amazon
web services

Masterclass

Amazon
Elastic MapReduce

Ian Massingham — Technical Evangelist

ianmas@amazon.com

@IanMmmm

intel

Masterclass

1 A technical deep dive that goes beyond the basics

2 Intended to educate you on how to get the best from AWS services

3 Show you how things work and how to get things done

Amazon Elastic MapReduce

Provides a managed Hadoop framework

Quickly & cost-effectively process vast amounts of data

Makes it easy, fast & cost-effective for you to process data

Run other popular distributed frameworks such as Spark

Low Cost

Easy to Use

Elastic

Amazon EMR

Flexible

Reliable

Secure

Amazon EMR: Example Use Cases

Clickstream Analysis

Amazon EMR can be used to analyze click stream data in order to segment users and understand user preferences. Advertisers can also analyze click streams and advertising impression logs to deliver more effective ads.

Genomics

Amazon EMR can be used to process vast amounts of genomic data and other large scientific data sets quickly and efficiently. Researchers can access genomic data hosted for free on AWS.

Log Processing

Amazon EMR can be used to process logs generated by web and mobile applications. Amazon EMR helps customers turn petabytes of un-structured or semi-structured data into useful insights about their applications or users.

Agenda

Hadoop Fundamentals

Core Features of Amazon EMR

How to Get Started with Amazon EMR

Supported Hadoop Tools

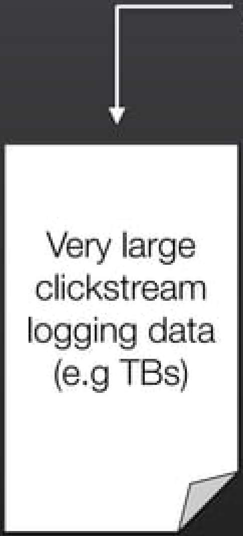
Additional EMR Features

Third Party Tools

Resources where you can learn more

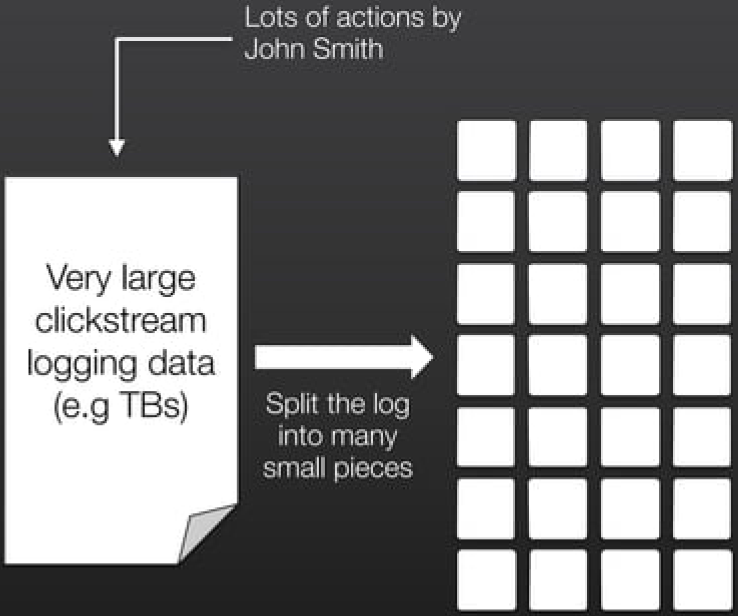
HADOOP FUNDAMENTALS

Very large
clickstream
logging data
(e.g TBs)



Lots of actions by John Smith

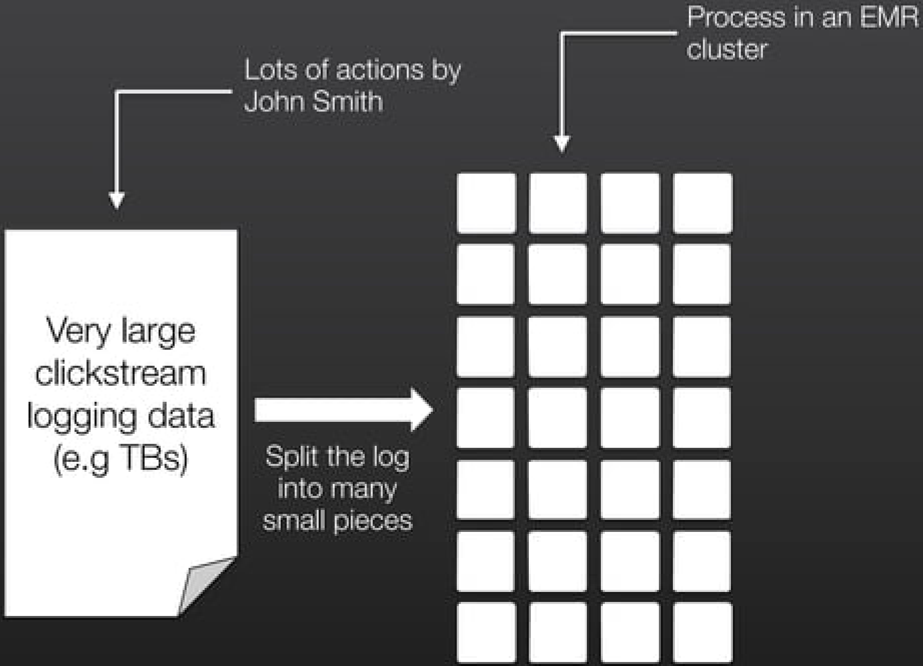
Very large clickstream logging data (e.g TBs)



Lots of actions by John Smith

Very large clickstream logging data (e.g TBs)

Split the log into many small pieces

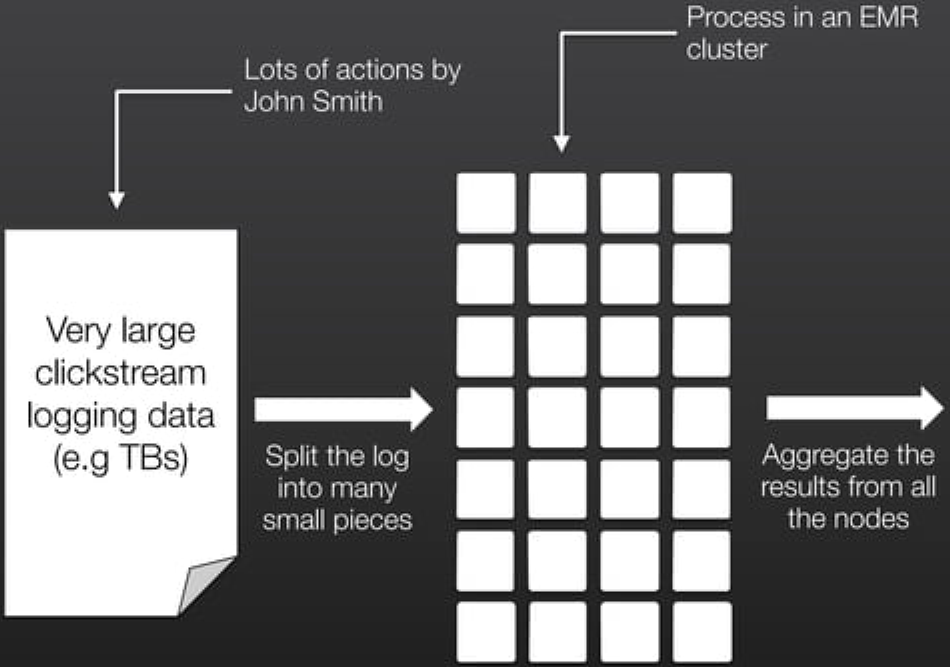


Lots of actions by John Smith

Process in an EMR cluster

Very large clickstream logging data (e.g TBs)

