MTA SZTAKI
Hungarian Academy of Sciences
Institute for Computer Science and Control

Automated Hadoop cluster deployment on clouds with Occopus

Enikő Nagy
József Kovács
Róbert Lovas
[robert.lovas@sztaki.mta.hu]

Óbuda University
John von Neumann
Faculty of Informatics

Topics

- Goals and background (projects)
- Occopus orchestration tool
- Hadoop deployment: implementation
- Performance
- Related work
- Future work

Introduction

- **Motivation:**
  - Easy to use Hadoop cluster deployment for **MTA Cloud and Agrodat.hu** data scientists

- **Main design goals**
  1. Portable
  2. Scalable
  3. Does not require any prepared image
  4. For advanced users: fine-tune the configuration of the Hadoop components
  5. Supports short or long-term usage scenarios

Agrodat.hu project

AGRODAT

Main objective: **knowledge centre** and **decision support** system

- based on data gathered by an innovative, ***complex sensor*** system and from international ***open repositories***
- relying on ***big data, cloud***, and ***HPC*** technologies

to support **precision agriculture**.

Duration: 2014-2017
Budget: appr. 8 MEUR
URL: www.agrodat.hu

Consortium:

NET

Hewlett Packard Enterprise

MTA SZTAKI
Hungarian Academy of Sciences
Institute for Computer Science and Control

SZÉCHENYI ISTVÁN EGYETEM
SZE · GYŐR

INSPIRE GEOPORTAL
Enhancing access to European spatial data
European Commission
GEOGRAPHIC LOCATION
AGRODAT
Type the search expression

Big Data center:

844 CPU Core
5274 GB Memory
564 TB SSD/HD

GPGPU:
21504 CUDA Core
488 Xeon Phi Core

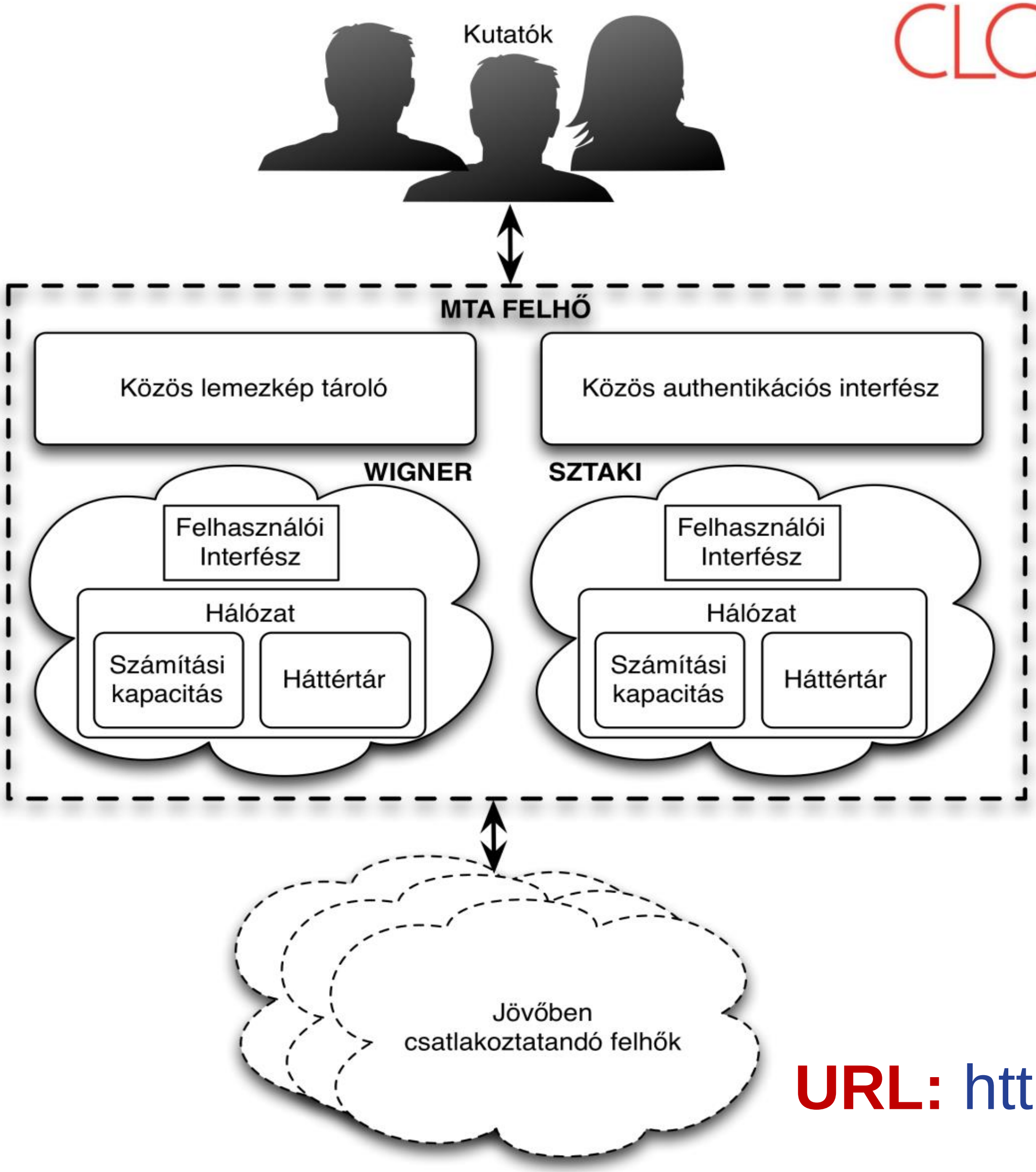
Network:
40 Gb Infiniband for HPC
10 Gb copper
1 Gb copper for mgm.
8/16 Gb FC for SAN
Connected to HBONE

Cloud middleware
OpenStack / HPE Helion

Power consumption
8-16 kW

MTA Cloud: federated resources

- Dedicated for academic research teams
- Fully operational from 1st Oct
- Wigner data center:
  - 800 vCPU
  - 2.5 TB RAM
  - 400 TB HDD
- MTA SZTAKI
  - 360 vCPU
  - 832 GB RAM
  - 164 TB HDD
- Based on OpenStack

