# Bright Cluster Manager

## Using Slurm for Data Aware Scheduling in the Cloud



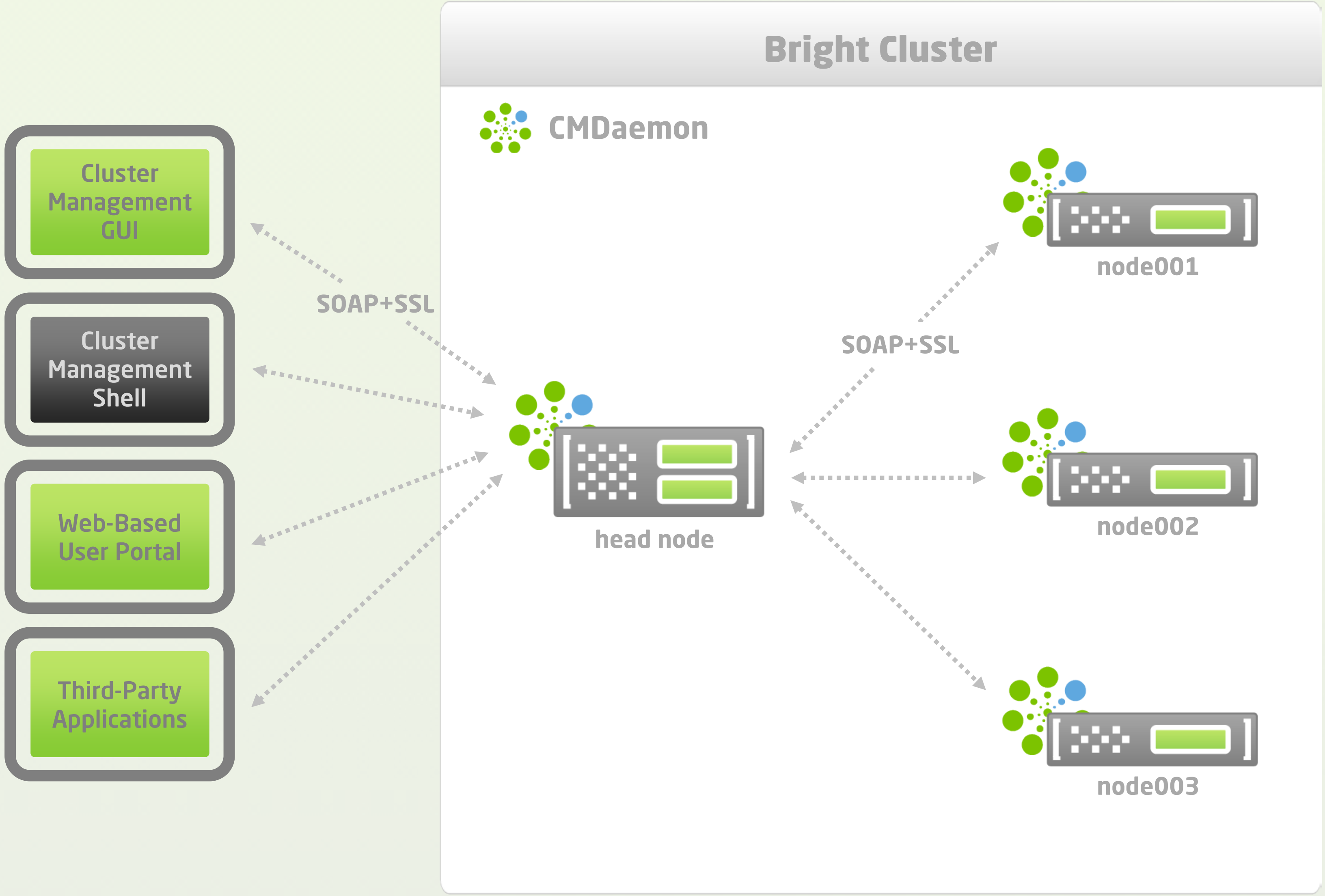
Martijn de Vries
CTO

# About Bright Computing

## Bright Computing

1. Develops and supports Bright Cluster Manager for HPC systems, server farms, grids and clouds
2. Incorporated in USA and The Netherlands (Offices in San Jose and Amsterdam)

# Architecture



# Management Interfaces

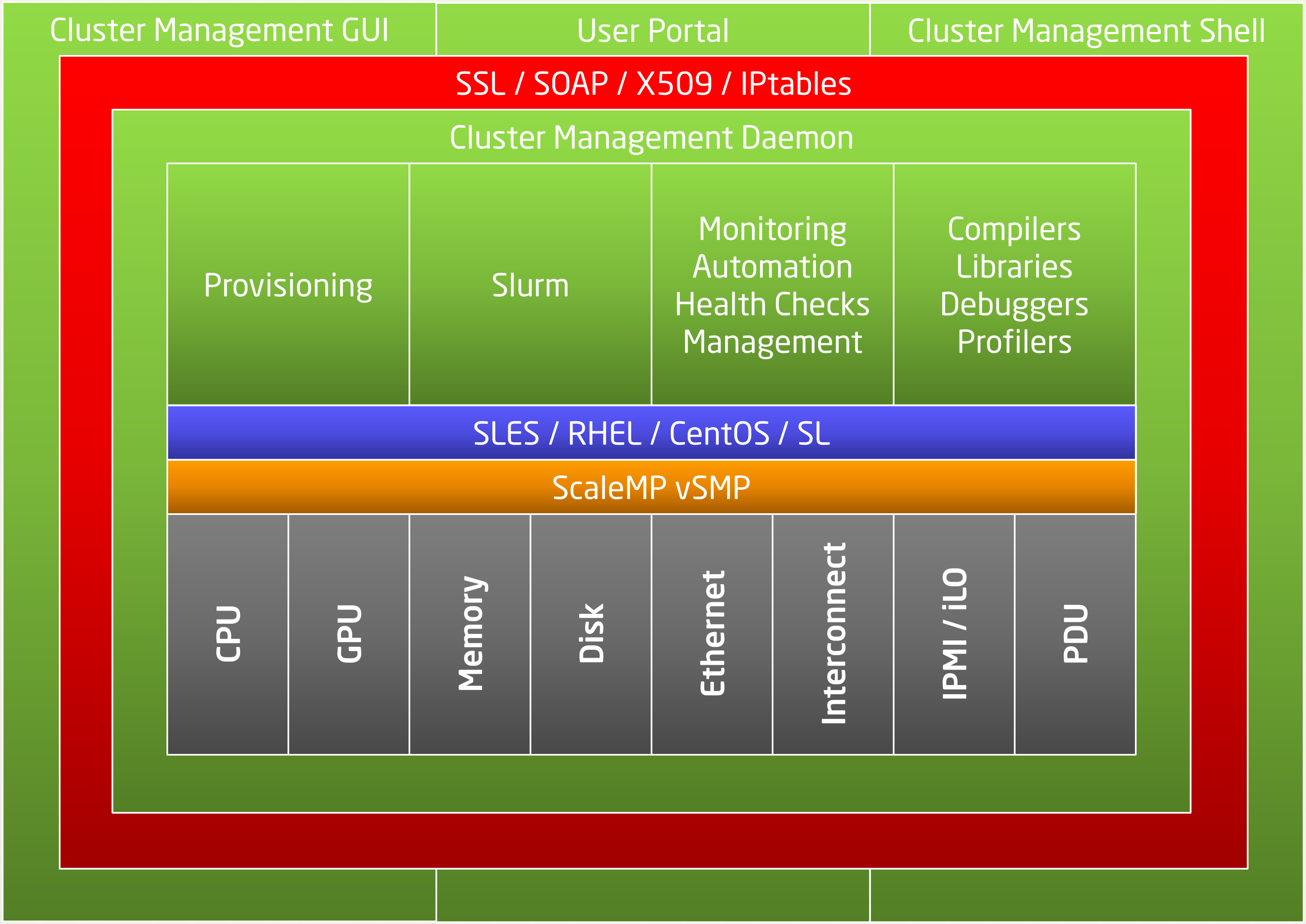
## Graphical User Interface (GUI)