Split the log into many small pieces



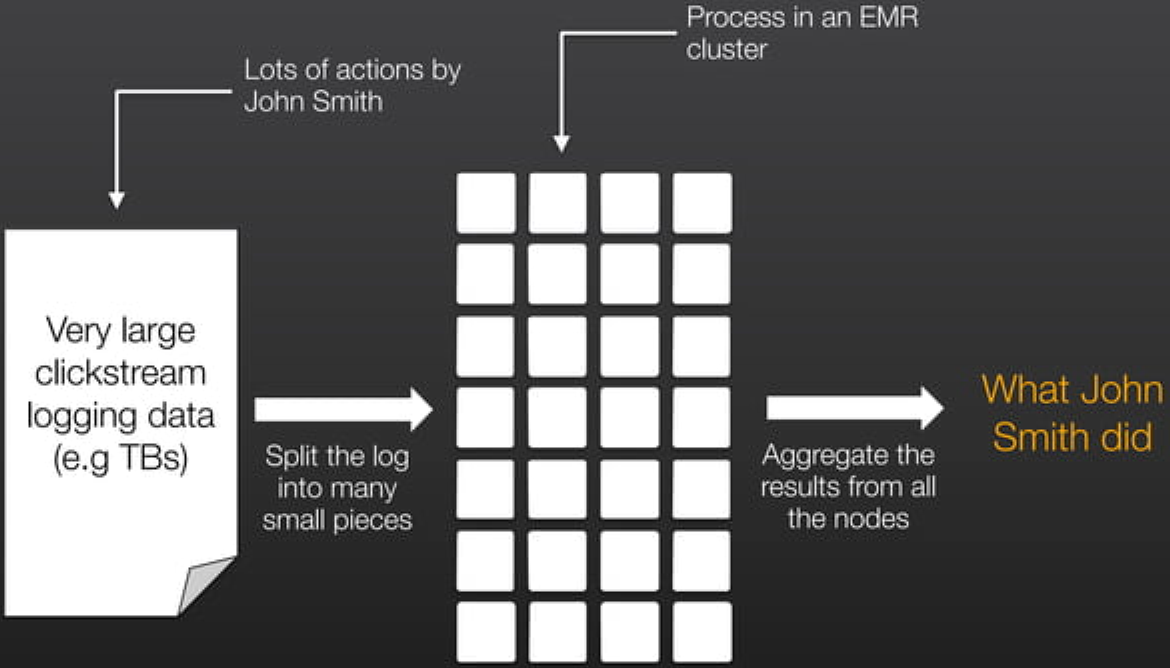
Lots of actions by John Smith

Process in an EMR cluster

Very large clickstream logging data (e.g TBs)

Split the log into many small pieces

Aggregate the results from all the nodes



Very large clickstream logging data (e.g TBs)

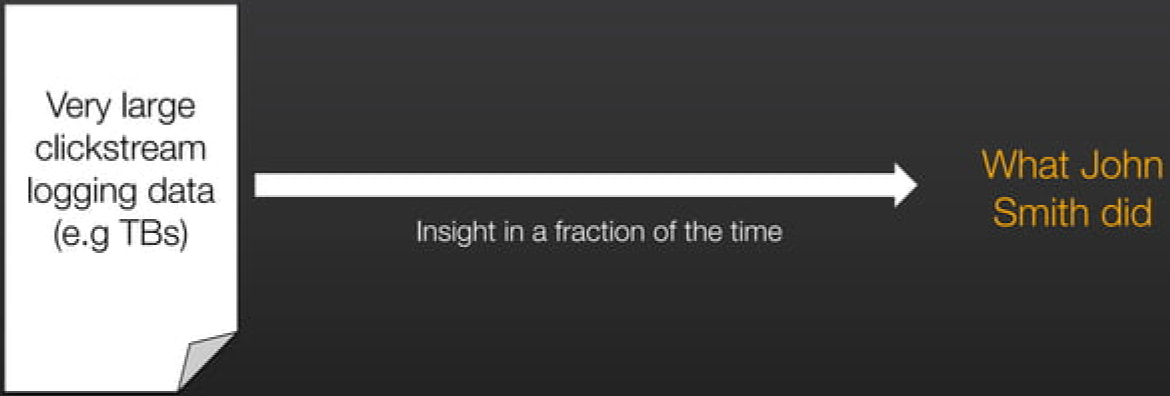
Lots of actions by John Smith

Split the log into many small pieces

Process in an EMR cluster

Aggregate the results from all the nodes

What John Smith did



Very large clickstream logging data (e.g TBs)

Insight in a fraction of the time

What John Smith did

CORE FEATURES OF AMAZON EMR

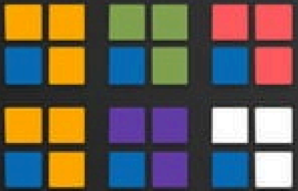
ELASTIC

Elastic

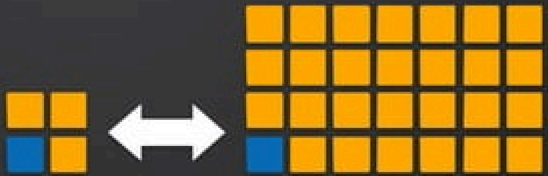
Provision as much capacity as you need

Add or remove capacity at any time

Deploy Multiple Clusters



Resize a Running Cluster



LOW COST

Low Cost

Low Hourly Pricing

Amazon EC2 Spot Integration

Amazon EC2 Reserved Instance Integration

Elasticity

Amazon S3 Integration

Low Cost

Accenture Hadoop Study:

Amazon EMR 'offers better price-performance'

High performance. Delivered.

Accenture Technology Labs

Where to deploy your Hadoop clusters?

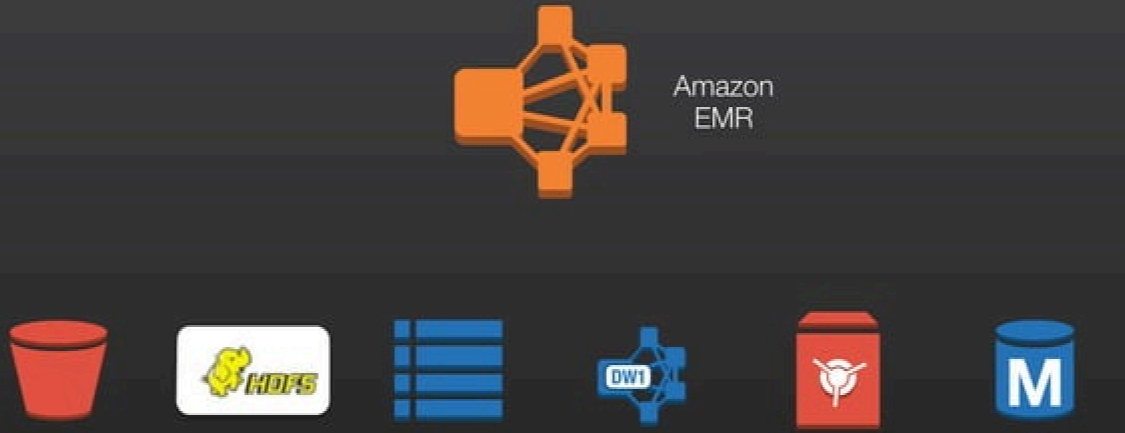
Executive Summary

accenture

consulting | technology | outsourcing

http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Technology-Labs-Where-To-Deploy-Your-Hadoop-CLusters-Executive-Summary.pdf

