H2O in Big Data Environments

Tom Kraljevic

H2O World Nov. 18, 2014

Outline

- Motivation
- Observations and Goals
- Hadoop (and YARN)
- Spark (and Sparkling Water)
- EC2
- Other environments
- Q & A

Motivation

- Information is the New Gold, and information comes from Data
- Last year at NY Strata: frenetic data collection (not necessarily with a plan)
- This year at NY Strata: people have a plan
  - Market mix modeling, Fraud detection
  - Advertising technology, Customer intelligence
- Reading data, building models and making predictions at scale

Observations and Goals

- Observations
  - Data has gravity
    - Govt. Regulation, Privacy, Sheer size
  - HDFS and S3 are terrific data stores
  - Procurement of new hardware resources is usually hard
    - Easier to use the Hadoop environment you already have, or start up new nodes in the cloud

- Goals
  - Move algorithms (i.e. H2O) to the data
    - H2O is pure Java, so it's easy to run anywhere
  - Make H2O best-in-class Machine Learning application / library
  - Make H2O work well in whatever environment you already have

Supported Hadoop Environments

- All major distributions
  - Cloudera
  - Hortonworks
  - MapR
- Hosted in the cloud
  - e.g. Altiscale
- We don't see this one too often (but it's worked when we have)
  - Open source Apache Hadoop

Hadoop (and YARN)

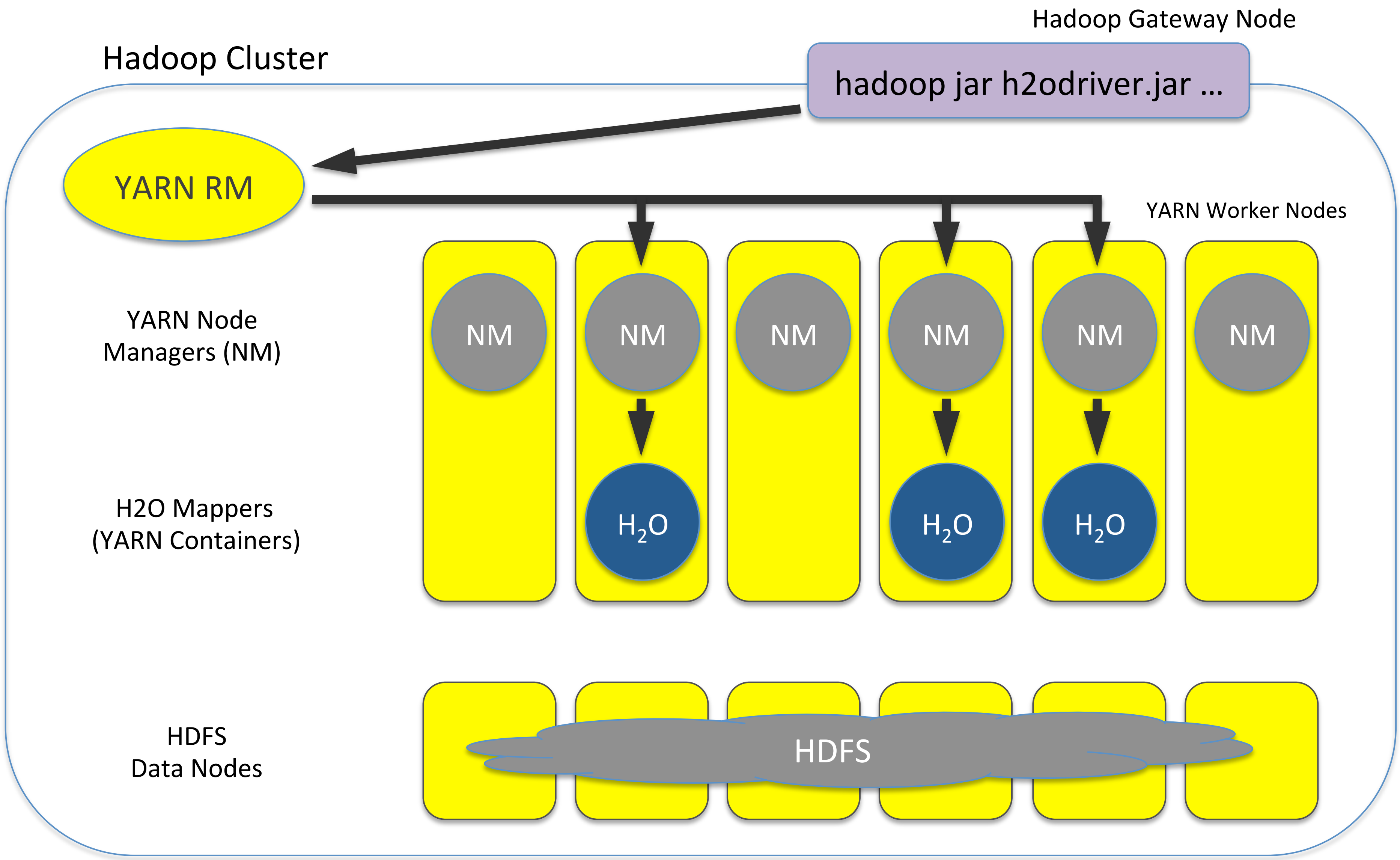
- You can launch H2O directly on Hadoop:

```
$ hadoop jar h2odriver.jar … –nodes 3 –mapperXmx 50g
```

