Configuration and Administration of Cray External Services Systems

Jeff Keopp, Cray Inc.
Harold Longley, Cray Inc.

What we will cover today

- What are Cray External Services systems?
- esMS, esLogin and esFS Overview
- Configuration and Administration
  - Operational Overview
  - *Break*
  - Device Configuration and Management
  - Node Provisioning
  - Image Management
  - *Break*
  - System Monitoring
  - esMS Failover
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

What are Cray External Services systems?

- Cray External Services systems expand the functionality of the Cray XE/XK and Cray XC systems by providing more powerful external login (esLogin) nodes and an external Lustre filesystem (esFS).

- A great advantage of these systems is that the external Lustre filesystem remains available to the external login nodes regardless of the state of the Cray XE/XK or Cray XC system.

- Consists of a group of service nodes
  - esMS
    - Management node of the External Services group
  - esLogin
    - Login, job submission, software development environment
    - Lustre client from esFS or Sonexion
  - esFS
    - Filesystem nodes, MDS and OSS Lustre server nodes

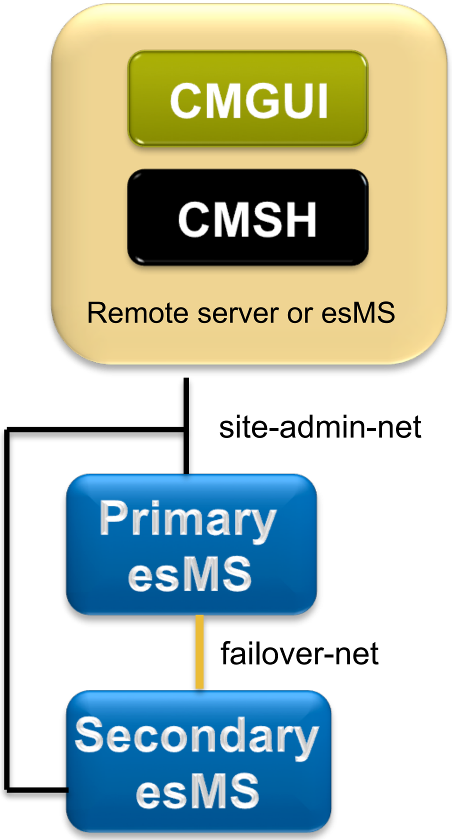
What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview
- Configuration and Administration
  - Operational Overview
  - *Break*
  - Device Configuration and Management
  - Node Provisioning
  - Image Management
  - *Break*
  - System Monitoring
  - esMS Failover
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

Cray External Services Systems

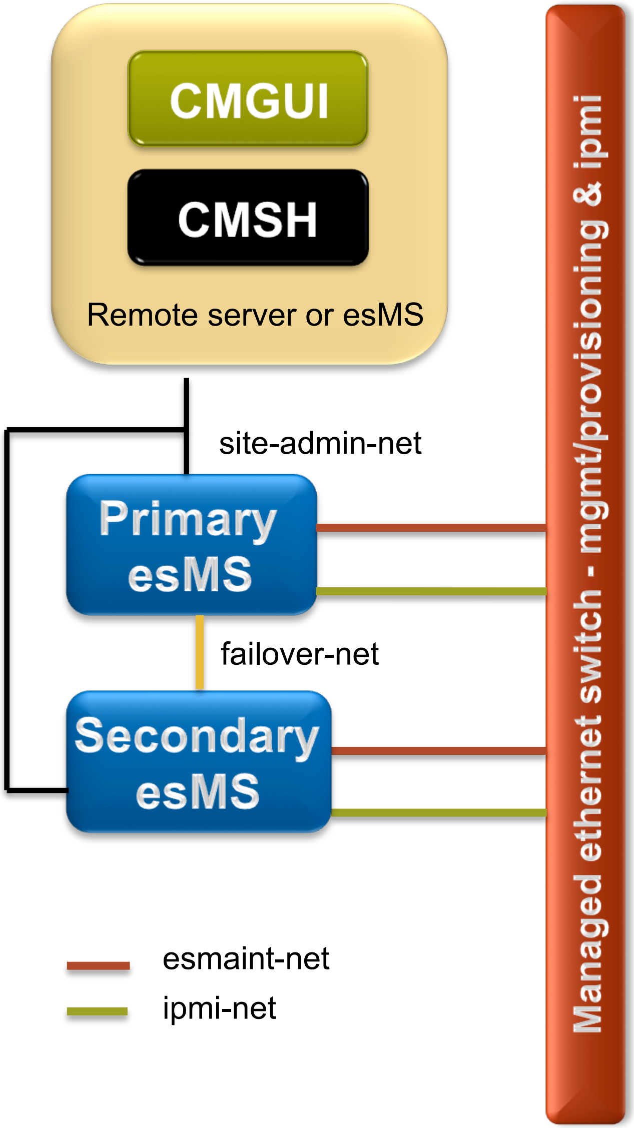
esMS, esLogin and esFS Overview

esMS Overview



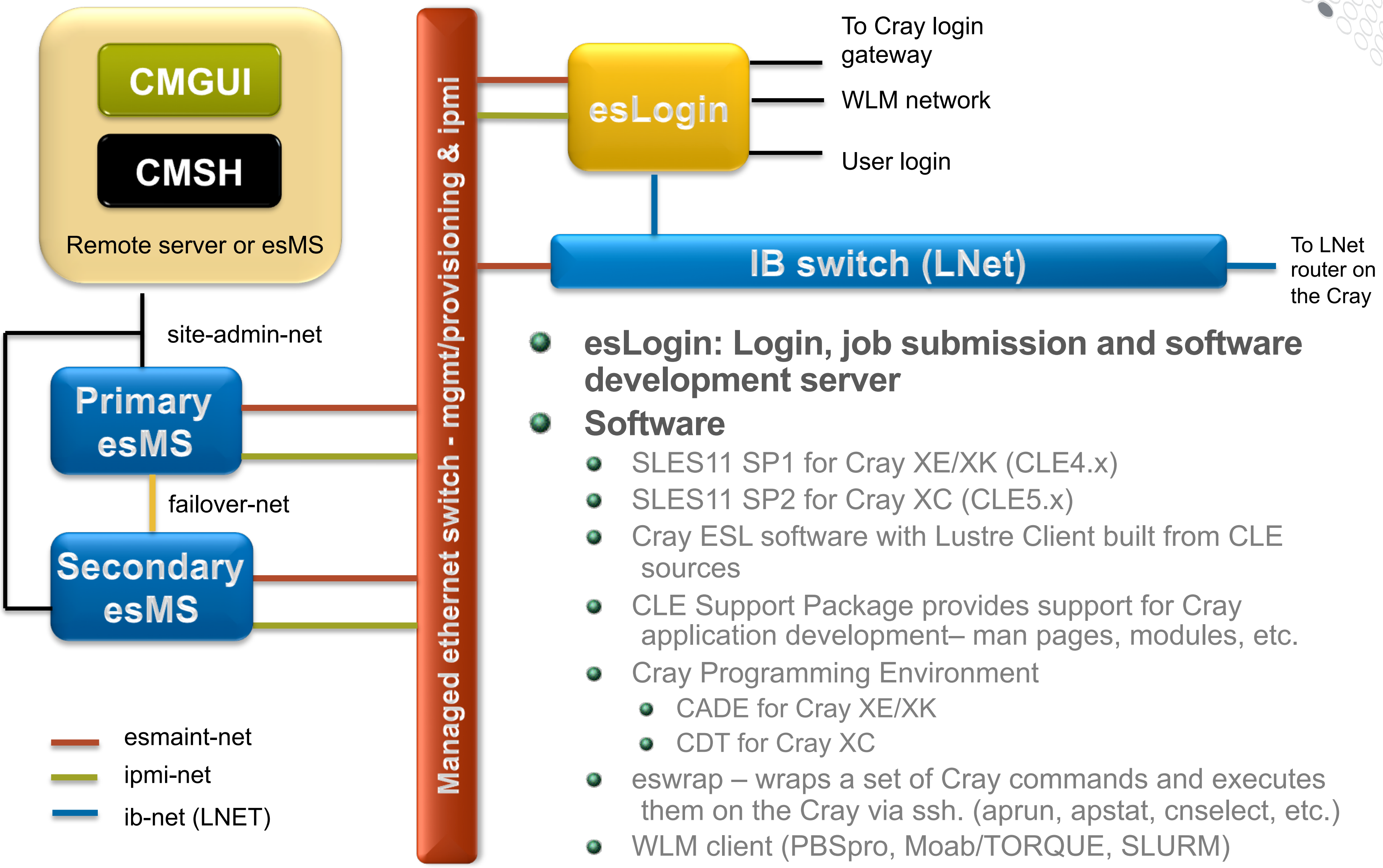
- **esMS: External Services Management Server**
  - Single server or high availability (HA) pair
    - HA pair is an active/passive configuration
  - Provides monitoring and provisioning of the external services
  - Restricted to system administrators, no regular users
- **Software**
  - SLES11 SP2
  - Bright Cluster Manager plus Cray ESM
- **System Management interfaces**
  - **CMGUI** – Cluster Manager Graphical Interface
    - May be run from esMS or remote server
    - Runs on Linux, Windows (MacOS in the future)
    - Can manage multiple systems
  - **CMSH** – Cluster Manager Shell Interface
    - Interactive or batch mode
    - Can execute shell commands within cmsh
  - **pythoncm** – Bright Cluster Manager's Python API
- **Manages Lustre Filesystem (lustre_control)**
  - Provides automated Lustre Failover via esfsmon coupled with lustre_control

esMS Networking



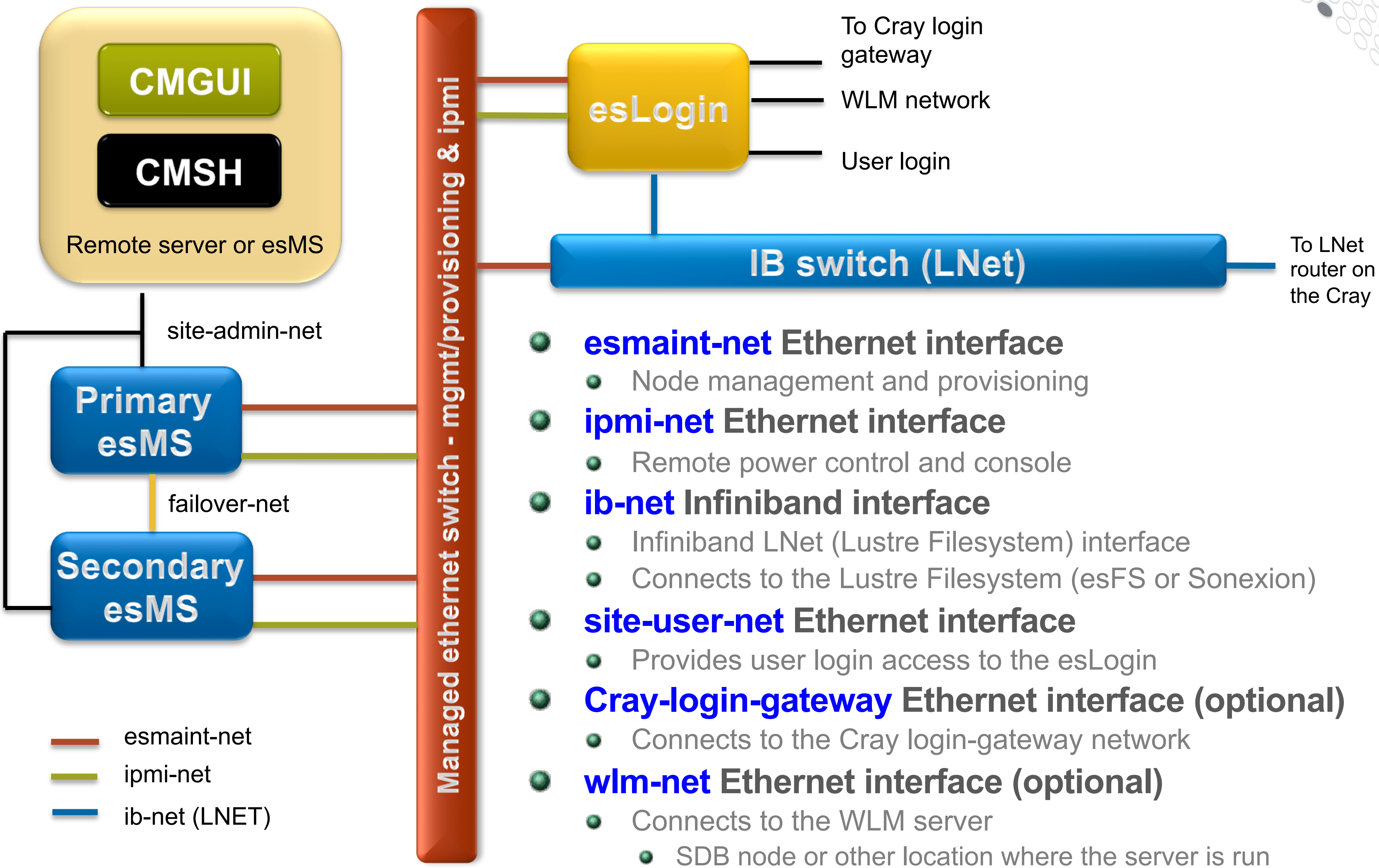
- **esmaint-net interface**
  - Node management and provisioning
  - Private network
- **ipmi-net interface**
  - Power control and remote console for the esLogin and esFS servers
  - Private network
- **site-admin-net interface**
  - Administrative access to the esMS
  - esMS ipmi/BMC is on this network for remote power control and console access
  - May be a customer network or a private network under the SMW
- **failover-net interface**
  - Direct heartbeat connection between HA esMS servers
  - Private network

esLogin Overview



- **esLogin: Login, job submission and software development server**
- **Software**
  - SLES11 SP1 for Cray XE/XK (CLE4.x)
  - SLES11 SP2 for Cray XC (CLE5.x)
  - Cray ESL software with Lustre Client built from CLE sources
  - CLE Support Package provides support for Cray application development– man pages, modules, etc.
  - Cray Programming Environment
    - CADE for Cray XE/XK
    - CDT for Cray XC
  - eswrap – wraps a set of Cray commands and executes them on the Cray via ssh. (aprun, apstat, cnselect, etc.)
  - WLM client (PBSpro, Moab/TORQUE, SLURM)

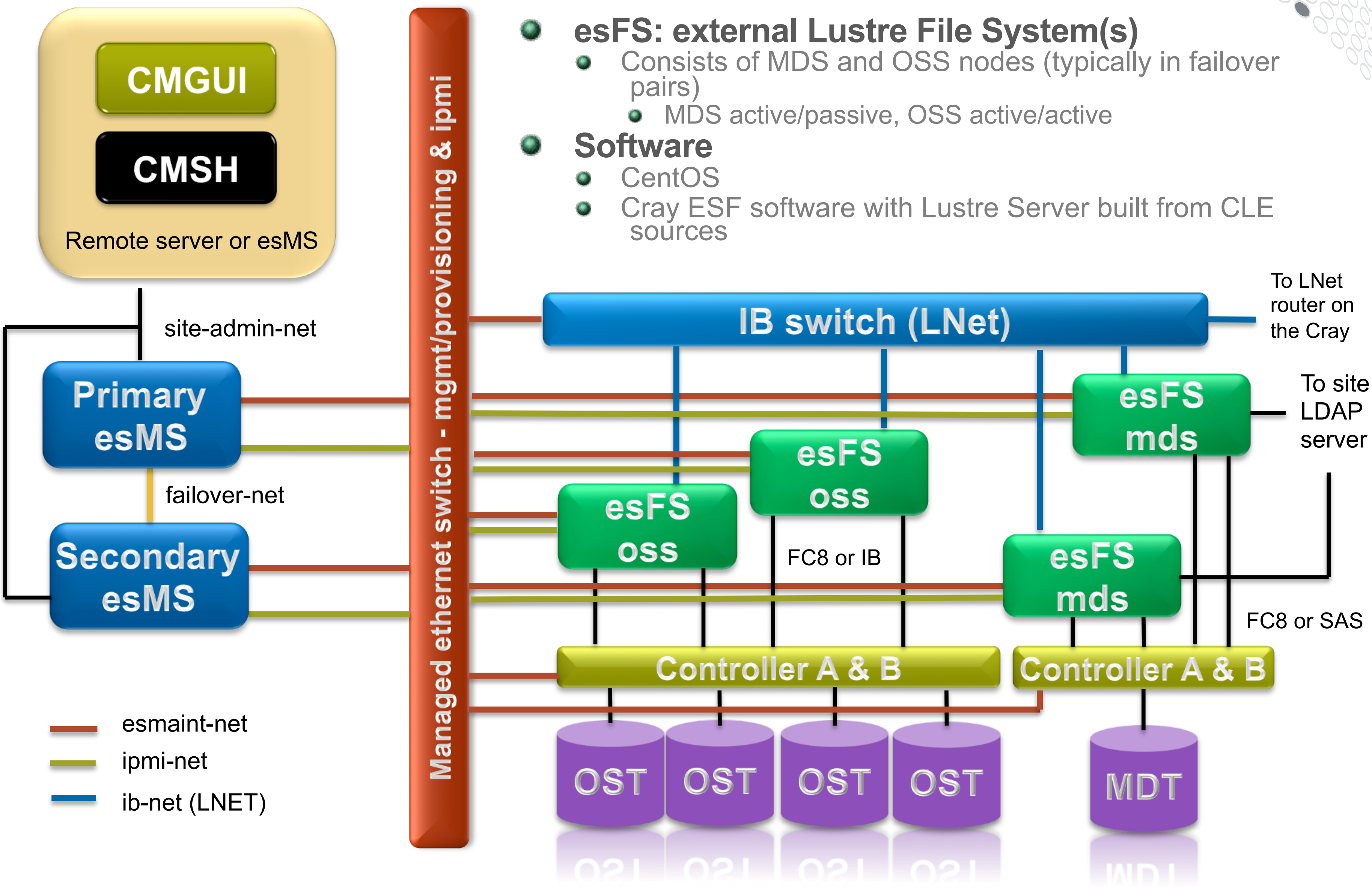
esLogin Networking



- **esmaint-net Ethernet interface**
  - Node management and provisioning
- **ipmi-net Ethernet interface**
  - Remote power control and console
- **ib-net Infiniband interface**
  - Infiniband LNet (Lustre Filesystem) interface
  - Connects to the Lustre Filesystem (esFS or Sonexion)
- **site-user-net Ethernet interface**
  - Provides user login access to the esLogin
- **Cray-login-gateway Ethernet interface (optional)**
  - Connects to the Cray login-gateway network
- **wlm-net Ethernet interface (optional)**
  - Connects to the WLM server
    - SDB node or other location where the server is run