FLEXIBLE DATA STORES



Amazon
EMR

Amazon
S3

Hadoop Distributed
File System

Amazon
DynamoDB

Amazon
Redshift

Amazon
Glacier

Amazon Relational
Database Service

Amazon S3 + Amazon EMR

Allows you to decouple storage and computing resources

Use Amazon S3 features such as server-side encryption

When you launch your cluster, EMR streams data from S3

Multiple clusters can process the same data concurrently

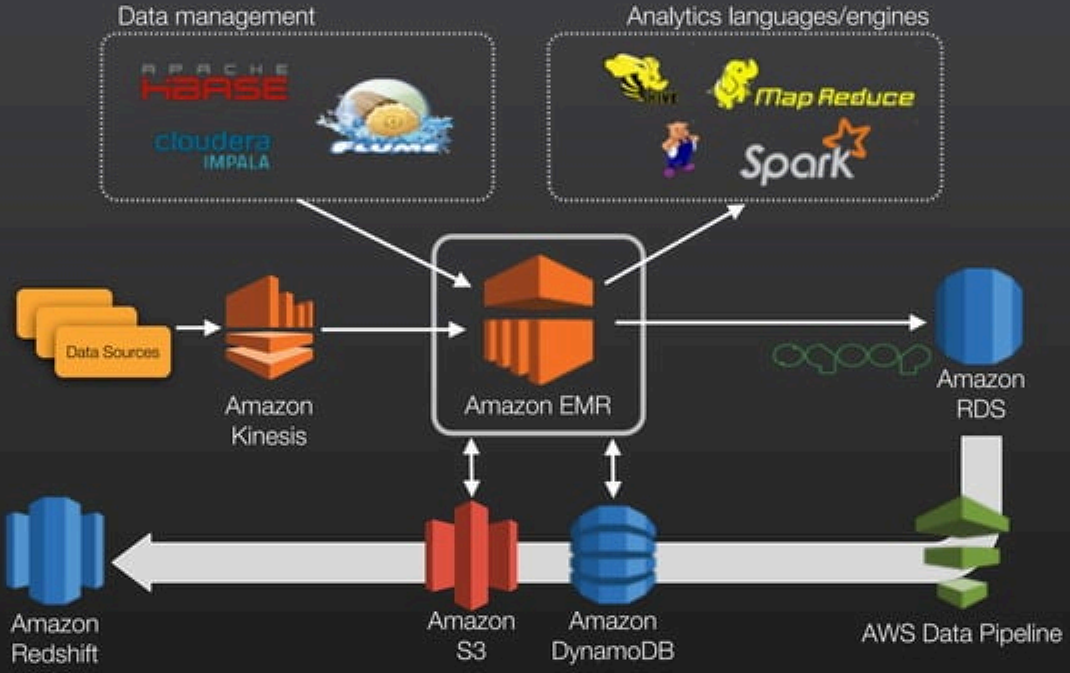
Hadoop Distributed
File System (HDFS)

Amazon
DynamoDB

AWS
Data Pipeline

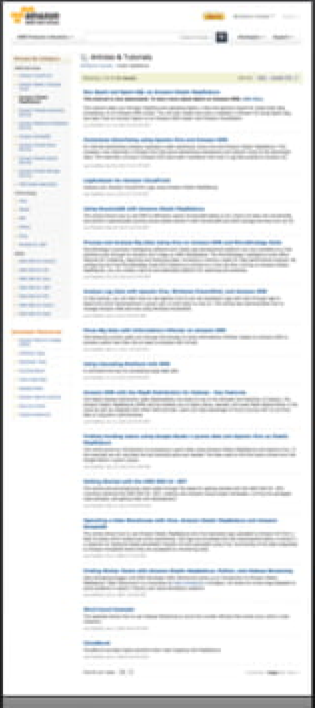
Amazon
RDS

Amazon
Redshift


Data management
APACHE HBASE
cloudera IMPALA
FLUME
Analytics languages/engines
HIVE
Map Reduce
Spark
Data Sources
Amazon Kinesis
Amazon EMR
sqoop
Amazon RDS
Amazon Redshift
Amazon S3
Amazon DynamoDB
AWS Data Pipeline

GETTING STARTED WITH AMAZON ELASTIC MAPREDUCE

Develop your data processing application



Articles & Tutorials

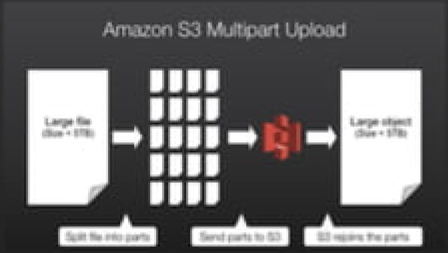
http://aws.amazon.com/articles/Elastic-MapReduce

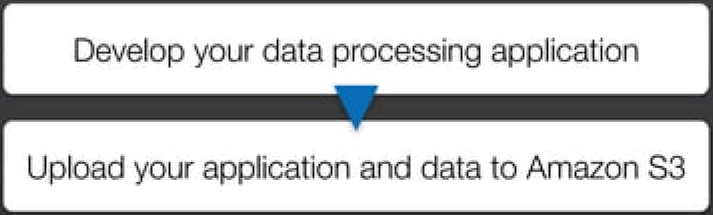
Develop your data processing application

Upload your application and data to Amazon S3

Develop your data processing application

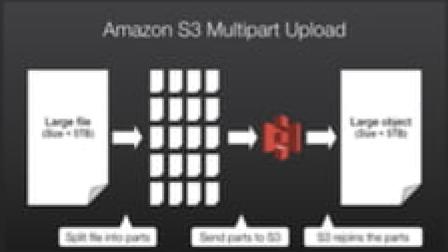
Upload your application and data to Amazon S3



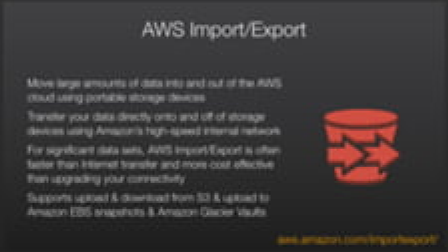


Develop your data processing application

Upload your application and data to Amazon S3



Amazon S3 Multipart Upload



AWS Import/Export

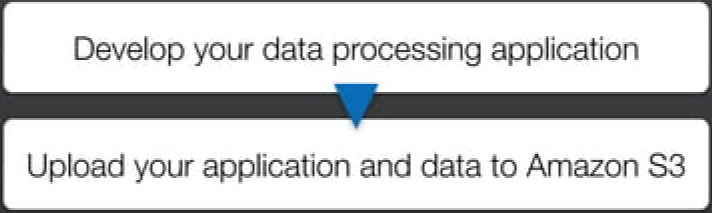
Move large amounts of data into and out of the AWS cloud using portable storage devices

Transfer your data directly onto and off of storage devices using Amazon's high-speed internal network

For significant data sets, AWS Import/Export is often faster than Internet transfer and more cost effective than upgrading your connectivity

Supports upload & download from S3 & upload to Amazon EBS snapshots & Amazon Glacier Vaults

aws.amazon.com/importexport/



Develop your data processing application

Upload your application and data to Amazon S3

Amazon S3 Multi

AWS Direct Connect

Makes it easy to establish a dedicated network connection from your premises to AWS

Establish private connectivity between AWS & your datacenter, office, or colocation environment

Reduce your network costs, increase bandwidth throughput, and provide a more consistent network experience