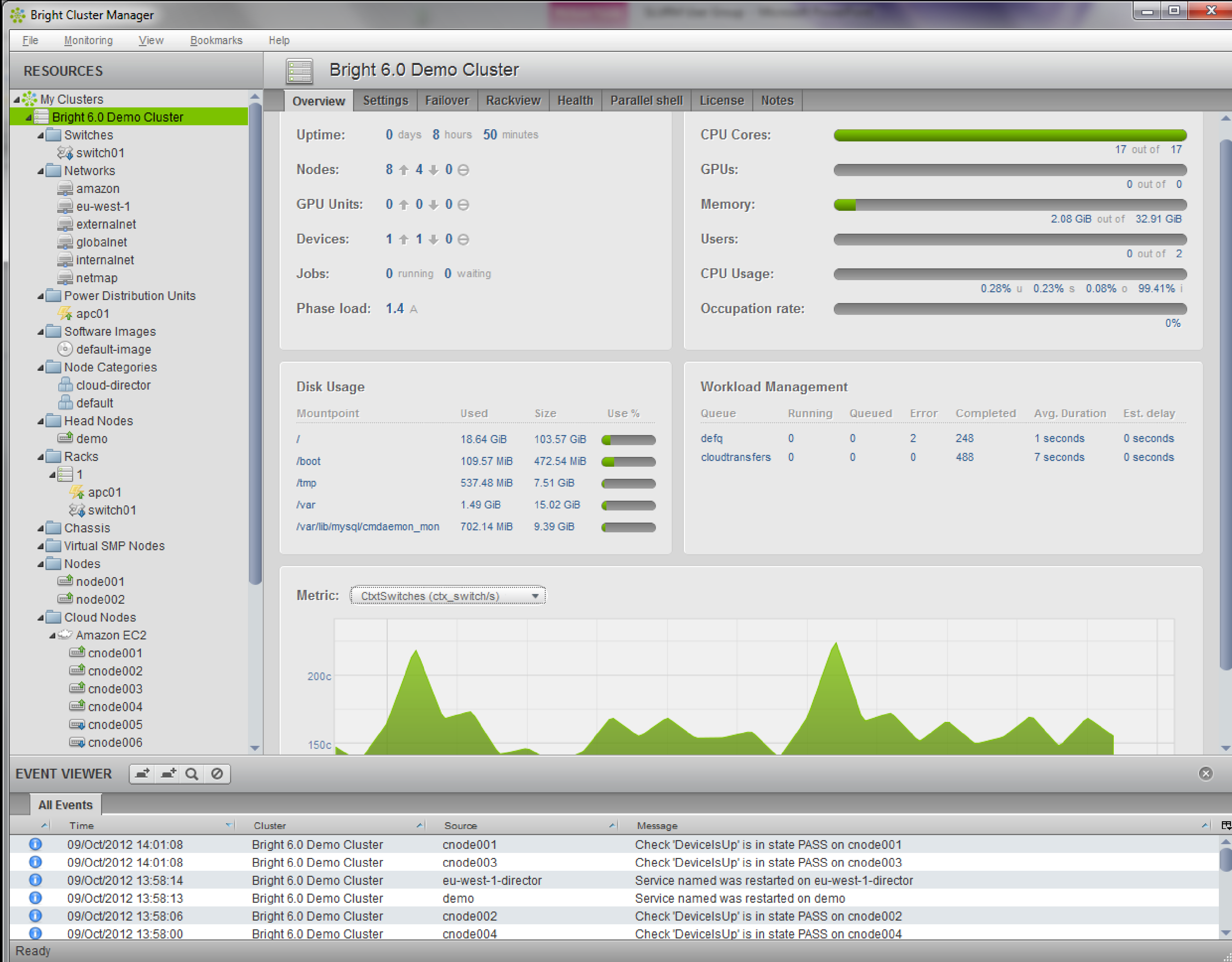
- Offers administrator full cluster control
- Standalone desktop application
- Manages multiple clusters simultaneously
- Runs on Linux & Windows
- Built on top of Mozilla XUL engine

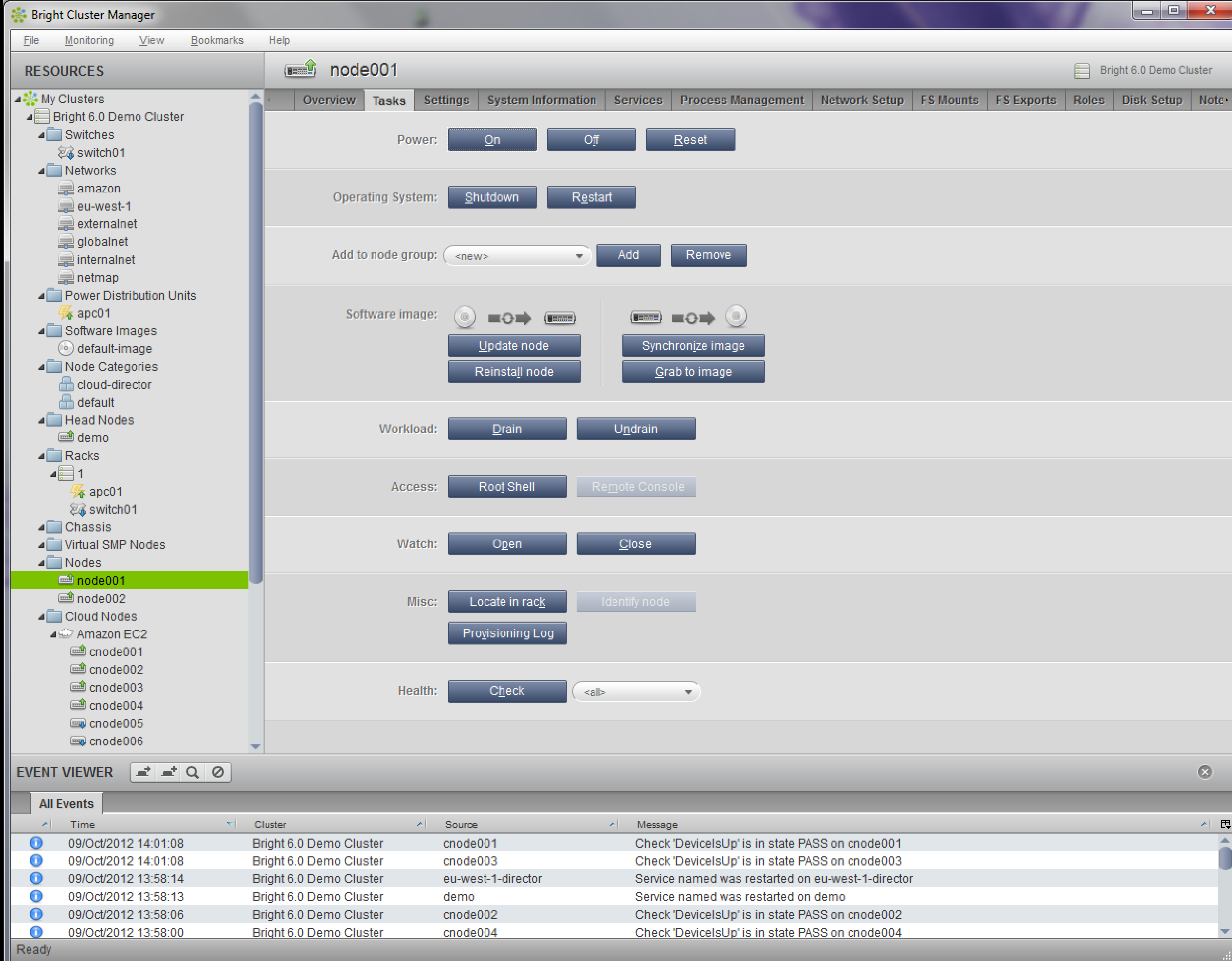

## Cluster Management Shell (CMSH)

- All GUI functionality also available through Cluster Management Shell
- Interactive and scriptable in batch mode