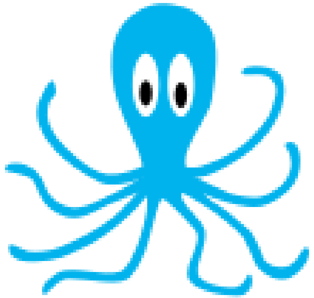


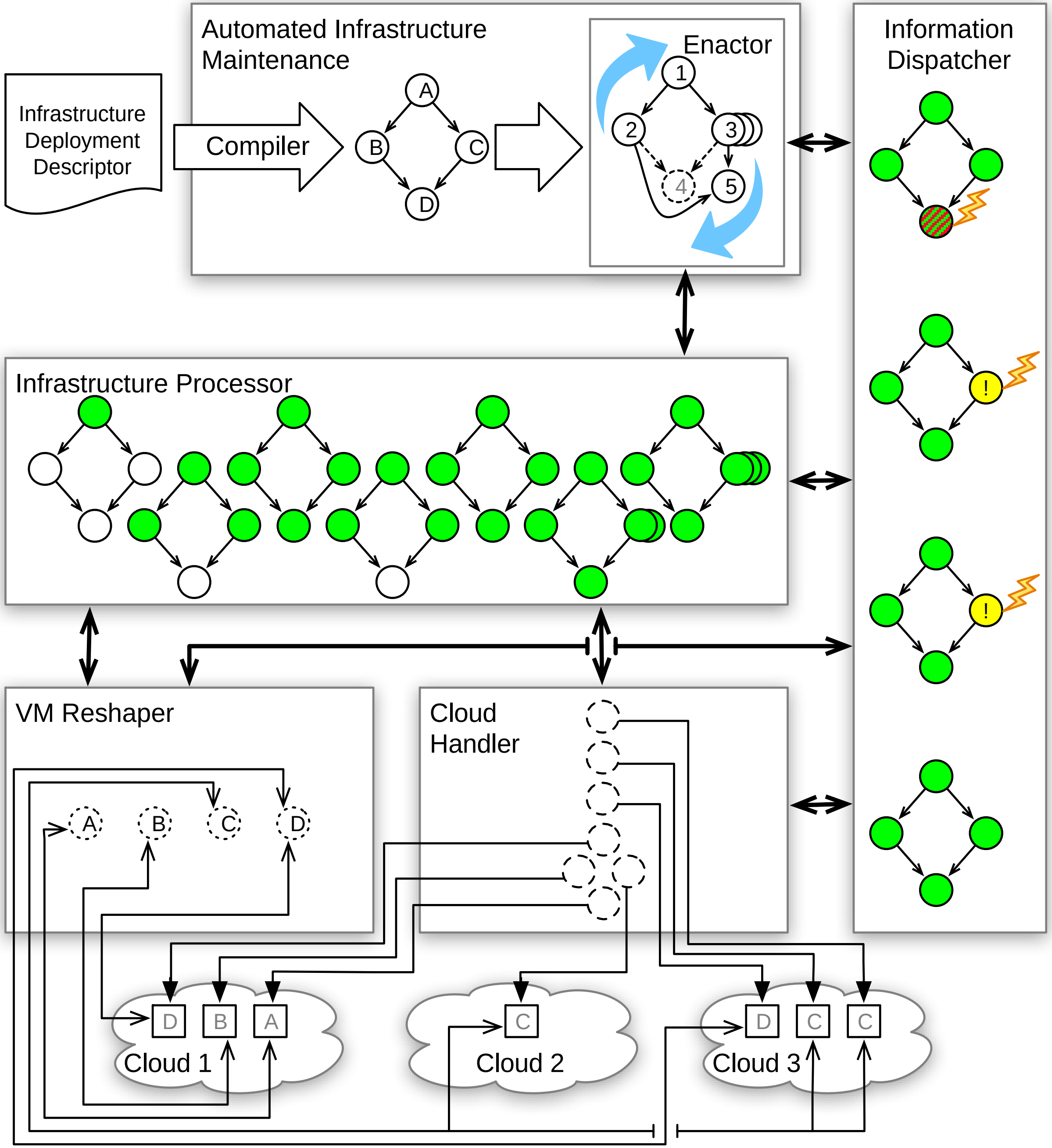
URL: http://cloud.mta.hu

Occopus

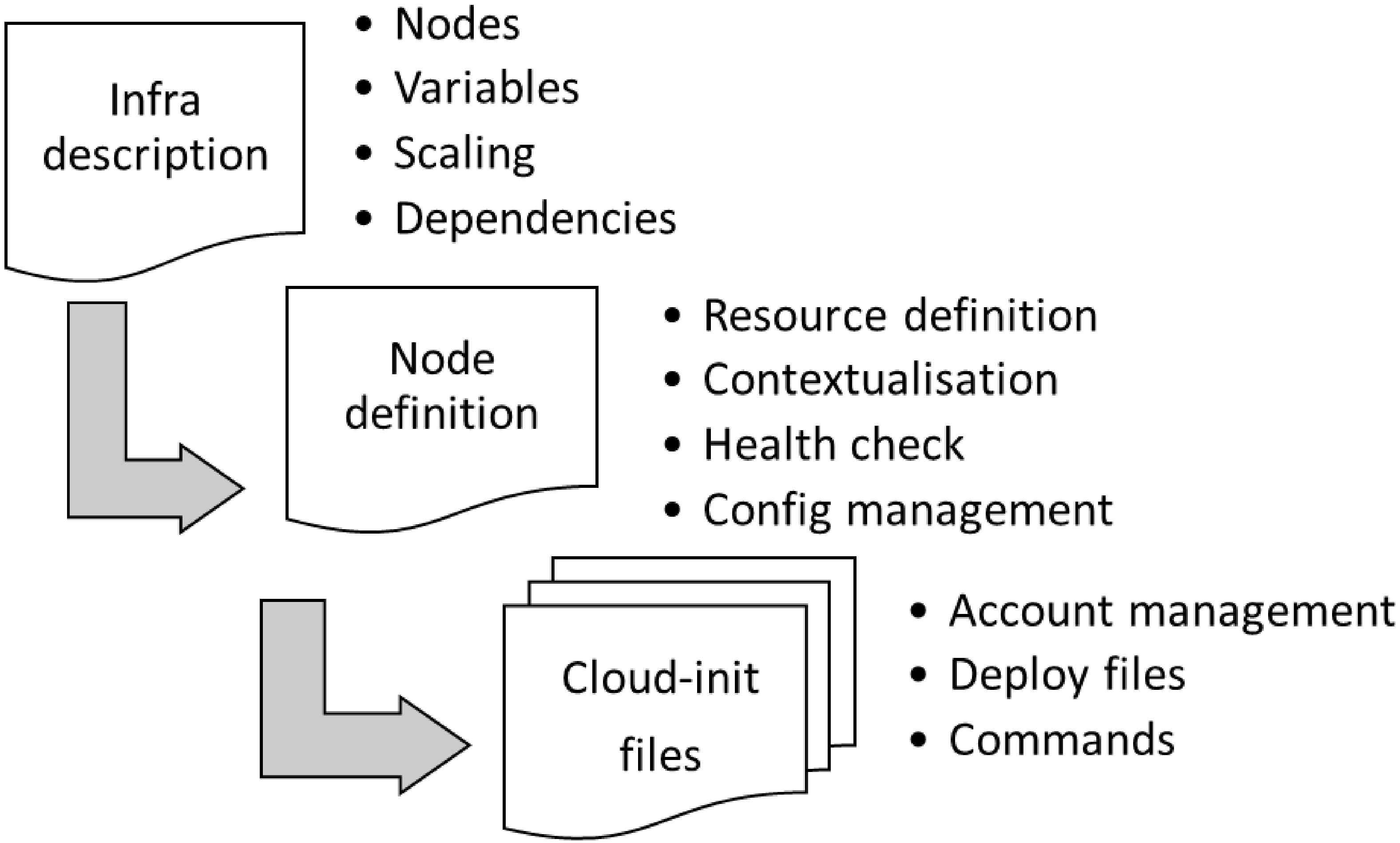
- Hybrid, cloud orchestrator tool by MTA SZTAKI

- Why Occopus?
  - Multi-cloud solution
  - Contextualization with cloud-init
  - Portable descriptor file
  - Enable scaling
  - No vendor lock-in





