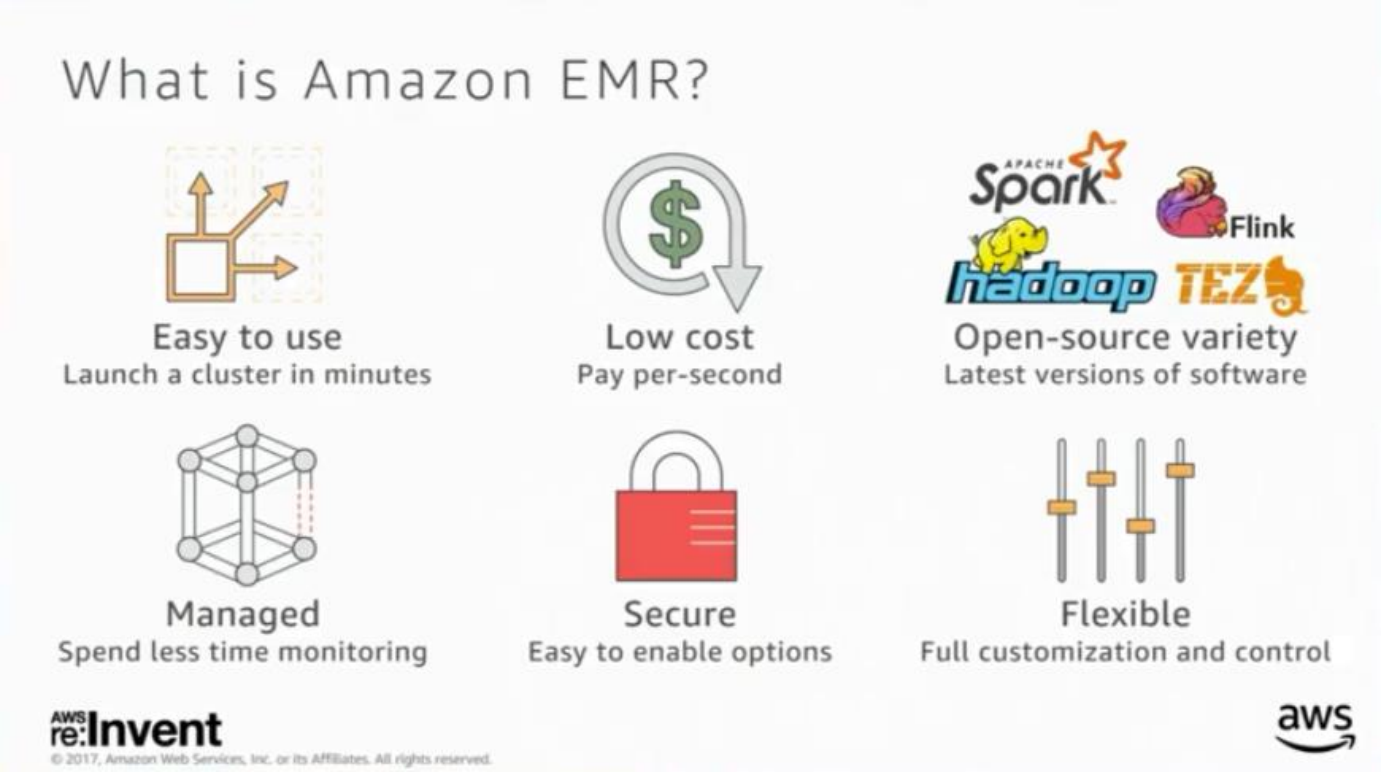
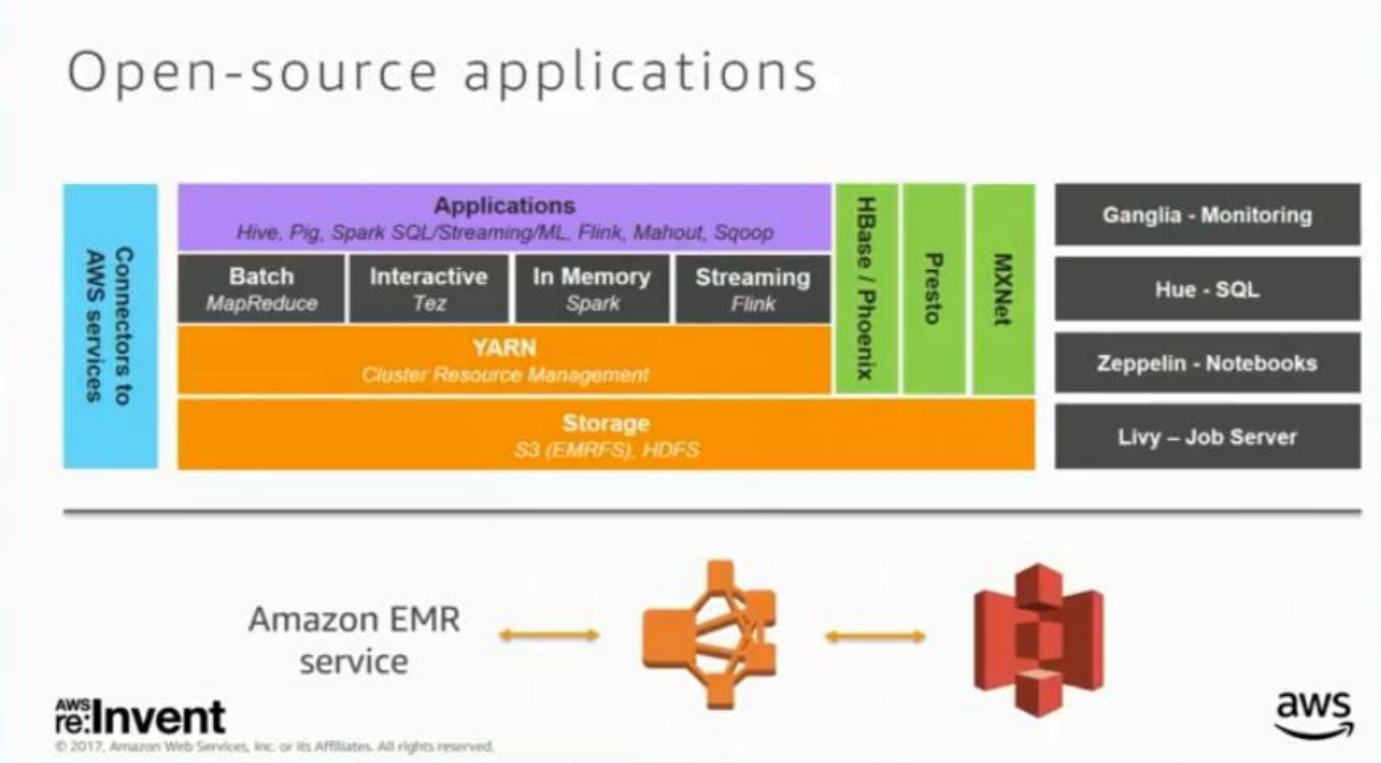
Amazon EMR is one of the largest Hadoop operators in the world, enabling customers to run ETL, machine learning, real-time processing, data science, and low-latency SQL at petabyte scale. In this session, we introduce you to Amazon EMR design patterns such as using Amazon S3 instead of HDFS, taking advantage of both long and short-lived clusters, and other Amazon EMR architectural best practices. We talk about lowering cost with Auto Scaling and Spot Instances, and security best practices for encryption and fine-grained access control. Finally, we dive into some of our recent launches to keep you current on our latest features.

Overview

- Intro and architectures
- Using Amazon EC2 Spot and Auto Scaling
- Security overview
- Ad-hoc and advanced workflows
- Apache Spark and Amazon EMR at Salesforce

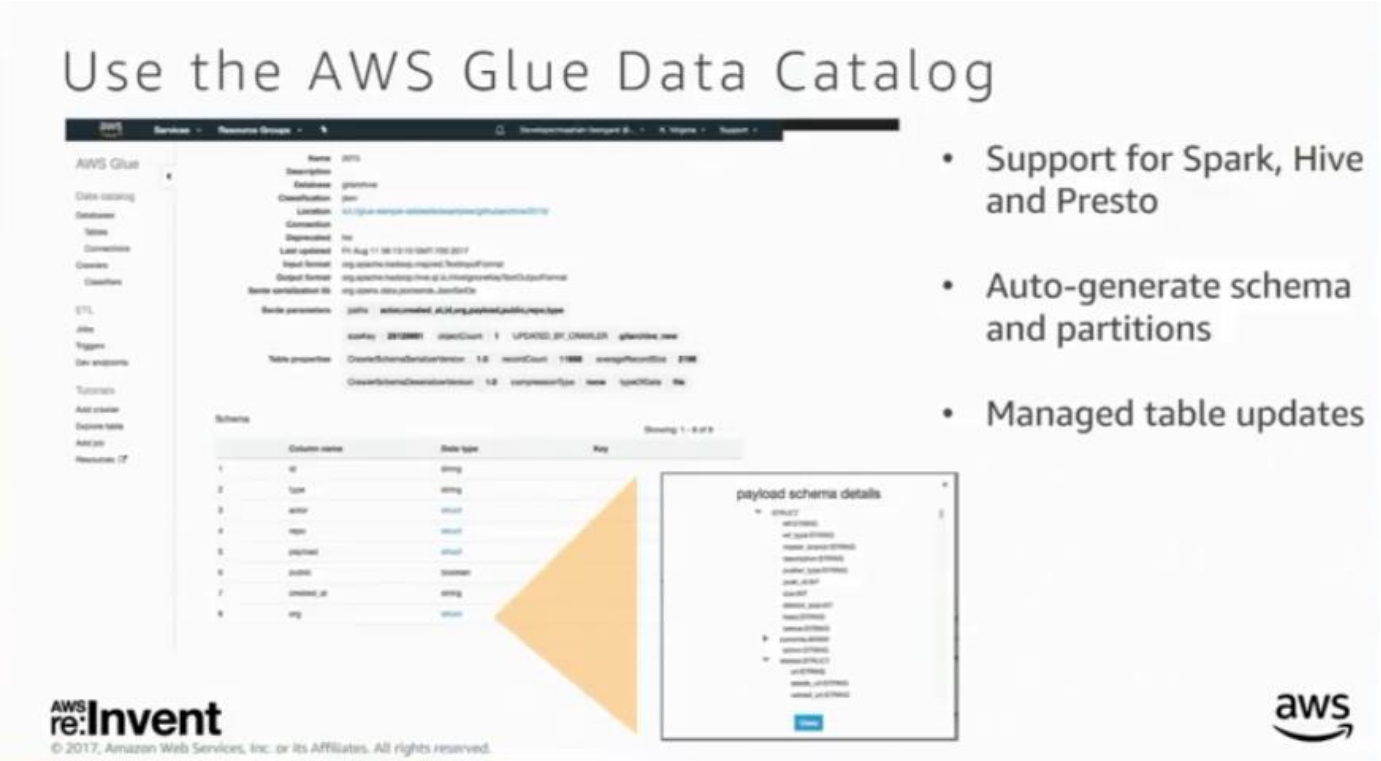


EMR allows you to launch from 1 – 1000s of Hadoop nodes in a cluster and spin them up for your use on AWS easily and start writing applications within the cluster, EMR is fully managed and now uses per second billing and has a variety of about 19 OSS projects that you can use like Spark, you can also change the default configurations etc.