The dedicated connection can be partitioned into multiple virtual interfaces using 802.1q VLANs

aws.amazon.com/directconnect

nport/Export

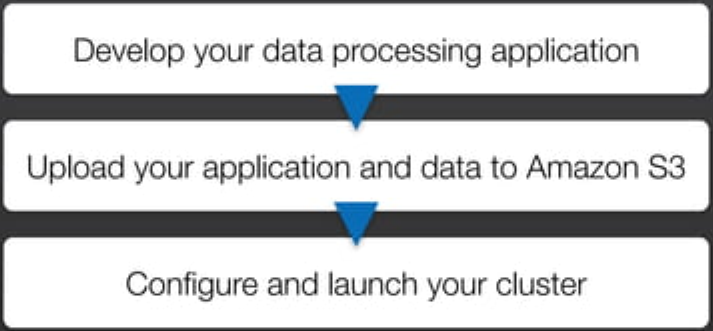
l out of the AWS

off of storage
internal network

t/Export is often
t cost effective

3 & upload to
Amazon EBS snapshots & Amazon Glacier Vaults

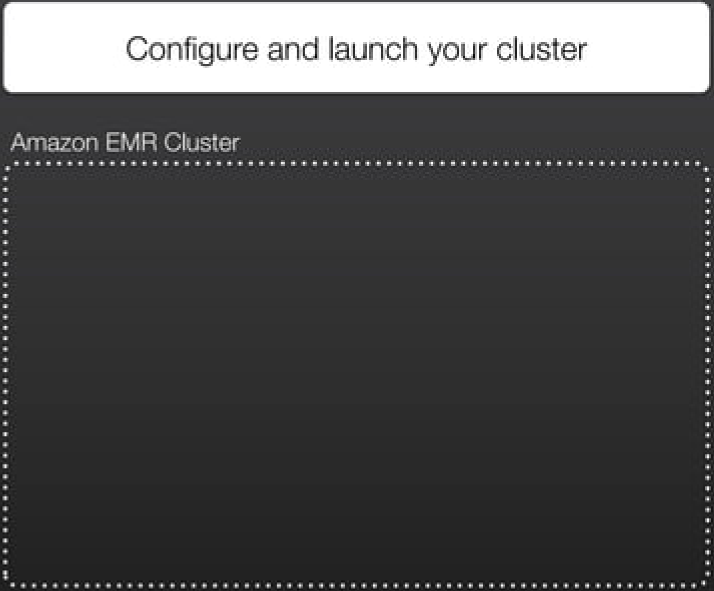
aws.amazon.com/importexport/



Develop your data processing application

Upload your application and data to Amazon S3

Configure and launch your cluster

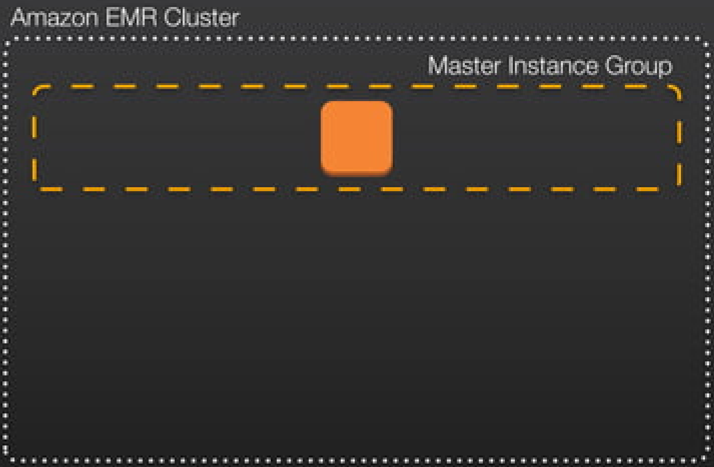


Configure and launch your cluster

Amazon EMR Cluster

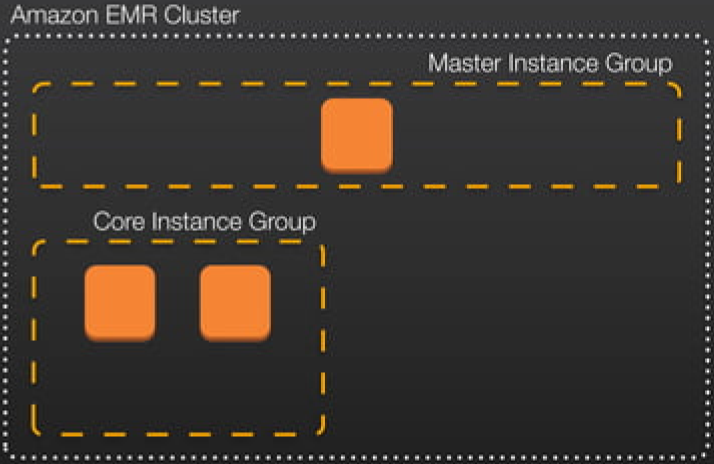
Start an EMR cluster using console, CLI tools or an AWS SDK

Configure and launch your cluster



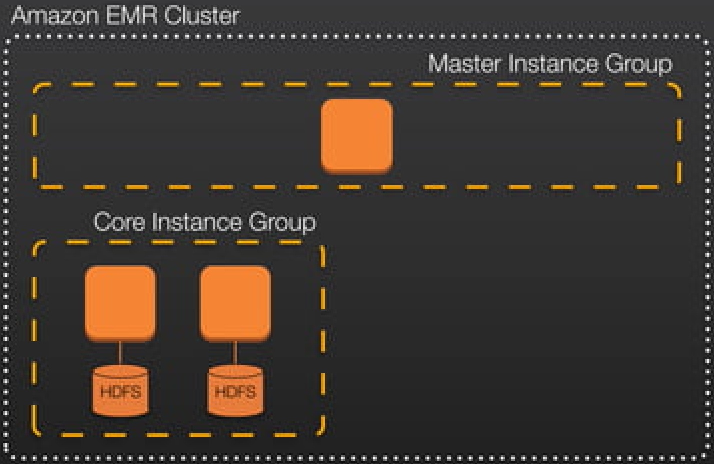
Master instance group created that controls the cluster