Describing an infrastructure by Occopus



Infra description

- Nodes
- Variables
- Scaling
- Dependencies

Node definition

- Resource definition
- Contextualisation
- Health check
- Config management

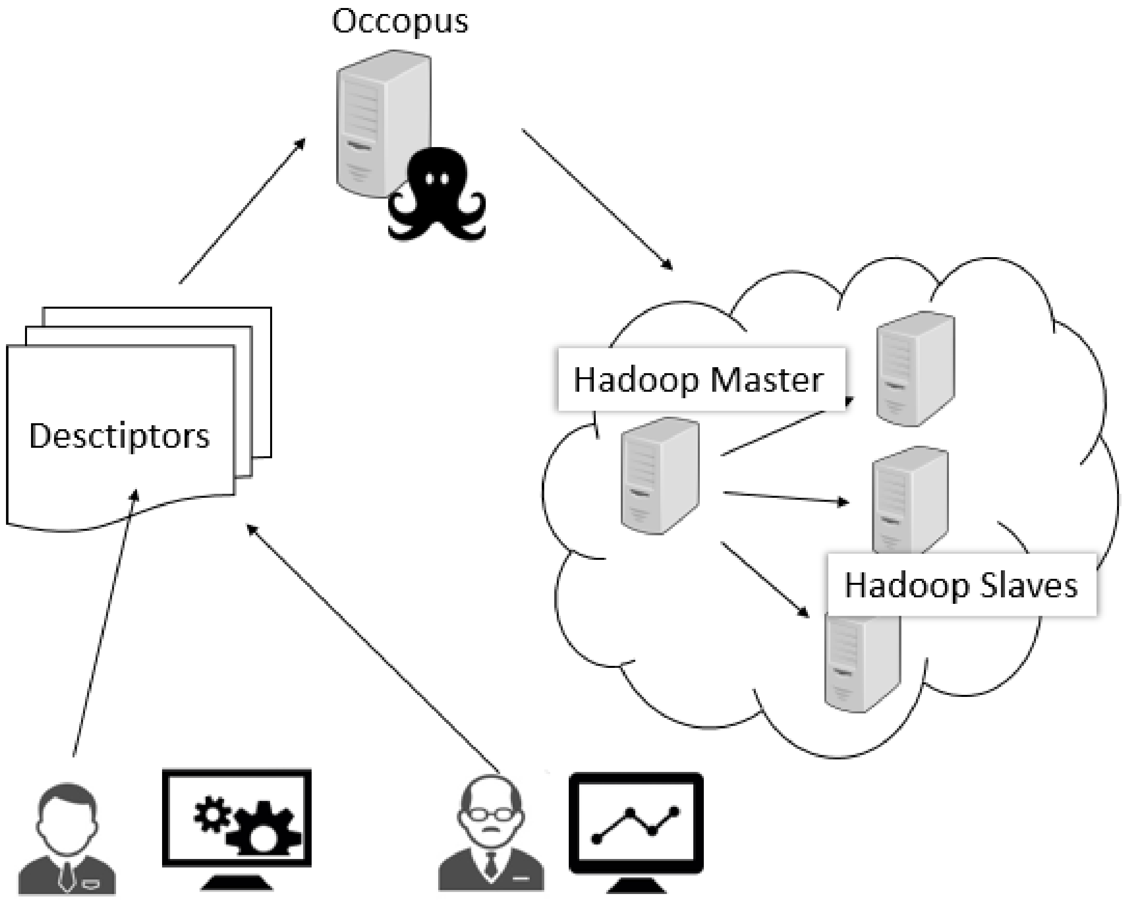
Cloud-init files

- Account management
- Deploy files
- Commands

Implementation

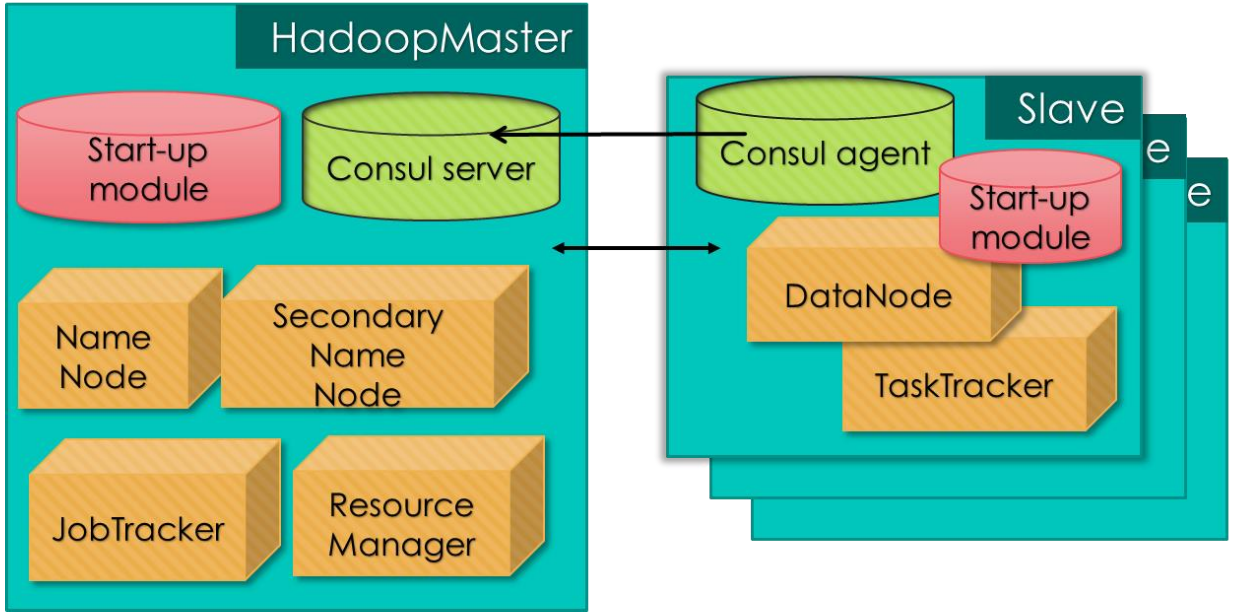
Architecture at large

- Advanced virtual infrastructure **operators** can **create** descriptors
- **End-users apply** them to create virtual infrastructures
- **Occopus** builds the Hadoop virtual infrastructure based on descriptors
  - 1st step: building Hadoop Master
  - 2nd step: simultaneous building of Hadoop Slaves
- **Hadoop Master node**
  - Supervise the mechanism of data storing in HDFS
  - Running parallel computations (MapReduce) on all that data
- **Hadoop Slave nodes**
  - Store data
  - Run the computations



Implementation II.

Overview of the Hadoop cluster architecture
Running Hadoop deamons and modules



On Hadoop Master node:

- **NameNode:** oversees and coordinates the data storage function
- **JobTracker:** oversees and coordinates the parallel processing of data using MapReduce
- **Secondary NameNode:** checkpoint in HDFS, helper node for NameNode
- **ResourceManager:** helps manage the distributed application by arbitrating all available cluster resources in the system
- **Start-up module:** prepare the Hadoop Master of the Hadoop cluster

Consul server:

- For Hadoop Slave node list, works as a Name Service

On Hadoop Slave nodes:

DataNode: serve read/write requests, perform block creation, deletion, replication

TaskTracker: accepts tasks - Map, Reduce and Shuffle operations - from a JobTracker.

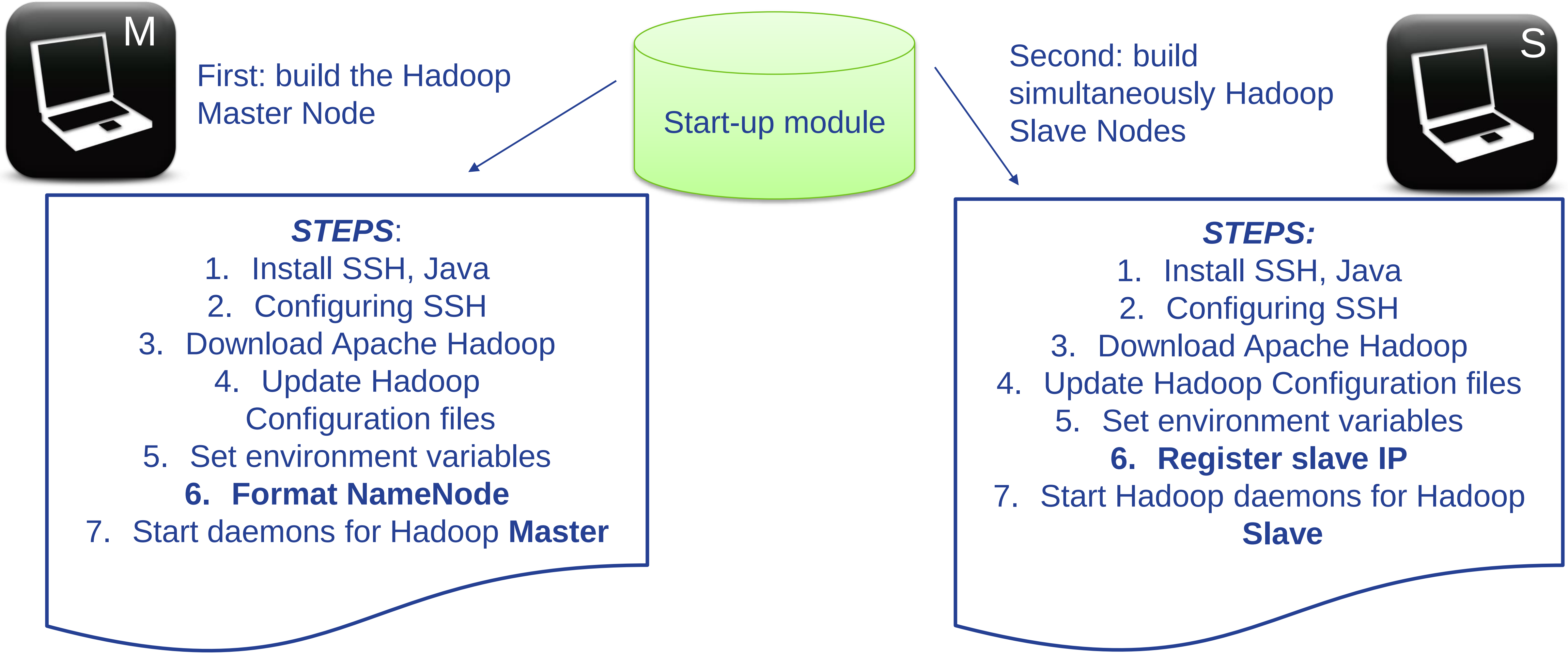
Start-up module:

- prepare the Hadoop Slave nodes of the Hadoop cluster

Consule agent

Implementation III.