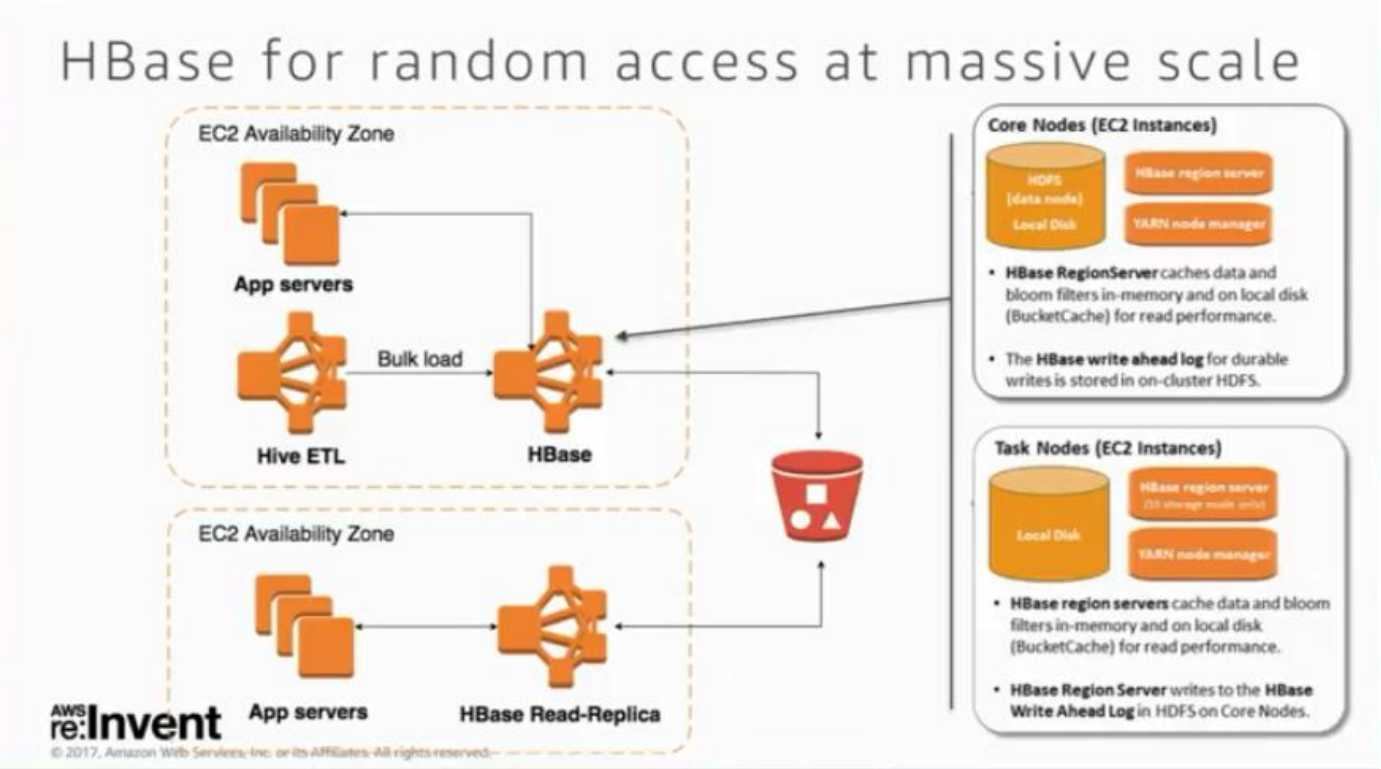


EMR is organized as above, a bunch of EC2 instances will spin up in your account and often times put processed data in S3. There is a variety of cluster management options like YARN that can run MapReduce, Spark, Tez, Flink, HBase and Phoenix. You can run NoSQL clusters, you can use Presto which is a distributed, low-latency SQL engine, MXNet for distributed training, a variety of frontend tools like Ganglia, Zeppelin, Hue SQL editors, also connectors to a variety of AWS services like using Spark to directly load Redshift using the Redshift Spark connector that uses the UNLOAD

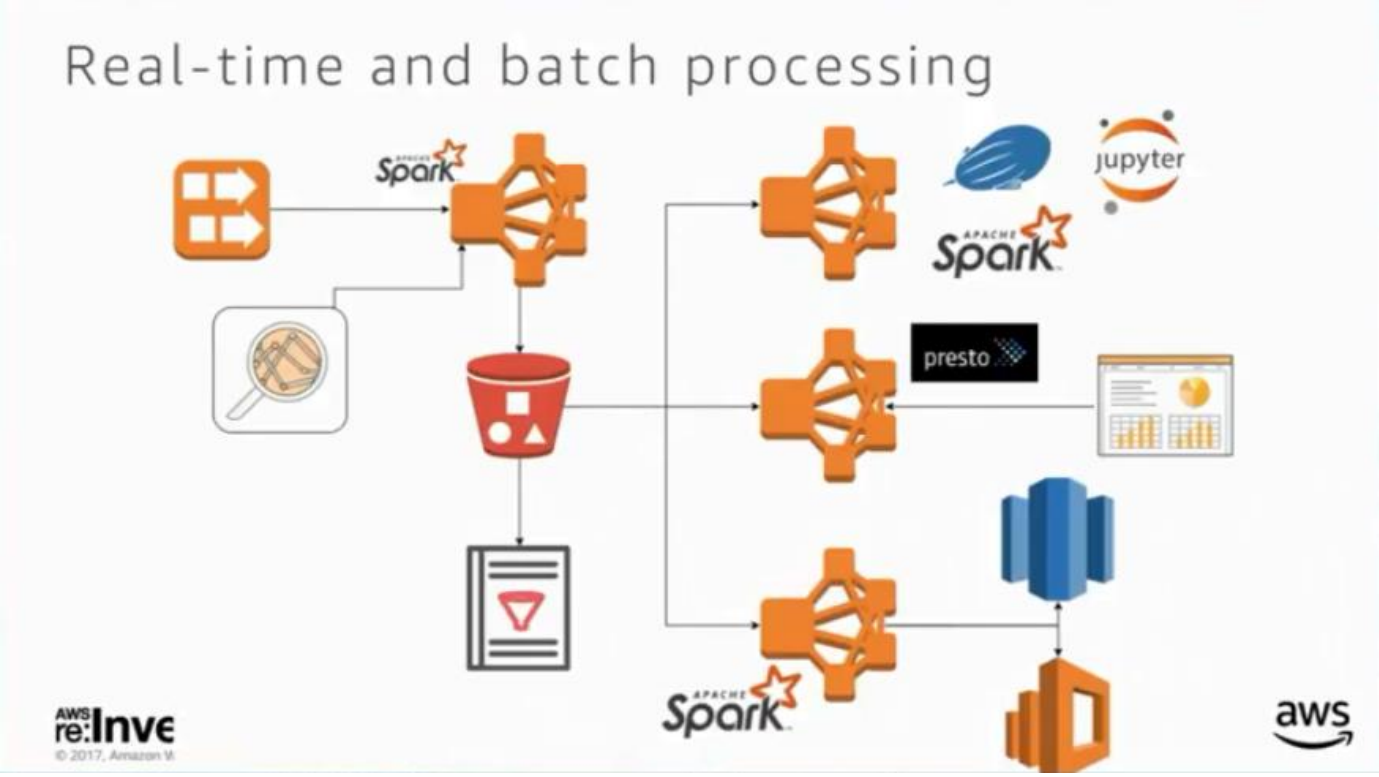
command for Redshift under the hood to get really good throughput to S3, you can use our Hive-DynamoDB connector to query and do big data analytics on your DynamoDB tables, you can use Sqoop to access data in your MySQL databases, etc.