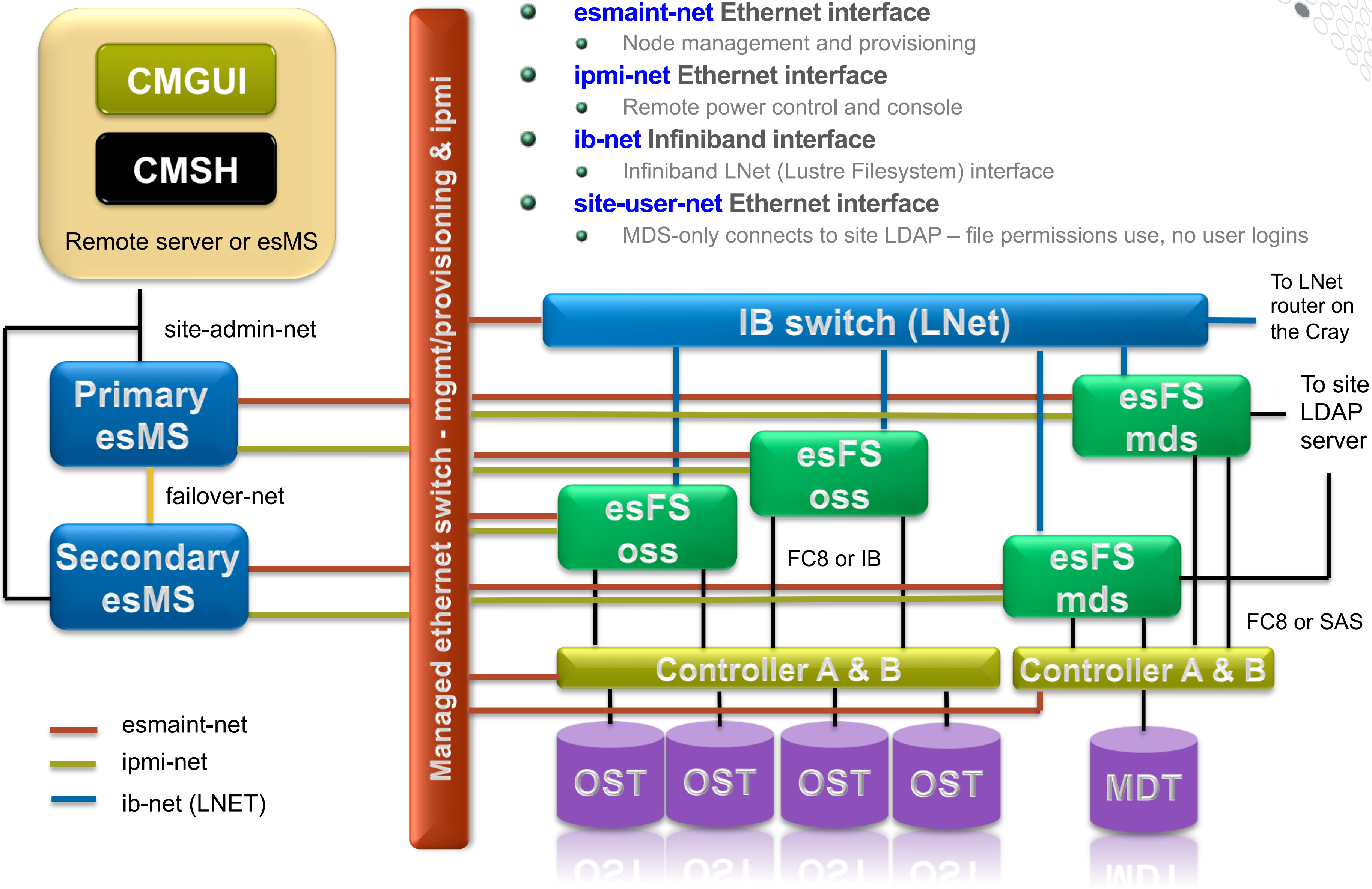
esFS Overview

- **esFS: external Lustre File System(s)**
  - Consists of MDS and OSS nodes (typically in failover pairs)
    - MDS active/passive, OSS active/active
- **Software**
  - CentOS
  - Cray ESF software with Lustre Server built from CLE sources

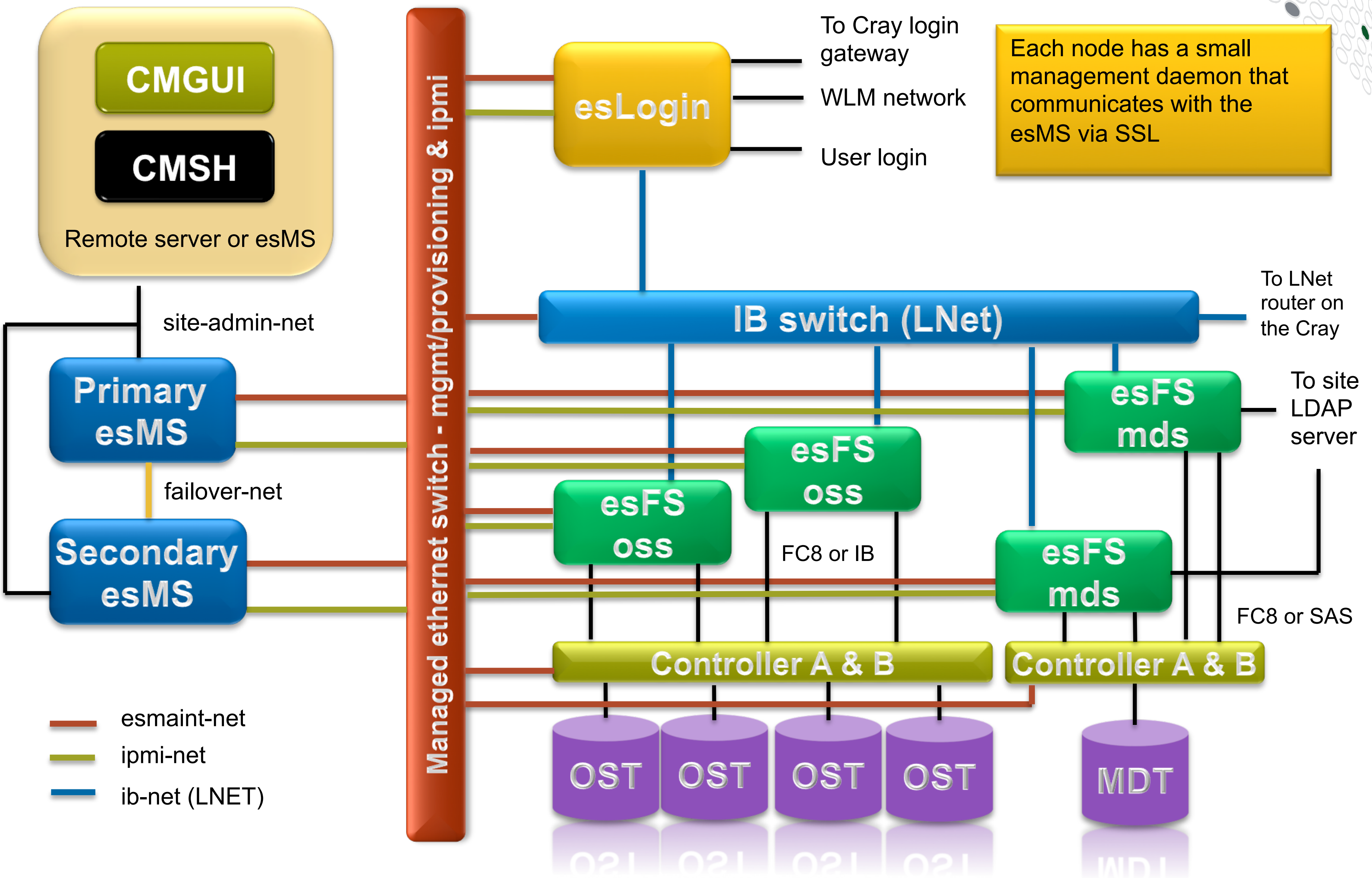


esFS Networking

- **esmaint-net Ethernet interface**
  - Node management and provisioning
- **ipmi-net Ethernet interface**
  - Remote power control and console
- **ib-net Infiniband interface**
  - Infiniband LNet (Lustre Filesystem) interface
- **site-user-net Ethernet interface**
  - MDS-only connects to site LDAP – file permissions use, no user logins



Cray External Services Systems Overview



Each node has a small management daemon that communicates with the esMS via SSL

What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview
  - *Break*
  - Device Configuration and Management
  - Node Provisioning
  - Image Management
  - *Break*
  - System Monitoring
  - esMS Failover
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

Cray External Services Systems

Configuration and Administration
Operational Overview

Operational Overview

- **esMS**
  - Manages, monitors and provisions Cray External Services nodes
  - Node configuration is stored in the management database
  - esLogin and esFS images are maintained on the esMS
    - /cm/images/*image-name*
    - Image modifications
      - May be made on the esMS or on a running node
      - May be captured from a running node to the image on the esMS
      - May be pushed from the esMS to running nodes
      - Changes made on a running node will be lost on reboot unless they are captured back to the image held on the esMS
  - Syslog messages are forwarded to the esMS from managed nodes
  - Console messages are logged on the esMS from managed nodes

Operational Overview

- **esLogin and esFS nodes**
  - When powered on, the nodes attempt a network boot (PXE)
  - If the MAC address is known to the esMS, dhcpd responds
  - The node makes a tftpboot request and the esMS responds with the node-installer image
  - Node-installer identifies the node, rsyncs the assigned image from the esMS, configures the node and pivots to the image on disk and calls /init

- **Only image diffs are downloaded, allowing faster reboots**
- **After the first time that the node is powered on, the esMS can remotely control power of the node**
- **Once software is installed, node can boot from disk if esMS is not available**

Operational Overview

- **Effects of loss of esMS in non-HA esMS configuration**
  - The esLogin and esFS nodes will continue to operate
  - They can reboot to the currently running image without the esMS present
  - Loss of management and monitoring
  - Loss of Automated Lustre Failover since this is managed from the esMS
  - Loss of Lustre Filesystem management since this is managed from the esMS via lustre_control

Where are administrative tasks performed?

- **Most tasks are performed on the esMS**
  - Bright Cluster Manager provides the management and monitoring infrastructure
  - Most administrative tasks are performed using the management interfaces to Bright Cluster Manager

- **Management Interfaces**
  - **CMGUI** – esMS Management Graphical User Interface
    - Runs on Linux, Windows (MacOS in the future)
    - Can monitor multiple Cray External Services systems
  - **CMSH** – esMS Management Shell
    - Interactive or batch mode
    - Can exec shell commands from within cmsh
  - **pythoncm** – Python API to the management daemon

Tasks performed in Bright Cluster Manager

- **Network configuration**
  - Examples of network parameters include:
    - Base address
    - Broadcast address
    - Gateway address
    - Netmask bits
    - Management allowed?
    - Node booting allowed?

- **Node configuration, includes the esMS (head node)**
  - Examples of configuration settings include:
    - Network interface configuration
    - Disk partitioning (slave nodes only)
    - Installation mode (slave nodes only)
    - /etc/fstab settings
    - Default gateway setting
    - Node Finalize script

Tasks performed in Bright Cluster Manager

- **Node management, includes esMS (head node)**
  - Examples of management operations include:
    - Operational and power status
    - Reboot/shutdown
    - Remote console
    - Power control
- **Image Management**
  - Operations are split between:
    - Bright Cluster Manager
    - Directly editing the image
    - Edit the image in a chroot environment
  - Examples of tasks performed in Bright Cluster Manager
    - Kernel version and parameters, Serial-Over-LAN (SOL) settings
    - Assigning images to node categories

Tasks performed outside Bright Cluster Manager

- **Adding/removing files or RPMs from images**
  - Use rpm, zypper or yum in chroot environment
- **Configuring services on/off**
  - Use chkconfig in chroot environment
- **Modifying files not managed by Bright Cluster Manager**
  - Files that are managed or partially managed by Bright Cluster Manager have comments surrounding the managed sections
    - Examples include /etc/hosts, /etc/resolv.conf and /etc/fstab
      - Additional entries may be added outside the commented section
- **Recording changes made to images**
  - There is no built-in image change log
  - Image change logging is left to the site to define

Bright Cluster Manager

- Organized in a hierarchy of resources or modes
  - **network** – info about networks used by the system
  - **partition** – there is only one partition (base) with info that is global to system
  - **category** – nodes must be in one category at a time
    - Inherits properties from the base partition
    - May override some of these inherited properties
  - **device** – all nodes, switches and storage array controllers
    - Inherits properties from the assigned category
    - May override some these inherited properties
  - **softwareimage** – slave image information
  - **nodegroup** - administrative grouping of nodes
    - nodes may be in multiple groups
  - **monitoring** – info about healthchecks, metrics and actions taken based on results
  - **rack** – rack info allowing a graphical display of selected parameters
  - **main** – license and version info about Bright Cluster Manager

CMSH Overview – Status and Commands

- **Checking Status**
  - Device mode using the “status” command
  - Can address all or subsets of nodes based on lists, categories or groups

```
test-esms1:~ # cmsh
[test-esms1]% device
[test-esms1->device]% status
node001 ................... [ CLOSED ]
test-esl1 ................. [   UP   ] restart-required
test-esms1 ................ [   UP   ]
test-esms2 ................ [   UP   ]
switch01 .................. [  DOWN  ] health check failed
[test-esms1->device]%

[test-esms1->device]% status test-esl1
test-esl1 ................. [   UP   ] restart-required
[test-esms1->device]%

[test-esms1->device]% category list
Name (key)               Software image
------------------------ ------------------------
default                  default-image
eslogin-test             ESL-XE-1.2.0-2013011109+
[test-esms1->device]% status -c eslogin-test
test-esl1 ................. [   UP   ] restart-required
[test-esms1->device]%
```

CMSH Overview – Status and Commands

Checking Power Status

- Device mode using the “power status” command
- Can address all or subsets of nodes based on lists, categories or groups

```
[test-esms1->device]% power status
ipmi0 .................... [  FAILED ] node001
ipmi0 .................... [   ON    ] test-esl1
ipmi0 .................... [   ON    ] test-esms1
ipmi0 .................... [   ON    ] test-esms2
No power control ......... [ UNKNOWN ] switch01

[test-esms1->device]% power -n test-esl1 status
ipmi0 .................... [   ON    ] test-esl1

[test-esms1->device]% power -c eslogin-test status
ipmi0 .................... [   ON    ] test-esl1
[test-esms1->device]%
```

CMSH Overview – Status and Commands

- **Power commands**
  - Power on/off/reset a list of nodes

```
[test-esms1->device]% power -n <node-list> [on | off | reset ]
```

  - Power on/off/reset nodes in one or more categories

```
[test-esms1->device]% power -c <category-list> [on | off | reset ]
```

  - Power on/off/reset nodes in one or more groups

```
[test-esms1->device]% power -g <group-list> [on | off | reset ]
```

CMSH Overview – Status and Commands

- Reboot commands
  - Reboot a list of nodes

```
[test-esms1->device]% reboot -n <node-list>
```

  - Reboot nodes in one or more categories

```
[test-esms1->device]% reboot -c <category-list>
```

  - Reboot nodes in one or more groups

```
[test-esms1->device]% reboot -g <group-list>
```

CMSH Overview – Status and Commands

- **Parallel shell commands (pexec)**
  - Execute commands on a list of nodes

```
[test-esms1->device]% pexec –n <node-list> “df -sh”
```

  - Execute commands on nodes in one or more categories

```
[test-esms1->device]% pexec –c <category-list> “lsscsi”
```

  - Execute commands on nodes in one or more groups

```
[test-esms1->device]% pexec –g <group-list> “uname –r; uptime”
```

CMSH Overview – Changing parameters (part 1)

- **CMSH - Getting and setting values**
  - Example:
    Set the software image for the ‘sles-test’ category to ‘eslogin-image’
    - List available software images

```
[test-esms1->category]% softwareimage list
Name (key)           Path                                     Kernel version
-------------------- ---------------------------------------- --------------------
CentOS6              /cm/images/CentOS6                       2.6.32-220.el6.x86_+
default-image        /cm/images/default-image                 2.6.32.12-0.7-defau+
default-image-mtj-t+ /cm/images/default-image-mtj-test        2.6.32.12-0.7-defau+
eslogin-bkup-image   /cm/images/eslogin-bkup-image            2.6.32.36-0.5-defau+
eslogin-image        /cm/images/eslogin-image                 2.6.32.54-0.3-defau+
sles11sp1.kiyi.5     /cm/images/sles11sp1.kiyi.5              2.6.32.49-0.3-defau+
[test-esms1->category]%
```

CMSH Overview – Changing parameters (part 2)