Start-up module (at large)



First: build the Hadoop Master Node

Second: build simultaneously Hadoop Slave Nodes

STEPS:

1. Install SSH, Java
2. Configuring SSH
3. Download Apache Hadoop
4. Update Hadoop Configuration files
5. Set environment variables
6. **Format NameNode**
7. Start daemons for Hadoop **Master**

STEPS:

1. Install SSH, Java
2. Configuring SSH
3. Download Apache Hadoop
4. Update Hadoop Configuration files
5. Set environment variables
6. **Register slave IP**
7. Start Hadoop daemons for Hadoop **Slave**

Consul service

Difficulties:

1. **Scaling down Hadoop cluster?**

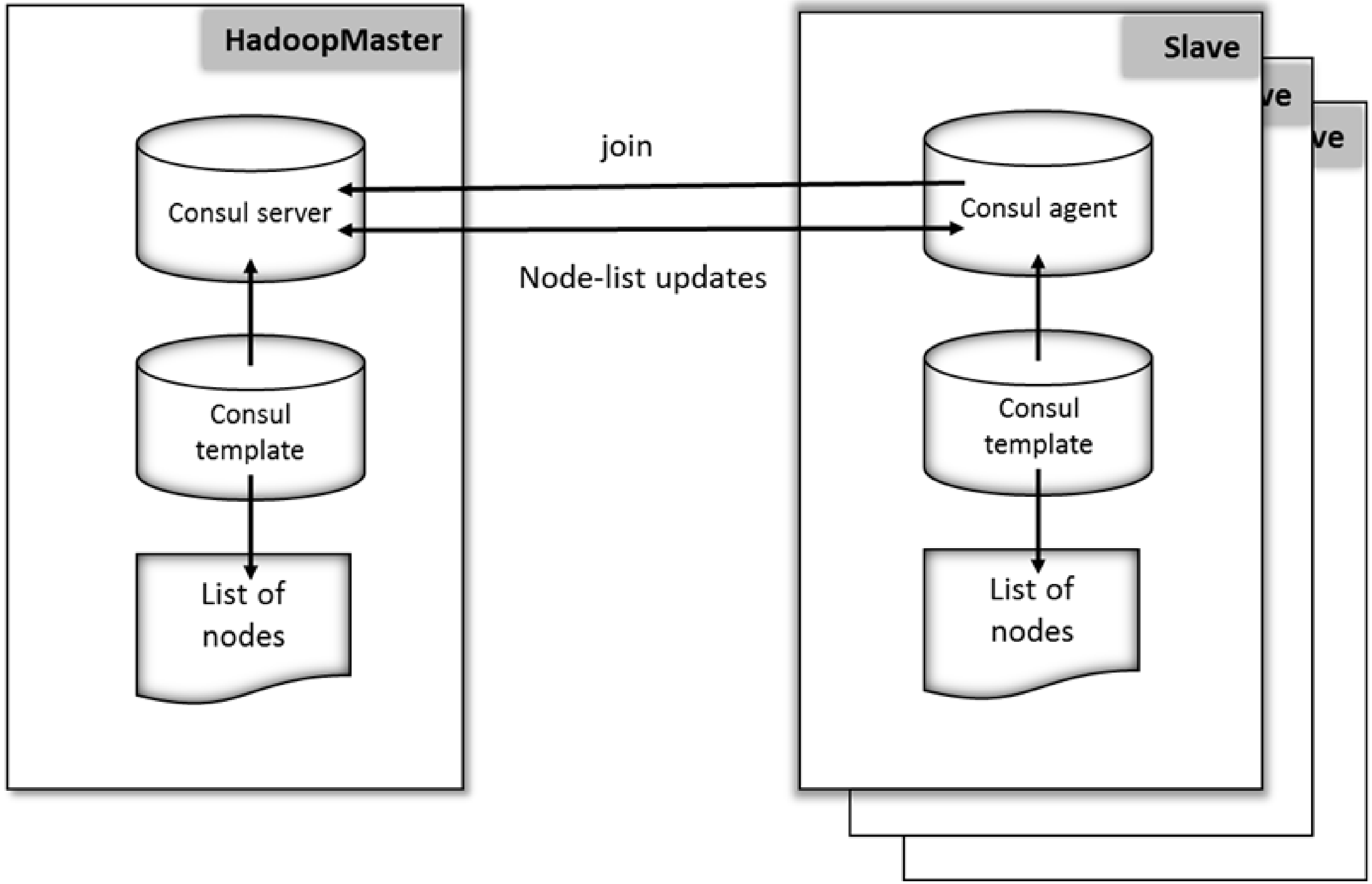
- 72 hours deadline ➔ service

2. **Deployment sequence**

- Consul first?
- Hadoop first?
- Initially with code?

3. Race situation

- Public IP queury



How to build Hadoop infrastructure with Occopus?

1. Install Occopus

- Follow the steps: *http://occopus.lpds.sztaki.hu/get-started*

2. Obtain Hadoop infrastructure descriptors

3. Make sure Occopus is activated:

- $ source ~/occopus/bin/activate

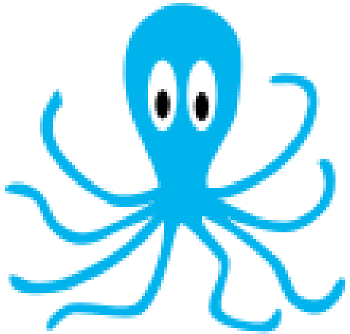
4. Import node definitions:

- $ occopus-import nodes/node_definitions.yaml

5. Start building process:

- $ occopus-build --parallelize infra-hadoop.yaml

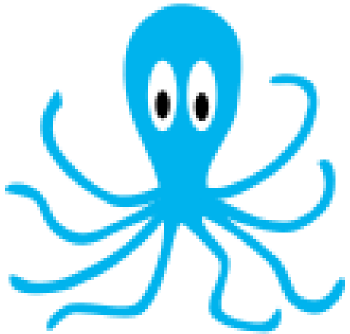
  (you will get INFRA_ID)



How to scale-up & scale-down with Occopus?

Scaling is a two-phase operation: register the scaling request + scale up/down the selected infrastructure by building new nodes /destroying old ones

1. **$ occopus-scale**
   - Registers scaling requests
   - Usage: *occopus-scale -n hadoop_slave -c COUNT -i INFRA_ID*
   - Count: positive/negative number expressing the direction and magnitude of scaling
2. **$ occopus-maintain**
   - Requests are handled and realized by this command
   - Usage: *occopus-maintain -i INFRA_ID*



For more information visit: http://occopus.lpds.sztaki.hu/

Results tested on various platforms

- **SZTAKI Cloud** (https://cfe2.lpds.sztaki.hu) ✓
  (Amazon – EC2)
- **MTA Cloud** (https://sztaki.cloud.mta.hu) ✓
  (NOVA)
- In progress: **Agrodat** Big Data center
- In progress: **Microsoft Azure**

