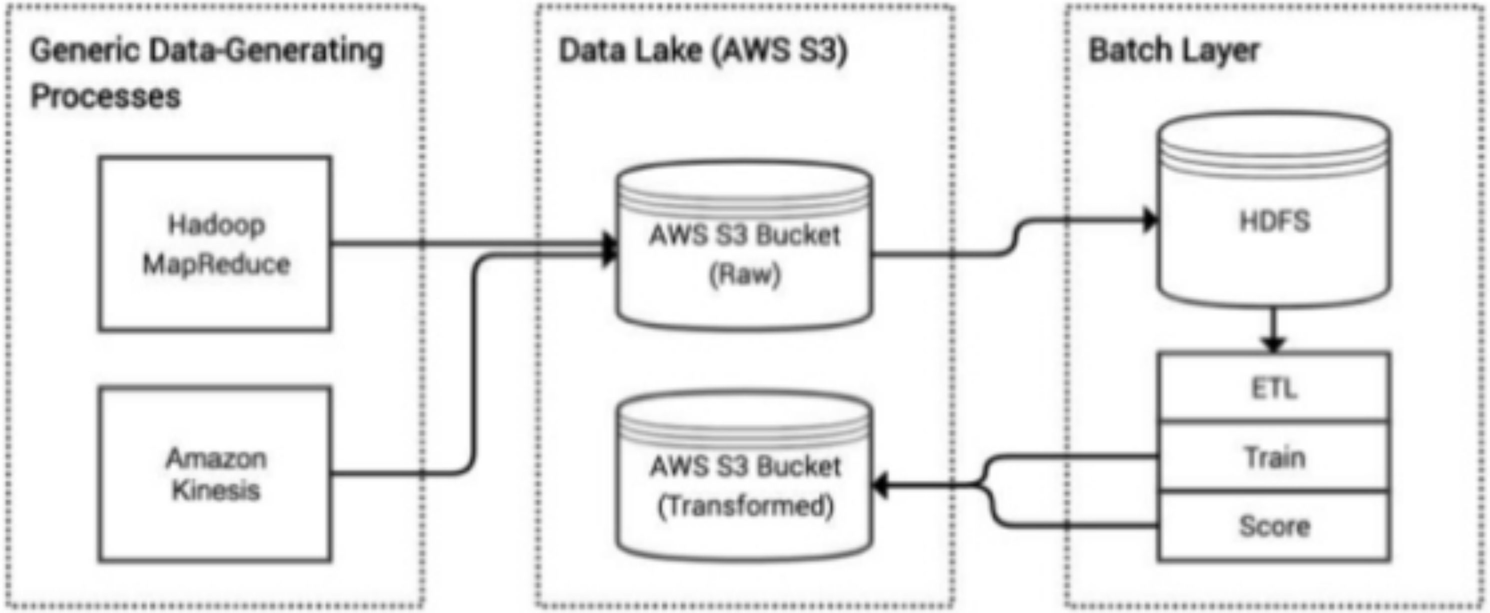
Score (Spark)

Recommendations

Producer (Python)