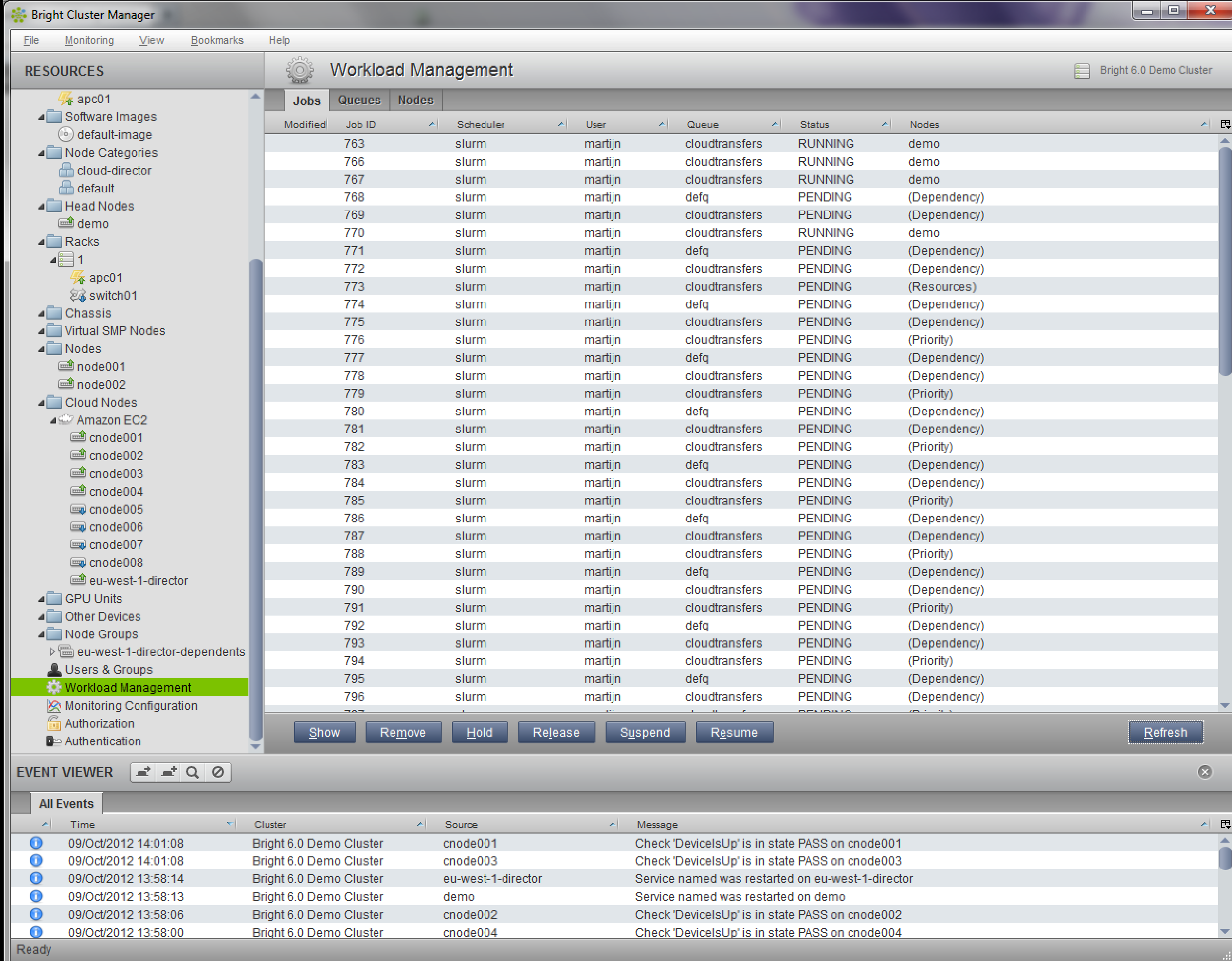
# Bright Cluster Manager – Elements




Bright Cluster Manager
File
Monitoring
View
Bookmarks
Help
RESOURCES
My Clusters
Bright 6.0 Demo Cluster
Switches
switch01
Networks
amazon
eu-west-1
externalnet
globalnet
internalnet
netmap
Power Distribution Units
apc01
Software Images
default-image
Node Categories
cloud-director
default
Head Nodes
demo
Racks
1
apc01
switch01
Chassis
Virtual SMP Nodes
Nodes
node001
node002
Cloud Nodes
Amazon EC2
cnode001
cnode002
cnode003
cnode004
cnode005
cnode006
Bright 6.0 Demo Cluster
Overview
Settings
Failover
Rackview
Health
Parallel shell
License
Notes
Uptime: 0 days 8 hours 50 minutes
Nodes: 8 4 0
GPU Units: 0 0 0
Devices: 1 1 0
Jobs: 0 running 0 waiting
Phase load: 1.4 A
CPU Cores:
17 out of 17
GPUs:
0 out of 0
Memory:
2.08 GiB out of 32.91 GiB
Users:
0 out of 2
CPU Usage:
0.28% u 0.23% s 0.08% o 99.41% i
Occupation rate:
0%
Disk Usage
Mountpoint Used Size Use %
/ 18.64 GiB 103.57 GiB
/boot 109.57 MiB 472.54 MiB
/tmp 537.48 MiB 7.51 GiB
/var 1.49 GiB 15.02 GiB
/var/lib/mysql/cmdaemon_mon 702.14 MiB 9.39 GiB
Workload Management
Queue Running Queued Error Completed Avg. Duration Est. delay
defq 0 0 2 248 1 seconds 0 seconds
cloudtransfers 0 0 0 488 7 seconds 0 seconds
Metric:
CtxtSwitches (ctx_switch/s)
200c
150c
EVENT VIEWER
All Events
Time Cluster Source Message
09/Oct/2012 14:01:08 Bright 6.0 Demo Cluster cnode001 Check 'DeviceIsUp' is in state PASS on cnode001
09/Oct/2012 14:01:08 Bright 6.0 Demo Cluster cnode003 Check 'DeviceIsUp' is in state PASS on cnode003
09/Oct/2012 13:58:14 Bright 6.0 Demo Cluster eu-west-1-director Service named was restarted on eu-west-1-director
09/Oct/2012 13:58:13 Bright 6.0 Demo Cluster demo Service named was restarted on demo
09/Oct/2012 13:58:06 Bright 6.0 Demo Cluster cnode002 Check 'DeviceIsUp' is in state PASS on cnode002
09/Oct/2012 13:58:00 Bright 6.0 Demo Cluster cnode004 Check 'DeviceIsUp' is in state PASS on cnode004
Ready