OpenNebula

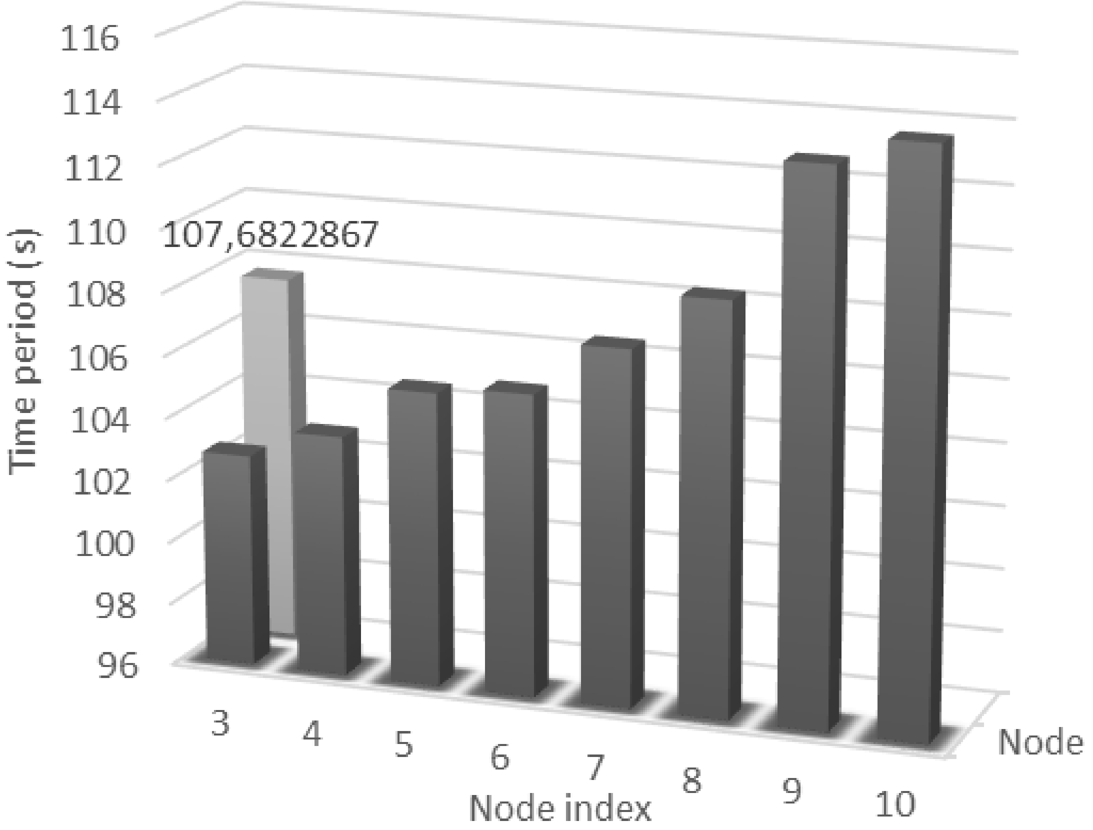
openstack

HPE Helion

Microsoft Azure

Performance

- Performance measurement:
  - Deployment implementation
- Scenario: scale up a cluster from 2 slave nodes to 10
- On MTA Cloud
- Diagram about the scaling-up performance (elasticity)
- Average mean time: 107,8 second



Related work

University of Westminster

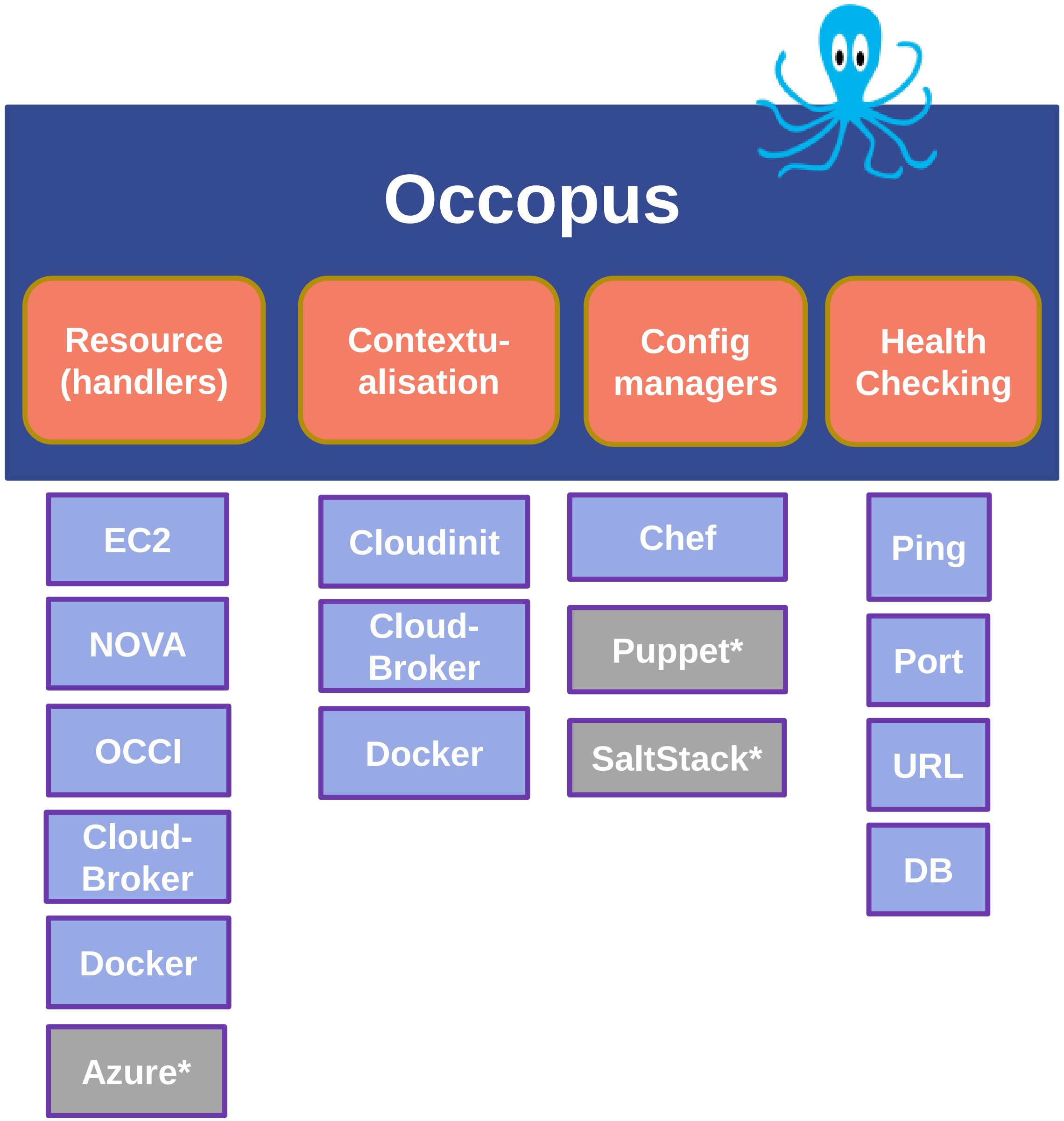
- WS-PGRADE portal – workflow based Hadoop deployment solution
- Works on EGI-FedCloud and with Cloudbroker
- Designed for short term Hadoop jobs

Amazon Web Services – Elastic MapReduce

- Vendor lock-in
- Commercial cloud
- Black-box services

Current state of Occopus

- Open-source (License: Apache v2)
- 6 releases so far (latest in August 2016)
- **Now: Release v1.2 (3rd production release)**
- Based on Python 2.7
- Webpage: http://occopus.lpds.sztaki.hu
- Git: https://github.com/occopus
- Documentation:
  - Users' Guide
  - Developers' Guide
  - Tutorials (e.g. building docker/swarm cluster)
- Package repository: http://pip.lpds.sztaki.hu/packages



* Coming soon…

Conclusion and future work

Conclusion:

- Portable
- Scalable
- Not depends on precompiled images
- Fully automatized infrastructure deployment by Occopus (one command)
- For data scientists

Future work:

- Automatic protection for scaling down too fast (data loss)
- Automatic scaling of overloaded Hadoop cluster
- Web UI – to help users

THANK YOU!

MTA SZTAKI
Hungarian Academy of Sciences
Institute for Computer Science and Control