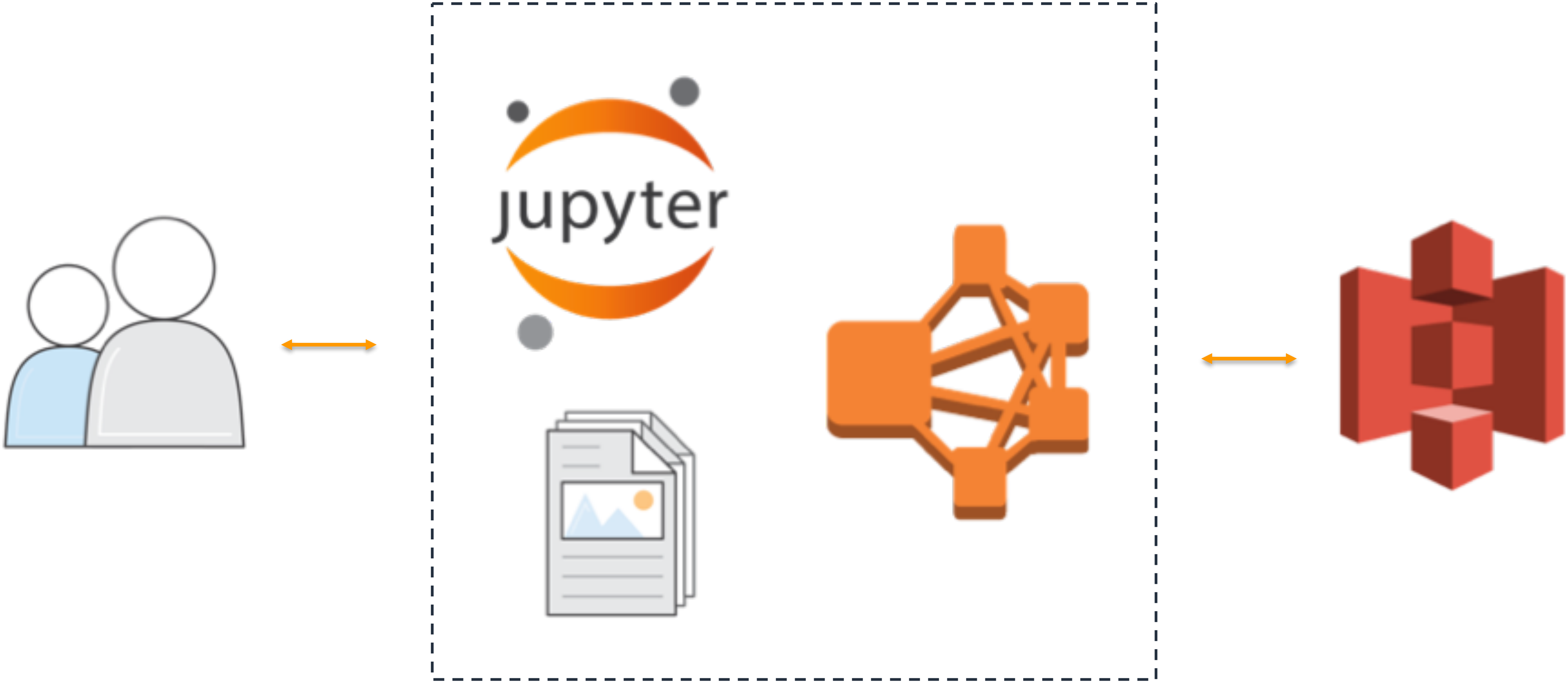
Active Listings (Kinesis / S3)

Zestimate



"Previously, given our scale limits using existing proprietary technology, it could take us an entire day or longer to compute a Zestimate," says [Jasjeet] Thind [VP of Data Science and Engineering]. "Now, **we can do it in hours nationwide using Spark on Amazon EMR**, which enables us to quickly perform calculations in parallel on multiple machines."

Ad hoc environment



Scale cluster to accommodate more users

Customer Story - DataXu

dataxu

DataXu spun out of MIT Labs to form a PB-scale digital marketing platform.

Challenge:

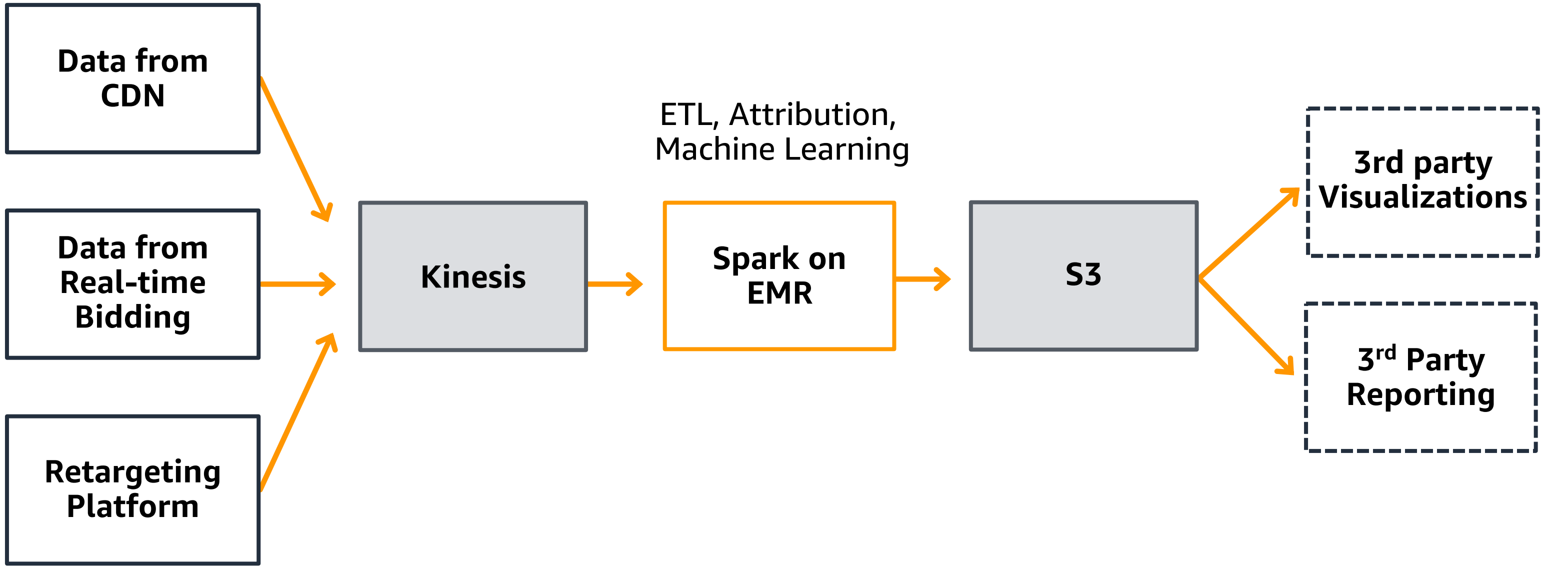
Deal with billions of impressions, PBs of data. Needed to help world's most valuable brands understand and engage with their consumers (doing real-time advertisement bidding).

Solution:

- Use Spark on EMR and Kinesis to stream and process real-time data from their bidding and retargeting platform
- Land all data in S3 data lake to serve up reports and visualizations