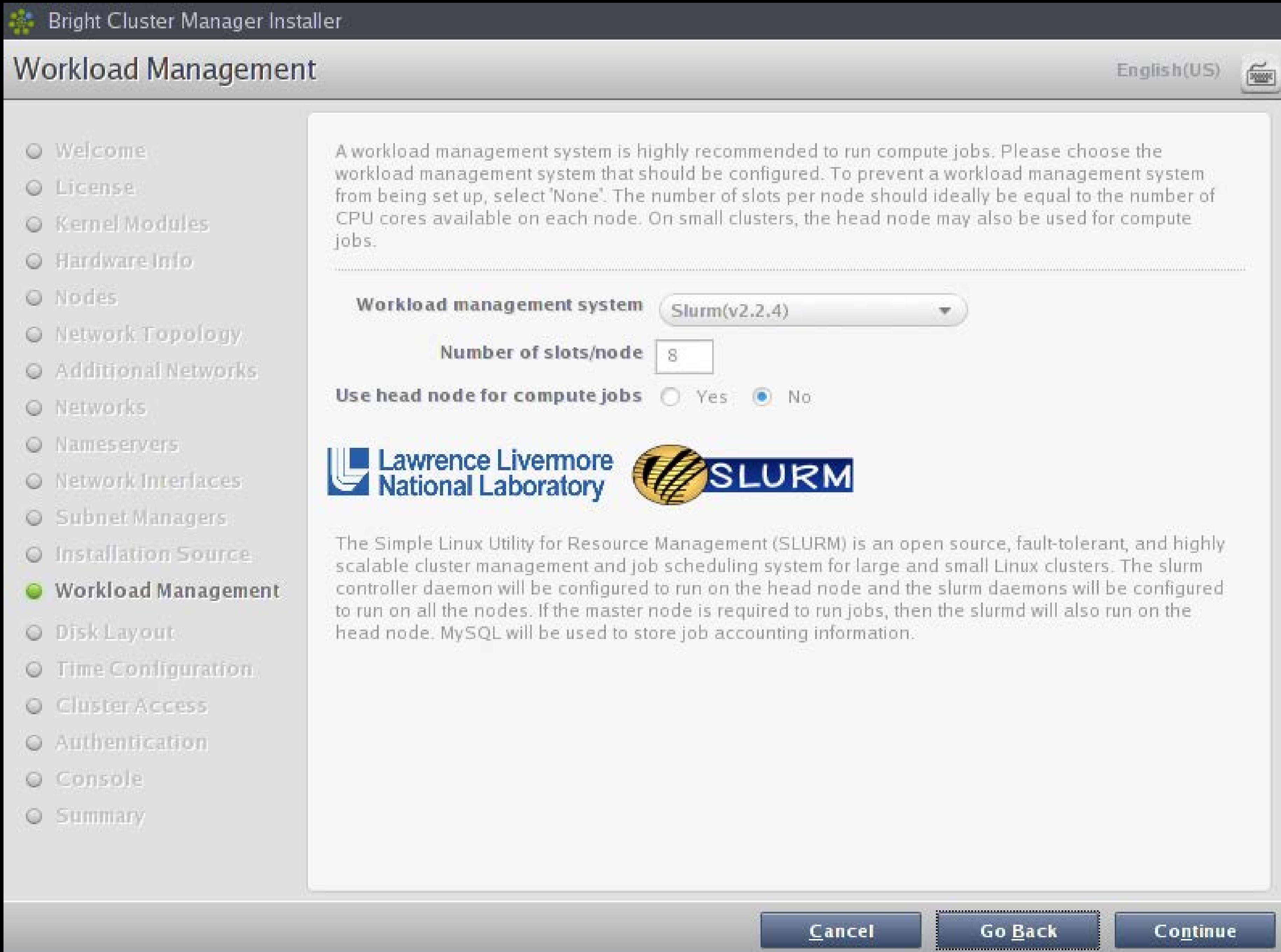


Bright Cluster Manager
File
Monitoring
View
Bookmarks
Help
RESOURCES
node001
Bright 6.0 Demo Cluster
My Clusters
Bright 6.0 Demo Cluster
Switches
switch01
Networks
amazon
eu-west-1
externalnet
globalnet
internalnet
netmap
Power Distribution Units
apc01
Software Images
default-image
Node Categories
cloud-director
default
Head Nodes
demo
Racks
1
apc01
switch01
Chassis
Virtual SMP Nodes
Nodes
node001
node002
Cloud Nodes
Amazon EC2
cnode001
cnode002
cnode003
cnode004
cnode005
cnode006
Overview
Tasks
Settings
System Information
Services
Process Management
Network Setup
FS Mounts
FS Exports
Roles
Disk Setup
Note
Power:
On
Off
Reset
Operating System:
Shutdown
Restart
Add to node group:
<new>
Add
Remove
Software image:
Update node
Reinstall node
Synchronize image
Grab to image
Workload:
Drain
Undrain
Access:
Root Shell
Remote Console
Watch:
Open
Close
Misc:
Locate in rack
Identify node
Provisioning Log
Health:
Check
<all>
EVENT VIEWER
All Events
Time
Cluster
Source
Message
09/Oct/2012 14:01:08
Bright 6.0 Demo Cluster
cnode001
Check 'DeviceIsUp' is in state PASS on cnode001
09/Oct/2012 14:01:08
Bright 6.0 Demo Cluster
cnode003
Check 'DeviceIsUp' is in state PASS on cnode003
09/Oct/2012 13:58:14
Bright 6.0 Demo Cluster
eu-west-1-director
Service named was restarted on eu-west-1-director
09/Oct/2012 13:58:13
Bright 6.0 Demo Cluster
demo
Service named was restarted on demo
09/Oct/2012 13:58:06
Bright 6.0 Demo Cluster
cnode002
Check 'DeviceIsUp' is in state PASS on cnode002
09/Oct/2012 13:58:00
Bright 6.0 Demo Cluster
cnode004
Check 'DeviceIsUp' is in state PASS on cnode004
Ready