Configure and launch your cluster



Core instance group created for life of cluster

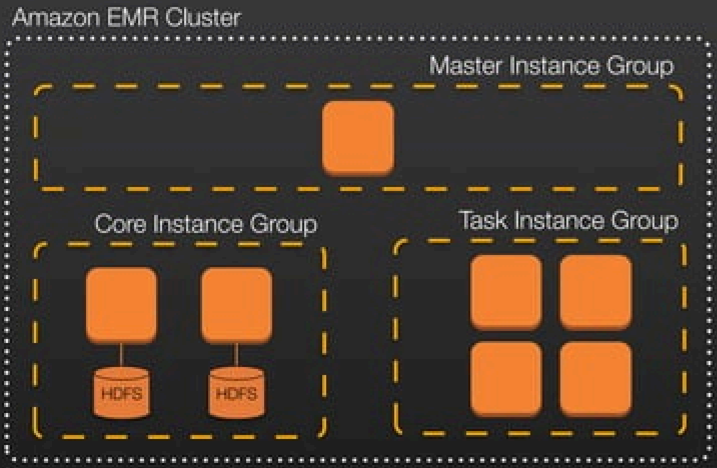
Configure and launch your cluster



Core instance group created for life of cluster

Core instances run DataNode and TaskTracker daemons

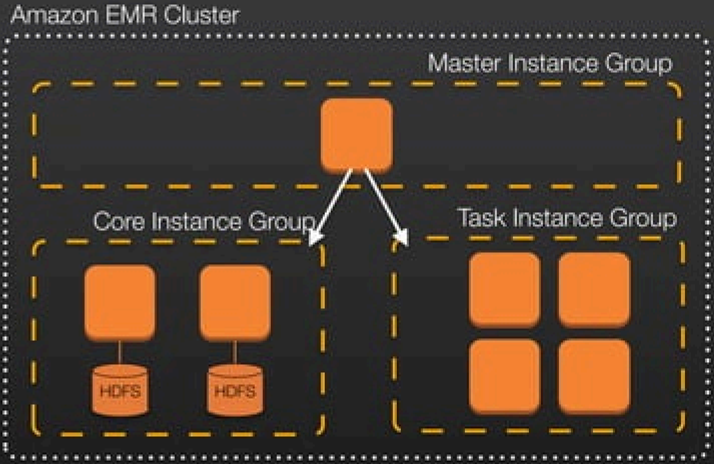
Configure and launch your cluster

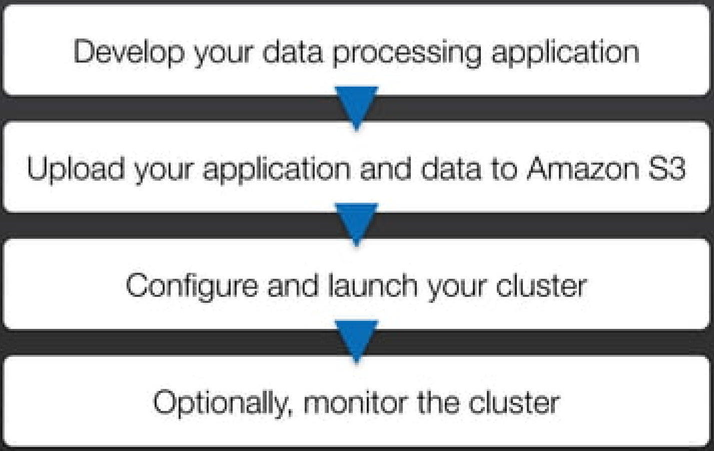


Optional task instances can be added or subtracted

Configure and launch your cluster

Master node coordinates distribution of work and manages cluster state



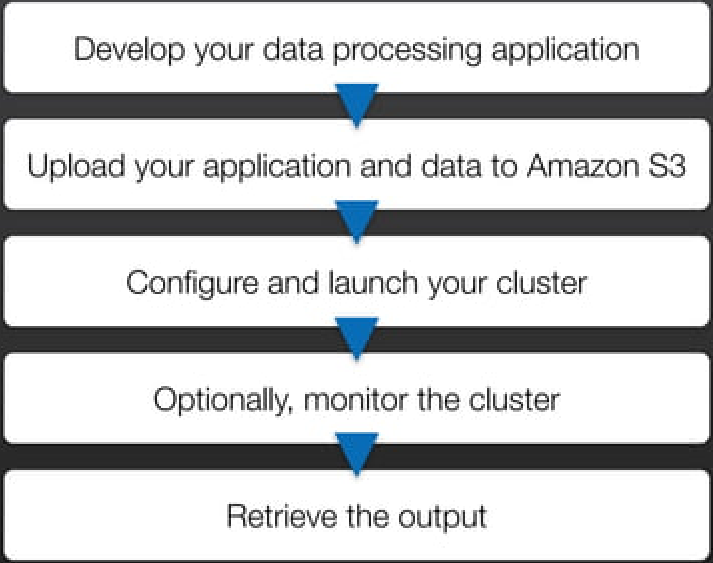


Develop your data processing application

Upload your application and data to Amazon S3

Configure and launch your cluster

Optionally, monitor the cluster



Develop your data processing application

Upload your application and data to Amazon S3

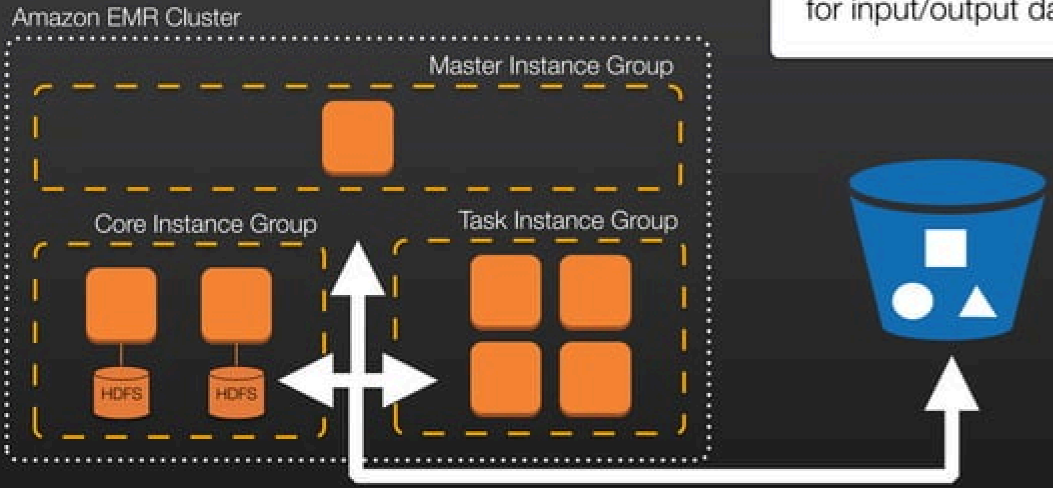
Configure and launch your cluster

Optionally, monitor the cluster

Retrieve the output

Retrieve the output

S3 can be used as underlying 'file system' for input/output data



DEMO:

GETTING STARTED WITH AMAZON EMR USING A SAMPLE HADOOP STREAMING APPLICATION

Hadoop Streaming

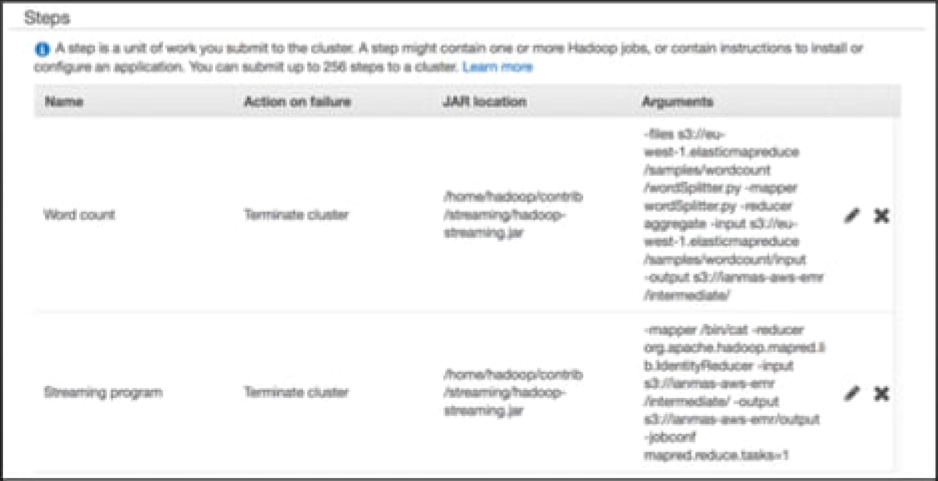
Utility that comes with the Hadoop distribution

Allows you to create and run Map/Reduce jobs with any executable or script as the mapper and/or the reducer

Reads the input from standard input and the reducer outputs data through standard output

By default, each line of input/output represents a record with tab separated key/value

Job Flow for Sample Application



Steps

A step is a unit of work you submit to the cluster. A step might contain one or more Hadoop jobs, or contain instructions to install or configure an application. You can submit up to 256 steps to a cluster. Learn more

Name	Action on failure	JAR location	Arguments
Word count	Terminate cluster	/home/hadoop/contrib/streaming/hadoop-streaming.jar	-files s3://eu-west-1.elasticmapreduce/samples/wordcount/wordSplitter.py -mapper wordSplitter.py -reducer aggregate -input s3://eu-west-1.elasticmapreduce/samples/wordcount/input -output s3://ianmas-aws-emr/intermediate/
Streaming program	Terminate cluster	/home/hadoop/contrib/streaming/hadoop-streaming.jar	-mapper /bin/cat -reducer org.apache.hadoop.mapred.lib.IdentityReducer -input s3://ianmas-aws-emr/intermediate/ -output s3://ianmas-aws-emr/output -jobconf mapred.reduce.tasks=1