- **CMSH - Getting and setting values (continued…)**
  - Set the ‘sles-test’ category softwareimage parameter to ‘eslogin-image’

```
[test-esms1->category]% set sles-test softwareimage eslogin-image
                        ^^^ ^^^^^^^^^ ^^^^^^^^^^^^^ ^^^^^^^^^^^^^
    syntax:         command category   parameter       value
```

  - Verify the change

```
[test-esms1->category*]% get sles-test softwareimage
eslogin-image
```

  - Save (commit) the change
    - **‘*’ in prompt indicates change is local to my cmsh session, commit it here**

```
[test-esms1->category*]% commit
Successfully committed 1 Categories
[test-esms1->category]%
```

- **NOTE: Changes are local to your cmsh session until committed!**

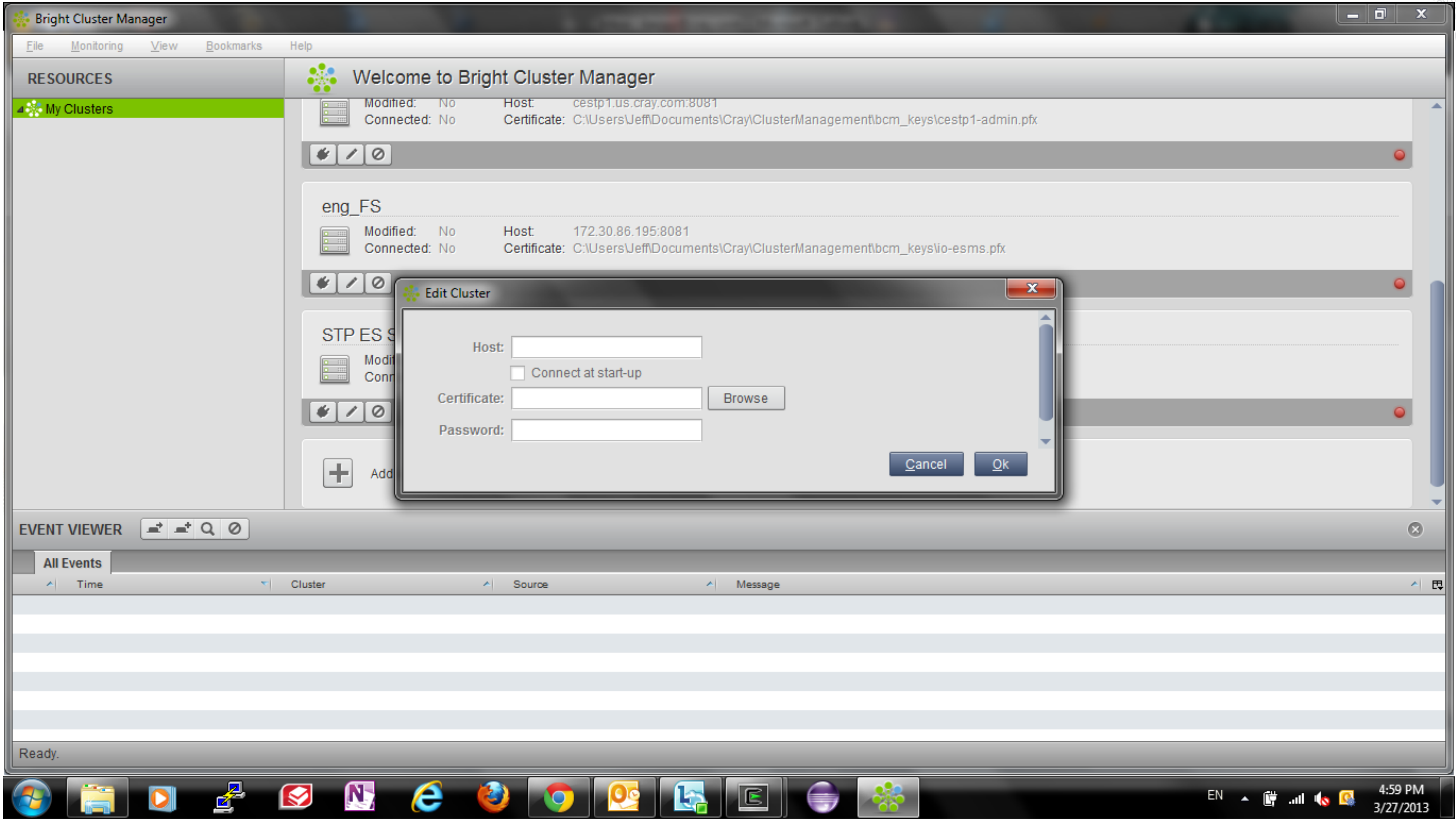
CMGUI Overview

- **The cmgui binary is located on the esMS**
  - ***/cm/shared/apps/cmgui/dist***
    - cmgui-<version>.tar.bz2 is for Linux
    - install.cmgui.<version>.exe is for Windows

- **Requires the admin.pfx file for the system**
  - ***/root/admin.pfx*** on the esMS
  - Copy to a location on the workstation you wish to run cmgui and name it something useful in case you have more than one system
    - For example, my esMS is named stp-esms1 so I'll rename the admin.pfx file to stp-esms1.pfx

- **cmgui may also be run directly on the esMS**
  - Type "cmgui" at the shell prompt

CMGUI – Connecting to a Cluster



Modified: No Host: cestp1.us.cray.com:8081
Connected: No Certificate: C:\Users\Jeff\Documents\Cray\ClusterManagement\bcm_keys\cestp1-admin.pfx

eng_FS

Modified: No Host: 172.30.86.195:8081
Connected: No Certificate: C:\Users\Jeff\Documents\Cray\ClusterManagement\bcm_keys\io-esms.pfx

STP ES S

Modi
Conn

Add

Edit Cluster

Host:

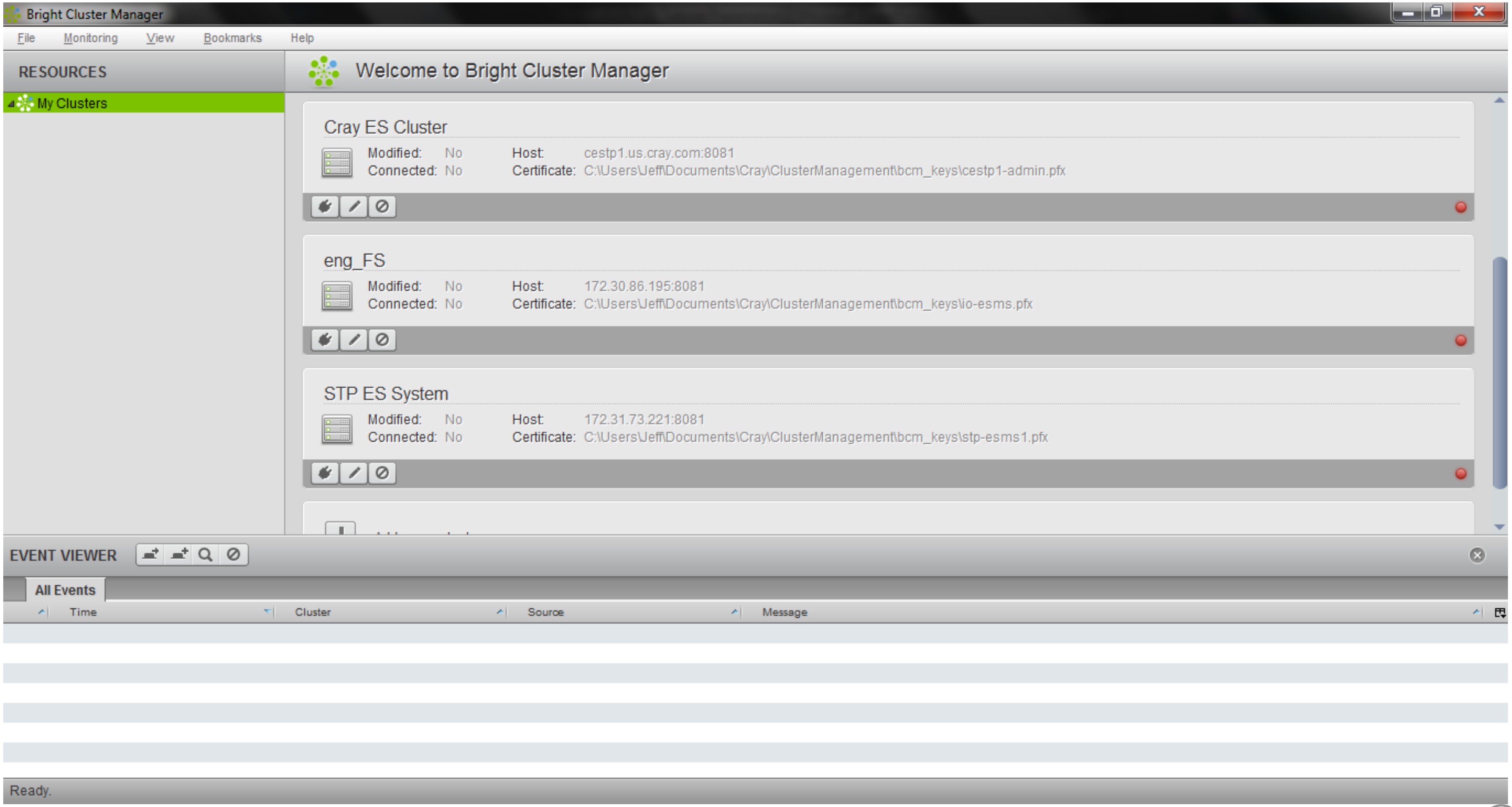
□ Connect at start-up

Certificate: Browse

Password:

Cancel Ok

CMGUI Overview – Connecting to a Cluster

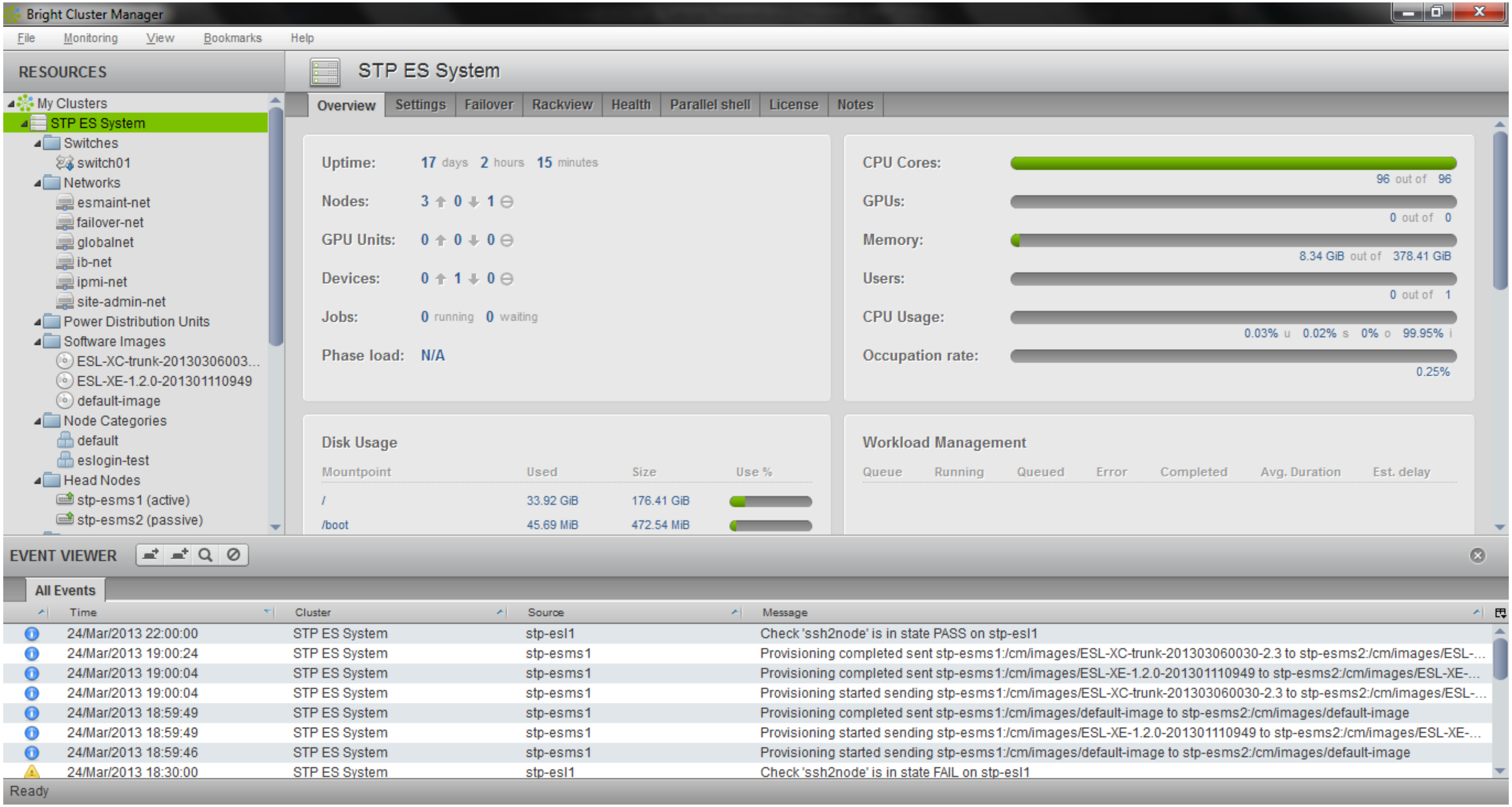

Bright Cluster Manager
File Monitoring View Bookmarks Help
RESOURCES
My Clusters
Welcome to Bright Cluster Manager

Cray ES Cluster
Modified: No Host: cestp1.us.cray.com:8081
Connected: No Certificate: C:\Users\Jeff\Documents\Cray\ClusterManagement\bcm_keys\cestp1-admin.pfx

eng_FS
Modified: No Host: 172.30.86.195:8081
Connected: No Certificate: C:\Users\Jeff\Documents\Cray\ClusterManagement\bcm_keys\io-esms.pfx

STP ES System
Modified: No Host: 172.31.73.221:8081
Connected: No Certificate: C:\Users\Jeff\Documents\Cray\ClusterManagement\bcm_keys\stp-esms1.pfx

CMGUI Overview – Main page



- My Clusters
  - STP ES System
    - Switches
      - switch01
    - Networks
      - esmaint-net
      - failover-net
      - globalnet
      - ib-net
      - ipmi-net
      - site-admin-net
    - Power Distribution Units
    - Software Images
      - ESL-XC-trunk-20130306003...
      - ESL-XE-1.2.0-201301110949
      - default-image
    - Node Categories
      - default
      - eslogin-test
    - Head Nodes
      - stp-esms1 (active)
      - stp-esms2 (passive)

STP ES System

Uptime: 17 days 2 hours 15 minutes

Nodes: 3 ↑ 0 ↓ 1 ⊖

GPU Units: 0 ↑ 0 ↓ 0 ⊖

Devices: 0 ↑ 1 ↓ 0 ⊖

Jobs: 0 running 0 waiting

Phase load: N/A

CPU Cores: 96 out of 96

GPUs: 0 out of 0

Memory: 8.34 GiB out of 378.41 GiB

Users: 0 out of 1

CPU Usage: 0.03% u 0.02% s 0% o 99.95% i

Occupation rate: 0.25%

Disk Usage

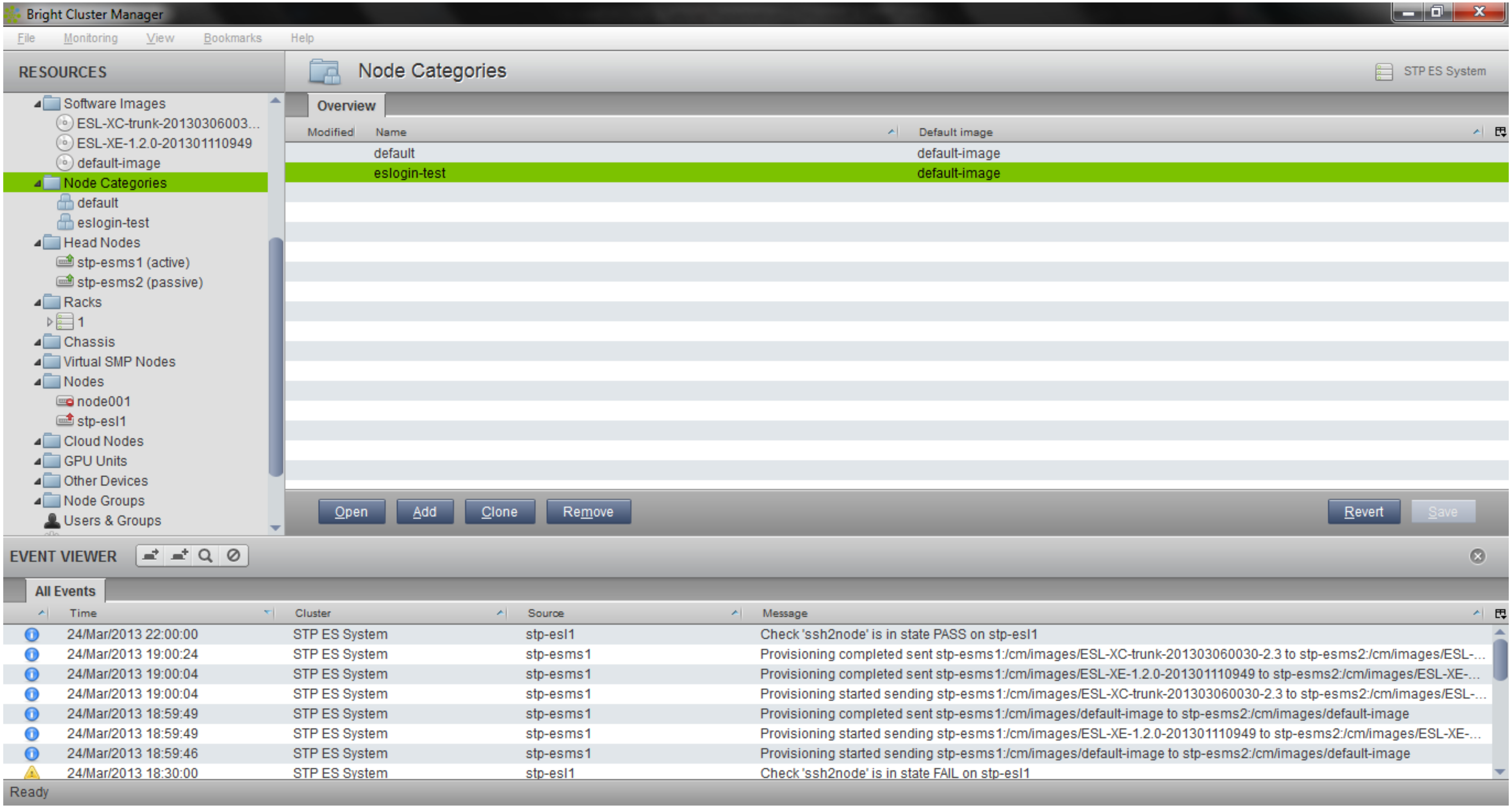
Mountpoint	Used	Size	Use %
/	33.92 GiB	176.41 GiB	
/boot	45.69 MiB	472.54 MiB	