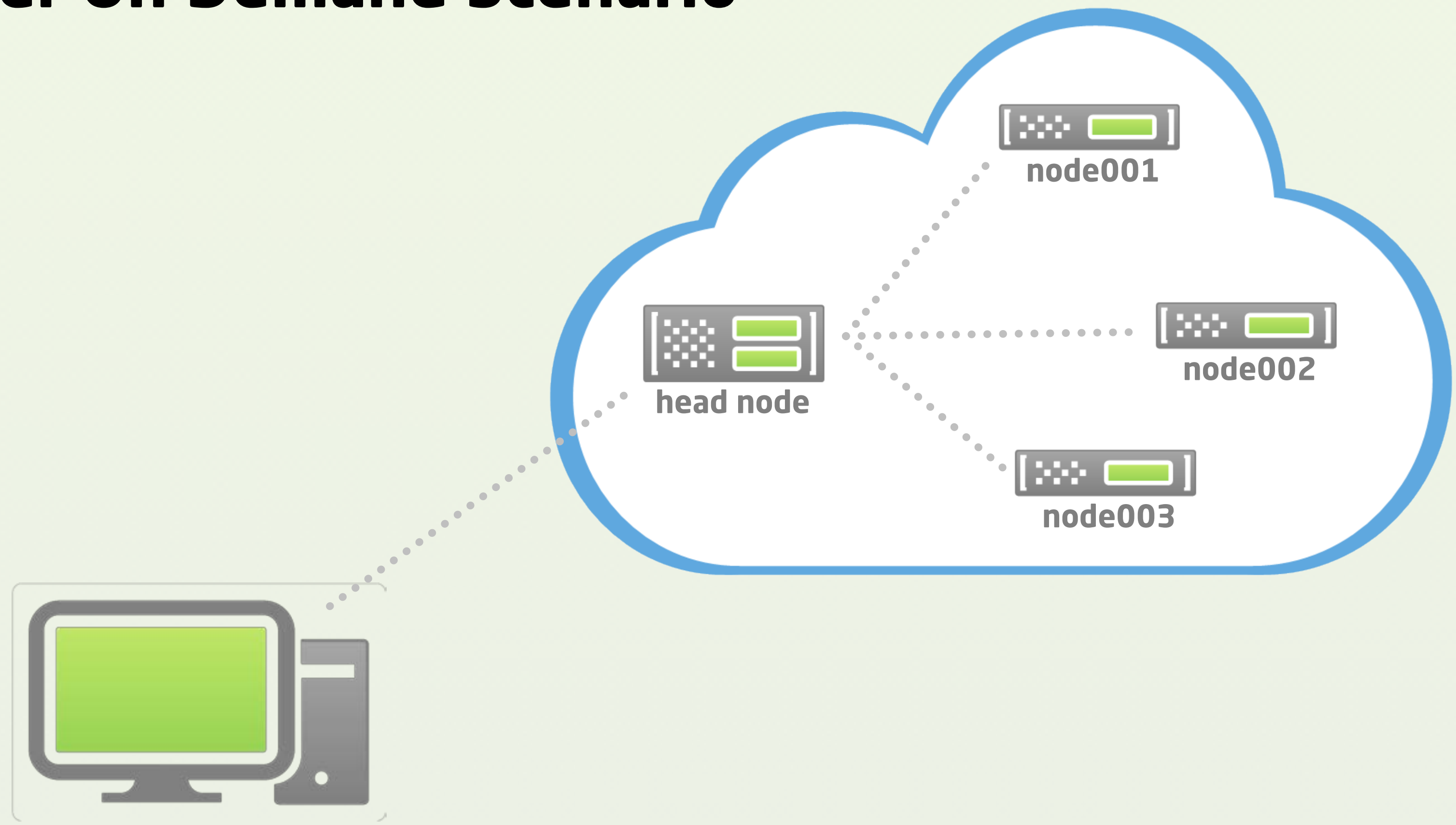

Bright Cluster Manager
File Monitoring View Bookmarks Help
RESOURCES
apc01
Software Images
default-image
Node Categories
cloud-director
default
Head Nodes
demo
Racks
1
apc01
switch01
Chassis
Virtual SMP Nodes
Nodes
node001
node002
Cloud Nodes
Amazon EC2
cnode001
cnode002
cnode003
cnode004
cnode005
cnode006
cnode007
cnode008
eu-west-1-director
GPU Units
Other Devices
Node Groups
eu-west-1-director-dependents
Users & Groups
Workload Management
Monitoring Configuration
Authorization
Authentication
Workload Management
Bright 6.0 Demo Cluster
Jobs Queues Nodes
Modified Job ID Scheduler User Queue Status Nodes
763 slurm martijn cloudtransfers RUNNING demo
766 slurm martijn cloudtransfers RUNNING demo
767 slurm martijn cloudtransfers RUNNING demo
768 slurm martijn defq PENDING (Dependency)
769 slurm martijn cloudtransfers PENDING (Dependency)
770 slurm martijn cloudtransfers RUNNING demo
771 slurm martijn defq PENDING (Dependency)
772 slurm martijn cloudtransfers PENDING (Dependency)
773 slurm martijn cloudtransfers PENDING (Resources)
774 slurm martijn defq PENDING (Dependency)
775 slurm martijn cloudtransfers PENDING (Dependency)
776 slurm martijn cloudtransfers PENDING (Priority)
777 slurm martijn defq PENDING (Dependency)
778 slurm martijn cloudtransfers PENDING (Dependency)
779 slurm martijn cloudtransfers PENDING (Priority)
780 slurm martijn defq PENDING (Dependency)
781 slurm martijn cloudtransfers PENDING (Dependency)
782 slurm martijn cloudtransfers PENDING (Priority)
783 slurm martijn defq PENDING (Dependency)
784 slurm martijn cloudtransfers PENDING (Dependency)
785 slurm martijn cloudtransfers PENDING (Priority)
786 slurm martijn defq PENDING (Dependency)
787 slurm martijn cloudtransfers PENDING (Dependency)
788 slurm martijn cloudtransfers PENDING (Priority)
789 slurm martijn defq PENDING (Dependency)
790 slurm martijn cloudtransfers PENDING (Dependency)
791 slurm martijn cloudtransfers PENDING (Priority)
792 slurm martijn defq PENDING (Dependency)
793 slurm martijn cloudtransfers PENDING (Dependency)
794 slurm martijn cloudtransfers PENDING (Priority)
795 slurm martijn defq PENDING (Dependency)
796 slurm martijn cloudtransfers PENDING (Dependency)
Show Remove Hold Release Suspend Resume Refresh
EVENT VIEWER
All Events
Time Cluster Source Message
09/Oct/2012 14:01:08 Bright 6.0 Demo Cluster cnode001 Check 'DeviceIsUp' is in state PASS on cnode001
09/Oct/2012 14:01:08 Bright 6.0 Demo Cluster cnode003 Check 'DeviceIsUp' is in state PASS on cnode003
09/Oct/2012 13:58:14 Bright 6.0 Demo Cluster eu-west-1-director Service named was restarted on eu-west-1-director
09/Oct/2012 13:58:13 Bright 6.0 Demo Cluster demo Service named was restarted on demo
09/Oct/2012 13:58:06 Bright 6.0 Demo Cluster cnode002 Check 'DeviceIsUp' is in state PASS on cnode002
09/Oct/2012 13:58:00 Bright 6.0 Demo Cluster cnode004 Check 'DeviceIsUp' is in state PASS on cnode004
Ready

# Integration with Slurm

- Slurm default choice for workload management system
- Slurm up and running at first boot
- Node & partition configuration
- Topology configuration
- HA configuration
- Workload management metrics
- Health checking
- Job monitoring and control
- Integrated in Cluster Management API


Bright Cluster Manager Installer
Workload Management
English(US)
Welcome
License
Kernel Modules
Hardware Info
Nodes
Network Topology
Additional Networks
Networks
Nameservers
Network Interfaces
Subnet Managers
Installation Source
Workload Management
Disk Layout
Time Configuration
Cluster Access
Authentication
Console
Summary
A workload management system is highly recommended to run compute jobs. Please choose the workload management system that should be configured. To prevent a workload management system from being set up, select 'None'. The number of slots per node should ideally be equal to the number of CPU cores available on each node. On small clusters, the head node may also be used for compute jobs.
Workload management system
Slurm(v2.2.4)
Number of slots/node
8
Use head node for compute jobs
Yes
No
Lawrence Livermore National Laboratory
SLURM
The Simple Linux Utility for Resource Management (SLURM) is an open source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters. The slurm controller daemon will be configured to run on the head node and the slurm daemons will be configured to run on all the nodes. If the master node is required to run jobs, then the slurmd will also run on the head node. MySQL will be used to store job accounting information.
Cancel
Go Back
Continue