Job Flow: Step 1

JAR location: `/home/hadoop/contrib/streaming/hadoop-streaming.jar`

Arguments:

```
-files s3://eu-west-1.elasticmapreduce/samples/wordcount/wordSplitter.py
-mapper wordSplitter.py
-reducer aggregate
-input s3://eu-west-1.elasticmapreduce/samples/wordcount/input
-output s3://ianmas-aws-emr/intermediate/
```

Step 1: mapper: wordSplitter.py

```python
#!/usr/bin/python
import sys
import re

def main(argv):
    pattern = re.compile("[a-zA-Z][a-zA-Z0-9]*")
    for line in sys.stdin:
        for word in pattern.findall(line):
            print "LongValueSum:" + word.lower() + "\t" + "1"

if __name__ == "__main__":
    main(sys.argv)
```

Step 1: mapper: wordSplitter.py

Read words from StdIn line by line

```
#!/usr/bin/python
import sys
import re

def main(argv):
    pattern = re.compile("[a-zA-Z][a-zA-Z0-9]*")
    for line in sys.stdin:
        for word in pattern.findall(line):
            print "LongValueSum:" + word.lower() + "\t" + "1"

if __name__ == "__main__":
    main(sys.argv)
```

Step 1: mapper: wordSplitter.py

```
#!/usr/bin/python
import sys
import re

def main(argv):
    pattern = re.compile("[a-zA-Z][a-zA-Z0-9]*")
    for line in sys.stdin:
        for word in pattern.findall(line):
            print "LongValueSum:" + word.lower() + "\t" + "1"

if __name__ == "__main__":
    main(sys.argv)
```

Output to StdOut tab delimited records in the format "LongValueSum:abacus   1"

Step 1: reducer: aggregate

Sorts inputs and adds up totals:

```
“Abacus      1”
“Abacus      1”
“Abacus      1”
```

becomes

```
“Abacus      3”
```

Step 1: input/ouput

The input is all the objects in the S3 bucket/prefix:

```
s3://eu-west-1.elasticmapreduce/samples/wordcount/input
```

Output is written to the following S3 bucket/prefix to be used as input for the next step in the job flow:

```
s3://ianmas-aws-emr/intermediate/
```

One output object is created for each reducer (generally one per core)

Job Flow: Step 2

JAR location: `/home/hadoop/contrib/streaming/hadoop-streaming.jar`

Arguments:

Accept anything and return as text

```
-mapper /bin/cat
-reducer org.apache.hadoop.mapred.lib.IdentityReducer
-input s3://ianmas-aws-emr/intermediate/
-output s3://ianmas-aws-emr/output
-jobconf mapred.reduce.tasks=1
```

Job Flow: Step 2

JAR location: `/home/hadoop/contrib/streaming/hadoop-streaming.jar`

Arguments:

Sort

```
-mapper /bin/cat
-reducer org.apache.hadoop.mapred.lib.IdentityReducer
-input s3://ianmas-aws-emr/intermediate/
-output s3://ianmas-aws-emr/output
-jobconf mapred.reduce.tasks=1
```

Job Flow: Step 2

JAR location: `/home/hadoop/contrib/streaming/hadoop-streaming.jar`

Arguments:

Take previous output as input

```
-mapper /bin/cat
-reducer org.apache.hadoop.mapred.lib.IdentityReducer
-input s3://ianmas-aws-emr/intermediate/
-output s3://ianmas-aws-emr/output
-jobconf mapred.reduce.tasks=1
```

Job Flow: Step 2

JAR location: `/home/hadoop/contrib/streaming/hadoop-streaming.jar`

Arguments:

Output location

```
-mapper /bin/cat
-reducer org.apache.hadoop.mapred.lib.IdentityReducer
-input s3://ianmas-aws-emr/intermediate/
-output s3://ianmas-aws-emr/output
-jobconf mapred.reduce.tasks=1
```

Job Flow: Step 2

JAR location: `/home/hadoop/contrib/streaming/hadoop-streaming.jar`

Arguments:

Use a single reduce task
to get a single output object

```
-mapper /bin/cat
-reducer org.apache.hadoop.mapred.lib.IdentityReducer
-input s3://ianmas-aws-emr/intermediate/
-output s3://ianmas-aws-emr/output
-jobconf mapred.reduce.tasks=1
```

SUPPORTED HADOOP TOOLS

Supported Hadoop Tools

Hive

An open source data warehouse & analytics package the runs on top of Hadoop. Operated by Hive QL, a SQL-based language which allows users to structure, summarize, and query data

Pig

An open source analytics package that runs on top of Hadoop. Pig is operated by Pig Latin, a SQL-like language which allows users to structure, summarize, and query data. Allows processing of complex and unstructured data sources such as text documents and log files.

HBase

Provides you an efficient way of storing large quantities of sparse data using column-based storage. HBase provides fast lookup of data because data is stored in-memory instead of on disk. Optimized for sequential write operations, and it is highly efficient for batch inserts, updates, and deletes.

Supported Hadoop Tools

Impala

A tool in the Hadoop ecosystem for interactive, ad hoc querying using SQL syntax.It uses a massively parallel processing (MPP) engine similar to that found in a traditional RDBMS.

This lends Impala to interactive, low-latency analytics. You can connect to BI tools through ODBC and JDBC drivers.

Presto

An open source distributed SQL query engine for running interactive analytic queries against data sources of all sizes ranging from gigabytes to petabytes.

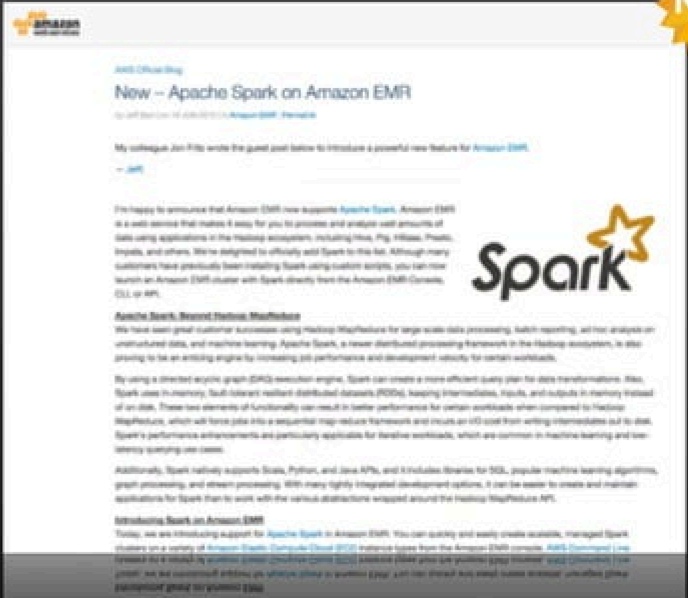
Hue

An open source user interface for Hadoop that makes it easier to run and develop Hive queries, manage files in HDFS, run and develop Pig scripts, and manage tables.

DEMO:
APACHE HUE ON EMR

aws.amazon.com/blogs/aws/new-apache-spark-on-amazon-emr/

New



AWS Official Blog

New – Apache Spark on Amazon EMR

by Jeff Barr | on 16 JUN 2015 | in Amazon EMR | Permalink

My colleague Jon Fritz wrote the guest post below to introduce a powerful new feature for Amazon EMR.