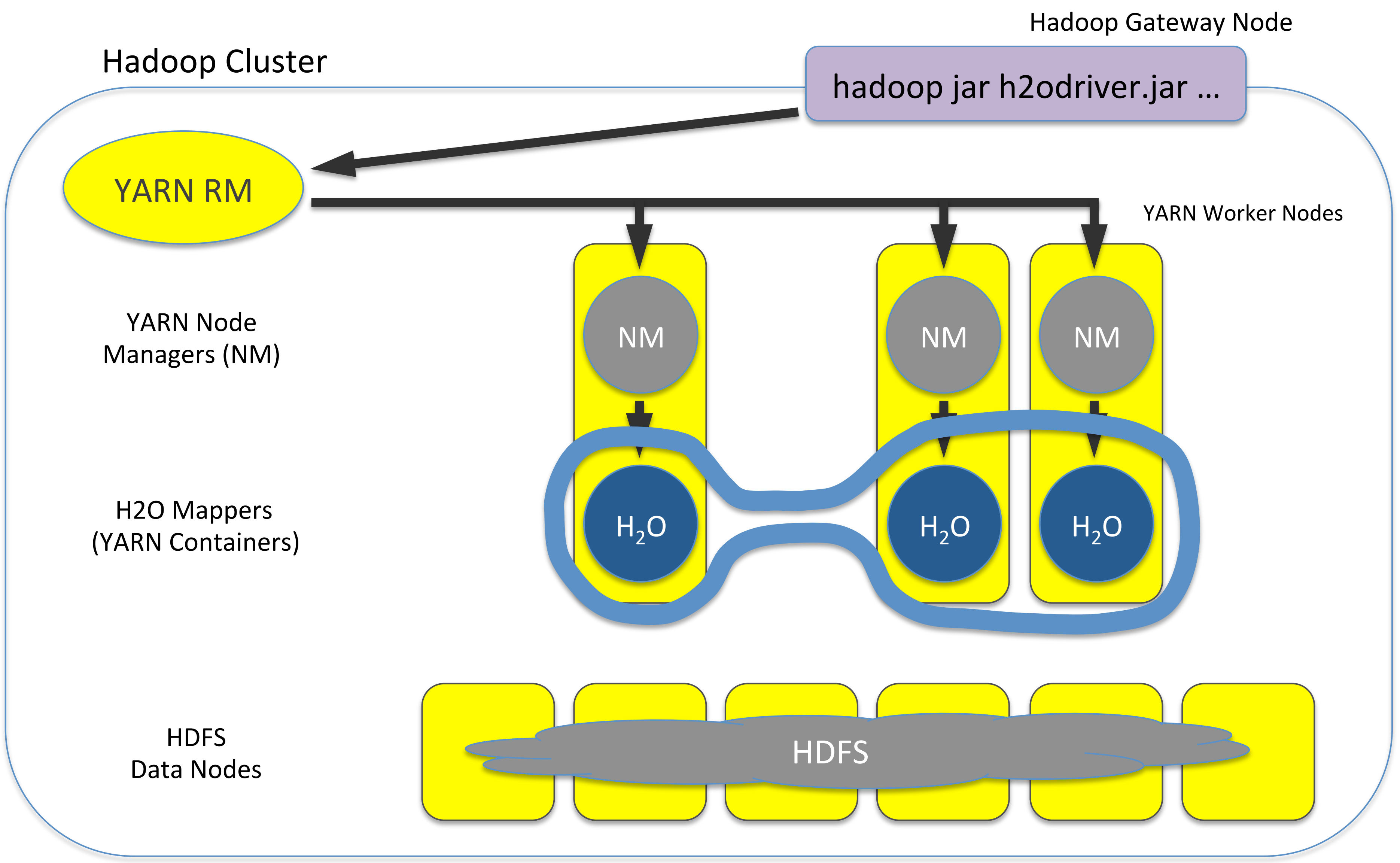
- H2O uses Hadoop MapReduce to get CPU and Memory on the cluster, not to manage work
  - H2O mappers stay at 0% progress forever
    - Until you shut down the H2O job yourself
  - All mappers (3 in this case) must be running at the same time
  - The mappers communicate with each other
    - Form an H2O cluster on-the-spot within your Hadoop environment
  - No Hadoop reducers(!)