Workload Management

Queue	Running	Queued	Error	Completed	Avg. Duration	Est. delay

Time	Cluster	Source	Message
24/Mar/2013 22:00:00	STP ES System	stp-esl1	Check 'ssh2node' is in state PASS on stp-esl1
24/Mar/2013 19:00:24	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XC-trunk-201303060030-2.3 to stp-esms2:/cm/images/ESL-...
24/Mar/2013 19:00:04	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XE-1.2.0-201301110949 to stp-esms2:/cm/images/ESL-XE-...
24/Mar/2013 19:00:04	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-XC-trunk-201303060030-2.3 to stp-esms2:/cm/images/ESL-...
24/Mar/2013 18:59:49	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/default-image to stp-esms2:/cm/images/default-image
24/Mar/2013 18:59:49	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-XE-1.2.0-201301110949 to stp-esms2:/cm/images/ESL-XE-...
24/Mar/2013 18:59:46	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/default-image to stp-esms2:/cm/images/default-image
24/Mar/2013 18:30:00	STP ES System	stp-esl1	Check 'ssh2node' is in state FAIL on stp-esl1

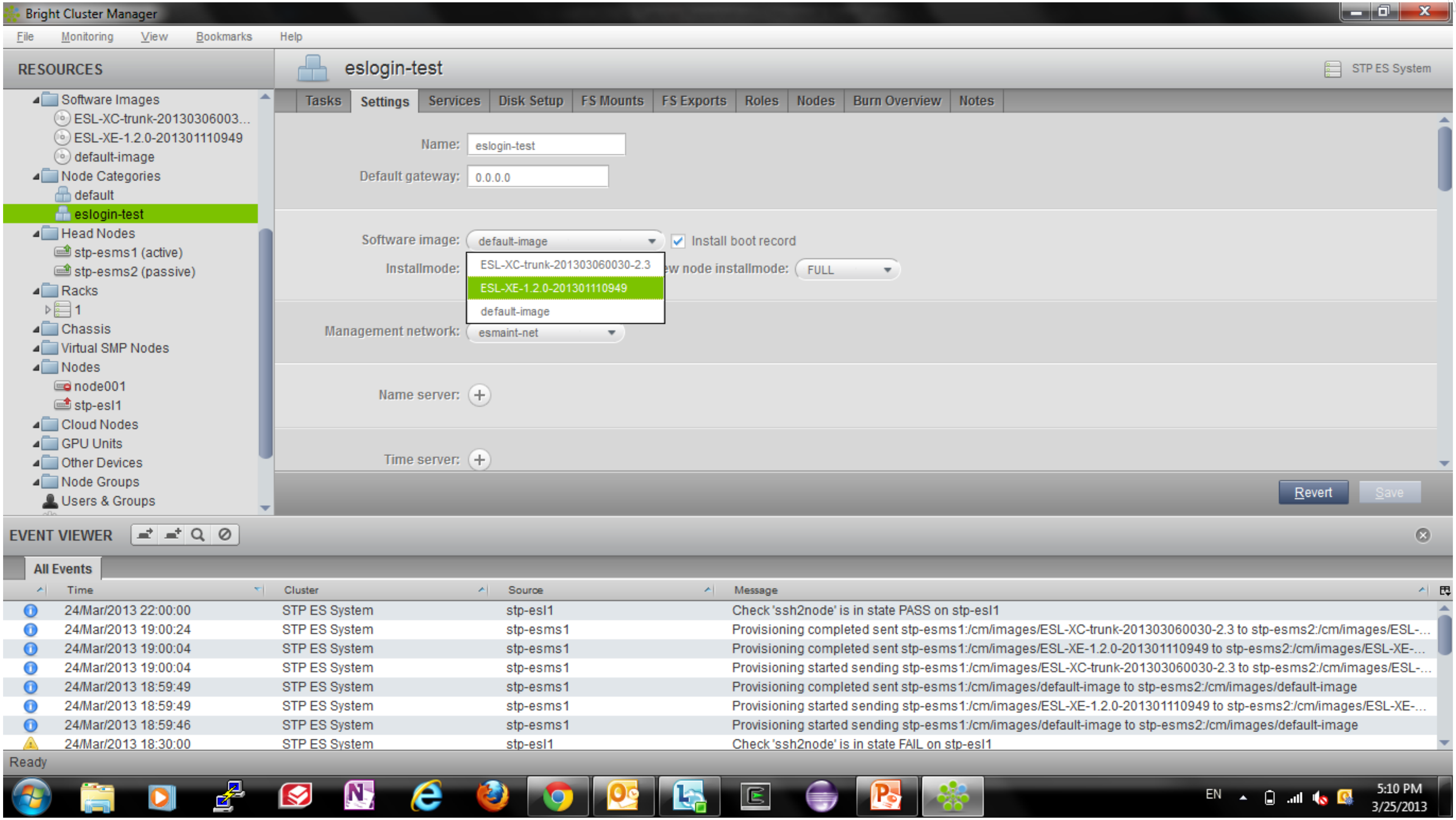
CMGUI – Viewing Node Categories



Modified	Name	Default image
	default	default-image
	eslogin-test	default-image

Time	Cluster	Source	Message
24/Mar/2013 22:00:00	STP ES System	stp-esl1	Check 'ssh2node' is in state PASS on stp-esl1
24/Mar/2013 19:00:24	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XC-trunk-201303060030-2.3 to stp-esms2:/cm/images/ESL-...
24/Mar/2013 19:00:04	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XE-1.2.0-201301110949 to stp-esms2:/cm/images/ESL-XE-...
24/Mar/2013 19:00:04	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-XC-trunk-201303060030-2.3 to stp-esms2:/cm/images/ESL-...
24/Mar/2013 18:59:49	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/default-image to stp-esms2:/cm/images/default-image
24/Mar/2013 18:59:49	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-XE-1.2.0-201301110949 to stp-esms2:/cm/images/ESL-XE-...
24/Mar/2013 18:59:46	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/default-image to stp-esms2:/cm/images/default-image
24/Mar/2013 18:30:00	STP ES System	stp-esl1	Check 'ssh2node' is in state FAIL on stp-esl1

CMGUI – Setting the software image for a category



Name: eslogin-test

Default gateway: 0.0.0.0

Software image: default-image

☑ Install boot record

Installmode:

- ESL-XC-trunk-201303060030-2.3
- ESL-XE-1.2.0-201301110949
- default-image

ew node installmode: FULL

Management network: esmaint-net

Name server: +

Time server: +

Time	Cluster	Source	Message
24/Mar/2013 22:00:00	STP ES System	stp-esl1	Check 'ssh2node' is in state PASS on stp-esl1
24/Mar/2013 19:00:24	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XC-trunk-201303060030-2.3 to stp-esms2:/cm/images/ESL-...
24/Mar/2013 19:00:04	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XE-1.2.0-201301110949 to stp-esms2:/cm/images/ESL-XE-...
24/Mar/2013 19:00:04	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-XC-trunk-201303060030-2.3 to stp-esms2:/cm/images/ESL-...
24/Mar/2013 18:59:49	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/default-image to stp-esms2:/cm/images/default-image
24/Mar/2013 18:59:49	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-XE-1.2.0-201301110949 to stp-esms2:/cm/images/ESL-XE-...
24/Mar/2013 18:59:46	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/default-image to stp-esms2:/cm/images/default-image
24/Mar/2013 18:30:00	STP ES System	stp-esl1	Check 'ssh2node' is in state FAIL on stp-esl1

What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview ✓
  - *Break*
  - Device Configuration and Management
  - Node Provisioning
  - Image Management
  - *Break*
  - System Monitoring
  - esMS Failover
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

Cray External Services Systems

Configuration and Administration
Device Configuration and Management

Device Configuration – Initial Installation

- **The initial installation is driven by an XML config file**
  - `cray-build-config.xml` for non-HA esMS
  - `cray-ha-build-config.xml` for HA esMS
    - The disk partitioning scheme is the only difference between these files
    - cray-ha-build-config.xml sets up DRBD partitions that will be shared between esMS servers
  - Defines at least the following:
    - Networks to be used (default set)
    - Nameservers
    - Time servers
    - IPMI credentials
    - Default category
    - Default image
    - Primary esMS
    - esMS disk partitioning
    - Default slave node
    - Cluster name

Device Configuration - Overview

- **A device in Bright Cluster Manager is any of the following**
  - Node – esMS, esLogin or esFS server
  - Switch – Ethernet, Infiniband, fibre channel, SAS, etc.
  - Storage array controller
- **A server device must be a member of a category**
- **Categories define parameters common to many devices**
  - Softwareimage
  - Default Gateway
  - Nameservers
  - Search Domains
  - Finalize script
  - Filesystem mounts
- **Settings in the device mode override those set in the category for that specific device**

Configuration - Networks

- Networks must be defined in Bright Cluster Manager
  - **Name**
    - Network name used by the system
  - **Domain Name**
    - Used by the internal system DNS
  - **Base Address**
  - **Broadcast Address**
  - **Netmask Bits**
  - **Dynamic Range Start/End**
    - The start and end IP addresses used if the esMS will issue IP addresses via DHCP on this network
  - **Gateway**
  - **MTU**
  - **IPv6**
    - “yes” or “no” - supports IPv6
  - **Lock Down DHCP**
    - Stops dhcpd from answering unknown MAC addresses on this network
  - **Type**
    - Sets network type: external, internal
  - **Management Allowed**
    - Allows node management over this network
  - **Node Booting**
    - Allows node provisioning over this network
  - **Revision**
    - User field for revision information
  - **Notes**
    - User field for notes

Default Network Definitions

- **Standard networks provided in default installation**
  - **esmaint-net** – Private management and provisioning network
    - Default: 10.141.0.0/16
    - Connects to esLogin, esFS, Ethernet switches, Infiniband switches and storage controllers
  - **ipmi-net** – Private IPMI network
    - Default: 10.148.0.0/16
    - Connects to remote power control and remote console for esLogin and esFS
  - **ib-net** – Infiniband network (LNet)
    - Default: 10.149.0.0/16
    - Connects to esLogin (Lustre Client) and esFS (Lustre Server) and LNet Router nodes on CLE system
  - **site-admin-net** – Administrative access to the esMS
    - May be a customer administrative network or a private network under the SMW
    - Classified as an “external” network
  - **site-user-net** – User login access and authentication (LDAP, etc.)
    - esLogin and esFS MDS servers
    - Classified as an “external” network

Default Firewall Configuration

- **Shorewall is used for firewall configuration**
  - esMS firewall
    - ICMP is allowed from the site-admin-net
    - ssh is allowed from the site-admin-net
    - ssl is allowed from the site-admin-net
    - ssh is rejected from the esmaint-net

Cluster Configuration - Partition

- Partition Parameters – Global settings
  - Administrator e-mail
    - List of email addresses to send alerts
  - BMC Password
  - BMC User ID
  - BMC User name
  - Cluster name
  - Default category
  - Default software image
  - External network
  - Externally visible IP
    - Only set for HA esMS
  - Failover (submode)
    - Failover esMS plus information about HA esMS configuration
  - Management network
  - Masternode
  - Name servers
  - Node basename
  - Node digits
  - Notes
    - User field for notes
  - Revision
    - User field for revision information
  - Search domains
  - Time servers
  - Time zone

Node Configuration – Categories (part 1)

- Nodes are in one category at a time
  - **Name** – category name
  - **Software image**
    - All nodes in the category use this image
  - **Default Gateway**
    - Defaults to esMS when set to 0.0.0.0
  - **Name servers**
    - Defaults to esMS
  - **Search Domain**
    - Defaults to those in base partition
  - **Time servers**
    - Defaults to those in base partition
  - **Filesystem exports** (submode)
  - **Filesystem mounts** (submode)
  - **Exclude Lists**
    - List files and directories to exclude from image synchronizations
      - Full Install (push everything)
      - Image update (push diffs)
      - Grab image (pull diffs)
      - Node reboot (push diffs)
  - **Finalize script**
    - Modifies image between load and boot
    - Can customize the image for each node

Node Configuration – Categories (part 2)

- **Nodes are in one category at a time**
  - **Disk Setup**
    - XML describing disk partitioning
  - **BMC User Name, User ID, Password**
    - Default to those in base partition
  - **Management Network**
    - Defaults to setting in base partition
  - **Install boot record**
    - Allows node to boot without esMS present
  - **IPMI power reset delay**
  - **Roles** (submode)
  - **Services** (submode)
  - **Install mode**
    - Auto – push image diffs
    - Full – push everything
    - Main – boot to maintenance shell
    - Nosync – don't push/pull image diffs
      - **NOTE**: if partition changes are detected, a full install will be performed.
  - **New node install mode**
    - Always a full install
  - **Notes** – User field for notes
  - **Revision** - User field for revision information

Node Configuration - Disk Setup

- Disk setup example of swap, /tmp, /var and /

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<diskSetup xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <device>
                 <blockdev>/dev/sda</blockdev>
                 <vendor>Dell</vendor>
                 <partition id="a1">
                            <size>64G</size>
                            <type>linux swap</type>
                 </partition>
                 <partition id="a2">
                            <size>64G</size>
                            <type>linux</type>
                            <filesystem>ext3</filesystem>
                            <mountPoint>/var</mountPoint>
                            <mountOptions>defaults,noatime,nodiratime</mountOptions>
                 </partition>
                 <partition id="a3">
                            <size>32G</size>
                            <type>linux</type>
                            <filesystem>ext3</filesystem>
                            <mountPoint>/tmp</mountPoint>
                            <mountOptions>defaults,noatime,nodiratime</mountOptions>
                 </partition>
                 <partition id="a4">
                            <size>max</size>
                            <type>linux</type>
                            <filesystem>ext3</filesystem>
                            <mountPoint>/</mountPoint>
                            <mountOptions>defaults,noatime,nodiratime</mountOptions>
                 </partition>
       </device>
</diskSetup>
```

Node Configuration – Categories

- **Filesystem Exports (esMS exports /cm/shared & /home)**
  - **All squash**
    - Default is “no”
  - **Anonymous GID**
    - Default is 65534
  - **Anonymous UID**
    - Default is 65534
  - **Async**
    - Default is “yes”
  - **Extra options**
  - **Hosts**
  - **Name**
    - Filesystem export name
  - **Path**
    - Filesystem path
  - **Revision**
    - User field for revision information
  - **Root squash**
    - Default is “no”
  - **Write**
    - Default is “no”
  - **FSID**
    - Default is “0”

Node Configuration – Categories

- Filesystem Mounts
  (slave nodes mount /cm/shared & /home)
  - Device
  - Dump
    - Default is “no”
  - Filesystem
    - Filesystem type
  - Filesystem check
    - Default is “0”
  - Mount options
  - Mount point
  - Revision
    - User field for revision information

Node Configuration – Devices (part 1) (nodes, switches and storage controllers)

- Many parameters are inherited from the category but may be overridden here
  - Activation
    - Timestamp of device creation
  - Hostname
  - IP
  - MAC
  - Management network
  - Category
  - Software image
  - Custom ping script and argument
  - Custom power script and argument
  - Custom remote console script and argument
  - Device height
  - Device position
  - Disk setup
  - Rack
  - Ethernet switch
  - Filesystem exports (submode)
  - Filesystem mounts (submode)
  - Finalize script
  - Initialize script
  - Install boot record
  - Install mode
  - Next install mode
  - Require FULL InstallConfirmation
  - Interfaces (submode)

Node Configuration – Devices (part 2) (nodes, switches and storage controllers)

- Many parameters are inherited from the category but may be overridden here
  - Network
  - Notes
    - User field for notes
  - PXE Label
  - Power control
  - IPMI power reset delay
  - PowerDistributionUnits
  - Provisioning Transport
  - Provisioning interface
  - Revision
    - User field for revision information
  - Roles (submode)
  - Services (submode)
  - Tag
  - Type
    - chassis, physicalnode,, headnode, ethernetswitch, ibswitch, genericdevice
  - Userdefined1
  - Userdefined2

Node Configuration – Devices (nodes, switches and storage controllers)

- Device Interfaces submode
  - **Additional hostnames**
  - **Card Type**
  - **DHCP** – yes/no
  - **Gateway** – override for this interface
  - **IP**
  - **MAC**
  - **Network** – network name from “network” mode
  - **Network device name** – ipmi0, eth0, ib0, etc.
  - **Revision** – user field for revision information
  - **StartIF** – when to start the interface (esMS only)
    - Active – if in failover pair
    - Passive – if in failover pair
    - Preferpassive
    - Always
  - **Type** – set when adding an interface
    - Alias
    - BMC
    - Physical

Node Configuration - Software Image

- Softwareimage parameters
  - **Creation time**
  - **Name**
  - **Path**
    - /cm/images/<name>
  - **Kernel modules** (submode)
  - **Kernel parameters**
  - **Kernel version**
  - **Locked**
  - **Notes** – user field for notes
  - **Revision** – user field for revision information
  - **Enable SOL**
    - SOL – Serial-Over-LAN
    - Set to “yes”
    - Provides remote console
  - **SOL Flow Control**
    - Set to “no”
  - **SOL Port**
    - ttys1
  - **SOL Speed**
    - Typically, 115200 – must match IPMI speed of the server

Administration - Node Groups

- **Allows nodes to be grouped together**
  - Simplifies operating on multiple nodes in different categories
  - A node can only be in **one category** at a time
  - A node can be in **multiple node groups** at a time

For example, if you have two Cray systems (cray1 and cray2) you could group all the esLogin nodes for each Cray system into two node groups (esl-cray1 and esl-cray2). This allows you to perform an operation on all them at one time.

Get device status for all esLogin nodes for both Cray systems