# Cluster On Demand Scenario



## Mixing Local and Cloud Resources

**Cloud does not work well for all HPC workloads**

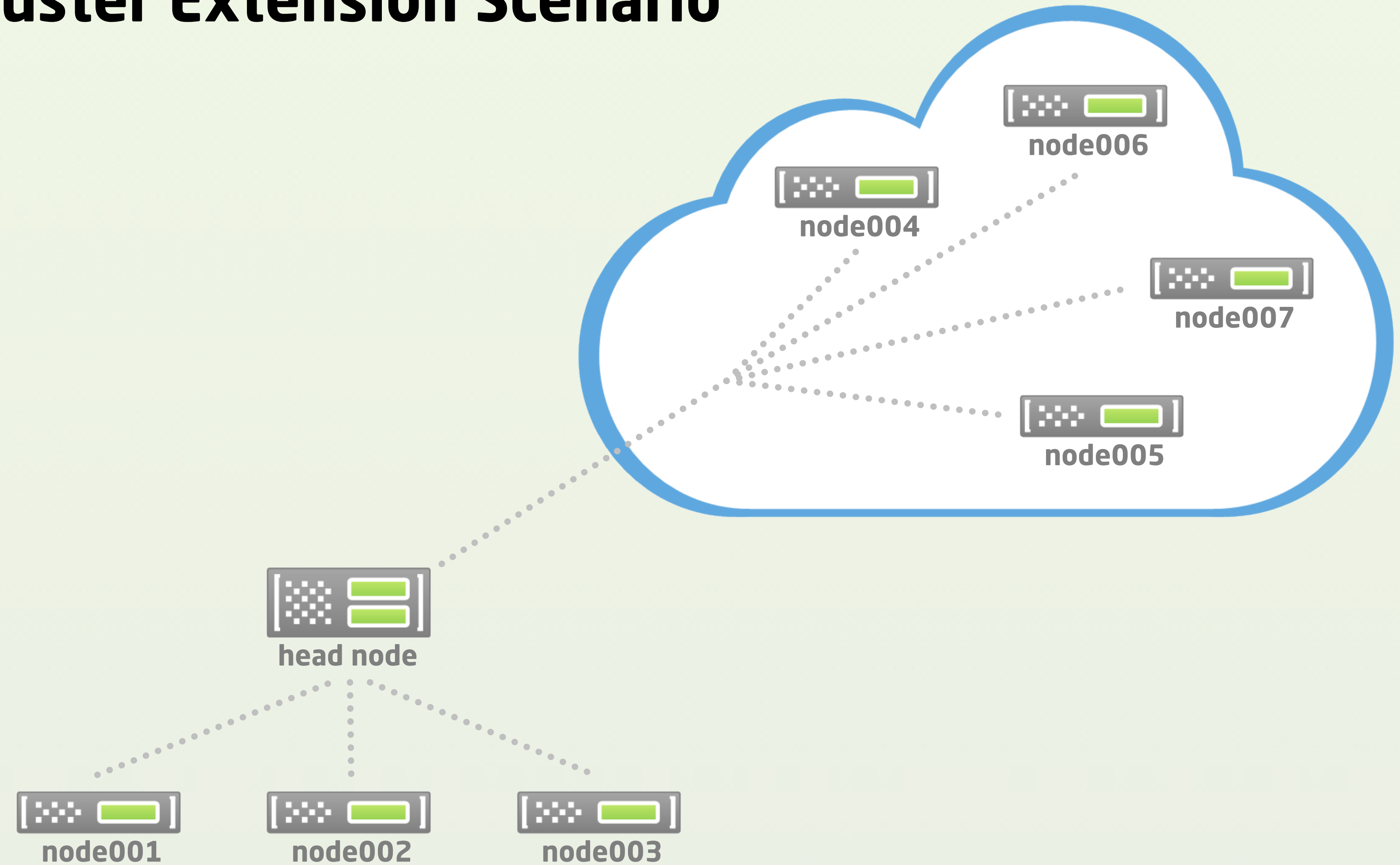
- Sensitive data/computations
- Problems getting huge amounts of data in/out
- Workload may depend on low latency / high bandwidth
- Workload may depend on non-standard compute resources
- Workload may depend on advanced shared storage (e.g. Lustre)

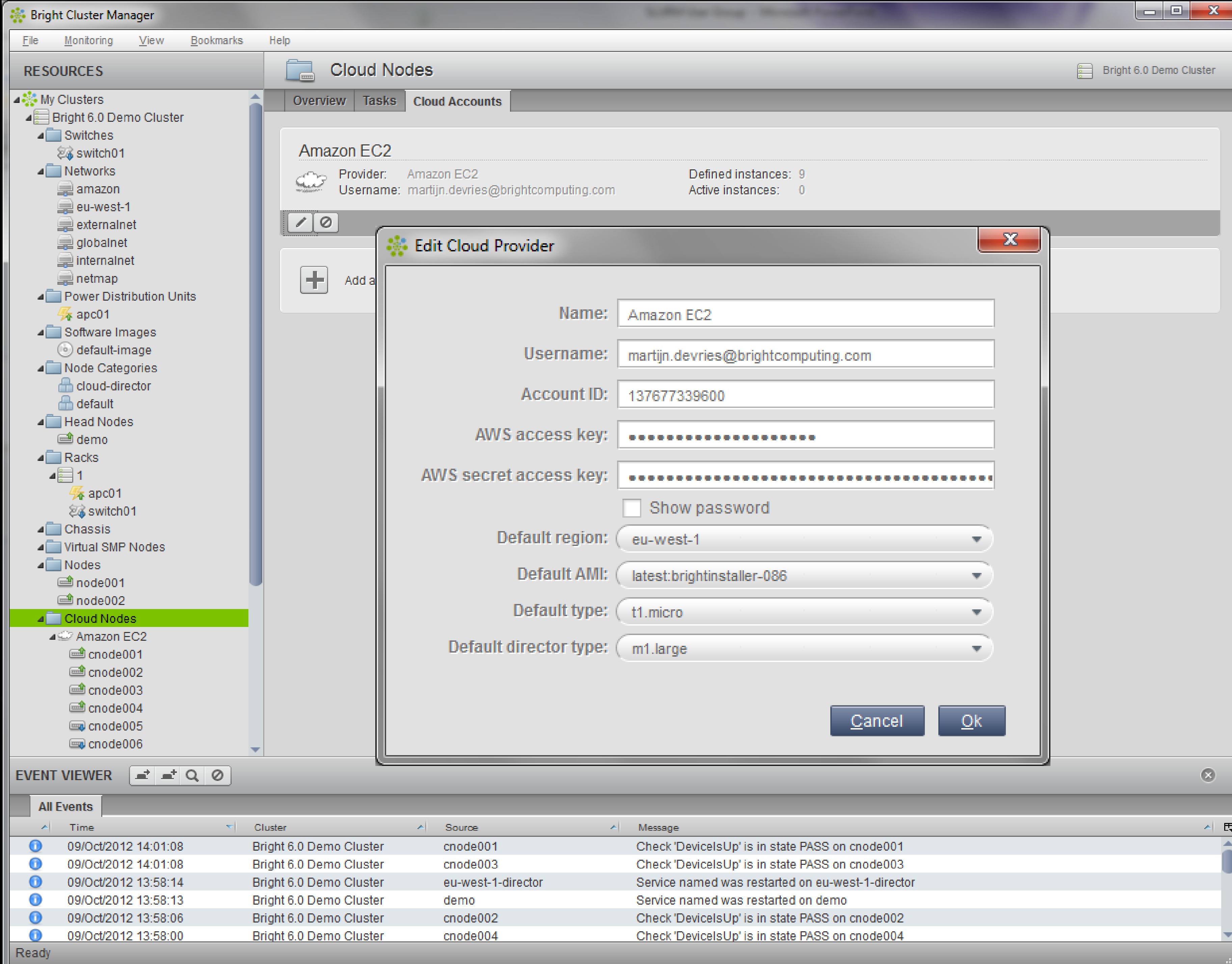
**Not everyone will replace HPC cluster with EC2 account**