- Special YARN memory settings for large mappers
  - yarn.nodemanager.resource.memory-mb
  - yarn.scheduler.maximum-allocation-mb

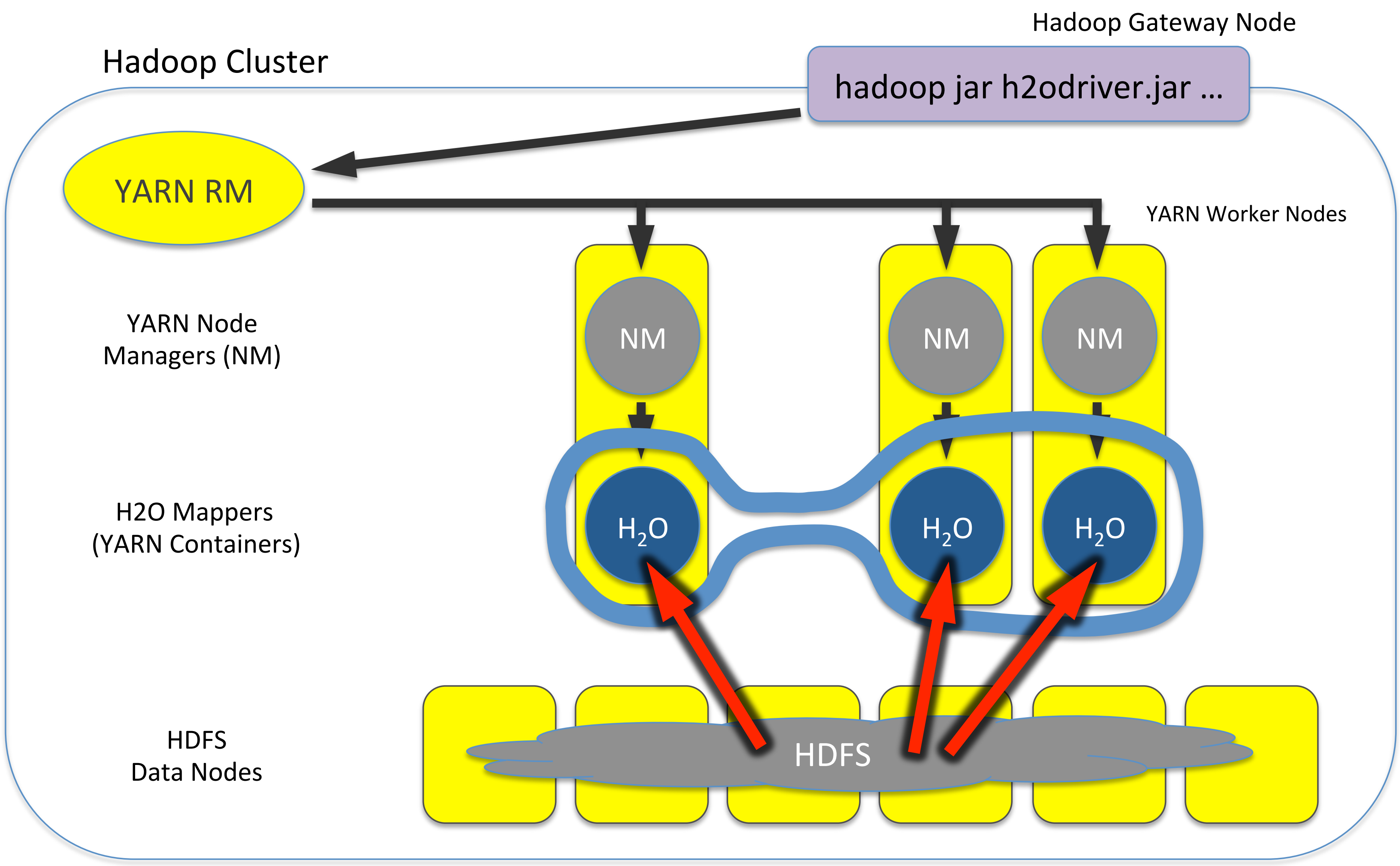