http://occopus.lpds.sztaki.hu

Robert.Lovas@sztaki.mta.hu

Óbuda University
John von Neumann
Faculty of Informatics

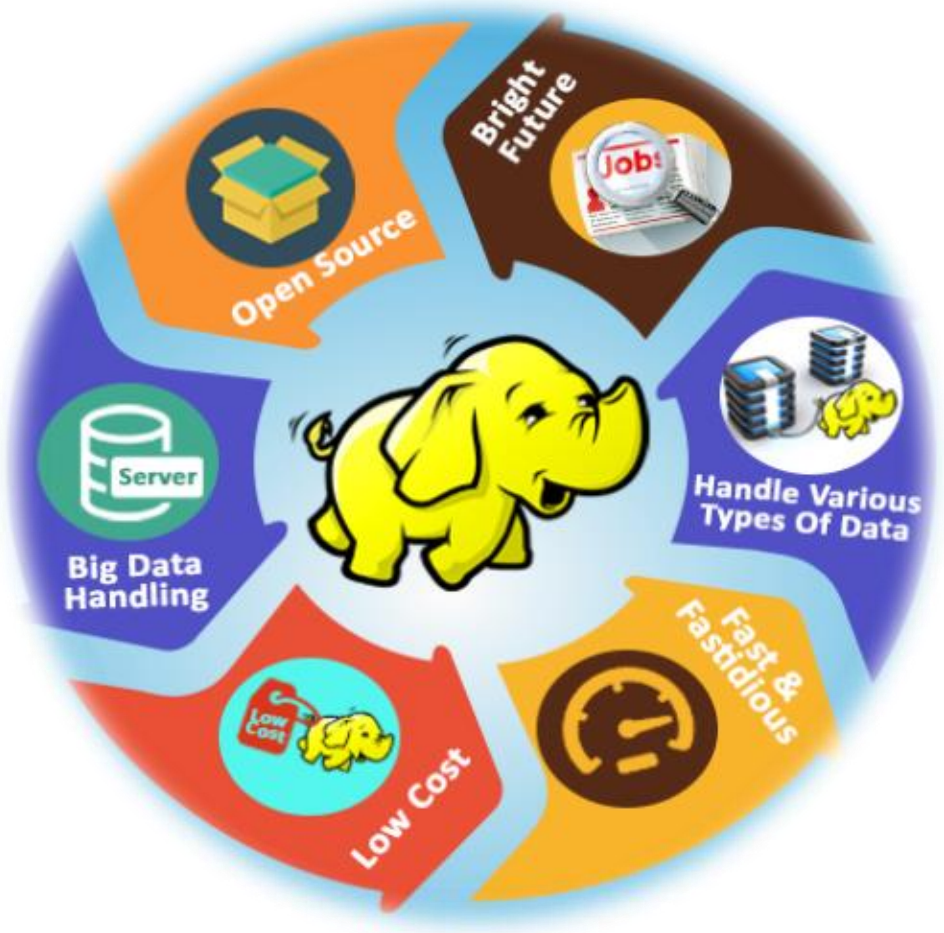
=== SPARE SLIDES ===

Big Data and cloud computing

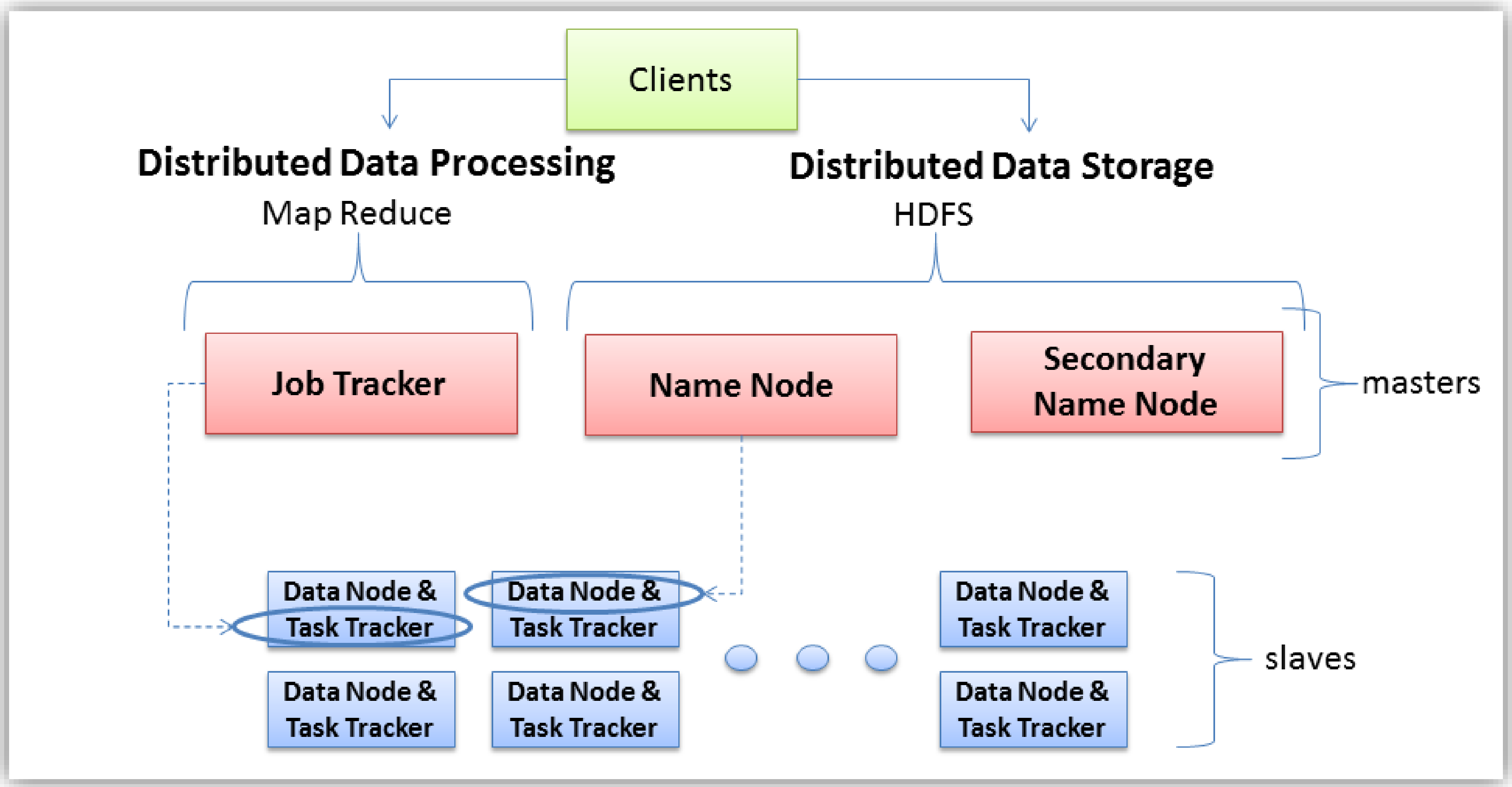
- Meaning of Big Data
- Benefits of Cloud
  - Cost effective
  - Increased storage
  - Scalability
  - Mobility
  - Always-on availability

Hadoop - 5 major advantages

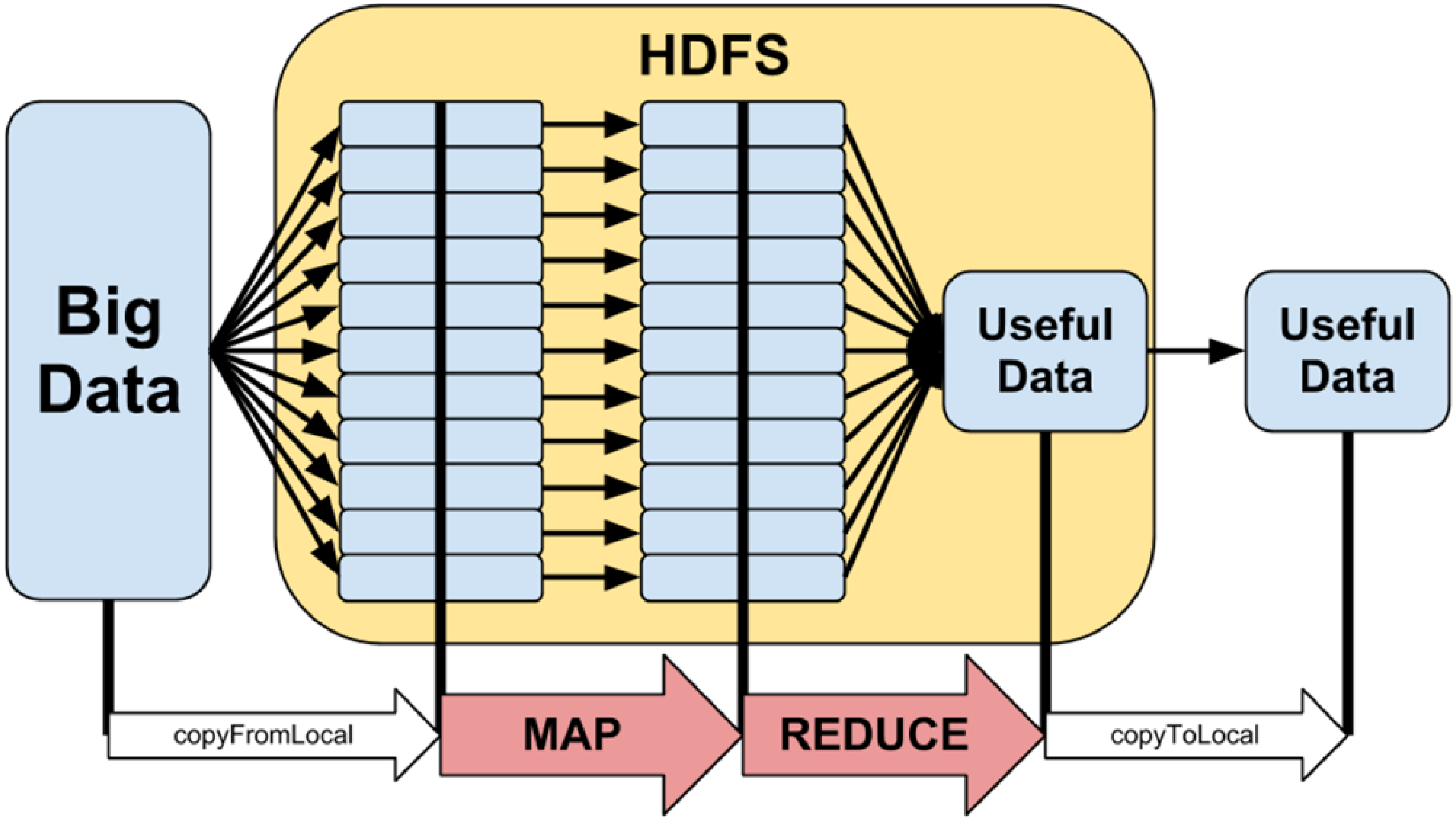
- Fast
- Flexible
- Resilient to failure
- Cost effective
- Scalable