```
test-esms1:~ # cmsh -c “device -g esl-cray1,esl-cray2 status”
```

Resource Management – Adding/Modifying Devices, Networks and Categories

- **The general process described here can be applied for adding/modifying nodes, networks and categories.**
  - Enter the desired resource/mode in cmsh or cmgui
  - List members
  - Clone a member
  - Change parameters of the cloned member, as needed
  - Commit changes

  - **DO NOT FORGET TO COMMIT THE CHANGES OR THEY WILL NOT BE APPLIED TO THE RUNNING SYSTEM!**

Device Management – Adding a node

- **Clone an existing node**
  - Edit the parameters that need changing (IP address, MAC, etc.)
  - Example: List devices and clone a node

```
test-esms1:~ # cmsh
[test-esms1]% device
[test-esms1->device]% list
Type                   Hostname (key)   MAC                 Category         Ip              Network
PowerDistributio
---------------------- ---------------- ------------------- ---------------- --------------- --------------
----------------
EthernetSwitch         switch01         00:00:00:00:00:00                    10.141.253.1    esmaint-net
HeadNode               test-esms1       D4:AE:52:B5:E2:64                    10.141.255.254  esmaint-net
HeadNode               test-esms2       D4:AE:52:B5:A4:68                    10.141.255.253  esmaint-net
PhysicalNode           node001          00:00:00:00:00:00   default          10.141.0.1      esmaint-net
[test-esms1->device]%
```

- Clone an existing node (cloning node002 from node001)

```
[test-esms1->device]% clone node001 node002
The IP of network interface: BOOTIF was not updated
The IP of network interface: ipmi0 was not updated
Warning: The Ethernet switch settings were not cloned, and have to be set manually
[test-esms1->device*[node002*]]%
```

Device Management – Adding a node

- **Setting interface values**
  - After cloning, the interface values are the same as the original node
  - List interfaces

```
[test-esms1->device*[node002*]]% interfaces
[test-esms1->device*[node002*]->interfaces]% list
Type         Network device name  IP                Network
------------ -------------------- ---------------- ----------------
bmc          ipmi0                10.148.0.1       ipmi-net
physical     BOOTIF [prov]        10.141.0.1       esmaint-net
[test-esms1->device*[node002*]->interfaces]%
```

  - Need to modify IP address since these are clones of node001

```
[test-esms1->device*[node002*]->interfaces]% set ipmi0 ip 10.148.0.2
[test-esms1->device*[node002*]->interfaces*]% set bootif ip 10.141.0.2
```

Device Management – Adding a node

- ## Verifying interface values

  - Check and commit interface changes

```
[test-esms1->device*[node002*]->interfaces]% list
Type          Network device name  IP                Network
------------- -------------------- ----------------- ----------------
bmc           ipmi0                10.148.0.2        ipmi-net
physical      BOOTIF [prov]        10.141.0.2        esmaint-net
[test-esms1->device*[node002*]->interfaces]% commit
[test-esms1->device[node002]->interfaces]%
```

- ## Set MAC address of BOOTIF

```
[test-esms1->device[node002]->interfaces]% exit
[test-esms1->device[node002]]% set mac 12:34:56:78:90:AB
[test-esms1->device*[node002*]]% get mac
12:34:56:78:90:AB
[test-esms1->device*[node002]*]% commit
[test-esms1->device[node002]]%
```

What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview ✓
  - *Break* ✓
  - Device Configuration and Management ✓
  - Node Provisioning
  - Image Management
  - *Break*
  - System Monitoring
  - esMS Failover
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

Cray External Services Systems

Node Provisioning

Node Provisioning

- **esMS contains a system configuration database**
  - Partition, network, category, device and software image data is used by the node-installer to generate node specific files at boot time on each node.

- **Four install modes control installation**
  - **AUTO** – (default) - partitioning changes and image differences are pushed to the node
  - **FULL** – disks are repartitioned and the full image is pushed to the node
  - **MAIN** – installer stops in a maintenance shell for debugging install issues
  - **NOSYNC** – no changes are pushed*
    - ***NOTE:** if the partitioning scheme on the local disk is different than what is assigned, then a full install is done. To truly disable changing image contents, set the excludelists to exclude /*.

Node Provisioning – Boot process

- **Nodes PXE boot a node-installer from the esMS**
  - By default, the esMS prevents dhcpd from responding to unknown MAC addresses
  - Verifies node identity by MAC address
  - Installs the node-installer process in memory on the node
- **Node-installer is now running on the booting node**
  - Verifies disk partitioning is correct per the disksetup XML for the node
  - Image is then pulled to the local disk based on the install mode via rsync
  - The node finalize script is run
    - Allows node-specific modifications that are not part of the configuration database or in the software image
  - Node-installer pivots to the local disk, calls /init and exits
- **Ensures all nodes are running the desired image on boot**

Software Images and Node Provisioning

- **Local node modifications are lost on boot unless steps are taken in Bright Cluster Manager to preserve them**
  - Five exclude lists exist for each node category to protect files from being overwritten or lost
    - **excludelistfullinstall** for pushing full installs on new nodes
    - **excludelistsyncinstall** for pushing image diffs when rebooting nodes
    - **excludelistupdate** for pushing image diffs to running node
    - **excludelistgrab** for pulling image diffs from running node to esMS
    - **excludelistgrabnew** for pulling image diffs to a new image on the esMS
  - The **FrozenFiles** section in **/cm/local/apps/cmd/etc/cmd.conf** can be used to identify files that should not be touched by Bright Cluster Manager

- **When there are no image diffs, reboot time is similar to rebooting a stand-alone server.**
  - The only additional time is for loading the node-installer and verifying the partitioning.

Software Image Management – exclude lists

- ## Exclude list format

```
# For details on the exclude patterns defined here please refer to
# the FILTER RULES section of the rsync man page.
#
# Files that match these patterns will not be installed onto the node.
- lost+found/
- /proc/*
- /sys/*
- /lustre/scratch
- /nfsserver/home
```

What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview ✓
  - *Break* ✓
  - Device Configuration and Management ✓
  - Node Provisioning ✓
  - Image Management
  - *Break*
  - System Monitoring
  - esMS Failover
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

Cray External Services Systems

Software Image Management

Software Images

- **Images are located on the esMS in /cm/images**
  - Full Linux image with Bright Cluster Manager and Cray RPMs added
  - Bright Cluster Manager generates an initrd for the image that is extended for the node-installer process
- **Some system configuration files (or sections of them) are generated by Bright Cluster Manager and must not be modified**
  - Bright will overwrite them again unless marked as a “Frozen File”
  - It is possible to prevent Bright Cluster Manager from touching files by listing them as “**FrozenFiles**” in **/cm/local/apps/cmd/etc/cmd.conf** in the image
- **Images are used by multiple nodes**
  - No node-specific data should be in the image
    - The node-installer process generates all node-specific data from the configuration database and finalize script

Software Images

- **Software modifications can be made to the image on the esMS and pushed to running nodes**

```
test-esms1:~ # chroot /cm/images/my-image rpm …
test-esms1:~ # chroot /cm/images/my-image zypper …
test-esms1:~ # chroot /cm/images/my-image chkconfig foo on
```

  - May need to bind mount the image /dev, /proc and /sys to esMS /dev, /proc and /sys for some RPM installations

- **The “imageupdate” command pushes the image diffs to the target node**

  - Example, update the node named node001

```
test-esms1:~ # cmsh
[test-esms1]% device use node001
[test-esms1->device[node001]]% imageupdate      <- NOTE: performs a dry run
[test-esms1->device[node001]]% synclog          <- NOTE: displays sync log for examination
[test-esms1->device[node001]]% imageupdate -w   <- NOTE: performs actual update to node
```

Software Images

- Software modifications can be made on a running node and captured to the image on the esMS

- The “grabimage” command pulls the image diffs from the target node to the esMS

  - Example, update the image on the esMS from the node named node001

```
test-esms1:~ # cmsh
[test-esms1]% device use node001
[test-esms1->device[node001]]% grabimage      <- NOTE: performs a dry run
[test-esms1->device[node001]]% synclog        <- NOTE: displays sync log for examination
[test-esms1->device[node001]]% grabimage -w   <- NOTE: performs actual update to image
```

Software Images – Recommended Practices

- **Updating Images**
  - Clone the image to update
    - Preserves current image in case the updates don’t perform as desired
  - Apply updates to the cloned image
  - Create a test category and assign the cloned image to it
  - Assign a test node to the test category
  - Reboot the test node and verify the image functions as desired
  - If the updates are successful, assign the updated image to the original category or categories
  - Assign the test node to its original category
  - Reboot or use imageupdate to apply the image updates to the desired nodes

  - **NOTE:** Due to limited disk space on the esMS (~3TB in /cm/images), it is recommended that sites archive and remove unused images from the esMS
    - Process is in the Admin Guide

What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview ✓
  - *Break* ✓
  - Device Configuration and Management ✓
  - Node Provisioning ✓
  - Image Management ✓
  - *Break*
  - System Monitoring
  - esMS Failover
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

Cray External Services Systems

System Monitoring

Monitoring

- **Bright Cluster Manager provides a flexible monitoring framework**
  - Two categories of monitors – Health Checks and Metrics
  - Many built-in monitors are automatically setup based on hardware discovery
- **Health Checks**
  - Run periodically by the management daemon
  - May run on the esMS, node or both
  - Return PASS, FAIL or UNKNOWN
  - Actions can be taken based on the return value
- **Metrics**
  - Run periodically by the management daemon
  - May run on the esMS or node
  - Return a numeric value
  - Actions can be taken based on crossing a threshold value

Monitoring

- **Creating a healthcheck**
  - Use **/cm/local/apps/cmd/scripts/healthchecks/testhealthcheck** as a template.
  - Echo informational messages to fd 3 (echo “some info” >&3). These show up in the cmsh/cmgui events display.
  - Healthcheck results are sent by echoing “PASS”, “FAIL”, or “UNKNOWN” to stdout. Stdout is not seen by the user.

- **Creating a metric**
  - Use **/cm/local/apps/cmd/scripts/metrics/testmetric** as a template
  - Echo informational messages to fd 3 (echo “some info” >&3). These show up in the cmsh/cmgui events display.
  - Metric results are sent by echoing the metric value to stdout. Stdout is not seen by the user.

Monitoring

- **Creating an action**
  - Use **/cm/local/apps/cmd/scripts/actions/testaction** as a template
- **Anything you can script can be a healthcheck, metric or action**
  - The templates list environment variables made available by Bright Cluster Manager

Monitoring - Health Checks

- **Healthchecks – return PASS, FAIL or UNKNOWN**
  - **Class of healthcheck** – cluster, cpu, disk, env, internal, mem, misc, net, os, prototype, workload
  - **Command** – path to the healthcheck command
  - **Description** – description of the healthcheck
  - **Disabled** – “yes” or “no”
  - **Extended environment** – “yes” or “no”
  - **Name** –healthcheck name
  - **Notes** – user field for notes
  - **Only when idle** – “yes” or “no”. Run only when the esMS is idle or not.
  - **Parameter permissions**
  - **Revision**
  - **Sampling method** – “samplingonmaster” or “samplingonnode”
  - **State flapping count** – default is “7”. If a state changed state 7 times in the last 12 samples, it is “flapping”
  - **Timeout**
  - **Valid for** – Ethernet, generic, Infiniband, headnode, node

Monitoring – Creating a Healthcheck

```
test-esms1:~ # cmsh
[test-esms1]% monitoring healthchecks
[test-esms1->monitoring->healthchecks]% add myhealthcheck
[test-esms1->monitoring->healthchecks*[myhealthcheck*]]% show
Parameter                      Value
------------------------------
------------------------------------------------
Class of healthcheck           misc
Command
Description
Disabled                       no
Extended environment           no
Name                           myhealthcheck
Notes                          <0 bytes>
Only when idle                 no
Parameter permissions          optional
Revision
Sampling method                samplingonnode
State flapping count           7
Timeout                        5
Valid for                      node,headnode
[test-esms1->monitoring->healthchecks*[myhealthcheck*]]%
```

- Use the “set” command to set the parameters, as needed
- **Don’t forget to “commit” after all parameters are set**

Monitoring - Metrics

- **Metrics – return a numeric value**
  - **Class of metric**– cluster, cpu, disk, env, internal, mem, misc, net, os, prototype, workload
  - **Command** – path to the metric command
  - **Cumulative** – “yes” or “no”
  - **Description** – description of the metric
  - **Disabled** – “yes” or “no”
  - **Extended environment** – “yes” or “no”
  - **Maximum** – max normal value
  - **Measurement unit**
  - **Minimum** – min normal value
  - **Name** – metric name
  - **Notes** – user field for notes
  - **Only when idle** – “yes” or “no”. Run only when the esMS is idle or not.
  - **Parameter permissions Retrieval method** – “cmdaemon” or “snmp”
  - **Revision** – user field for revision information
  - **Sampling method** – “samplingonmaster” or “samplingonnode”
  - **State flapping count** – default is “7”. If a state changed state 7 times in the last 12 samples, it is “flapping”
  - **Timeout**
  - **Valid for** – Ethernet, generic, Infiniband, headnode, node

Monitoring – Creating a Metric

```
test-esms1:~ # cmsh
[[test-esms1]% monitoring metrics
[test-esms1->monitoring->metrics]% add mymetric
[test-esms1->monitoring->metrics*[mymetric*]]% show
Parameter                      Value
------------------------------ ------------------------------------------------
Class of metric                misc
Command
Cumulative                     no
Description
Disabled                       no
Extended environment           no
Maximum                        <range not set>
Measurement Unit
Minimum                        <range not set>
Name                           mymetric
Notes                          <0 bytes>
Only when idle                 no
Parameter permissions          optional
Retrieval method               cmdaemon
Revision
Sampling method                samplingonnode
State flapping count           7
Timeout                        5
Valid for                      node,headnode
[test-esms1->monitoring->metrics*[mymetric*]]%
```

- **Use the "set" command to set the parameters, as needed**
- **Don't forget to "commit" after all parameters are set**

Monitoring – Actions

- **Actions - optional actions taken as a result of a healthcheck or metric**
  - **Command** – path to the command to execute when this action is called
  - **Description** – text description of the action
  - **Name** – the name of the action
  - **Revision** – optional revision field
  - **Run on** – where the action is run: “master” or “node”
  - **Timeout** – action timeout
  - **isCustom** – “yes” or “no”

Monitoring – Creating an Action

- Called as a result of a healthcheck or metric

```
test-esms1:~ # cmsh
[test-esms1]% monitoring actions
[test-esms1->monitoring->actions]% add myaction
[test-esms1->monitoring->actions*[myaction*]]% show
Parameter                      Value
------------------------------ ------------------------------------------------
Command
Description
Name                           myaction
Revision
Run on                         headnode
Timeout                        5
isCustom                       yes
[test-esms1->monitoring->actions*[myaction*]]%
```

- Use the “set” command to set the parameters, as needed
- Don’t forget to “commit” after all parameters are set

Monitoring Configuration

- Can be configured for any of the following:
  - Headnode
  - Any node category

Monitoring – Healthcheck Setup Parameters (part 1)

- **Check Interval**
  - Number of seconds between checks
- **Disabled**
  - “yes” or “no”
- **Fail Actions**
  - optional action to take on healthcheck “FAIL” state
- **Fail severity**
  - numeric severity value for “FAIL” state
- **GapThreshold**
  - the number of missing samples allowed before recording a null
- **Healthcheck**
  - name of the healthcheck
- **HealthCheckParam**
  - parameters passed to the healthcheck
- **LogLength**
  - how many results to log
- **Only when idle**
  - “yes” or “no”

Monitoring – Healthcheck Setup Parameters (part 2)

- **Pass Actions**
  - optional action to take on healthcheck "pass" state
- **Revision**
  - optional revision
- **Stateflapping Actions**
  - optional action to take on state flapping condition
- **Store**
  - "yes" or "no", to store results in the healthcheck log
- **ThresholdDuration**
  - number of samples in the threshold zone before a threshold event is decided to have occurred
- **Unknown Actions**
  - optional action to take on healthcheck "unknown" state
- **Unknown severity**
  - numeric severity value for "unknown" state

Monitoring – Configuring a Healthcheck