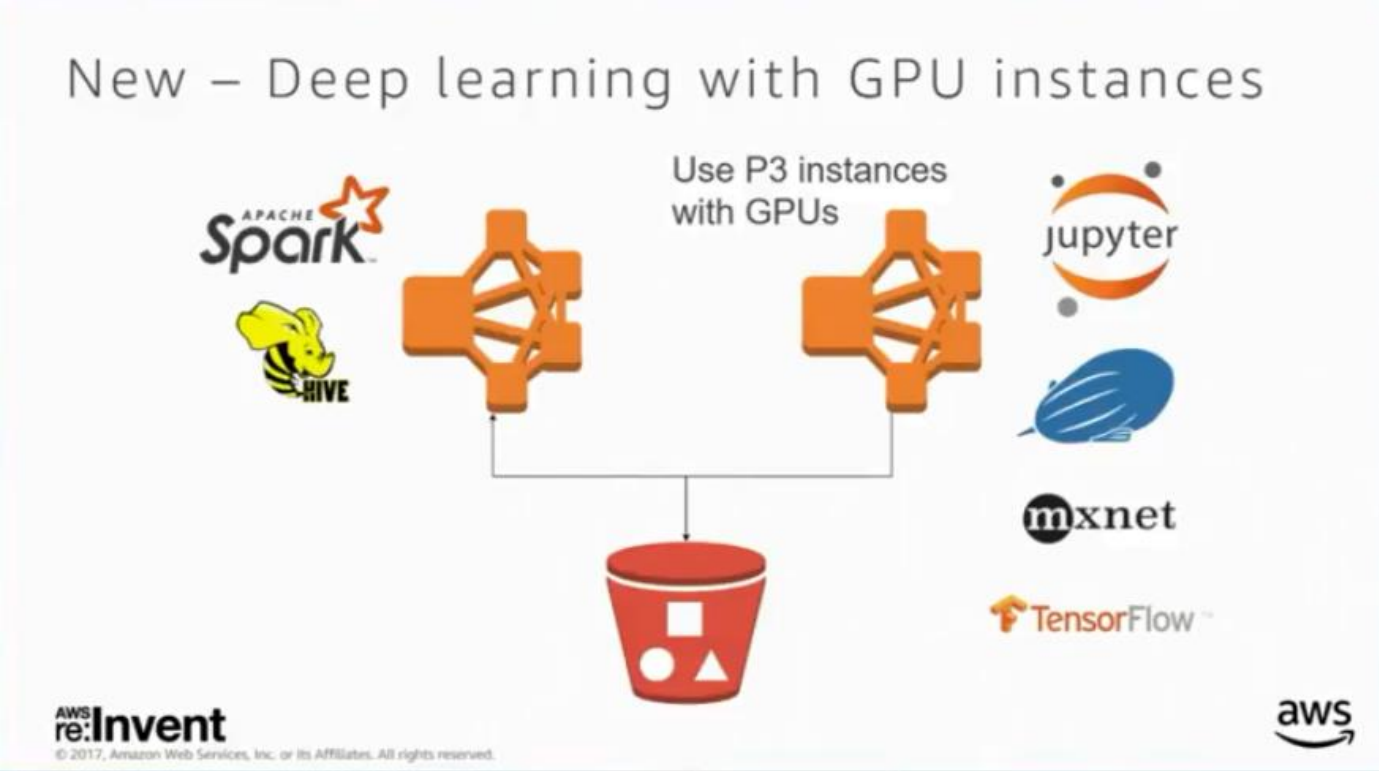


AWS Glue is a new connector that you can use.



A lot of customers run HBase using HDFS as the underlying file system, now you can replace HDFS with S3 as used by FINRA.





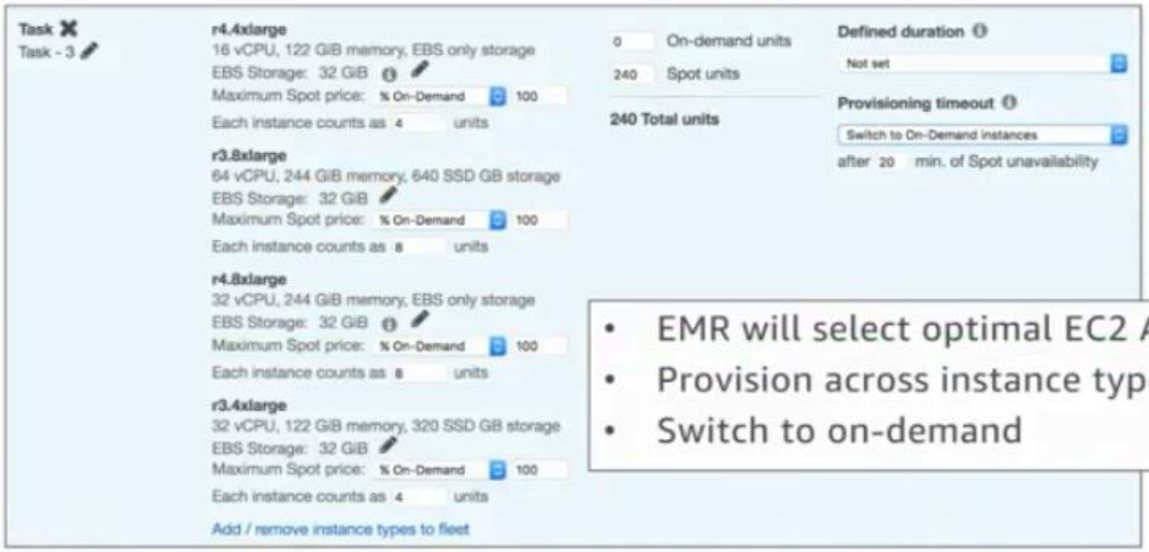
Tips to

Lower your costs

Transient or long-running clusters

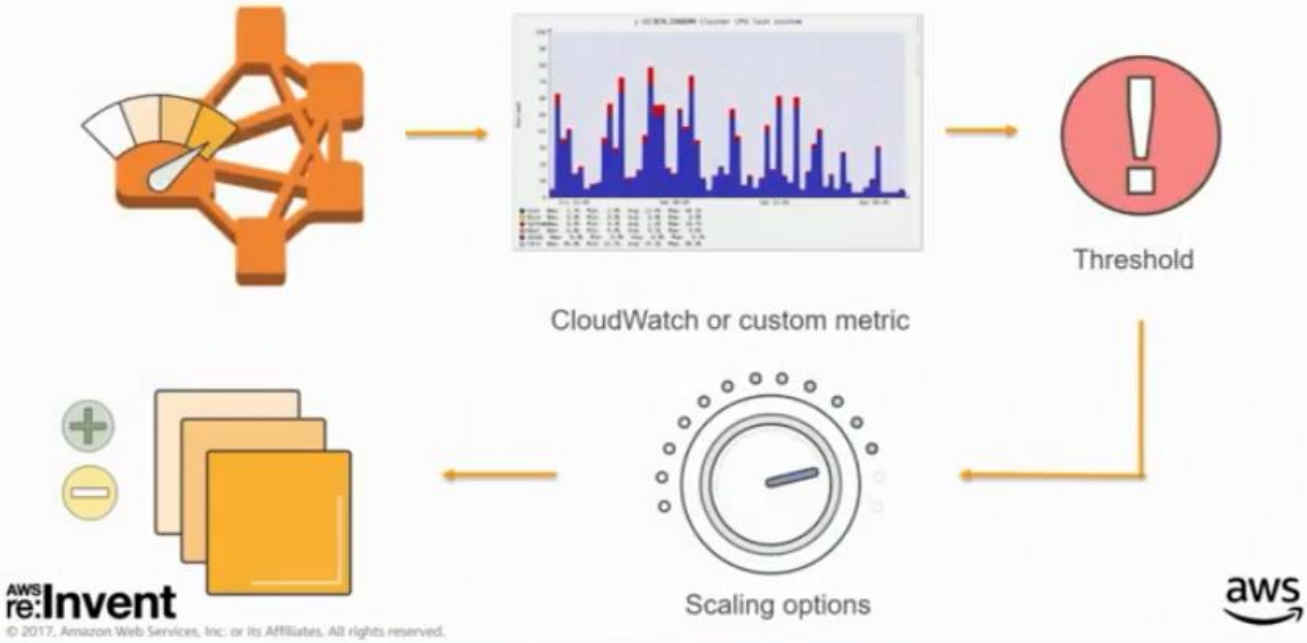
- Amazon Linux AMI with preinstalled customizations for faster cluster creation
- Auto Scaling to minimize costs for long-running clusters