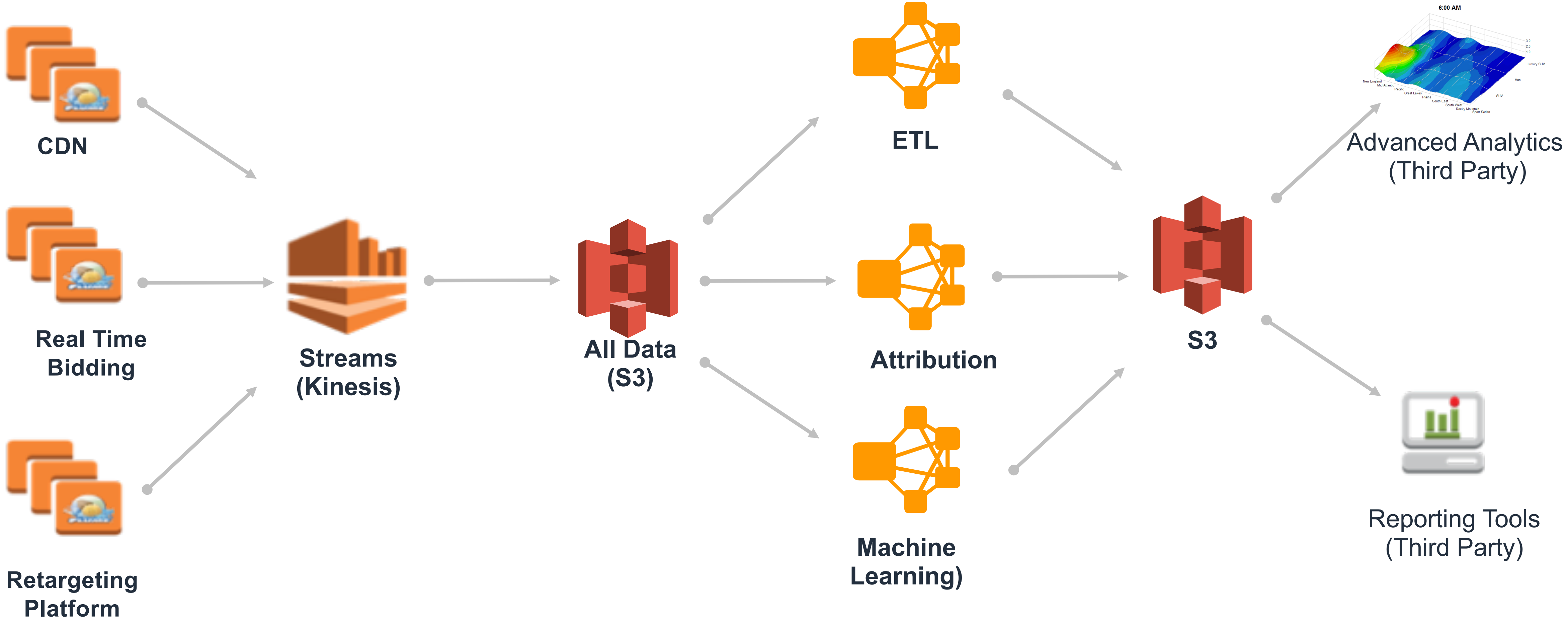
DataXu uses AWS for real-time analytics

dataxu



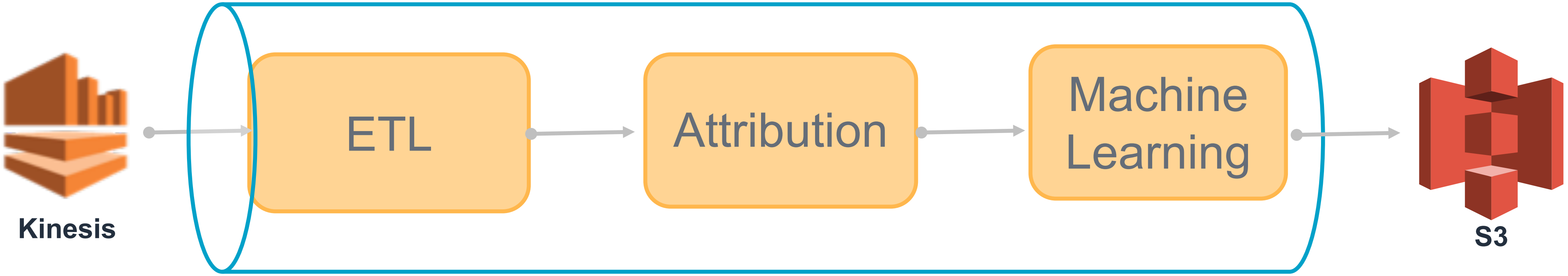
- Stream real-time data from CDN, their real-time bidding platform, and retargeting platform
- Spark on EMR does ETL, attribution, and Machine Learning
- Land data in S3 data lake
- Use third party data visualizations and reporting