— Jeff;

I'm happy to announce that Amazon EMR now supports Apache Spark. Amazon EMR is a web service that makes it easy for you to process and analyze vast amounts of data using applications in the Hadoop ecosystem, including Hive, Pig, HBase, Presto, Impala, and others. We're delighted to officially add Spark to this list. Although many customers have previously been installing Spark using custom scripts, you can now launch an Amazon EMR cluster with Spark directly from the Amazon EMR Console, CLI, or API.

Apache Spark: Beyond Hadoop MapReduce

We have seen great customer successes using Hadoop MapReduce for large scale data processing, batch reporting, ad hoc analysis on unstructured data, and machine learning. Apache Spark, a newer distributed processing framework in the Hadoop ecosystem, is also proving to be an enticing engine by increasing job performance and development velocity for certain workloads.

By using a directed acyclic graph (DAG) execution engine, Spark can create a more efficient query plan for data transformations. Also, Spark uses in-memory, fault-tolerant resilient distributed datasets (RDDs), keeping intermediates, inputs, and outputs in memory instead of on disk. These two elements of functionality can result in better performance for certain workloads when compared to Hadoop MapReduce, which will force jobs into a sequential map-reduce framework and incurs an I/O cost from writing intermediates out to disk. Spark's performance enhancements are particularly applicable for iterative workloads, which are common in machine learning and low-latency querying use cases.

Additionally, Spark natively supports Scala, Python, and Java APIs, and it includes libraries for SQL, popular machine learning algorithms, graph processing, and stream processing. With many tightly integrated development options, it can be easier to create and maintain applications for Spark than to work with the various abstractions wrapped around the Hadoop MapReduce API.

Introducing Spark on Amazon EMR

Today, we are introducing support for Apache Spark in Amazon EMR. You can quickly and easily create scalable, managed Spark clusters on a variety of Amazon Elastic Compute Cloud (EC2) instance types from the Amazon EMR console, AWS Command Line

AWS CLI

Create a Cluster with Spark

```
$ aws emr create-cluster --name "Spark cluster" \
   --ami-version 3.8 --applications Name=Spark \
   --ec2-attributes KeyName=myKey --instance-type m3.xlarge \
   --instance-count 3 --use-default-roles
```

```
$ ssh -i myKey hadoop@masternode
```

invoke the spark shell with

```
$ spark-shell
```

or

```
$ pyspark
```

pyspark

Working with the Spark Shell

Counting the occurrences of a string a text file stored in Amazon S3 with spark

```
$ pyspark
>>> sc
<pyspark.context.SparkContext object at 0x7fe7e659fa50>
>>> textfile = sc.textFile("s3://elasticmapreduce/samples/hive-ads/tables/impressions/
dt=2009-04-13-08-05/ec2-0-51-75-39.amazon.com-2009-04-13-08-05.log")
>>> linesWithCartoonNetwork = textfile.filter(lambda line: "cartoonnetwork.com" in
line).count()
15/06/04 17:12:22 INFO lzo.GPLNativeCodeLoader: Loaded native gpl library from the
embedded binaries
<snip>
<Spark program continues>
>>> linesWithCartoonNetwork
9
```

docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-spark.html

ADDITIONAL EMR FEATURES

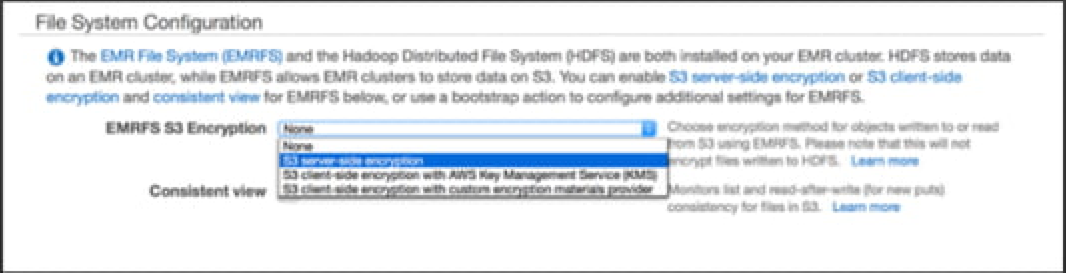
CONTROL NETWORK ACCESS TO YOUR EMR CLUSTER

Using SSH local port forwarding

```
ssh -i EMRKeyPair.pem -N \
    -L 8160:ec2-52-16-143-78.eu-west-1.compute.amazonaws.com:8888 \
    hadoop@ec2-52-16-143-78.eu-west-1.compute.amazonaws.com
```

docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-web-interfaces.html

MANAGE USERS, PERMISSIONS AND ENCRYPTION



File System Configuration

The EMR File System (EMRFS) and the Hadoop Distributed File System (HDFS) are both installed on your EMR cluster. HDFS stores data on an EMR cluster, while EMRFS allows EMR clusters to store data on S3. You can enable S3 server-side encryption or S3 client-side encryption and consistent view for EMRFS below, or use a bootstrap action to configure additional settings for EMRFS.

EMRFS S3 Encryption: None

Choose encryption method for objects written to or read from S3 using EMRFS. Please note that this will not encrypt files written to HDFS. Learn more

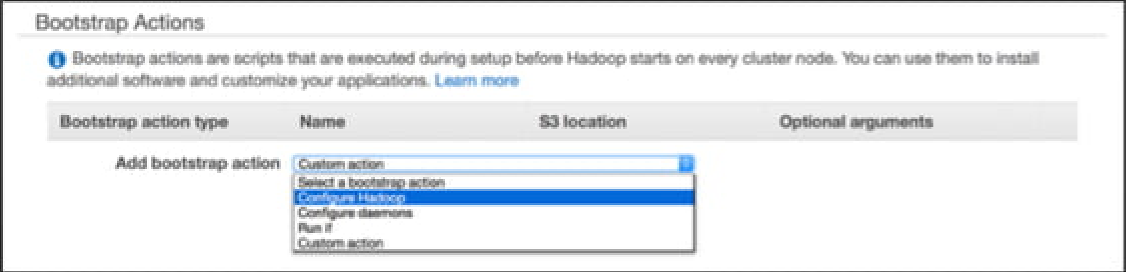
Consistent view

Monitors list and read-after-write (for new puts) consistency for files in S3. Learn more

None
S3 server-side encryption
S3 client-side encryption with AWS Key Management Service (KMS)
S3 client-side encryption with custom encryption materials provider

docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-plan-access.html

INSTALL ADDITIONAL SOFTWARE WITH BOOTSTRAP ACTIONS



Bootstrap Actions

Bootstrap actions are scripts that are executed during setup before Hadoop starts on every cluster node. You can use them to install additional software and customize your applications. Learn more

Bootstrap action type	Name	S3 location	Optional arguments

Add bootstrap action: Custom action

docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-plan-bootstrap.html

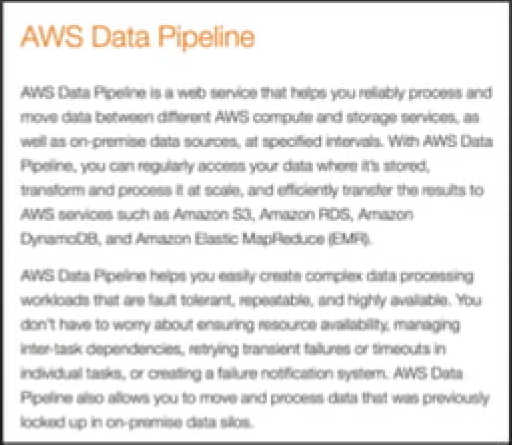
EFFICIENTLY COPY DATA TO EMR FROM AMAZON S3

Run on a cluster master node:

```
$ hadoop jar /home/hadoop/lib/emr-s3distcp-1.0.jar -
Dmapreduce.job.reduces=30 --src s3://s3bucketname/ --dest hdfs://
$HADOOP_NAMENODE_HOST:$HADOOP_NAMENODE_PORT/data/ --outputCodec 'none'
```

docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/UsingEMR_s3distcp.html

SCHEDULE RECURRING WORKFLOWS



AWS Data Pipeline

AWS Data Pipeline is a web service that helps you reliably process and move data between different AWS compute and storage services, as well as on-premise data sources, at specified intervals. With AWS Data Pipeline, you can regularly access your data where it's stored, transform and process it at scale, and efficiently transfer the results to AWS services such as Amazon S3, Amazon RDS, Amazon DynamoDB, and Amazon Elastic MapReduce (EMR).