EC2 Spot and instance fleets



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

Use Auto Scaling

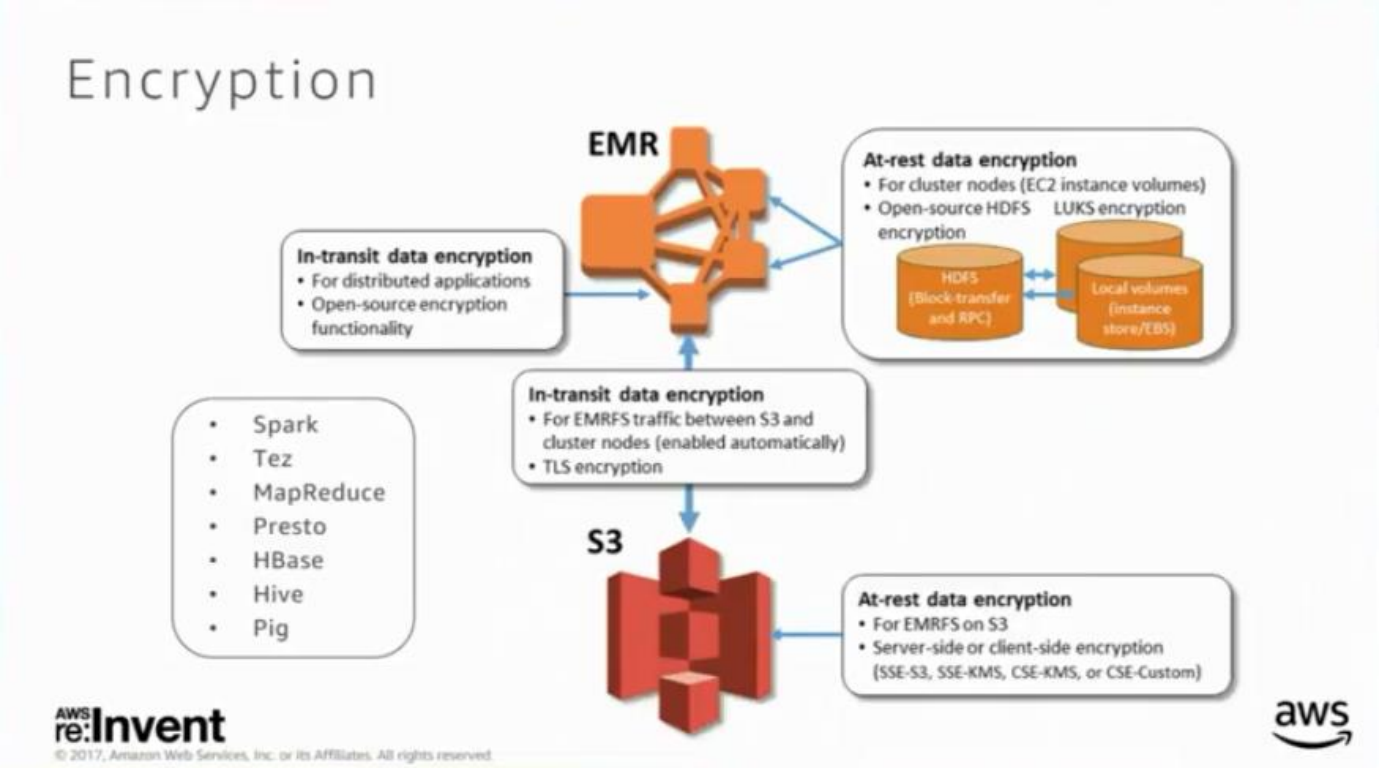


Auto Scaling

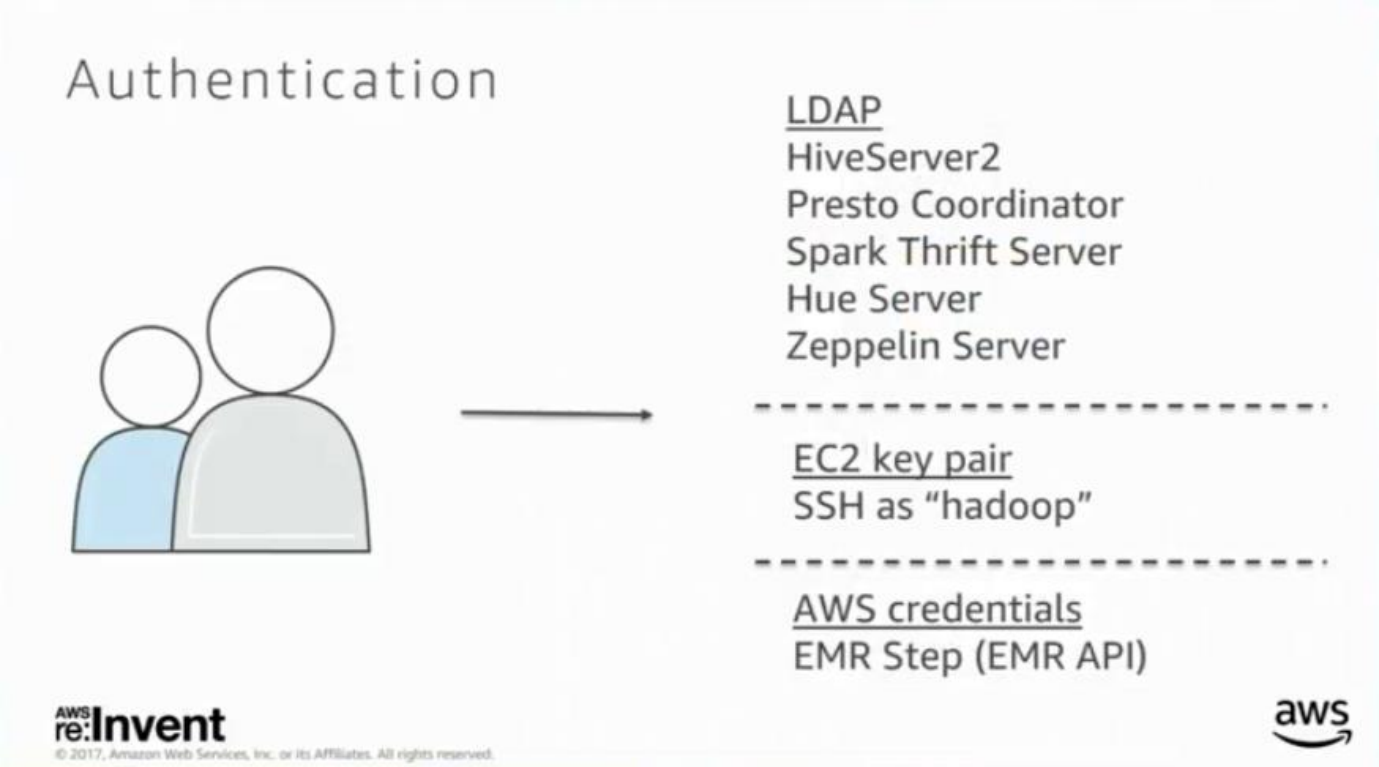
- EMR scales-in at YARN task completion
- Selectively removes nodes with no running tasks
- yarn.resourcemanager.decommissioning.timeout
    - Default timeout is one hour
- Spark scale-in contributions
    - Spark specific blacklisting of tasks
    - Unregistering cached data and shuffle blocks
    - Advanced error handling

Tips to

Secure your cluster

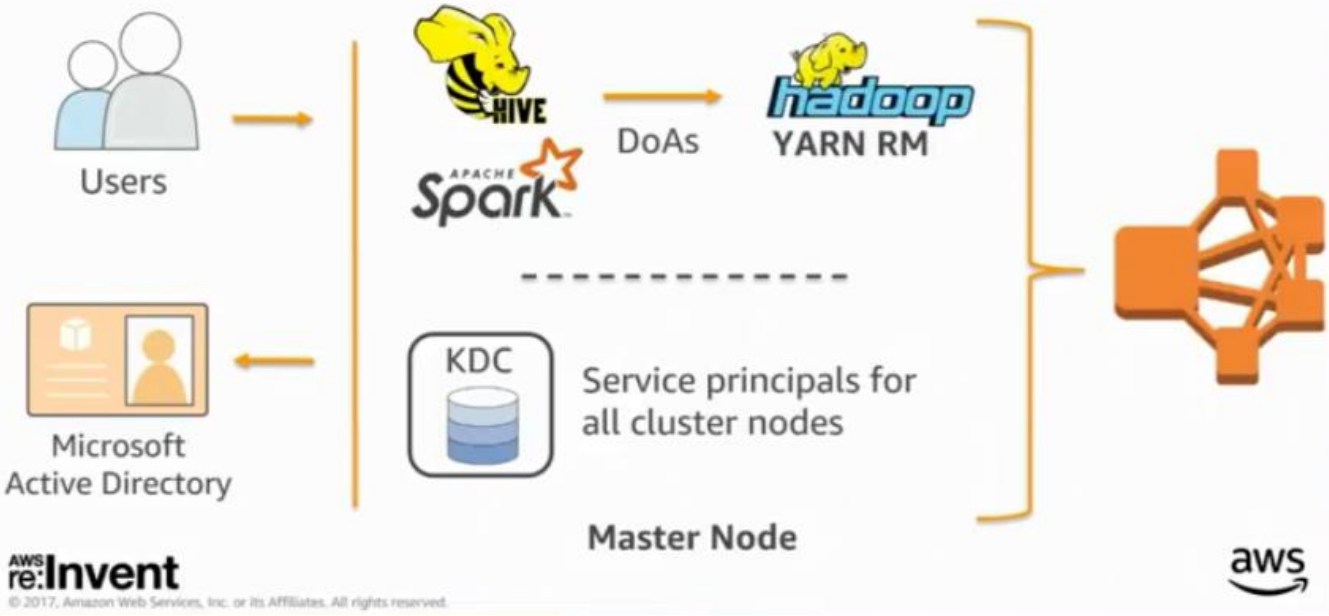


EMR supports end-to-end encryption for some plugins, S3, local disks of your cluster, etc.



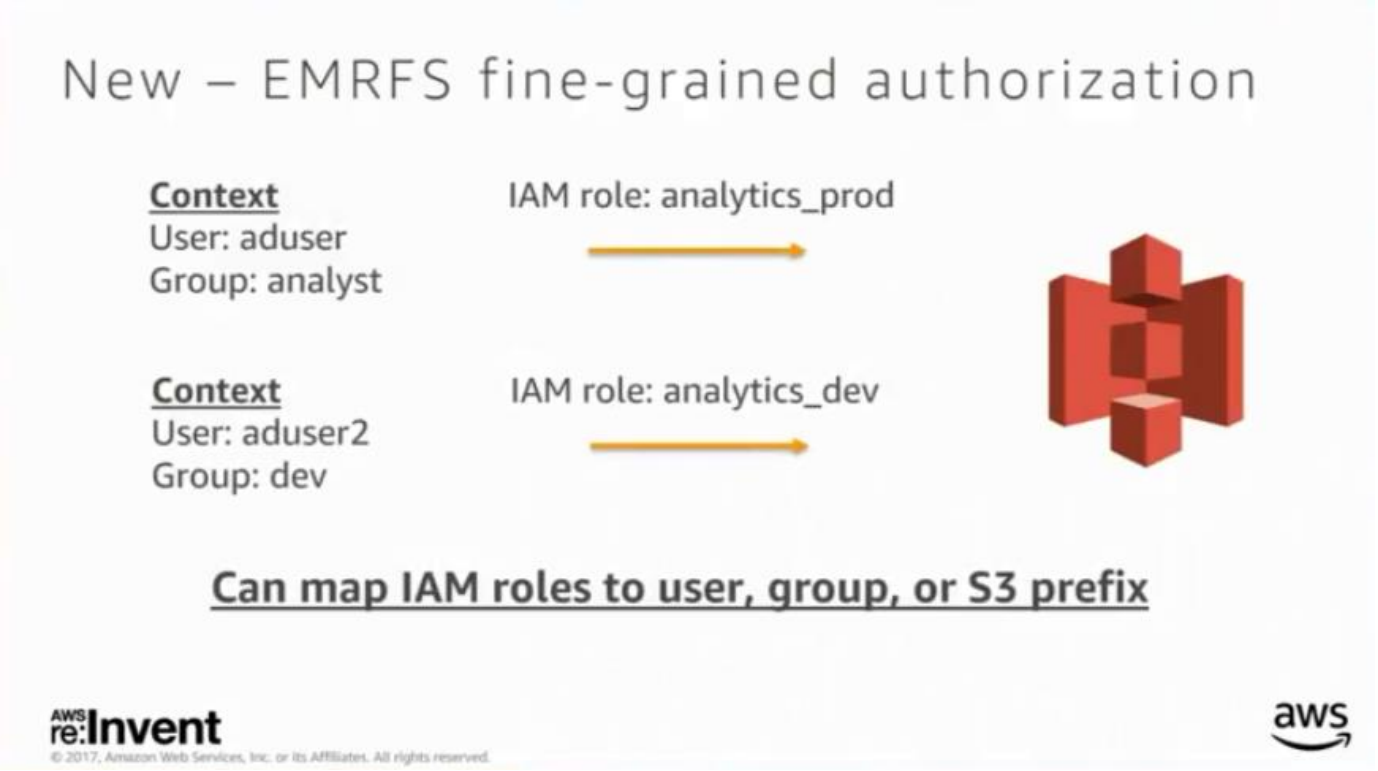
You can authenticate your users using the following, LDAP, EC2 and credentials