H2O on YARN Deployment



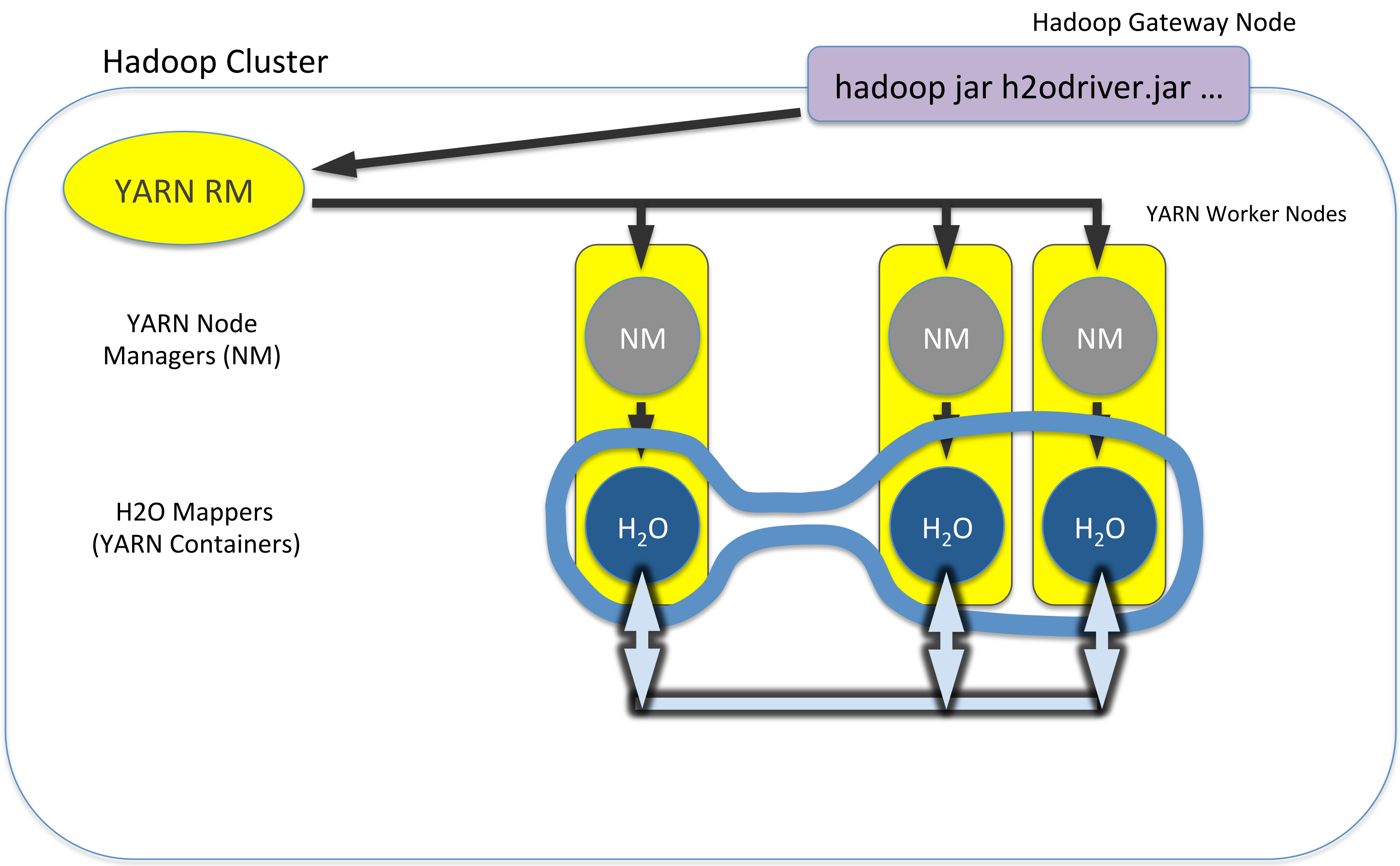
Now You Have an H2O Cluster



Read Data from HDFS ***Once***



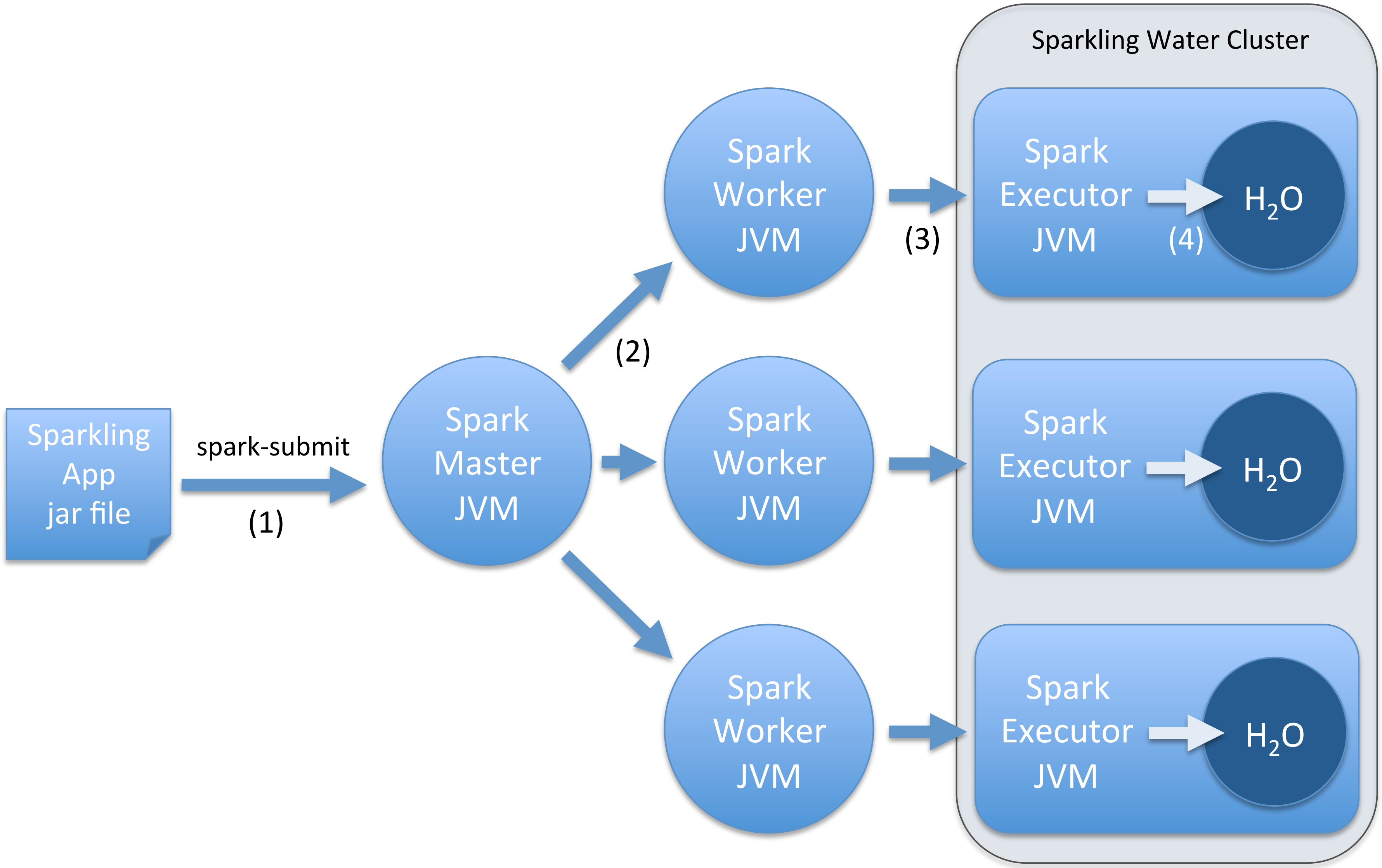
Build Models *in-Memory*