- Allow local cluster to be extended with cloud resources to give best of both worlds
- Allow workload suitable for cloud to be off-loaded
- Allow traditional HPC users to try out and migrate to cloud

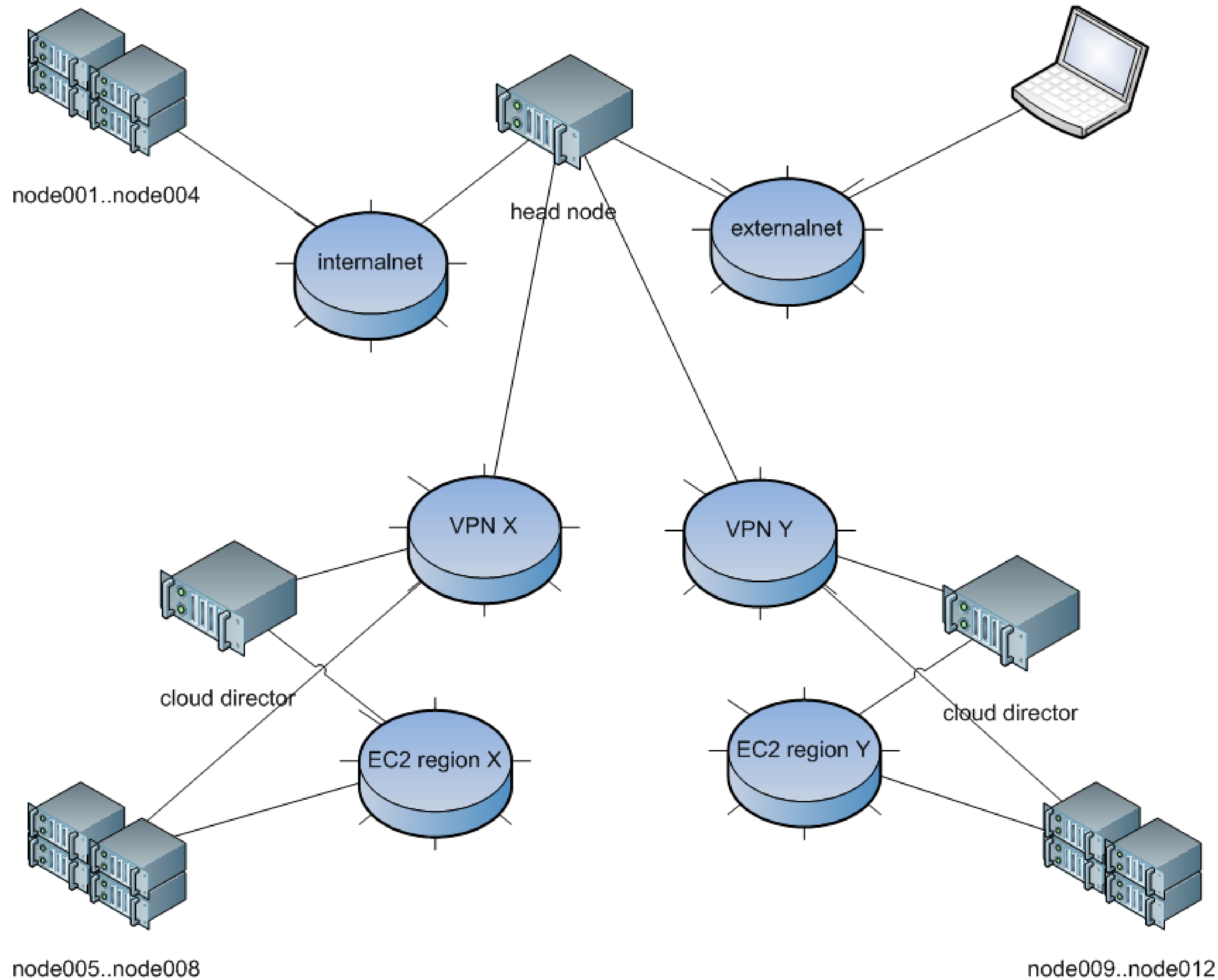
Bright Computing

# Cluster Extension Scenario




Bright Cluster Manager
File
Monitoring
View
Bookmarks
Help
RESOURCES
My Clusters
Bright 6.0 Demo Cluster
Switches
switch01
Networks
amazon
eu-west-1
externalnet
globalnet
internalnet
netmap
Power Distribution Units
apc01
Software Images
default-image
Node Categories
cloud-director
default
Head Nodes
demo
Racks
1
apc01
switch01
Chassis
Virtual SMP Nodes
Nodes
node001
node002
Cloud Nodes
Amazon EC2
cnode001
cnode002
cnode003
cnode004
cnode005
cnode006
Cloud Nodes
Bright 6.0 Demo Cluster
Overview
Tasks
Cloud Accounts
Amazon EC2
Provider: Amazon EC2
Username: martijn.devries@brightcomputing.com
Defined instances: 9
Active instances: 0
Add a
Edit Cloud Provider
Name: Amazon EC2
Username: martijn.devries@brightcomputing.com
Account ID: 137677339600
AWS access key:
AWS secret access key:
Show password
Default region: eu-west-1
Default AMI: latest:brightinstaller-086
Default type: t1.micro
Default director type: m1.large
Cancel
Ok
EVENT VIEWER
All Events
Time
Cluster
Source
Message
09/Oct/2012 14:01:08 Bright 6.0 Demo Cluster cnode001 Check 'DevicesIsUp' is in state PASS on cnode001
09/Oct/2012 14:01:08 Bright 6.0 Demo Cluster cnode003 Check 'DevicesIsUp' is in state PASS on cnode003
09/Oct/2012 13:58:14 Bright 6.0 Demo Cluster eu-west-1-director Service named was restarted on eu-west-1-director
09/Oct/2012 13:58:13 Bright 6.0 Demo Cluster demo Service named was restarted on demo
09/Oct/2012 13:58:06 Bright 6.0 Demo Cluster cnode002 Check 'DevicesIsUp' is in state PASS on cnode002
09/Oct/2012 13:58:00 Bright 6.0 Demo Cluster cnode004 Check 'DevicesIsUp' is in state PASS on cnode004
Ready

# Cloud Network Map



Bright Computing

## Cloud Director acts as a head node in the cloud

- Provides gateway between local and cloud nodes
- Provisions software image to cloud nodes
- Serves shared storage for cloud nodes
- Mirrors network services for the cloud nodes (e.g. LDAP, DNS)

## Cloud node booting process

- Instances are created with 1GB EBS and *n*GB ephemeral/EBS disk
- Bright Node Installer AMI goes on EBS disk
- Node Installer continues with normal procedure to bring up node
- Software image gets provisioned onto second disk

# Uniformity

## Cloud nodes behave the same way as local nodes

- Same method of provisioning
- Same software image and user environment
- Same workload management set-up
- Same management interface that allows to control cluster
- Same monitoring & health checking

## Everything can talk to everything

- Accomplished using VPN, routing, network mapping
- VPN set-up automated and does not require firewall set-up (requires just **outgoing** access on 1194/udp)
- Single global DNS namespace