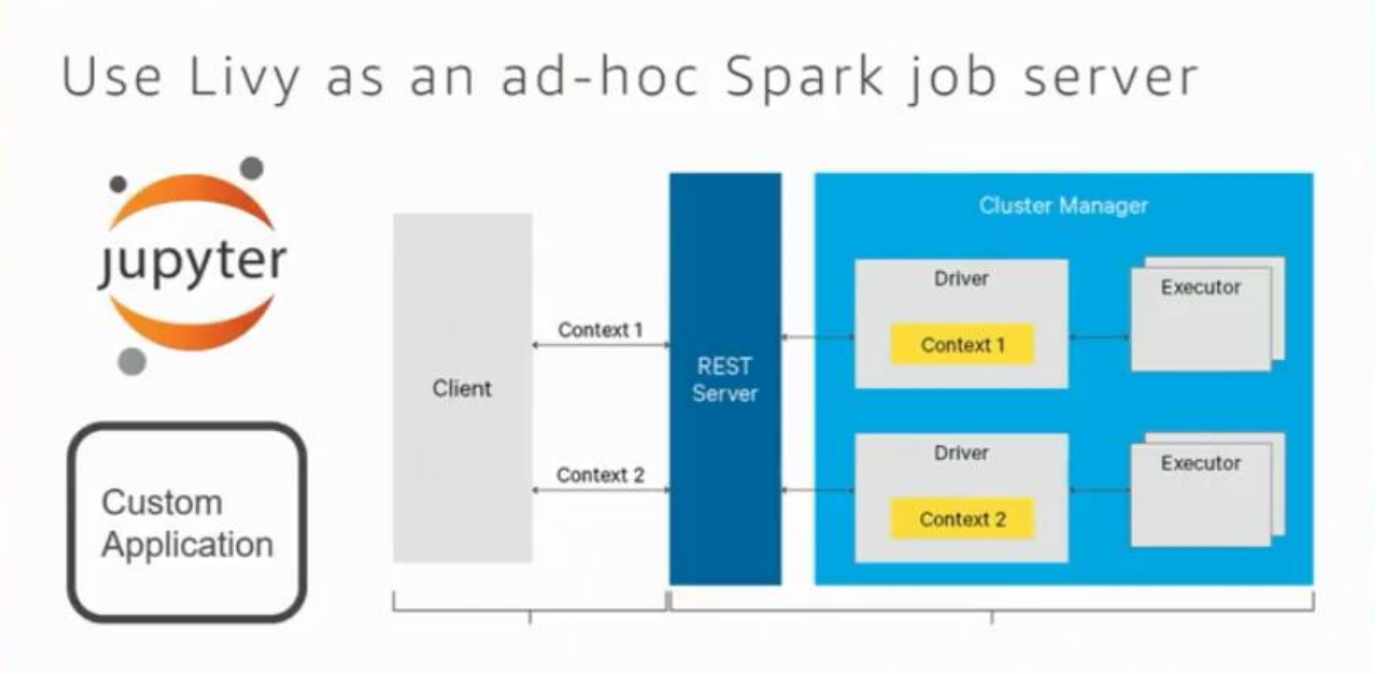
New – Authentication with Kerberos



Authorization

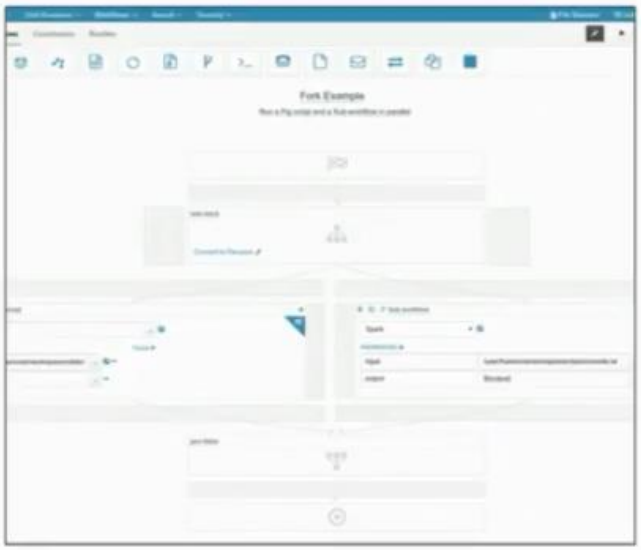
- Storage-based
  - EMRFS/S3
  - HDFS
- HiveServer2 and Presto (SQL-based)
- HBase
- YARN queues
- Fine-grained access control by cluster tag (IAM)
- Apache Ranger on edge node (using CloudFormation)





Livy is an HTTP endpoint that you can interact with in an ad-hoc way to create and manage Spark sessions for you that you can submit jobs to and use with your notebooks

Oozie and Airflow for DAGs of jobs



Select Amazon EMR customers

Apache Spark and Amazon EMR at Salesforce

Anya Bida, Senior Member of Technical Staff at Salesforce
abida@salesforce.com @anyabida1

Fastest Growing Top 5 Enterprise Software Company

"Innovator of the Decade"
Forbes
September 2016

FORTUNE 100 BEST COMPANIES TO WORK FOR
2009 • 2010 • 2011
2012 • 2013 • 2014
2015 • 2016 • 2017

Forbes
The world's most innovative companies
2011 • 2012 • 2013
2014 • 2015 • 2016 • 2017

Overview

Our Goal

Getting started with EMR

Spark primer

Monitor multiple viewpoints

Use AWS Identity and Access Management (IAM) roles

Isolate Environments

Complete ML Pipeline

ETL

Feature Engineering

Model Training

Model Evaluation

Deploy & Operationalize Models

Score & Update Models

Support Batch & Real Time

Selecting a tool

ETL

Feature Engineering

Model Training

Model Evaluation

Deploy & Operationalize Models

Score & Update Models

Support Batch & Real Time

- ✔ Supports each step of our ML pipeline
- ✔ Scales for small & large jobs
- ✔ Good ML Libraries
- ✔ Active user base
- ✔ Ability to deploy production ready code

APACHE Spark

We wanted Spark…now how to deploy it?

EC2

- Support for batch / streaming
- Integrates with our tooling
- Spin up / down clusters
- Larger / smaller clusters
- Support for different versions of Hadoop, Spark
- Storage & Compute options

Need: Management

Amazon EMR

Provision EMR

Simplest approach

Amazon EMR

Clone | Terminate | AWS CLI export

```
aws emr create-cluster
--applications Name=Hadoop
Name=Spark Name=Ganglia
--tags
--ec2-attributes
--release-label
--log-uri
```

```
...
--name
--instance-groups
--region
```