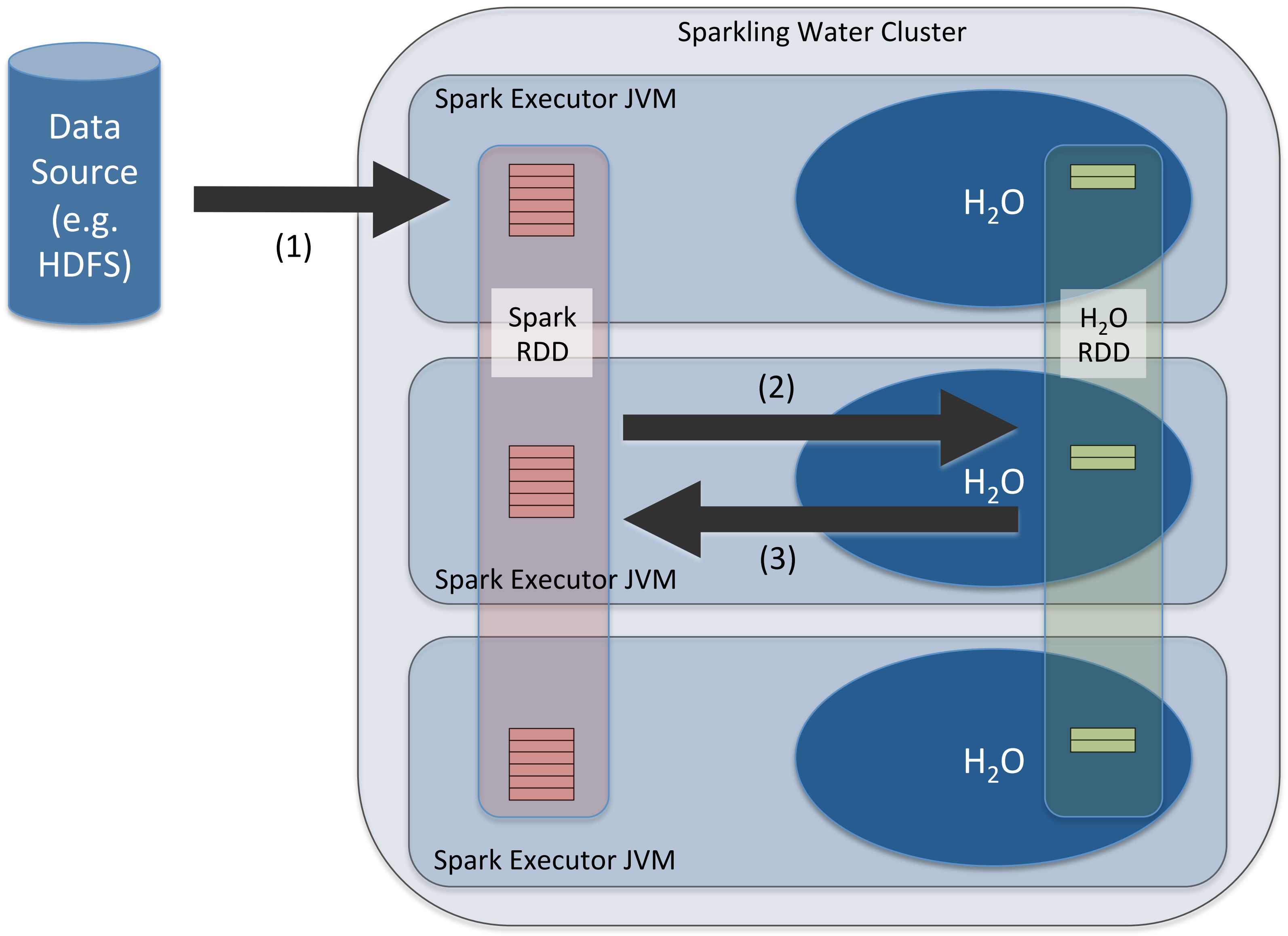
Spark (and Sparkling Water)

- H2O runs as an application on a Spark cluster using spark-submit
  - Standard Spark 1.1
  - Includes H2O on Spark on YARN
- H2O and Spark nodes share a JVM process
- H2ORDD facilitates easy data sharing between Spark (e.g. Spark SQL, MLlib) and H2O (e.g. Deep Learning)
- Scala support