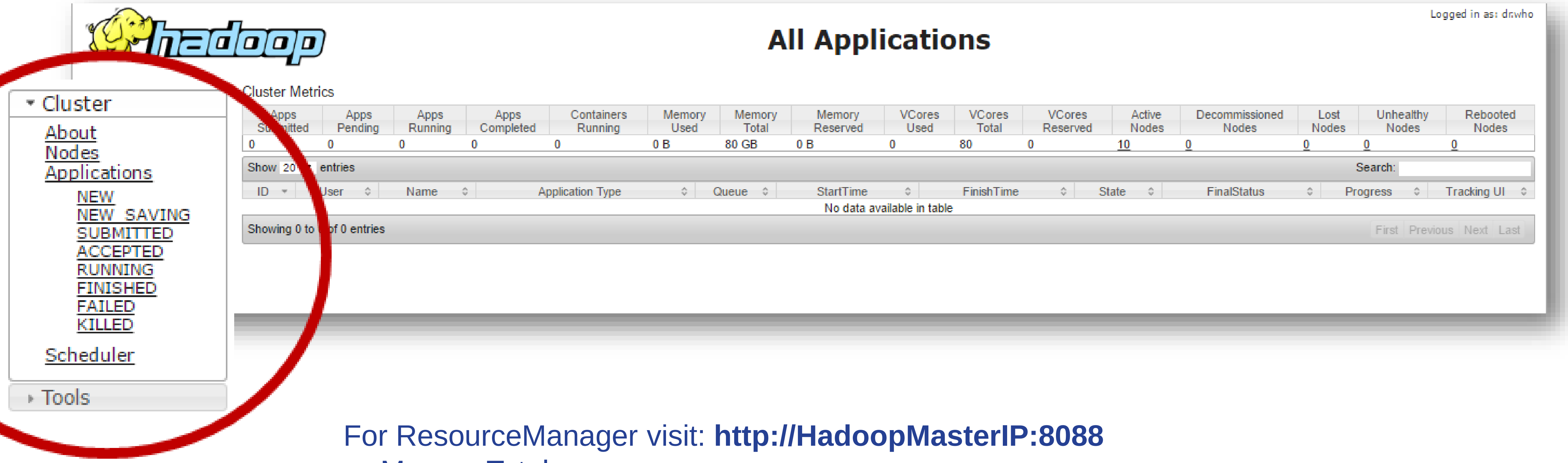
Hadoop - Roles



Hadoop- HDFS and MapReduce process



Hadoop Master Web User interfaces 1.



Logged in as: dr.who

All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
0	0	0	0	0	0 B	80 GB	0 B	0	80	0	10	0	0	0	0

Show 20 entries Search:

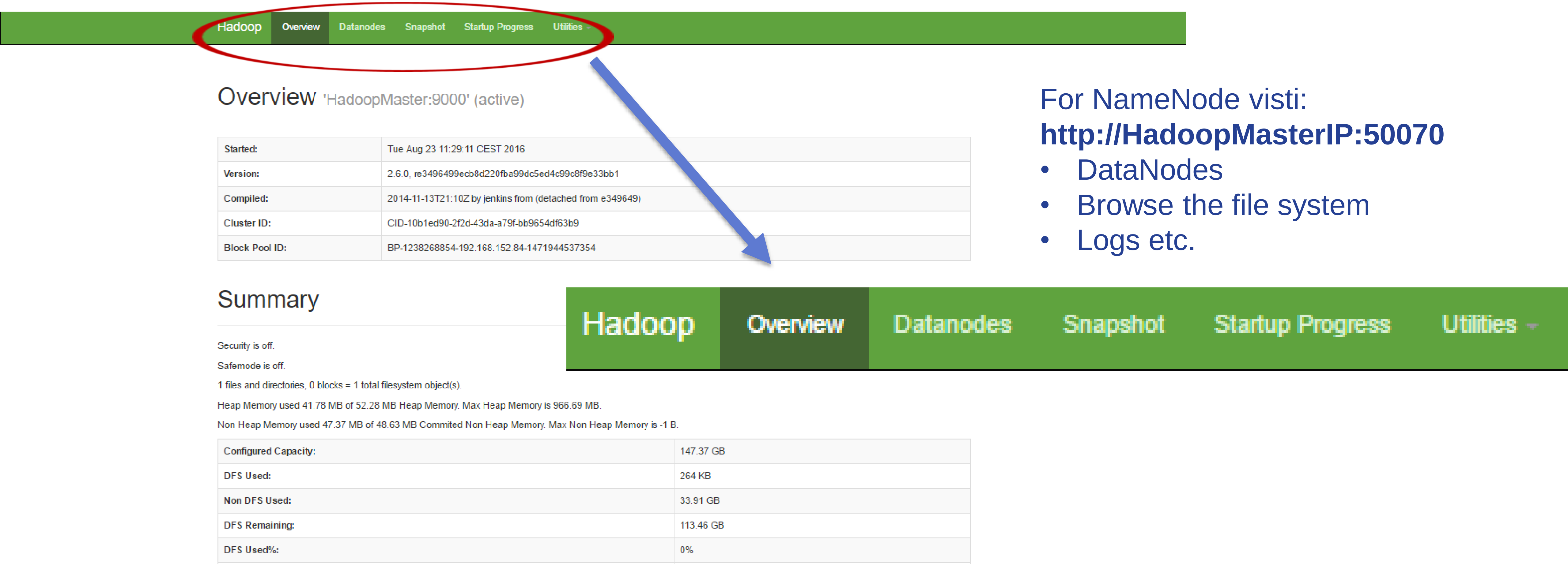
ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
No data available in table										

Showing 0 to 0 of 0 entries First Previous Next Last

For ResourceManager visit: **http://HadoopMasterIP:8088**

- MemoryTotal
- VCoresTotal
- Active Nodes etc.

Hadoop Master Web User interfaces 2.



Overview 'HadoopMaster:9000' (active)

Started:	Tue Aug 23 11:29:11 CEST 2016
Version:	2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1
Compiled:	2014-11-13T21:10Z by jenkins from (detached from e349649)
Cluster ID:	CID-10b1ed90-2f2d-43da-a79f-bb9654df63b9
Block Pool ID:	BP-1238268854-192.168.152.84-1471944537354

Summary

Security is off.

Safemode is off.

1 files and directories, 0 blocks = 1 total filesystem object(s).

Heap Memory used 41.78 MB of 52.28 MB Heap Memory. Max Heap Memory is 966.69 MB.

Non Heap Memory used 47.37 MB of 48.63 MB Commited Non Heap Memory. Max Non Heap Memory is -1 B.

Configured Capacity:	147.37 GB
DFS Used:	264 KB
Non DFS Used:	33.91 GB
DFS Remaining:	113.46 GB
DFS Used%:	0%

For NameNode visti:
http://HadoopMasterIP:50070

- DataNodes
- Browse the file system
- Logs etc.