```
test-esms1:~ # cmsh
[test-esms1]% monitoring setup healthconf headnode
[test-esms1->monitoring->setup[HeadNode]->healthconf]% add myhealthcheck
[test-esms1->monitoring->setup*[HeadNode*]->healthconf*[myhealthcheck*]]%
show
Parameter                      Value
------------------------------
------------------------------------------------
Check Interval                 120
Disabled                       no
Fail Actions
Fail severity                  10
GapThreshold                   2
HealthCheck                    myhealthcheck
HealthCheckParam
LogLength                      3000
Only when idle                 no
Pass Actions
Revision
Stateflapping Actions
Store                          yes
ThresholdDuration              1
Unknown Actions
Unknown severity               10
[test-esms1->monitoring->setup*[HeadNode*]->healthconf*[myhealthcheck*]]%
```

- Use the “set” command to set the parameters, as needed
- Don’t forget to “commit” after all parameters are set

Monitoring – Metric Setup Parameters (part 1)

- **Consolidators**
  - submode to consolidate results by day, hour and week
- **Disabled**
  - “yes” or “no”
- **GapThreshold**
  - the number of missing samples allowed before recording a null
- **LogLength**
  - how many results to log
- **Metric**
  - name of the metric
- **MetricParam**
  - optional parameter passed to the metric
- **Only when idle**
  - “yes” or “no”
- **Revision**
  - user field for revision information
- **Sampling Interval**

Monitoring – Metric Setup Parameters (part 2)

- **Stateflapping Actions**
  - optional action to take on state flapping condition
- **Store**
  - “yes” or “no”, to store results in the log
- **ThresholdDuration**
  - number of samples in the threshold zone before a threshold event is decided to have occurred
- **Thresholds**
  - submode defining triggers based on crossing threshold values
  - **Actions**
  - **Bound**
  - **Name**
  - **Revision**
  - **Severity**
  - **UpperBound**

Monitoring – Configuring a Metric

```
test-esms1:~ # cmsh
[test-esms1]% monitoring setup metricconf headnode
[test-esms1->monitoring->setup[HeadNode]->metricconf]% add mymetric
[test-esms1->monitoring->setup*[HeadNode*]->metricconf*[mymetric*]]%
show
Parameter                      Value
------------------------------
------------------------------------------------
Consolidators                  <3 in submode>
Disabled                       no
GapThreshold                   2
LogLength                      3000
Metric                         mymetric
MetricParam
Only when idle                 no
Revision
Sampling Interval              120
Stateflapping Actions
Store                          yes
ThresholdDuration              1
Thresholds                     <0 in submode>
[test-esms1->monitoring->setup*[HeadNode*]->metricconf*[mymetric*]]%
```

- Use the “set” command to set the parameters, as needed
- Don’t forget to “commit” after all parameters are set

Monitoring – Checking Healthcheck Status using “latesthealthdata”

```
[test-esms1->device]% help latesthealthdata
Name:
          latesthealthdata - Get latest health data

Usage:
          latesthealthdata [OPTIONS] [device]

Options:
          -v, --verbose
               Show multiline info messages

          -d, --delimiter <string>
               Use <string> as delimiter between
               columns

Examples:
          latesthealthdata -d "|"      Latesthealthdata for the current device
          latesthealthdata node001     Latesthealthdata for node001

[test-esms1->device]%
```

Monitoring – Checking Healthcheck Status using “latesthealthdata”

```
test-esms1:~ # cmsh
[test-esms1]% device use test-esms1
[test-esms1->device[test-esms1]]% latesthealthdata
Health Check                 Severity Value            Age (sec.) Info Message
---------------------------- -------- ---------------- ----------
----------------------------------------
DeviceIsUp                   0        PASS             44
ManagedServicesOk            0        PASS             404
mounts                       0        PASS             279884
exports                      0        PASS             279884
smart                        0        PASS             279884     sdb: Smart
command failed              +
ldap                         0        PASS             279884
failover                     0        PASS             279884
interfaces                   0        PASS             279884
oomkiller                    0        PASS             279884
cmsh                         0        PASS             279884
mysql                        0        PASS             279884
failedprejob                 0        PASS             279884
diskspace:2% 10% 20%         0        PASS             279884
ntp                          0        PASS             1484
schedulers                   0        PASS             279884
chrootprocess                0        PASS             279884
esfsmon:scratch              0        PASS             44         NO standby MDS.
MDS failover disabled.
[test-esms1->device[test-esms1]]%
```

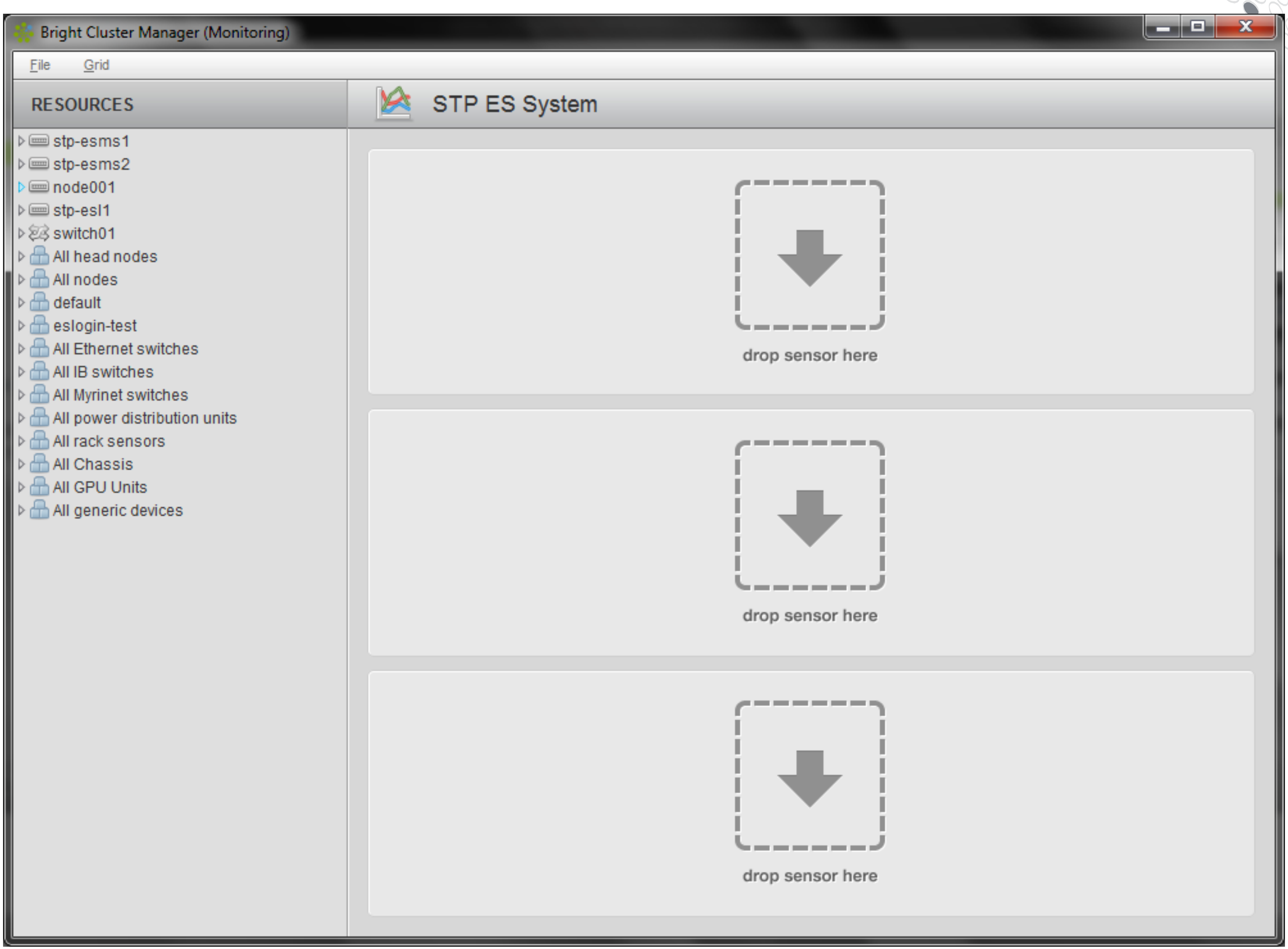
Monitoring – Checking Metric Status using “latestmetricdata”

```
[test-esms1->device]% help latestmetricdata
Name:
          latestmetricdata - Get latest metric data

Usage:
          latestmetricdata [OPTIONS] [device]

Options:
          -v, --verbose
               Show multiline info messages

          -d, --delimiter <string>
               Use <string> as delimiter between
               columns

Examples:
          latestmetricdata -d "|"      Latestmetricdata for the current device
          latestmetricdata node001     Latestmetricdata for node001

[test-esms1->device]%
```

Monitoring – Checking Metric Status using “latestmetricdata”

```
[test-esms1->device[test-esms1]]% latestmetricdata
Metric                                     Value             Age (sec.) Info Message
------------------------------------------ ---------------- ----------
----------------------------------------
AlertLevel:max                             0                 110
AlertLevel:sum                             0                 110
AvgExpFactor                               0                 110
BufferMemory                               1.03167e+09       110
BytesRecv:eth0                             4905.66           110
BytesRecv:eth1                             1388.45           110
BytesRecv:eth2                             44.4333           110
BytesRecv:eth3                             145.733           110
BytesRecv:ib0                              0                 110
BytesRecv:ib1                              0                 110
BytesSent:eth0                             4722.88           110
BytesSent:eth1                             0                 110
BytesSent:eth2                             24.1667           110
BytesSent:eth3                             145.733           110
BytesSent:ib0                              0                 110
BytesSent:ib1                              0                 110
CMDMemUsed                                 7.44612e+07       110
CPUCoresAvailable                          96                110
CPUIdle                                    3198.94           110
CPUIrq                                     0                 110
CPUNice                                    0                 110
CPUSoftIrq                                 0.0166667         110
CPUSystem                                  0.516667          110
….
[test-esms1->device[test-esms1]]%
```

Health and Metric History - cmgui



- stp-esms1
- stp-esms2
- node001
- stp-esl1
- switch01
- All head nodes
- All nodes
- default
- eslogin-test
- All Ethernet switches
- All IB switches
- All Myrinet switches
- All power distribution units
- All rack sensors
- All Chassis
- All GPU Units
- All generic devices

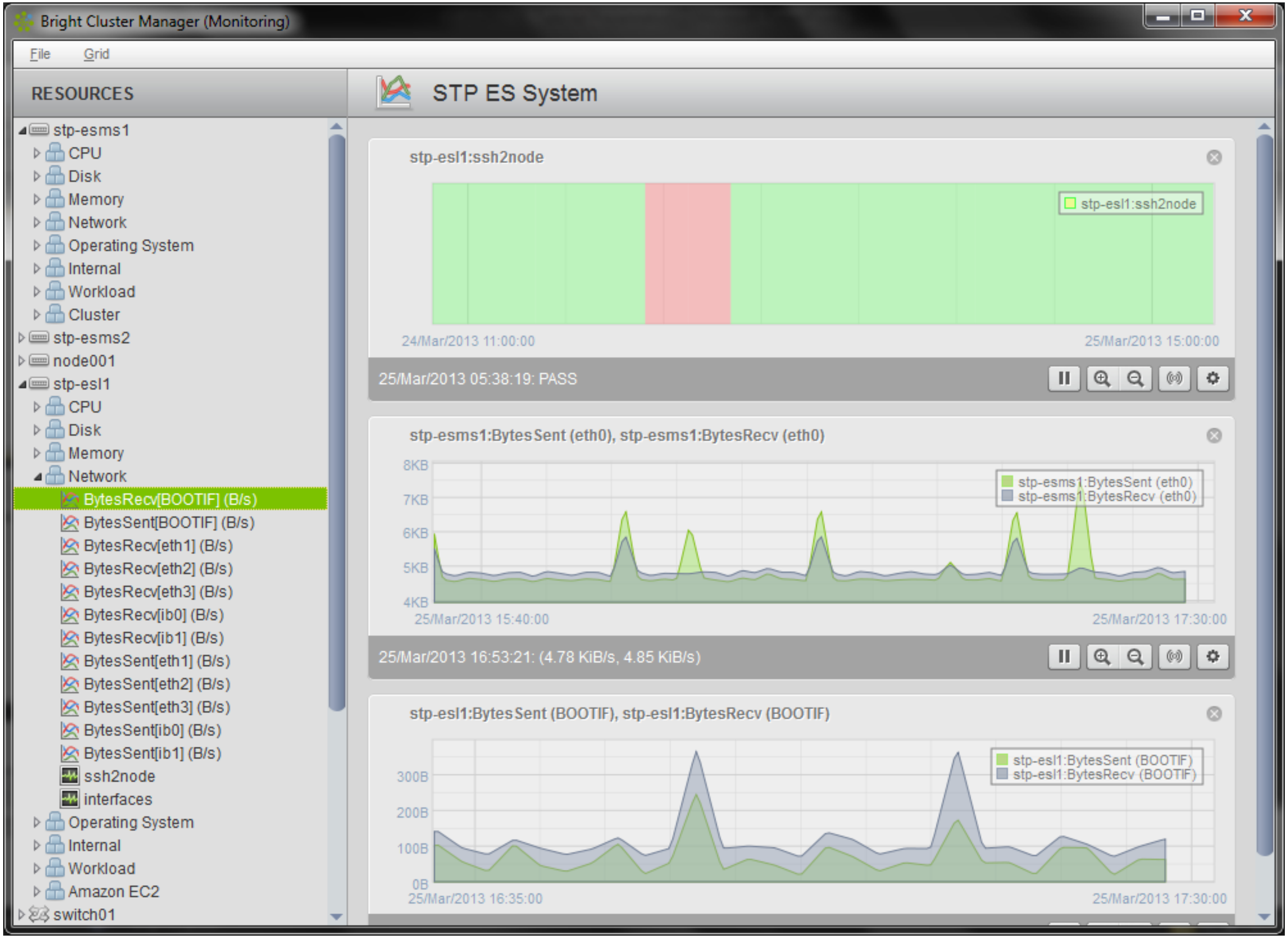
STP ES System

drop sensor here

drop sensor here

drop sensor here

Health and Metric History - cmgui



Bright Cluster Manager (Monitoring)

File Grid

RESOURCES

- stp-esms1
  - CPU
  - Disk
  - Memory
  - Network
  - Operating System
  - Internal
  - Workload
  - Cluster
- stp-esms2
- node001
- stp-esl1
  - CPU
  - Disk
  - Memory
  - Network
    - BytesRecv[BOOTIF] (B/s)
    - BytesSent[BOOTIF] (B/s)
    - BytesRecv[eth1] (B/s)
    - BytesRecv[eth2] (B/s)
    - BytesRecv[eth3] (B/s)
    - BytesRecv[ib0] (B/s)
    - BytesRecv[ib1] (B/s)
    - BytesSent[eth1] (B/s)
    - BytesSent[eth2] (B/s)
    - BytesSent[eth3] (B/s)
    - BytesSent[ib0] (B/s)
    - BytesSent[ib1] (B/s)
    - ssh2node
    - interfaces
  - Operating System
  - Internal
  - Workload
  - Amazon EC2
- switch01

STP ES System

stp-esl1:ssh2node

25/Mar/2013 05:38:19: PASS

stp-esms1:BytesSent (eth0), stp-esms1:BytesRecv (eth0)

25/Mar/2013 16:53:21: (4.78 KiB/s, 4.85 KiB/s)

stp-esl1:BytesSent (BOOTIF), stp-esl1:BytesRecv (BOOTIF)

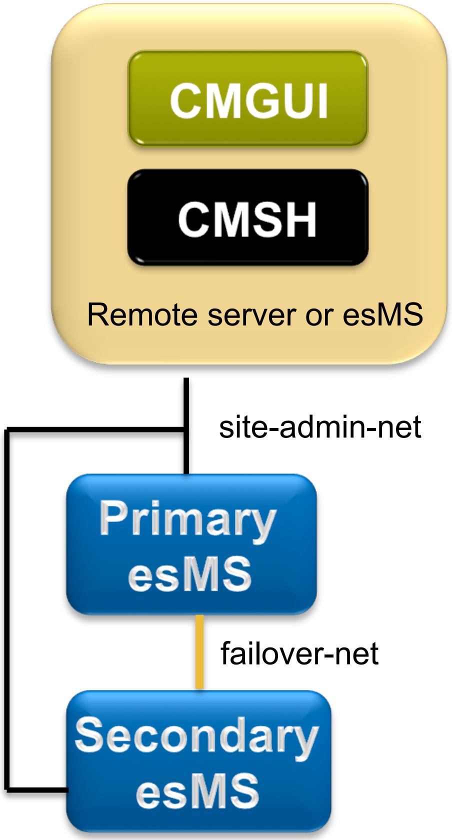
What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview ✓
  - *Break* ✓
  - Device Configuration and Management ✓
  - Node Provisioning ✓
  - Image Management ✓
  - *Break* ✓
  - System Monitoring ✓
  - esMS Failover
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

Cray External Services Systems

esMS Failover

High Availability esMS Configuration



- **High Availability (HA) esMS servers are in an active/passive configuration**
  - The following services are running on both esMS servers:
    - **CMDaemon**: providing functionality on both esMS servers (provisioning)
    - **DHCP**: load balanced setup
    - **LDAP**: running in replication mode (the active esMS LDAP database is pulled by the passive)
    - **MySQL**: running in master-slave replication mode (the active esMS MySQL database is pulled by the passive)
    - **NTP**
    - **DNS**
  - When an HA esMS setup is created from a single esMS setup, the above services are automatically reconfigured to run in the HA environment over two esMS servers.

High Availability esMS Configuration

- **DRBD provides shared storage between the esMS servers**
  - DRBD (Distributed Replicated Block Device) is similar to RAID1 over a network (mirrored file system)
  - Only the active esMS mounts the DRBD file systems to prevent a split brain scenario

Device	Size	File System
/dev/drbd1	16G	/var/log/conman
/dev/drbd2	~367G	/cm/shared
/dev/drbd3	1G	/var/esfsmon
/dev/drbd4	1G	/cm/node-installer/certificates
/dev/drbd5	20G	/home

High Availability esMS Configuration

- **Both esMS servers provision slave nodes**
  - Images are kept in sync between esMS servers in three ways
    - Automatically every 5 minutes (configurable in /cm/local/apps/cmd/etc/cmd.conf)
    - When a node makes a boot request
    - Manually using the “updateprovisioners” command in cmsh softwareimage mode

- **Automated failover is possible if the esMS has the ability to control power of the other**
  - Initiated if the passive esMS loses contact with the active esMS **AND** more than half the slave nodes report that they cannot contact the active esMS.
  - The passive esMS will STONITH the unresponsive active esMS to prevent a split brain scenario
  - Failover will not take place if the active esMS is gracefully shutdown

High Availability esMS Configuration

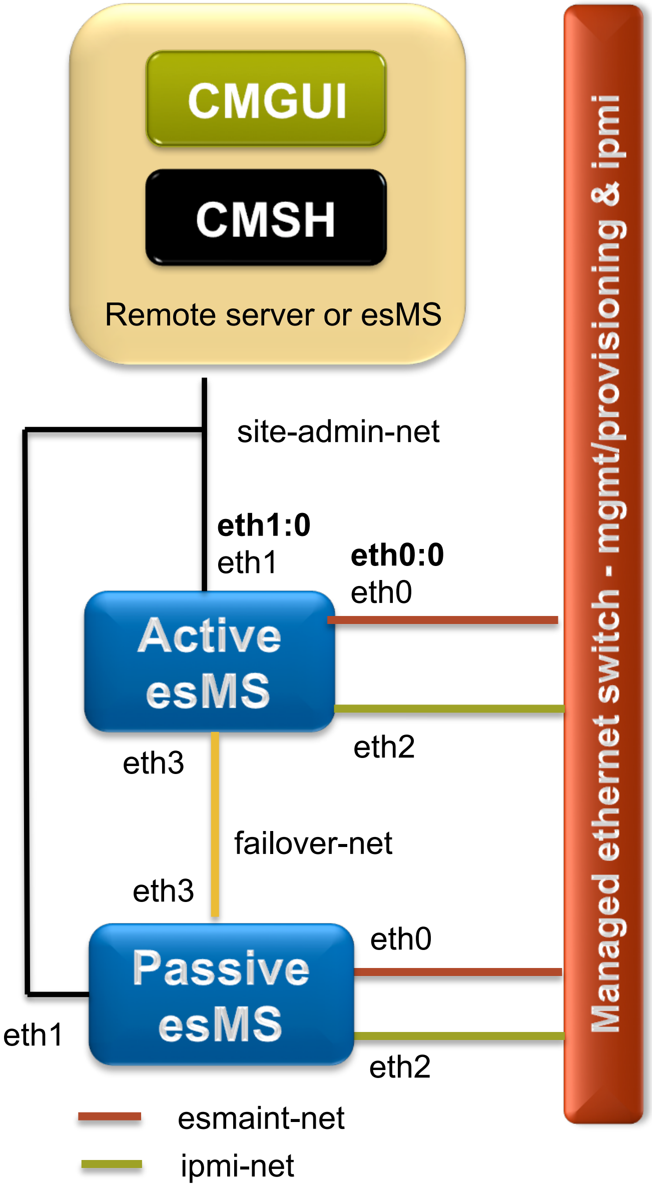
- **Manual Failover**
  - The “cmha makeactive” command executed on the passive esMS will initiate a manual failover

- **Software Maintenance**
  - It is important to perform software updates on both esMS servers to keep them in sync
  - Only the slave software images, management database and log files are kept in sync automatically

- **HA Status**
  - The “cmha status” command will display HA status