Sparkling Water Application Life Cycle



Sparkling Water Data Distribution



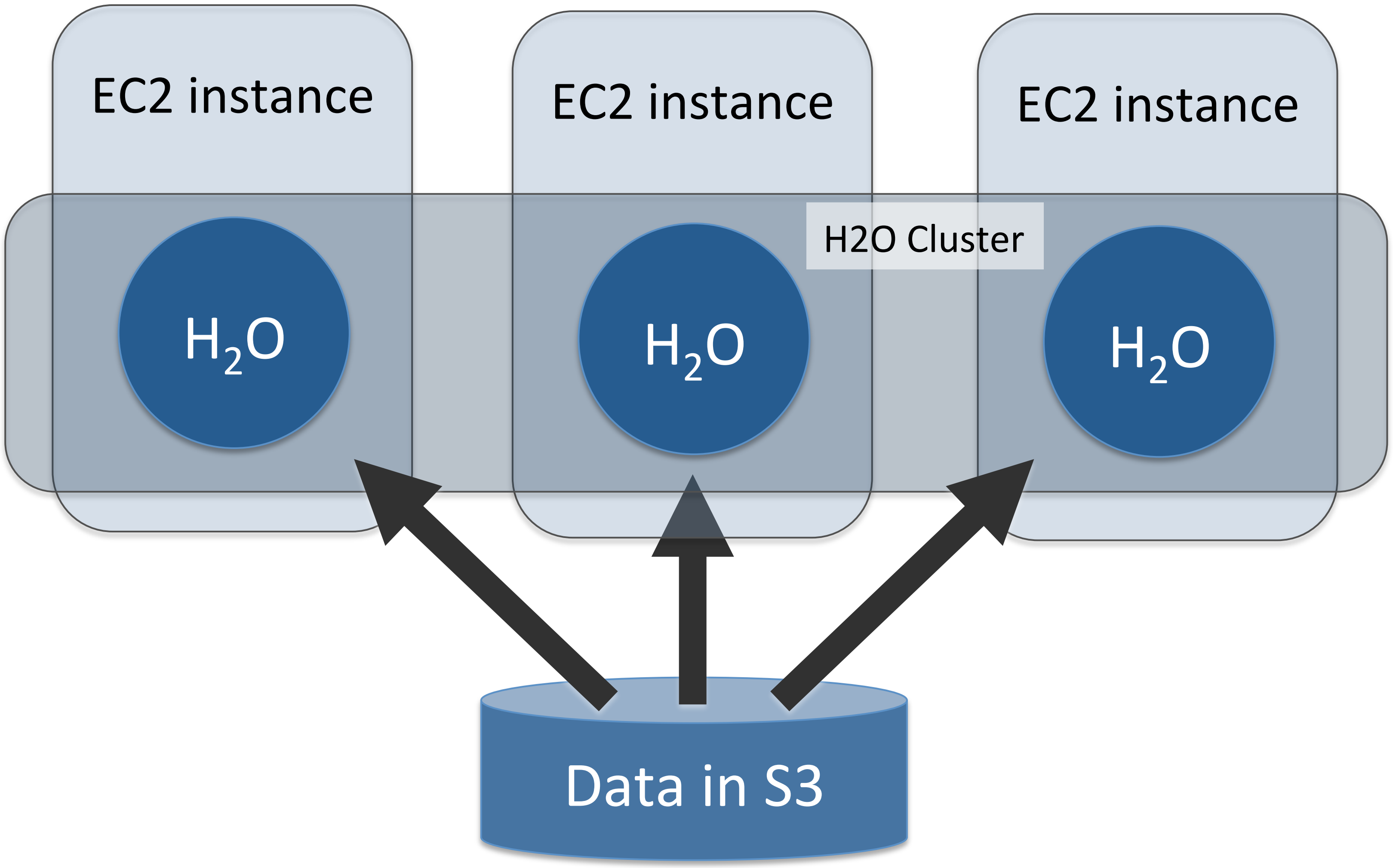
For More Info on Sparkling Water...

- Visit Michal at the H2O World Hacker's Corner
- Training and Blogs
  - http://learn.h2o.ai/content/hackers_station/start_with_sparkling_water.html
  - http://h2o.ai/blog/2014/11/sparkling/water/on/yarn/example/
  - http://h2o.ai/blog/2014/09/sparkling-water-tutorials/
  - http://h2o.ai/blog/2014/09/how-sparkling-water-brings-h2o-to-spark/
  - http://h2o.ai/blog/2014/09/Sparkling-Water/

EC2

- Security Group settings for TCP/UDP port management
  - port 54321 (TCP, User to H2O)
  - port 54322 (TCP and UDP, H2O to H2O)
- IAM role for easy and secure access to S3 data buckets (H2O to S3 data)
- Give each H2O node a fully-specified flatfile with the IP and port of each node in the cloud
- Use decent instances (cloud doesn't imply powerful)
  - m3.xlarge (or higher)
  - c3.xlarge (or higher)

EC2 Instances and Data Access



Other Environments

- Rumored to work, but untested (so far) by us…
  - Google Compute Engine
  - Microsoft Azure

Q & A

Thanks for attending H2O World!

http://h2o.ai