DataXu Flows



Ecosystem of tools and services

DataXu Spark Pipeline



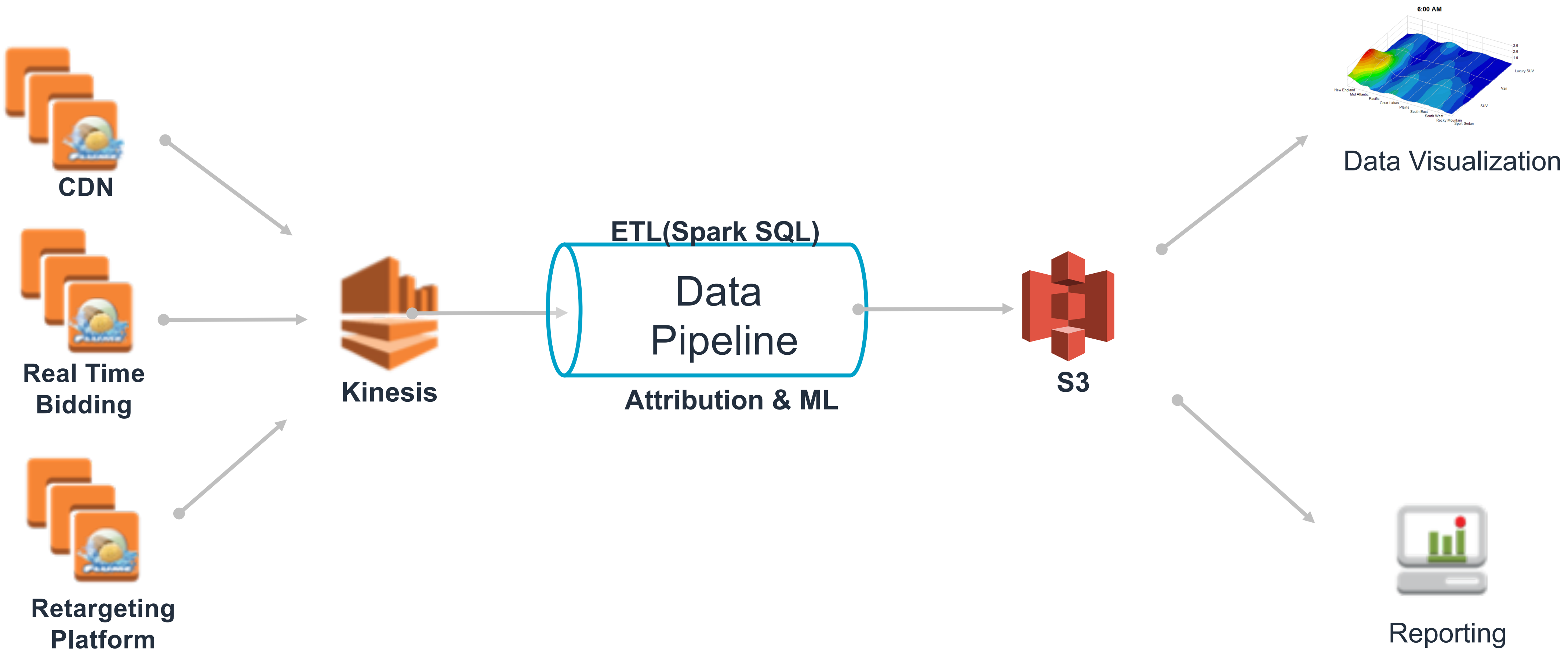
Streaming

Spark SQL

Spark MLib

Parquet, DataSets, DataFrames
Common Backend
Amazon EMR

DataXu Flows – New Generation



CDN

Real Time Bidding

Retargeting Platform

Kinesis

ETL(Spark SQL)

Data Pipeline

Attribution & ML

S3

Data Visualization

Reporting

AWS

Ecosystem of tools and services

EMR powers the most cloud Hadoop & Spark projects

FINRA
NETFLIX
Expedia
Nasdaq
REDFIN
kik
Pfizer
Avalara
bazaarvoice:
bloomreach
Bristol-Myers Squibb Company
coffee meets bagel
QUALITY DATING MADE EASY
Discovery COMMUNICATIONS
drive.ai
ENERNOC
Eisai
GRUPO BIMBO
HARRY'S
JIMDO
KOCH INDUSTRIES INC
MATOMY MEDIA GROUP
move
Nextdoor
orderbird.
patsnap
tinder
Ringier
ROVIO
SAMSUNG
SCOPELY
smartsheet
splunk>
Sysco
Tapjoy
transurban
trulia
UberMedia
wex
yelp
ACTIVISION BLIZZARD
CROWDSTRIKE
JustGiving
QANTAS
ROVIO
Burt.
alphasense
AdRoll
euclid

Thank you!