Provision EMR

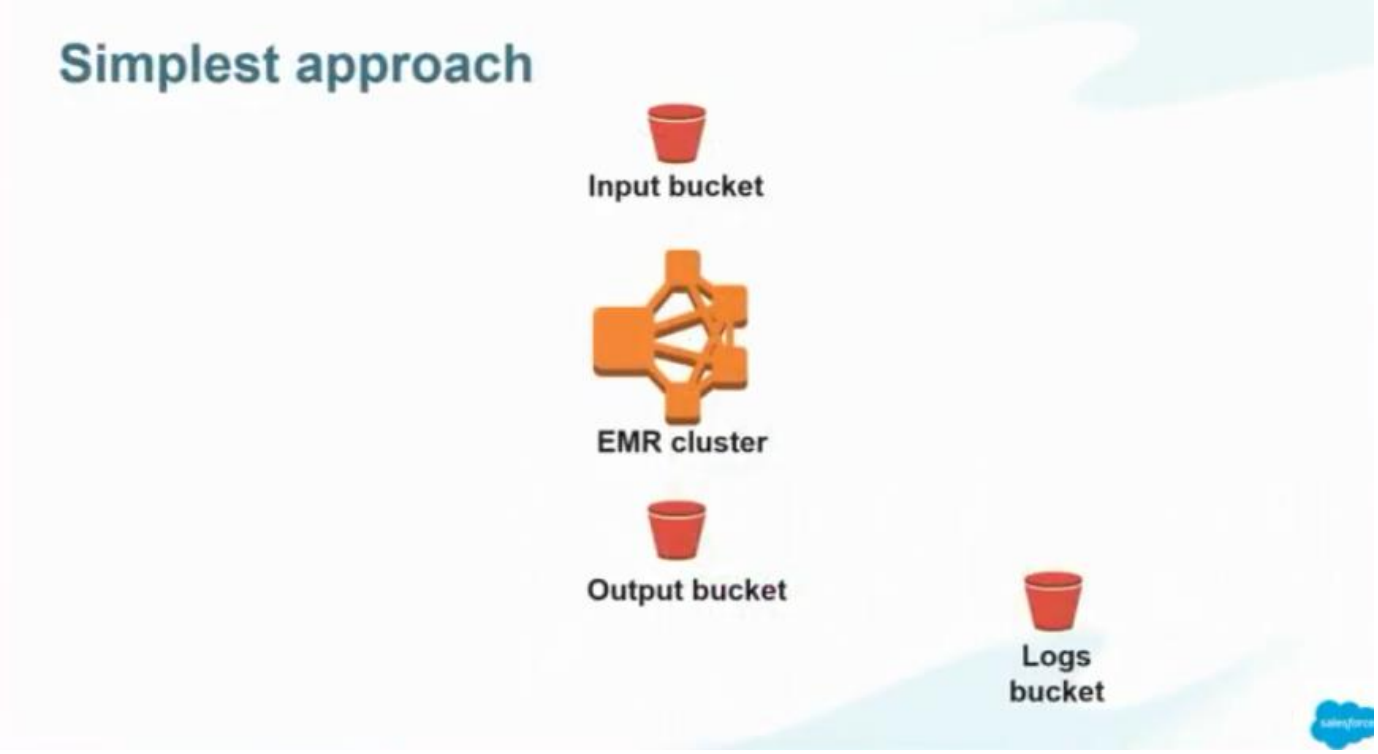
Simplest approach

Amazon EMR

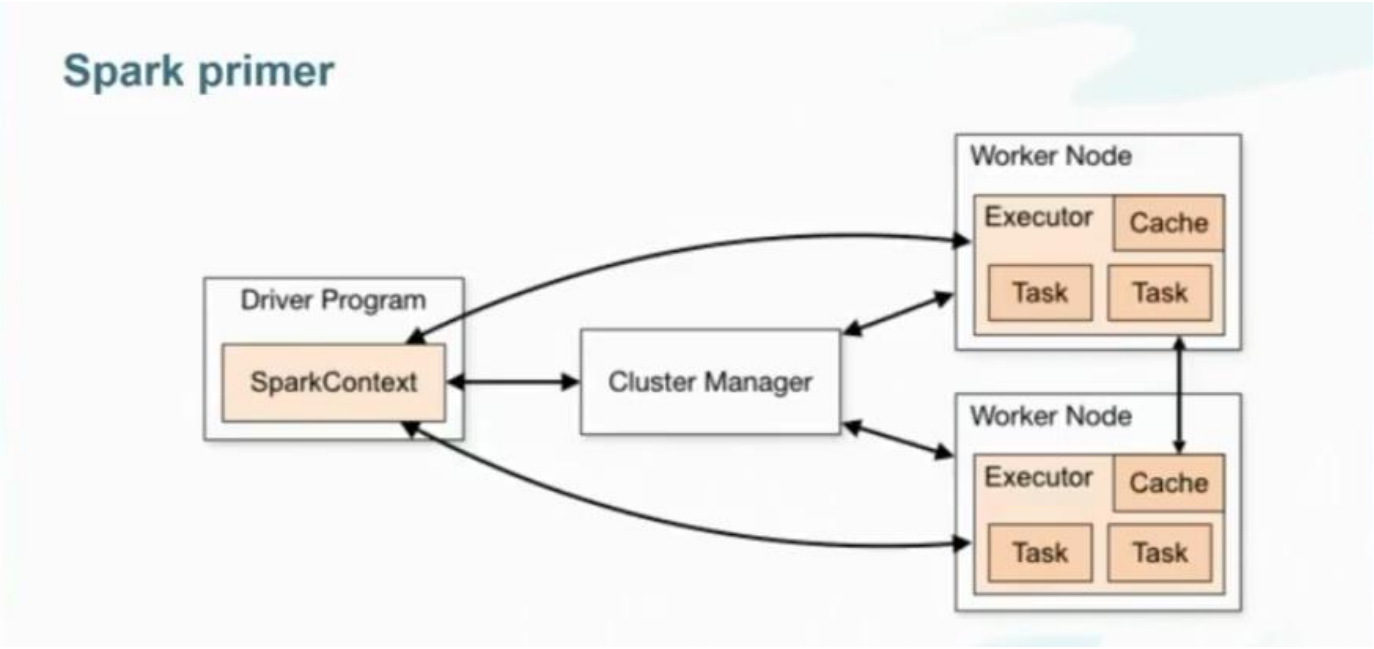
Clone | Terminate | AWS CLI export

```
aws emr create-cluster
--applications Name=Hadoop
Name=Spark Name=Ganglia

# tag for cost analysis by project
--tags
--ec2-attributes
--release-label

# send logs to S3
--log-uri
```

```
...
# use naming conventions for service
discovery
# <region>-<project>-<version>-<env>
--name

# CORE nodes used for writes to HDFS
# TASK nodes used for compute - try spot
instances here for starters
--instance-groups

--region
```

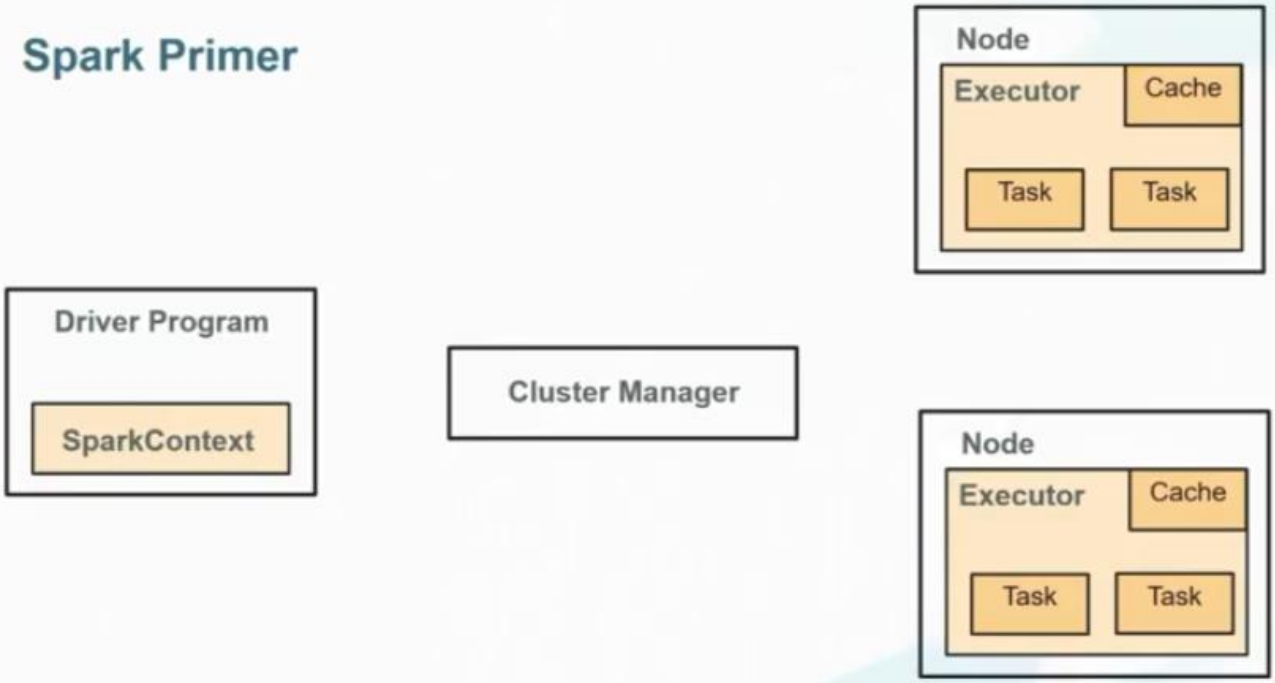


We have our data in the Input bucket in S3 and we will read data from there, then we will run the jobs in our EMR cluster and write to our Output bucket and send logs to the Logs bucket



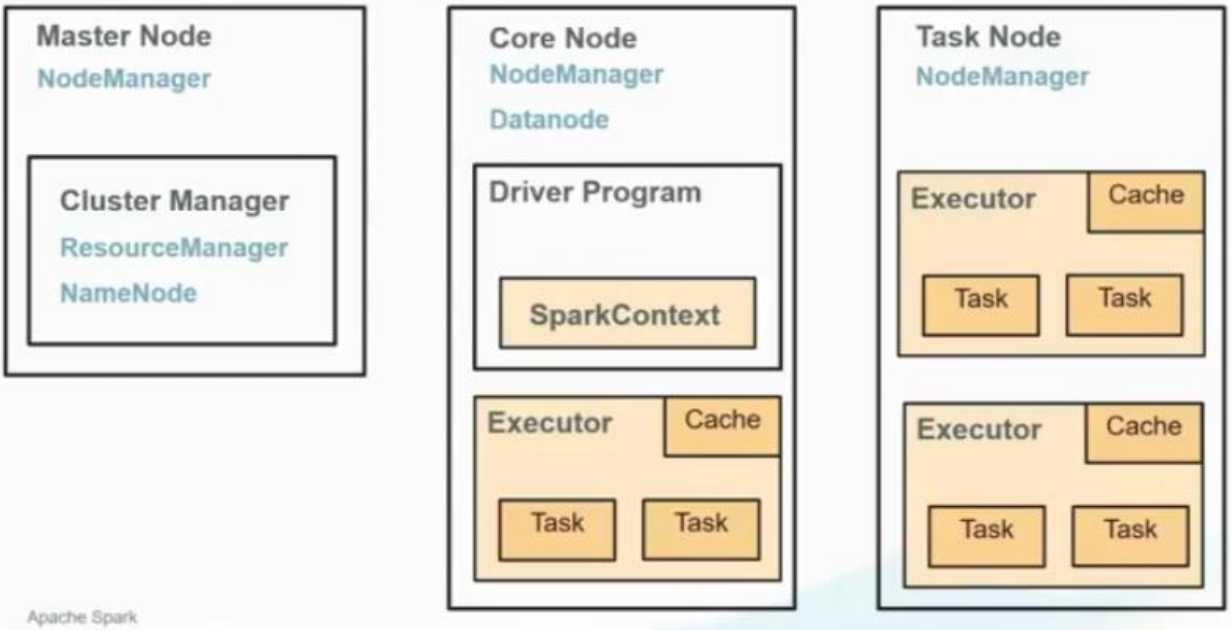
The Driver program is where we write the application user code, it will talk to the cluster manager to get some resources provisioned, the work gets run on the worker nodes

Spark Primer



Apache Spark

Spark on Amazon EMR



Apache Spark

Properties related to Dynamic Allocation

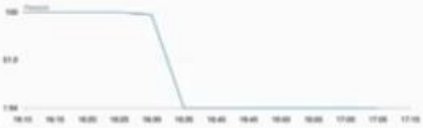
Property	Value
Spark.dynamicAllocation.enabled	true
Spark.shuffle.service.enabled	true
spark.dynamicAllocation.minExecutors	*5*
spark.dynamicAllocation.maxExecutors	*17*
spark.dynamicAllocation.initalExecutors	*0*
sparkdynamicAllocation.executorIdleTime	*60s*
spark.dynamicAllocation.schedulerBacklogTimeout	*5s*
spark.dynamicAllocation.sustainedSchedulerBacklogTimeout	*5s*

Optional

Where do I find Metrics?