```
test-esms1:~ # cmha status
Node Status: running in active master mode

Failover status:
test-esms1* -> test-esms2
  backupping    [  OK  ]
  mysql         [  OK  ]
  ping          [  OK  ]
  status        [  OK  ]
test-esms2 -> test-esms1*
  backupping    [  OK  ]
  mysql         [  OK  ]
  ping          [  OK  ]
  status        [  OK  ]
test-esms1:~ #
```

esMS Failover Networking



- **High Availability Networking**
  - The site-admin-net interface (eth1) is aliased with a shared IP address (eth1:0)
    - Use this IP address to log into the active esMS
    - esMS hostname recommendations
      - Add “-esms1” and “-esms2” suffix to the primary and secondary esMS hostnames (eth1 IP addresses)
      - Use “-esms” suffix for the shared IP (eth1:0) hostname
      - For example, stp-esms1, stp-esms2 and stp-esms
  - The esmaint-net interface (eth0) is aliased with a shared IP address (eth0:0)
    - Maintains connection with the slave nodes
  - Only the active esMS will activate these alias interfaces

What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview ✓
  - *Break* ✓
  - Device Configuration and Management ✓
  - Node Provisioning ✓
  - Image Management ✓
  - *Break* ✓
  - System Monitoring ✓
  - esMS Failover ✓
  - Automated Lustre Failover and lustre_control
  - Troubleshooting

Cray External Services Systems

Automated Lustre Failover and lustre_control

Lustre Filesystems – lustre_control

- **The lustre_control utility is used for managing the Lustre Filesystems**
- **The following operations can be performed on a single Lustre Filesystem or all Lustre Filesystems**
  - Format/Reformat
  - Start
  - Stop
  - Tune
  - Check status
  - Failover/failback

- **lustre_control is executed on the esMS**
  - Some components reside on the esMS and esFS nodes
  - Filesystems are defined in fs_defs files and installed by lustre_control

Lustre Filesystems – lustre_control

```
lustre_control

Valid commands are:
    install             Install file system definitions.
    remove              Remove file system definitions
    start               Start services on Lustre servers.
    stop                Stop services on Lustre servers.
    reformat            Format Lustre devices.
    write_conf          Regenerate Lustre configuration logs.
    failover            Failover services from their specified primary
                        server(s) to their respective backup server(s).
    failback            Failback services onto their specified primary
                        server(s) from their respective backup server(s).
    status              Reports the status of Lustre services.
    verify_config       Report differences between expected e2label and
                        actual
                        e2label for Lustre devices.
    dump_csv_config     Print to stdout the file system configuration in a
                        comma-separated value format
    help                Display detailed usage information for a particular
                        command.
    set_tune            Set Lustre tunable parameters.

See 'lustre_control help COMMAND' for more information on a specified
command, or 'man lustre_control' for more detailed information.
```

Automated Lustre Failover Monitor - esfsmon

- **esfsmon – automated Lustre failover**
  - Implemented as the esfsmon healthcheck for each Lustre filesytem
    - Takes the filesystem name as a parameter when setting up the healthcheck in Bright Cluster Manager
      - Allows monitoring of multiple Lustre filesystems
    - **/cm/local/apps/cmd/scripts/healthchecks/esfsmon_healthcheck**
    - **/cm/local/apps/cmd/etc/esfsmon.conf** is the configuration file
    - Tests on the nodes are executed by the CMDaemon on that node
      - No ssh connection overhead
  - Executes tests in parallel to half the filesystem nodes at a time
    - Uses node categories to target nodes
    - Data is gathered from the esMS and directly on the nodes
  - Calls esfsmon_action if a failure is detected
    - **/cm/local/apps/cmd/scripts/actions/esfsmon_action**
      - Does some housekeeping
      - STONITH the failed node
      - Calls lustre_control to execute the failover

Automated Lustre Failover Monitor - esfsmon

- **esfsmon – 3 operational modes**
  - **Normal** – performs all tests and will execute the failover operation on test failure
    - Failures are logged to /var/log/messages
  - **Runsafe** – performs all tests but will not execute the failover operation on test failure
    - Failures are logged to /var/log/messages
    - Turned on by existence of /var/esfsmon/esfsmon_runsafe_<filesystem> file
  - **Suspended** – no tests are performed
    - Turned on by existence of /var/esfsmon/esfsmon_suspend_<filesystem> file

Automated Lustre Failover Monitor – esfsmon.conf

- **esfsmon.conf – automated Lustre failover configuration**
  - **/cm/local/apps/cmd/etc/esfsmon.conf** is the configuration file
  - Sourced by **esfsmon_healthcheck**, **esfsmon_action** and **esfsmon_failback** to get common parameters

  - Defines the node categories used for each Lustre filesystem
    - Healthcheck operations are performed by category
    - There are six node categories used for each esFS system
      - The suffix of the category name is the filesystem. The following shows category names for the "scratch" filesystem
        - **MDS primary node** - esfs-mds-scratch
        - **MDS failover node** - esfs-mds-fo-scratch
        - **MDS failed node** – esfs-mds-failed-scratch - if there is a node assigned to this category, the system has a failed MDS node and MDS failover is disabled
        - **OSS even numbered nodes** - esfs-oss-even-scratch
        - **OSS odd numbered nodes** – esfs-oss-odd-scratch
        - **OSS failed nodes** – esfs-oss-failed-scratch - if there is a node assigned to this category, the system has a failed OSS node and runsafe mode is set

Automated Lustre Failover Monitor – esfsmon_failback

- **esfsmon_failback**
  - Places a node which was failed over by esfsmon back into service
  - Performs housekeeping and calls lustre_control to failback the node
  - Using ‘lustre_control failback’ instead of esfsmon_failback results in nodes that are not in their proper category and esfsmon will not test them

What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview ✓
  - *Break* ✓
  - Device Configuration and Management ✓
  - Node Provisioning ✓
  - Image Management ✓
  - *Break* ✓
  - System Monitoring ✓
  - esMS Failover ✓
  - Automated Lustre Failover and lustre_control ✓
  - Troubleshooting

Cray External Services Systems

Troubleshooting

Log Files – located on the esMS

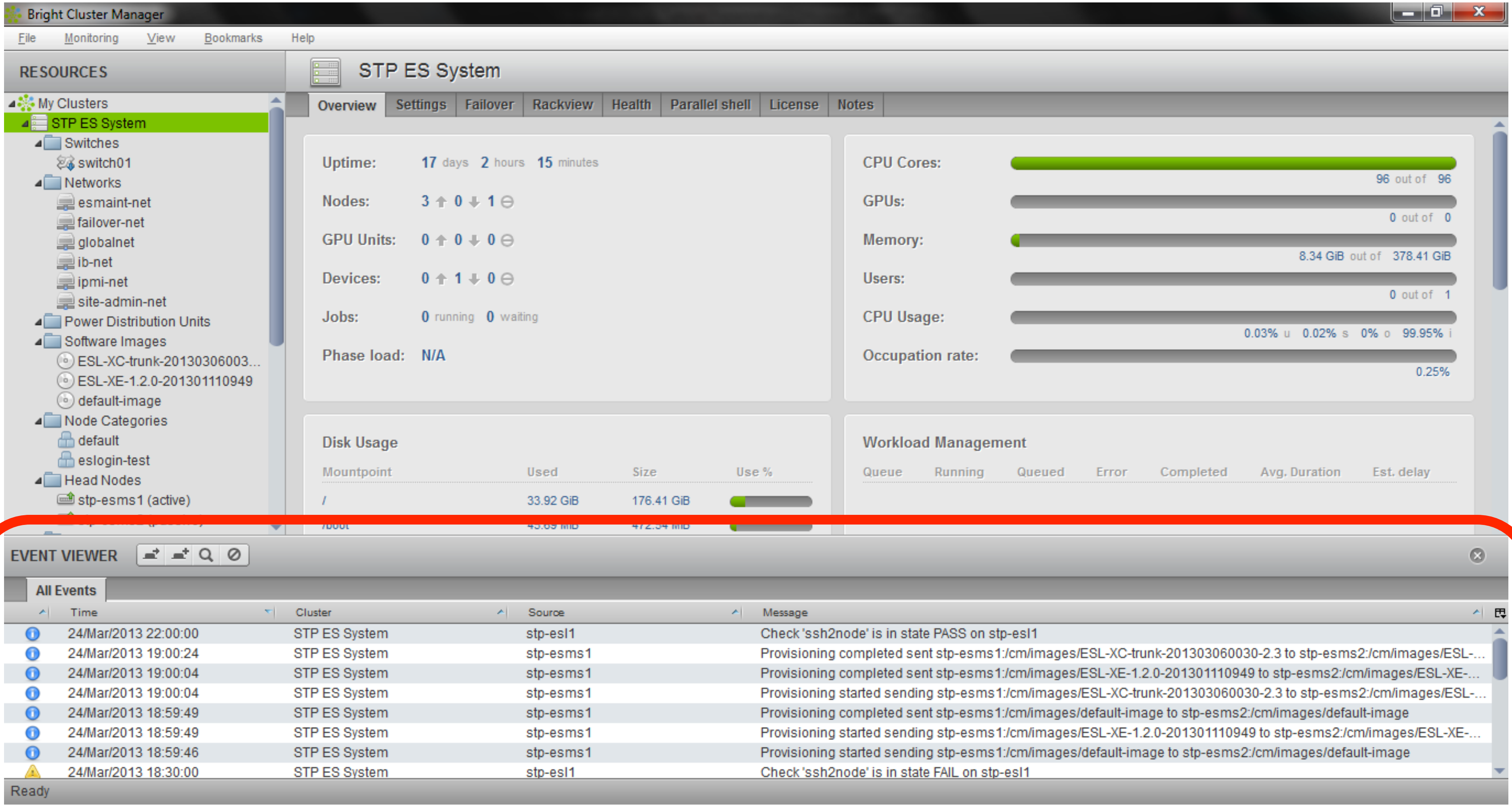
- Syslog – All slave nodes forward syslog to the esMS
  - /var/log/messages
- CMDaemon – Management daemon log
  - /var/log/cmdaemon
- Node-installer – Node Installer log
  - /var/log/node-installer
- Conman – esLogin and esFS Node console logs
  - /var/log/conman/
- Software Installation Logs
  - /var/adm/cray/logs
- Bright Cluster Manager Event Log
  - Stored in the Bright Cluster Manager database
  - Accessed via “events” command in cmsh or event viewer in cmgui

Event Log – viewing in cmsh

- **Get the last 5 events…**