# Workload Driven Cloud Resizing

- Nodes are created in the cloud:
  - Manually by administrator using CMGUI/CMSH
  - Automatically based on workload by cloud-resize utility
- Cloud-resize called periodically from crond
- Three inputs to cloud-resize:
  - Current workload
  - Current number of cloud nodes
  - Policy (Python module)
- When more cloud nodes are needed (as determined by policy), more nodes are created in the cloud based on configured node properties
- When more nodes come online (~2-5m), Slurm will schedule jobs onto nodes

# Workload Management in the Cloud

- Typical setup: one workload management queue per region
- Jobs that may run in the cloud should be submitted to one of the cloud queues
- Alternatively, cloud nodes and regular nodes can be combined in same queue.
- Cloud nodes also have workload management *features* which can be used as job-constraints
- Example:

```
#!/bin/sh
#SBATCH -J TestJob
#SBATCH --ntasks=16
#SBATCH --constraint=us-east-1
```

- Workload management system will schedule jobs onto cloud nodes the same way as on local nodes
- Nodes NFS mount /home and /cm/shared:
  - Local nodes mount from head node
  - Cloud nodes mount from a cloud director

# Data Locality Problem

**Problem:**

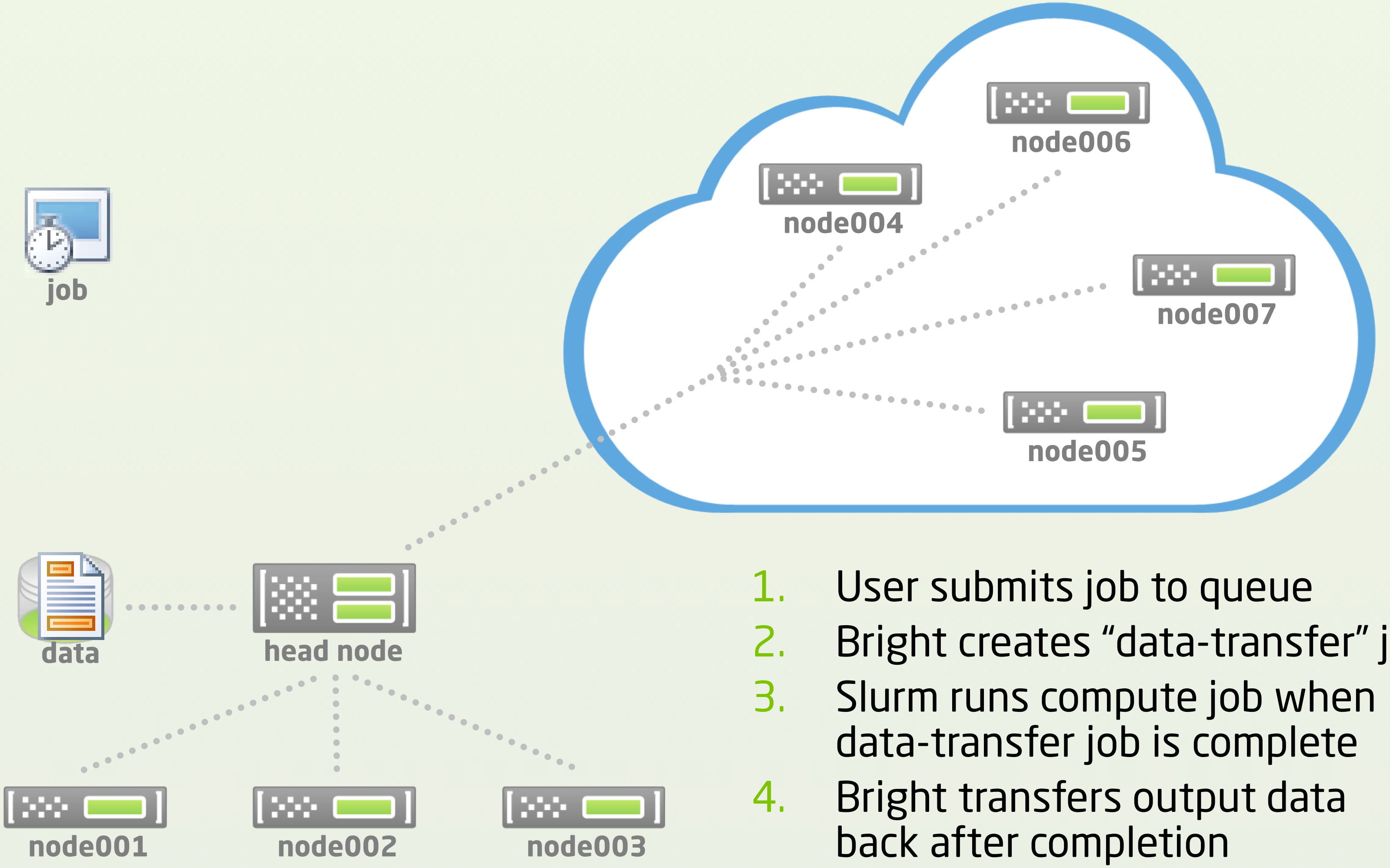
- Jobs usually require input data and produce output data
- Input and/or output data may require significant transfer time
- Resources charged by the hour, so input/output data should be transferred while resources are not yet allocated
- Data moving mechanics should be hidden from users as much as possible

**Solution:**

- Bright introduces job submission utility *cmsub* which allows data dependencies of jobs to be made explicit in Slurm
- Useful for cloud, but can also be useful for e.g.
  - Fetching data from tape archive
  - Staging data to local compute nodes to overcome throughput limitations of parallel filesystem (needed for exascale)

Cloud Bursting

# Data-Aware Scheduling to the Cloud




1. User submits job to queue
2. Bright creates “data-transfer” job
3. Slurm runs compute job when data-transfer job is complete
4. Bright transfers output data back after completion

# Slurm Job with Data Dependencies

Bright Computing

- Example

```
#!/bin/sh

#SBATCH -J Data-Transfer-Test
#SBATCH --ntasks=1

#CMSUB --input=/home/martijn/data-transfer-test/inputfile.txt
#CMSUB --regions=eu-west-1

# Do the heavy work of reversing the lines
tac inputfile.txt >outputfile-$SLURM_JOB_ID.txt

# Schedule output file to be transferred back
CM_SCHEDULE_TRANSFER(/home/martijn/data-transfer-
   test/outputfile-$SLURM_JOB_ID.txt)

echo Processed data on `hostname`
```

# Data Aware Workload Management

Bright Computing

- User submits job to workload management system using cmsub
- The cmsub utility will:
  - Submit input data transfer job to Slurm
  - Submit compute job with dependency on input transfer job
  - Submit output data transfer job with dependency user job


- Data transfer jobs run on head node, so compute nodes need not be allocated while data is being transferred in/out of cloud
- Option to remove or keep data in the cloud after job completed
- Cmsub prevents multiple transfers of same data
- Partial data transfers are handled elegantly
- Users may also take responsibility for transferring data outside of cmsub

# Future Directions

Bright Computing

- Scheduling priorities of data transfers and compute jobs should be interdependent
- Order in which data should be transferred depends on:
  - Estimated transfer time (data size, target location)
  - Estimated job run time
  - Job priority
  - Resources requested by job
- Simple example:
  - Job 1: run time: 1h input data: 10GB (10h)
  - Job 2: run time: 10h input data: 1GB (1h)
  - Naïve scheduling: 10h + 1h + 10h = 21h
  - Optimal scheduling: 1h + 10h + 1h = 12h
- Making things worse: what about priority for output data?

# Questions?

Martijn de Vries

martijn.devries@brightcomputing.com