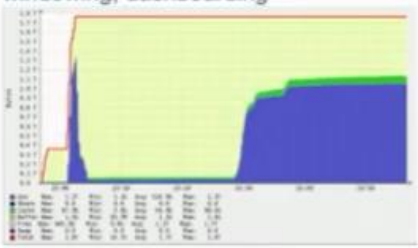
Cloudwatch

YARNMemoryAvailablePercentage



Ganglia

windowing, dashboarding



Logs?

Summary | Application history | Monitoring | Hardware | Events | Steps | Configurations | Bootstrap actio

YARN applications > application_1506102332083_0001 (Spark)

Jobs | Stages | Executors

Jobs > Job 0 > Stage 0 (attempt 0)

Total time across all tasks: 2.5 min

Locality level summary: Rack local: 7

Input (size / records): 402.7 MiB / 5,265,807

Shuffle write (size / records): 45.1 MiB / 938

Summary metrics for 7 completed tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	8 s	15 s	25 s	28 s	28 s
GC time	0.1 s	0.2 s	0.5 s	0.7 s	0.7 s
Input (size / records)	18.6 MiB / 264,344	64.0 MiB / 566,465	64.0 MiB / 905,146	64.0 MiB / 905,525	64.0 MiB / 906,751
Result serialization time				1 ms	1 ms
Shuffle write (size / records)	2.3 MiB / 119	2.8 MiB / 119	7.8 MiB / 140	8.3 MiB / 140	8.6 MiB / 140
Task deserialization time	4 ms	7 ms	0.4 s	0.4 s	0.4 s

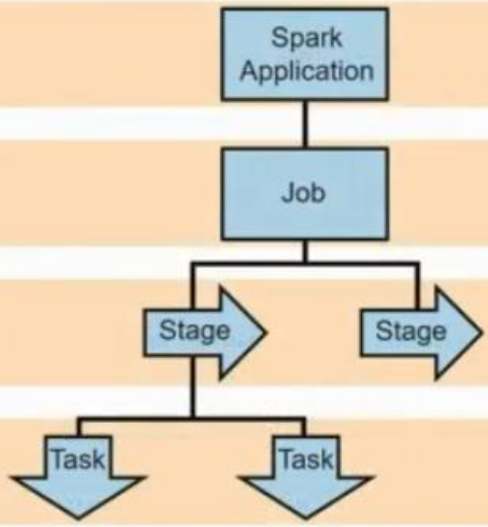
Anatomy of a Spark Job

Spark Context/Spark Session Object

Actions (e.g., collect, saveAsTextFile)

Wide transformations; (sort, groupByKey)

Computation to evaluate one partition
(combine narrow transforms)



High Performance Spark, Karau & Warren, O'Reilly

Simplest approach

Reads from S3
- Jar files too!

Write Intermediate files
- MEM or Disk?
- Local? HDFS? Amazon S3?

Writes to S3
- Data available after cluster is terminated

RDD Re-use

Persist to improve speed, Checkpoint to improve fault tolerance

	Cache	Persist	Checkpoint	Local Checkpoint
local mem cache	MEM	MEM		MEM
local disk		DISK		DISK
HDFS / S3			Specify dir	
If exec is decommed, are writes available?	No	No	Yes	No
If job finishes are writes available?	No	No	Yes	No
Preserve lineage graph?	Yes	Yes	No	No

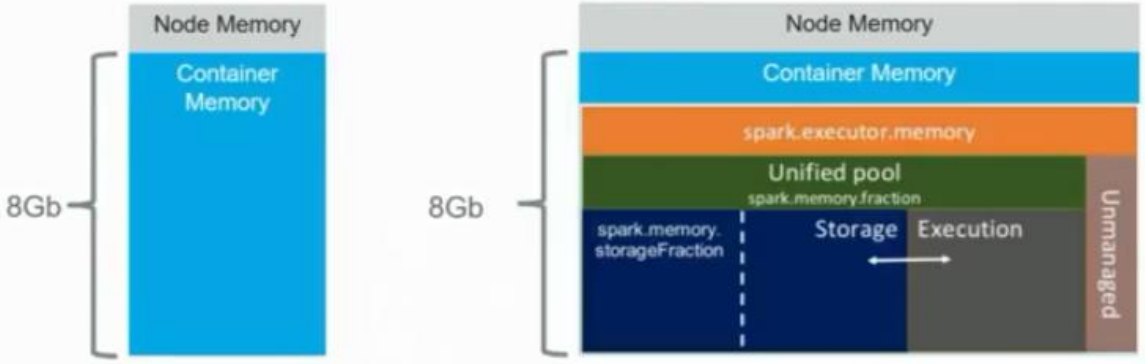
Overview

- ✔ Our Goal
- ✔ Getting started with EMR
- ✔ Spark primer
- Monitor multiple viewpoints
- Use IAM Roles
- Isolate Environments

Monitor multiple viewpoints

https://light.co/camera

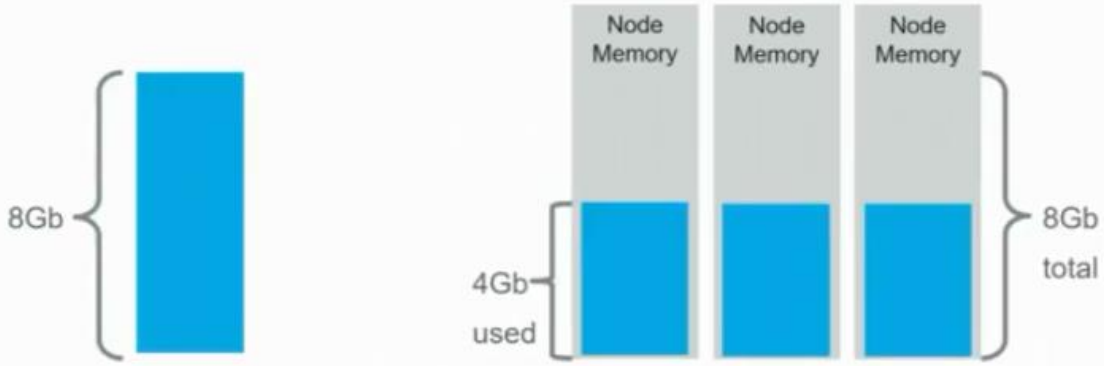
Understand resource allocation



Understanding Memory Management in Spark For Fun And Profit Shivnath Babu (Duke University, Unravel Data Systems)
Mayuresh Kunjir (Duke University)

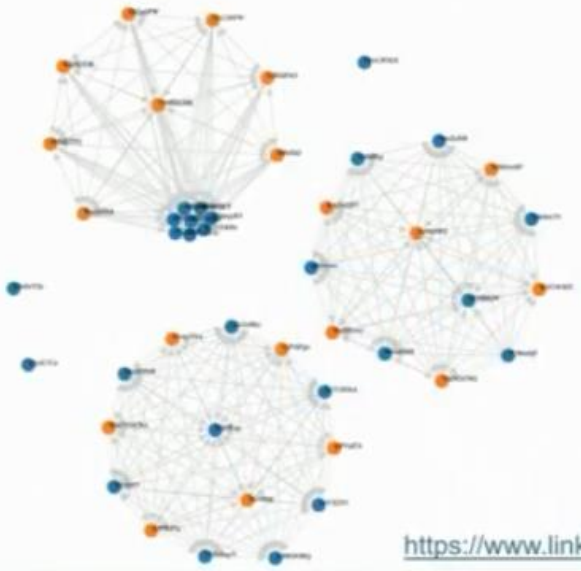
Can my 8Gb container launch on this cluster?

Scale-out Rule: Num Containers Pending



Monitor multiple viewpoints

Connectivity viewer



Vaibhav Tandon

https://www.linkedin.com/in/vaibhavt/

Monitor multiple viewpoints

Connectivity viewer

Vaibhav Tandon

https://www.linkedin.com/in/vaibhavt/

Monitor multiple viewpoints

Connectivity viewer

Vaibhav Tandon

https://www.linkedin.com/in/vaibhavt/

Overview

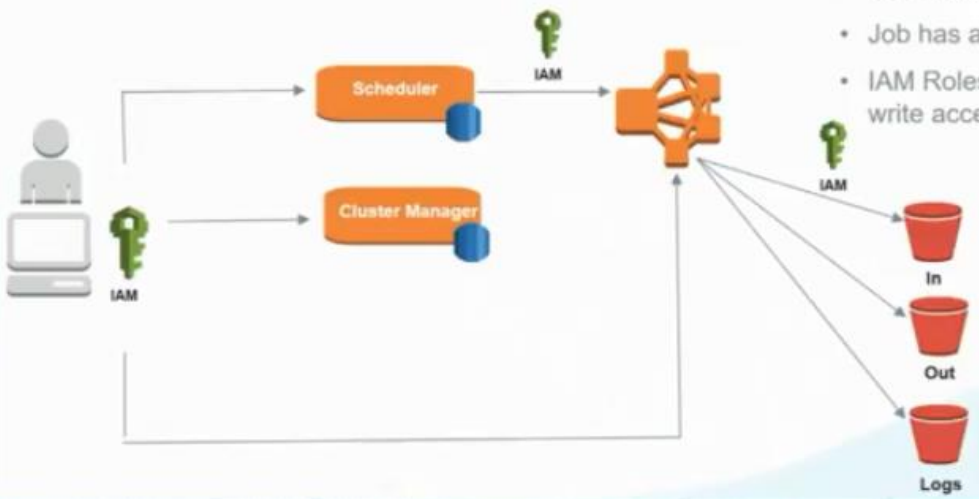
- ✔ Our Goal
- ✔ Getting started with EMR
- ✔ Spark primer
- ✔ Monitor multiple viewpoints
- ✔ Use IAM Roles
- Isolate Environments

Use IAM roles

Every user, service, & job should have specific, auditable permissions.
New: EMRFS fine-grained access control!!