How to run a Hadoop MapReduce job?
1. Inputs – on Hadoop Master node

After building-up a virtual Hadoop infrastructure we can run MapReduce job on it, follow these steps:

1.SSH to HadoopMaster node

2.Use commands as hduser

3.Sample text-files as input:

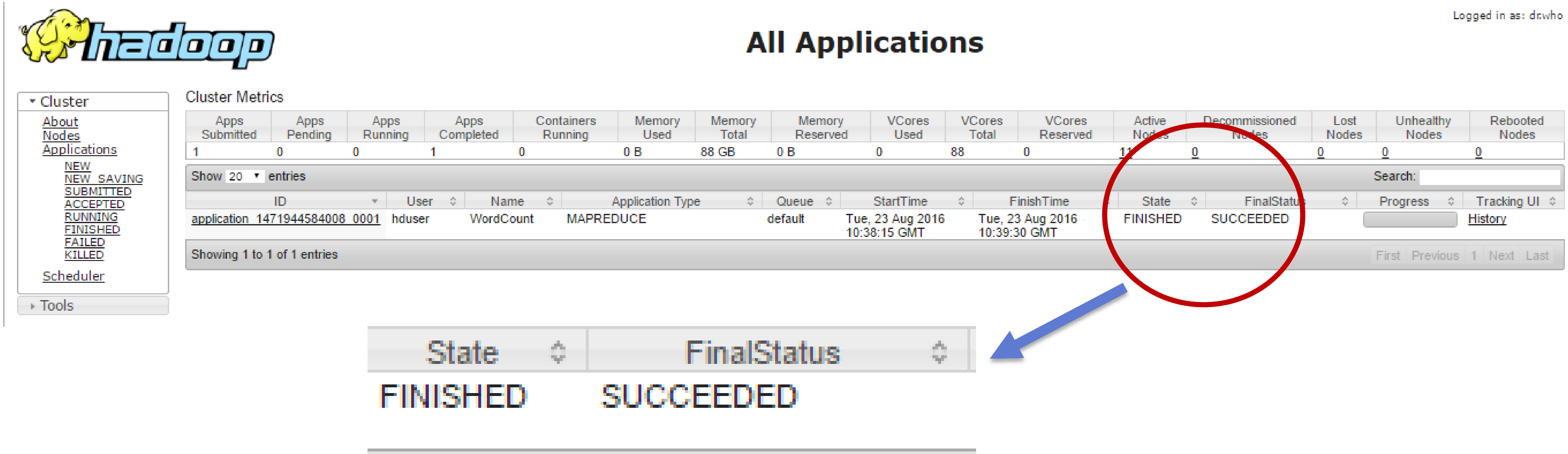
- Hduser:$ cat input/**file01**
  - Hello World, Bye World!
- A **wordcount.jar file**

4.Import inputs to HDFS:

- $HADOOP_HOME/bin/hadoop fs -mkdir /input
- $HADOOP_HOME/bin/hadoop fs -put /home/hduser/input/file01.txt /input

How to run a Hadoop MapReduce job?
2. Run a Hadoop job – on Hadoop Master node

- Use this command as **hduser on Hadoop Master node**:
- **$HADOOP_HOME/bin/hadoop jar /home/hduser/input/wordcount.jar org.myorg.WordCount /input /output**
- To check, read console or visite: http://HadoopMasterIP:8088:



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
1	0	0	1	0	0 B	88 GB	0 B	0	88	0	11	0	0	0	0

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1471944584008_0001	hduser	WordCount	MAPREDUCE	default	Tue, 23 Aug 2016 10:38:15 GMT	Tue, 23 Aug 2016 10:39:30 GMT	FINISHED	SUCCEEDED		History

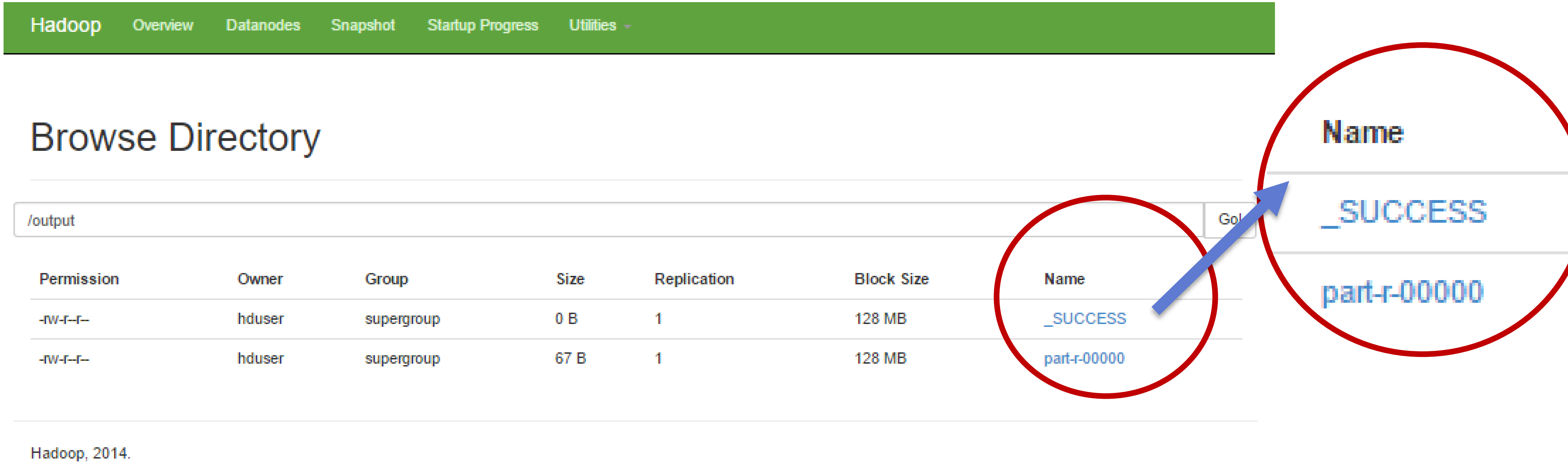
Showing 1 to 1 of 1 entries

State	FinalStatus
FINISHED	SUCCEEDED

How to run a Hadoop MapReduce job?

2. Run a Hadoop job – on Hadoop Master node

- To check the **output** of the MapReduce job visit web UI of the NameNode: http://HadoopMasterIP:50070
- **Choose**: Utilities -> Browse the file system -> select /output -> **download** part-r-00000



Browse Directory

/output

Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	hduser	supergroup	0 B	1	128 MB	_SUCCESS
-rw-r--r--	hduser	supergroup	67 B	1	128 MB	part-r-00000

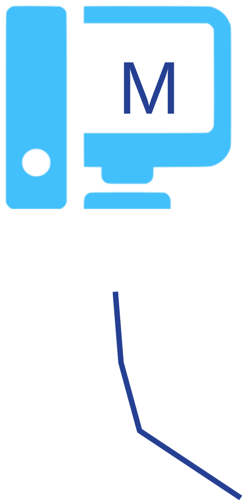
Hadoop, 2014.

Implementation

Inputs

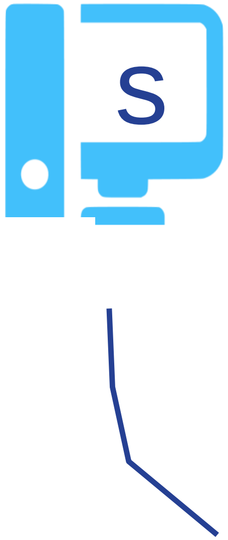
Hadoop configuration files:

- **Cores-site.xml**
  - Site-specific configuration for a given Hadoop installation
- **Hdfs-site.xml**:
  - Configurations for NameNode and DataNode
- **Yarn-site.xml**
  - Configurations for ResourceManager, NodeManager and History Server
- **Mapred-site.xml**
  - Configurations for MapReduce Applications and MapReduce JobHistory Server
- **Hadoop-env.sh**
  - Control the Hadoop scripts by setting site-specific values



SSH public key
SSH private key
Hadoop environment variables
Configuration files:

- Core-site.xml
- Hdfs-site.xml
- Yarn-site.xml
- Mapred-site.xml
- Hadoop-env.sh



SSH public key
SSH private key
Hadoop environment variables
HadoopMaster IP
Configuration files:

- Core-site.xml
- Hdfs-site.xml
- Yarn-site.xml
- Mapred-site.xml
- Hadoop-env.sh