AWS Data Pipeline helps you easily create complex data processing workloads that are fault tolerant, repeatable, and highly available. You don't have to worry about ensuring resource availability, managing inter-task dependencies, retrying transient failures or timeouts in individual tasks, or creating a failure notification system. AWS Data Pipeline also allows you to move and process data that was previously locked up in on-premise data silos.

aws.amazon.com/datapipeline

MONITOR YOUR CLUSTER

docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/UsingEMR_ViewingMetrics.html

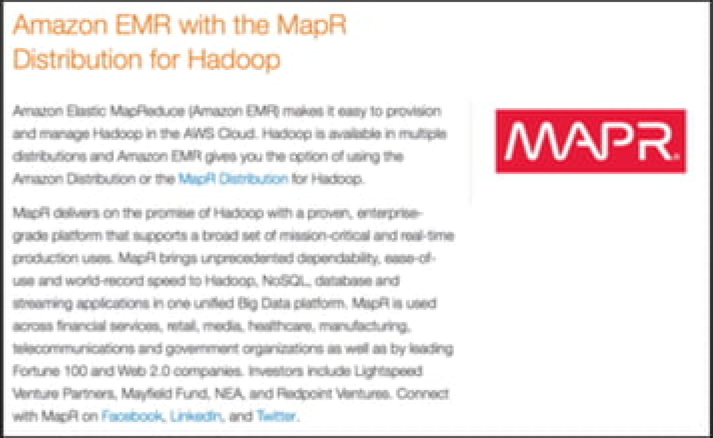
DEBUG YOUR APPLICATIONS

Log files generated by EMR Clusters include:

- Step logs
- Hadoop logs
- Bootstrap action logs
- Instance state logs

docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-plan-debugging.html

USE THE MAPR DISTRIBUTION



Amazon EMR with the MapR Distribution for Hadoop

Amazon Elastic MapReduce (Amazon EMR) makes it easy to provision and manage Hadoop in the AWS Cloud. Hadoop is available in multiple distributions and Amazon EMR gives you the option of using the Amazon Distribution or the MapR Distribution for Hadoop.

MapR delivers on the promise of Hadoop with a proven, enterprise-grade platform that supports a broad set of mission-critical and real-time production uses. MapR brings unprecedented dependability, ease-of-use and world-record speed to Hadoop, NoSQL, database and streaming applications in one unified Big Data platform. MapR is used across financial services, retail, media, healthcare, manufacturing, telecommunications and government organizations as well as by leading Fortune 100 and Web 2.0 companies. Investors include Lightspeed Venture Partners, Mayfield Fund, NEA, and Redpoint Ventures. Connect with MapR on Facebook, LinkedIn, and Twitter.

aws.amazon.com/elasticmapreduce/mapr/

TUNE YOUR CLUSTER FOR COST & PERFORMANCE

Supported EC2 instance types

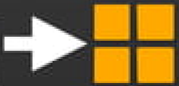
- General Purpose
- Compute Optimized
- Memory Optimized
- Storage Optimized - D2 instance family New

  D2 instances are available in four sizes with 6TB, 12TB, 24TB, and 48TB storage options.
- GPU Instances

TUNE YOUR CLUSTER FOR COST & PERFORMANCE

Scenario #1
Job Flow

Allocate
4 on demand
instances

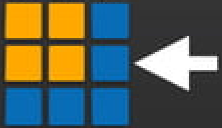


Duration:

Cost *without* Spot
4 instances *14 hrs * $0.50
Total = $28

Time Savings: 50%
Cost Savings: ~22%

Scenario #2
Job Flow

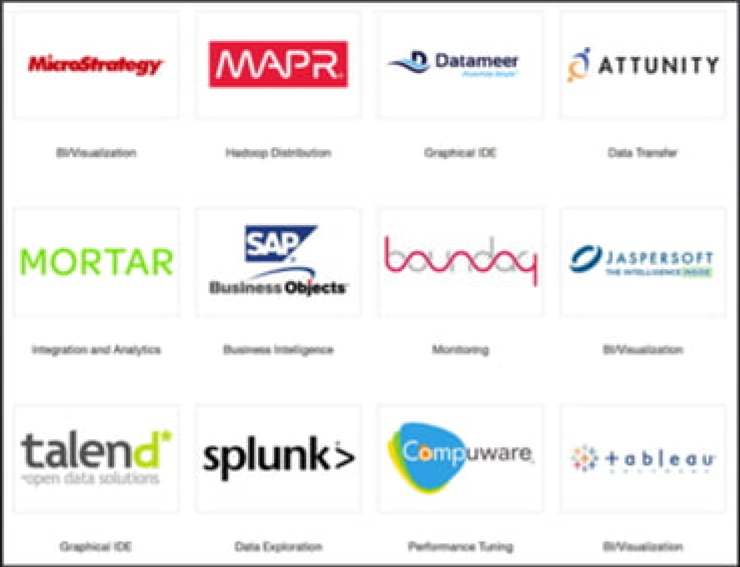


Add
5 additional
spot instances

Duration:

Cost *with* Spot
4 instances *7 hrs * $0.50 = $14 +
5 instances * 7 hrs * $0.25 = $8.75
Total = $22.75

THIRD PARTY TOOLS



MicroStrategy	MAPR	Datameer	ATTUNITY
BI/Visualization	Hadoop Distribution	Graphical IDE	Data Transfer
MORTAR	SAP Business Objects	boundary	JASPERSOFT THE INTELLIGENCE INSIDE
Integration and Analytics	Business Intelligence	Monitoring	BI/Visualization
talend open data solutions	splunk>	Compuware	tableau software
Graphical IDE	Data Exploration	Performance Tuning	BI/Visualization

RESOURCES YOU CAN USE TO LEARN MORE

aws.amazon.com/emr

Getting Started with Amazon EMR Tutorial guide:

docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-get-started.html

Customer Case Studies for Big Data Use-Cases

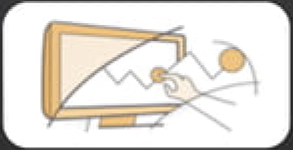
aws.amazon.com/solutions/case-studies/big-data/

Amazon EMR Documentation:

aws.amazon.com/documentation/emr/

AWS Training & Certification

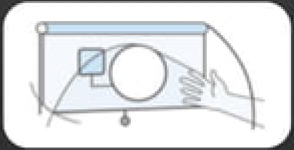
Self-Paced Labs



Try products, gain new skills, and get hands-on practice working with AWS technologies

aws.amazon.com/training/
self-paced-labs

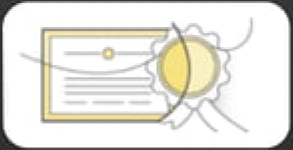
Training



Build technical expertise to design and operate scalable, efficient applications on AWS

aws.amazon.com/training

Certification



Validate your proven skills and expertise with the AWS platform

aws.amazon.com/certification

Follow us for more events & webinars

amazon web services

Ian Massingham — Technical Evangelist

@IanMmmm

@AWS_UKI for local AWS events & news

@AWScloud for Global AWS News & Announcements