IAM Roles

- User has an IAM Role
- Job has an IAM Role
- IAM Roles determine read / write access to data



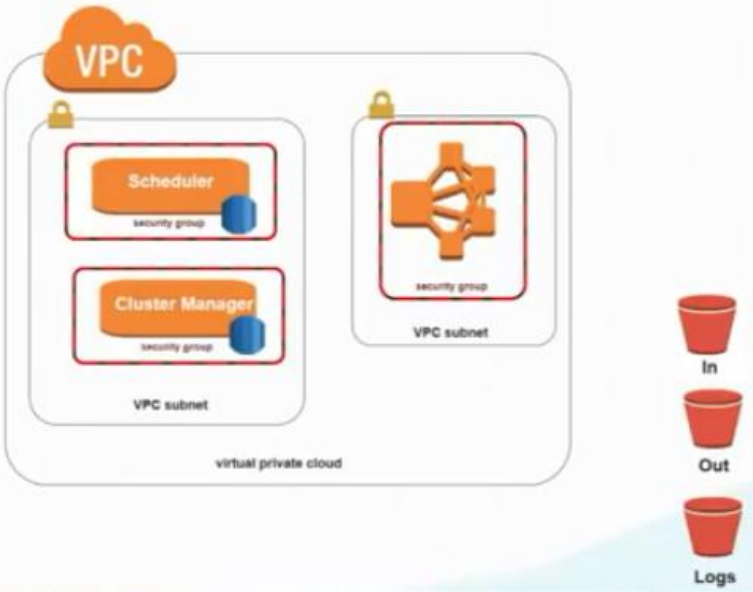

```
aws emr create-cluster
...
--service-role
```

Overview

- ✔ Our Goal
- ✔ Getting started with EMR
- ✔ Spark primer
- ✔ Monitor multiple viewpoints
- ✔ Use IAM Roles
- Isolate Environments

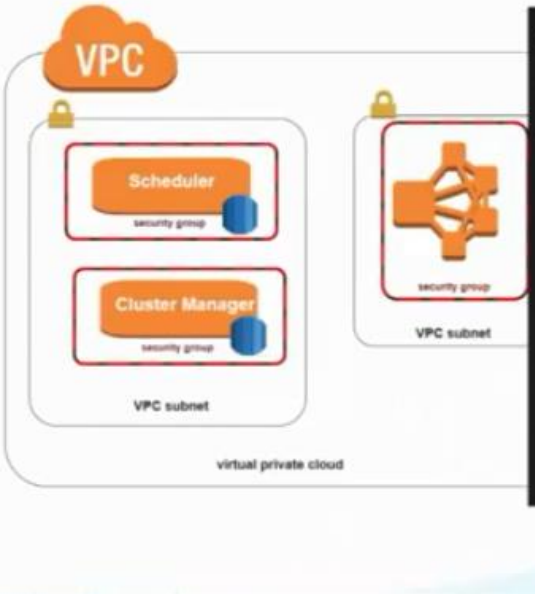
Isolate environments

Need: Build and release? Multitenancy?

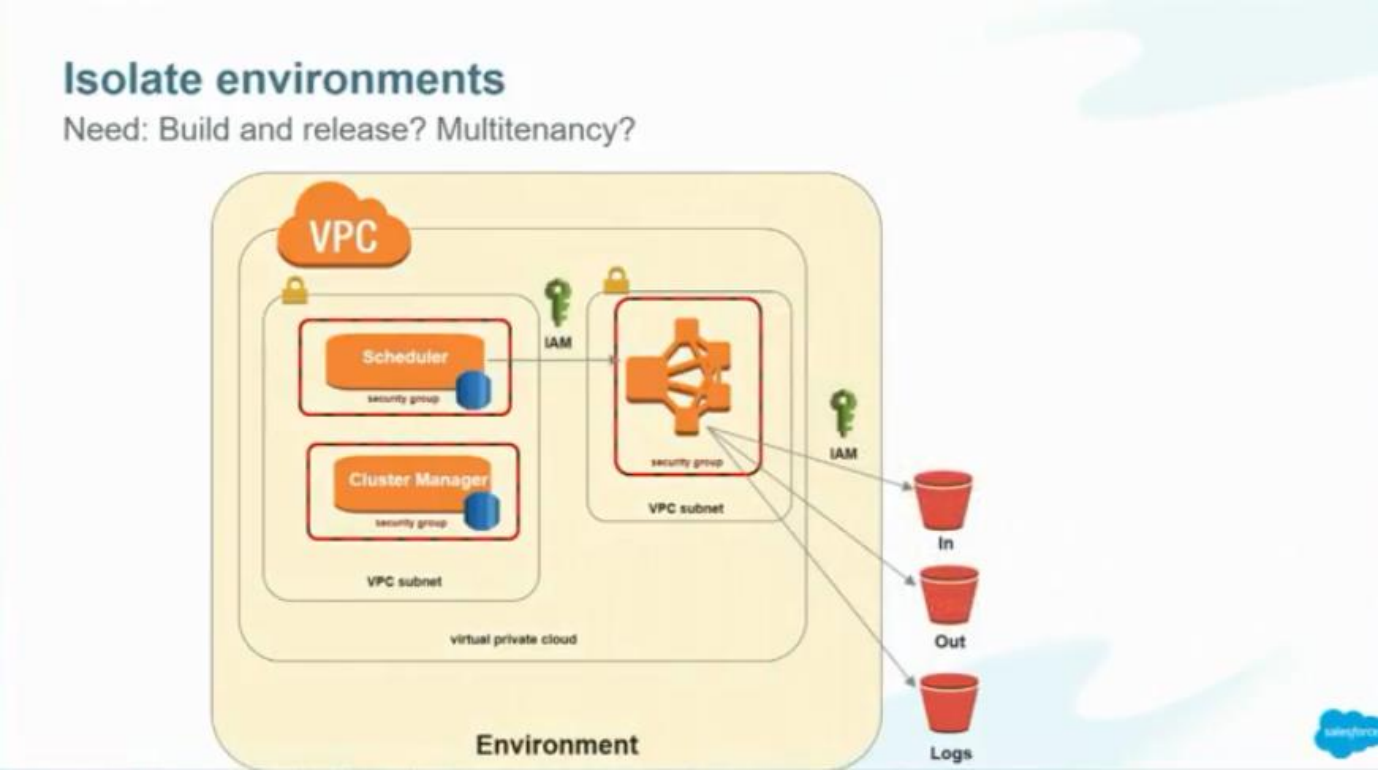


Isolate environments

Need: Build and release? Multitenancy?



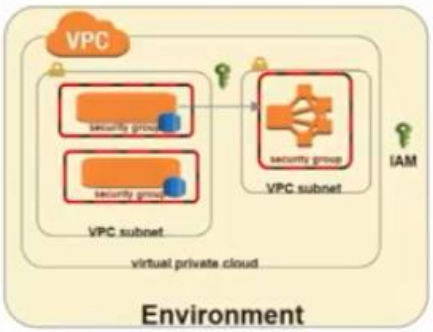

```
aws emr create-cluster
...
--ec2-attributes '{
    "KeyName":"",
    "InstanceProfile":"",
    "ServiceAccessSecurityGroup":"",
    "SubnetId":"",
    "EmrManagedSlaveSecurityGroup":"",
    "EmrManagedMasterSecurityGroup":""}'}
```



We need to templatize this environment for each environment we have

Isolate environments

Need: Build and release? Multitenancy?

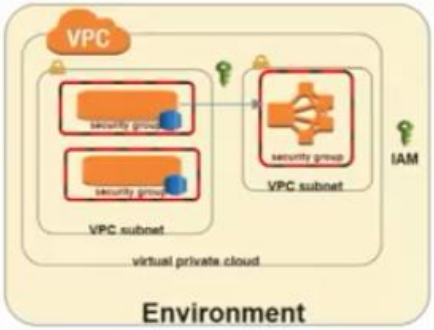


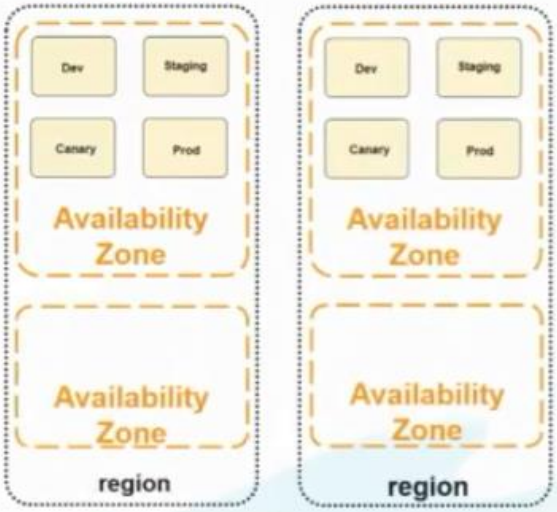
Automation

- Use Cloudformation or Terraform
- Upgrades use the same provisioning script + DNS Upsert

Isolate environments

Need: Build and release? Multitenancy?





Overview

- ✔ Our Goal
- ✔ Getting started with EMR
- ✔ Spark primer
- ✔ Monitor multiple viewpoints
- ✔ Use IAM Roles
- ✔ Isolate Environments

AWS re:Invent

THANK YOU!

Jonathan Fritz – jonfritz@amazon.com
Anya Bida – abida@salesforce.com

aws.amazon.com/emr
aws.amazon.com/blogs/big-data
aws.amazon.com/blogs/ai

AWS re:Invent

aws