```
[test-esms1->device[node001]]% events 5
Tue Aug 21 10:59:07 2012 [notice] esms: Starting periodic update of software
image(s) on provisioning node(s).
Tue Aug 21 10:59:39 2012 [notice] test-esms1: Finished updating software image(s)
on provisioning node(s).
Tue Aug 21 11:00:21 2012 [notice] node001: Check 'DeviceIsUp' is in state PASS on
node001
Tue Aug 21 13:30:00 2012 [warning] node003: Check 'rogueprocess' is in state FAIL
on node003
For details type: events details 4818
Tue Aug 21 14:01:00 2012 [notice] node003: Check 'rogueprocess' is in state PASS
on node003
[test-esms1->device[node001]]%
```

Event Viewer - cmgui



Uptime: 17 days 2 hours 15 minutes

Nodes: 3 ⬆ 0 ⬇ 1 ⊖

GPU Units: 0 ⬆ 0 ⬇ 0 ⊖

Devices: 0 ⬆ 1 ⬇ 0 ⊖

Jobs: 0 running 0 waiting

Phase load: N/A

CPU Cores: 96 out of 96

GPUs: 0 out of 0

Memory: 8.34 GiB out of 378.41 GiB

Users: 0 out of 1

CPU Usage: 0.03% u 0.02% s 0% o 99.95% i

Occupation rate: 0.25%

Disk Usage

Mountpoint	Used	Size	Use %
/	33.92 GiB	176.41 GiB	
/boot	45.69 MiB	472.54 MiB	

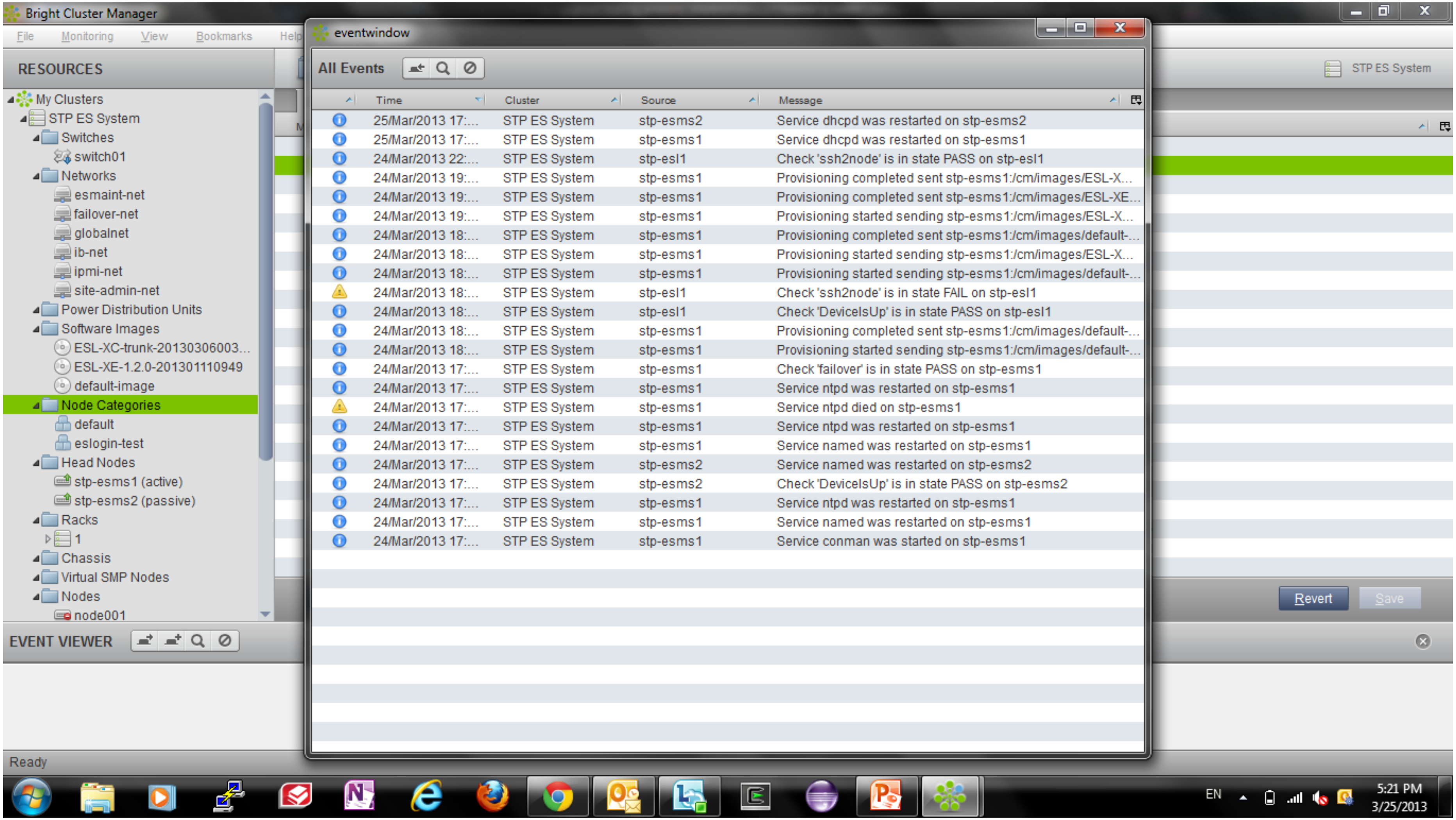
Workload Management

Queue	Running	Queued	Error	Completed	Avg. Duration	Est. delay

	Time	Cluster	Source	Message
ⓘ	24/Mar/2013 22:00:00	STP ES System	stp-esl1	Check 'ssh2node' is in state PASS on stp-esl1
ⓘ	24/Mar/2013 19:00:24	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XC-trunk-201303060030-2.3 to stp-esms2:/cm/images/ESL-...
ⓘ	24/Mar/2013 19:00:04	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XE-1.2.0-201301110949 to stp-esms2:/cm/images/ESL-XE-...
ⓘ	24/Mar/2013 19:00:04	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-XC-trunk-201303060030-2.3 to stp-esms2:/cm/images/ESL-...
ⓘ	24/Mar/2013 18:59:49	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/default-image to stp-esms2:/cm/images/default-image
ⓘ	24/Mar/2013 18:59:49	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-XE-1.2.0-201301110949 to stp-esms2:/cm/images/ESL-XE-...
ⓘ	24/Mar/2013 18:59:46	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/default-image to stp-esms2:/cm/images/default-image
⚠	24/Mar/2013 18:30:00	STP ES System	stp-esl1	Check 'ssh2node' is in state FAIL on stp-esl1

Ready

Event Viewer – cmgui (detached)



	Time	Cluster	Source	Message
ⓘ	25/Mar/2013 17:...	STP ES System	stp-esms2	Service dhcpd was restarted on stp-esms2
ⓘ	25/Mar/2013 17:...	STP ES System	stp-esms1	Service dhcpd was restarted on stp-esms1
ⓘ	24/Mar/2013 22:...	STP ES System	stp-esl1	Check 'ssh2node' is in state PASS on stp-esl1
ⓘ	24/Mar/2013 19:...	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-X...
ⓘ	24/Mar/2013 19:...	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/ESL-XE...
ⓘ	24/Mar/2013 19:...	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-X...
ⓘ	24/Mar/2013 18:...	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/default-...
ⓘ	24/Mar/2013 18:...	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/ESL-X...
ⓘ	24/Mar/2013 18:...	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/default-...
⚠	24/Mar/2013 18:...	STP ES System	stp-esl1	Check 'ssh2node' is in state FAIL on stp-esl1
ⓘ	24/Mar/2013 18:...	STP ES System	stp-esl1	Check 'DeviceIsUp' is in state PASS on stp-esl1
ⓘ	24/Mar/2013 18:...	STP ES System	stp-esms1	Provisioning completed sent stp-esms1:/cm/images/default-...
ⓘ	24/Mar/2013 18:...	STP ES System	stp-esms1	Provisioning started sending stp-esms1:/cm/images/default-...
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms1	Check 'failover' is in state PASS on stp-esms1
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms1	Service ntpd was restarted on stp-esms1
⚠	24/Mar/2013 17:...	STP ES System	stp-esms1	Service ntpd died on stp-esms1
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms1	Service ntpd was restarted on stp-esms1
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms1	Service named was restarted on stp-esms1
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms2	Service named was restarted on stp-esms2
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms2	Check 'DeviceIsUp' is in state PASS on stp-esms2
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms1	Service ntpd was restarted on stp-esms1
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms1	Service named was restarted on stp-esms1
ⓘ	24/Mar/2013 17:...	STP ES System	stp-esms1	Service conman was started on stp-esms1

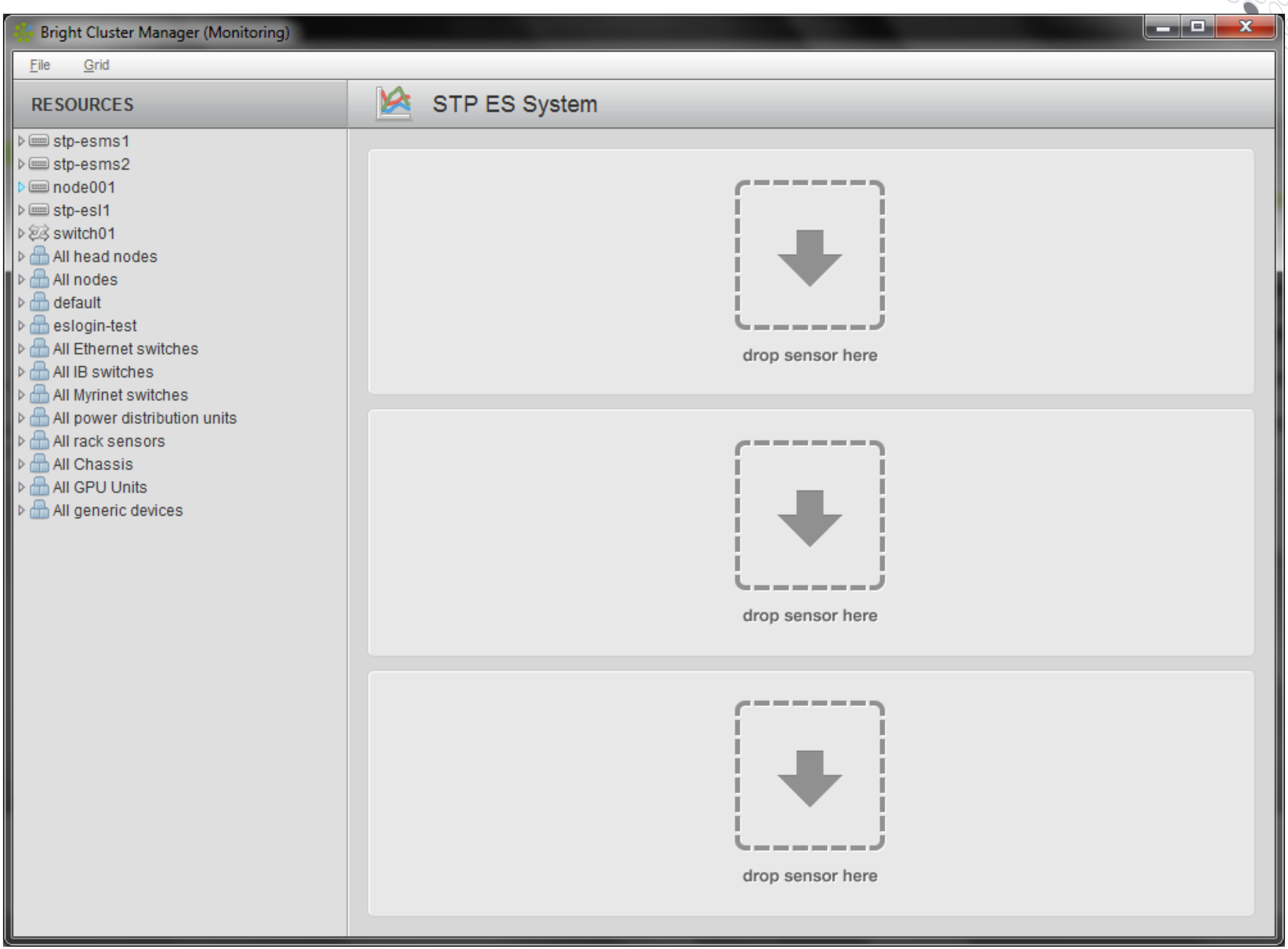
Health and Metric History - cmsh

- **By default, the last 3000 data values are stored**
- **Can be accessed by the “dumphealthdata” and “dumpmetricdata” commands in cmsh**
  - Example, get ‘ssh2node’ data for the past day on node004

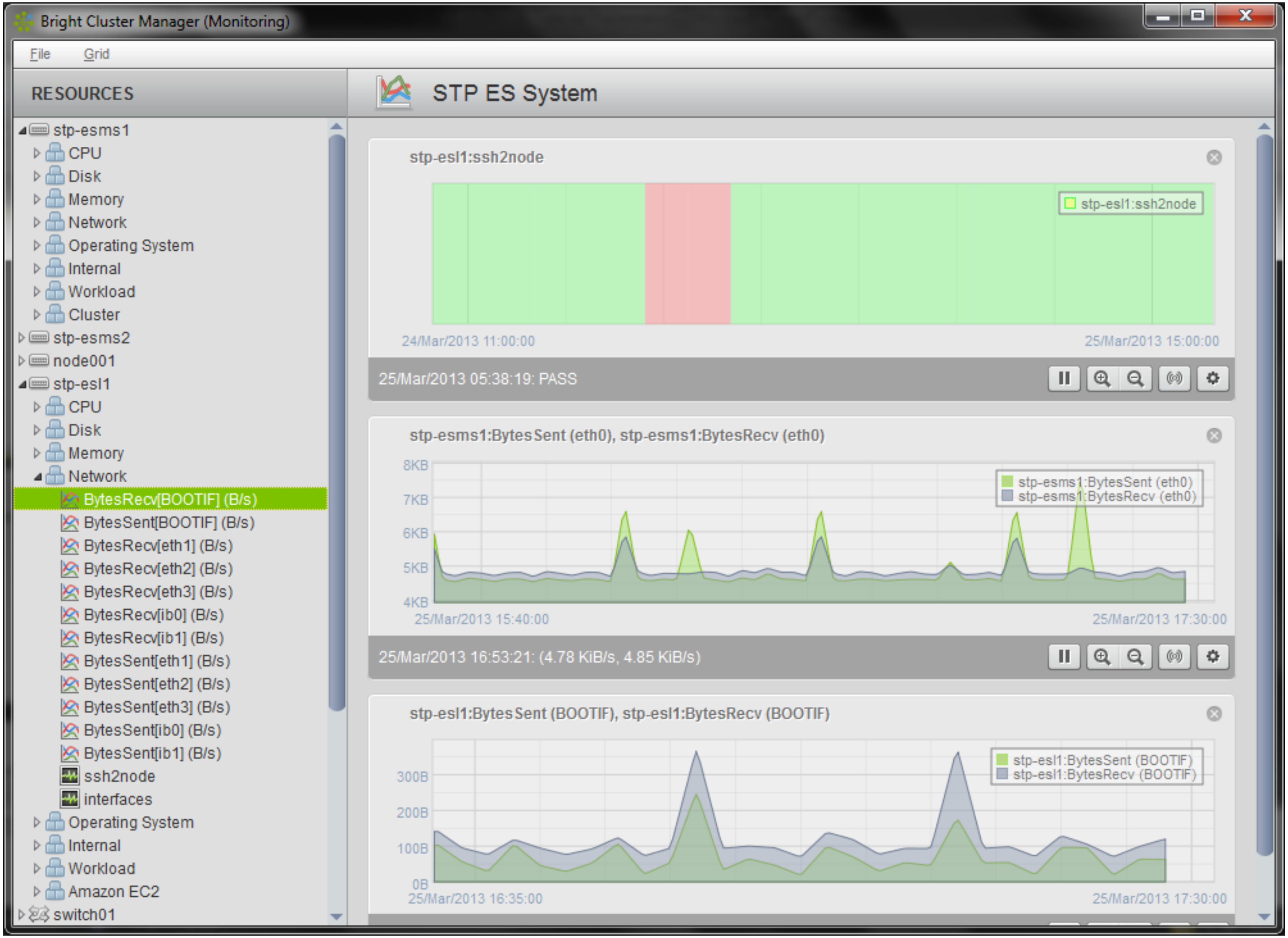
```
test-esms1:~ # cmsh
[test-esms1]% device use node004
[test-esms1->device[node004]]% dumphealthdata -1d now ssh2node
# From Mon Aug 20 16:34:52 2012 to Tue Aug 21 16:34:52 2012
Time                           Value             Info Message
-------------------------- ----------------
------------------------------------------
Mon Aug 20 16:34:52 2012   PASS              Not UP according to
CMDaemon
Tue Aug 21 10:00:03 2012   PASS
Tue Aug 21 10:49:48 2012   PASS              Not UP according to
CMDaemon
Tue Aug 21 11:00:20 2012   PASS
Tue Aug 21 11:30:00 2012   PASS
Tue Aug 21 16:30:00 2012   PASS
[test-esms1->device[node004]]%
```

- **Can be plotted in the cmgui monitoring window**

Health and Metric History - cmgui



Health and Metric History - cmgui



Bright Cluster Manager (Monitoring)

File Grid

RESOURCES

- stp-esms1
  - CPU
  - Disk
  - Memory
  - Network
  - Operating System
  - Internal
  - Workload
  - Cluster
- stp-esms2
- node001
- stp-esl1
  - CPU
  - Disk
  - Memory
  - Network
    - BytesRecv[BOOTIF] (B/s)
    - BytesSent[BOOTIF] (B/s)
    - BytesRecv[eth1] (B/s)
    - BytesRecv[eth2] (B/s)
    - BytesRecv[eth3] (B/s)
    - BytesRecv[ib0] (B/s)
    - BytesRecv[ib1] (B/s)
    - BytesSent[eth1] (B/s)
    - BytesSent[eth2] (B/s)
    - BytesSent[eth3] (B/s)
    - BytesSent[ib0] (B/s)
    - BytesSent[ib1] (B/s)
    - ssh2node
    - interfaces
  - Operating System
  - Internal
  - Workload
  - Amazon EC2
- switch01

STP ES System

stp-esl1:ssh2node

25/Mar/2013 05:38:19: PASS

stp-esms1:BytesSent (eth0), stp-esms1:BytesRecv (eth0)

25/Mar/2013 16:53:21: (4.78 KiB/s, 4.85 KiB/s)

stp-esl1:BytesSent (BOOTIF), stp-esl1:BytesRecv (BOOTIF)

Troubleshooting

- Depending on the problem, check the most likely logs
  - **Booting issues**
    - node-installer log - /var/log/node-installer
  - **Management issues**
    - cmdaemon log - /var/log/cmdaemon
    - syslog - /var/log/messages
    - event log
      - ‘events’ command in cmsh
      - ‘event viewer’ in cmgui
    - Check relevant monitor histories
      - ‘dumphealthdata’ for health checks
      - ‘dumpmetricdata’ for metrics
  - **Operational issues**
    - syslog - /var/log/messages
    - event log
      - ‘events’ command in cmsh
      - ‘event viewer’ in cmgui
    - Check relevant monitor histories
      - ‘dumphealthdata’ for health checks
      - ‘dumpmetricdata’ for metrics

Troubleshooting – Common Issues

- **Service fails to start after a grabimage and reboot**
  - Check that no *.pid files were grabbed to the image on the esMS
    - If any were grabbed, remove them and add the files or path to the excludelistgrab and excludelistgrabnew lists for the related categories

- **Unexpected interfaces attempt to start or fail to start**
  - Check that no /etc/sysconfig/network/ifcfg-* files exist in the image other than ifcfg-lo

- **Nodes don’t appear to respond correctly**
  - Check that the IP addresses are not duplicated with another node

Troubleshooting – Recovery Preparations

- **Make backups of the following for recovery**
  - buildconfig.xml used to initially install the esMS
  - Software Images
  - Management Database
  - Site customizations
    - Configuration files
    - Added software

- **Backing up the Management Database**

```
test-esms1:~ # /etc/init.d/cmd stop
test-esms1:~ # cmd -x mybackupbuildconfig.xml
test-esms1:~ # /etc/init.d/cmd start
```

Troubleshooting – Recovery Process

- **Install esMS per esMS Installation Guide**
  - Use buildconfig.xml that was used to initially install the esMS
  - Install software images that were archived
  - Restore Management Database
  - Restore site customizations
    - Configuration files
    - Added software

- **Restoring the Management Database**

```
test-esms1:~ # /etc/init.d/cmd stop
test-esms1:~ # cmd -i mybackupbuildconfig.xml
test-esms1:~ # /etc/init.d/cmd start
```

What we will cover today

- What are Cray External Services systems? ✓
- esMS, esLogin and esFS Overview ✓
- Configuration and Administration
  - Operational Overview ✓
  - *Break* ✓
  - Device Configuration and Management ✓
  - Node Provisioning ✓
  - Image Management ✓
  - *Break* ✓
  - System Monitoring ✓
  - esMS Failover ✓
  - Automated Lustre Failover and lustre_control ✓
  - Troubleshooting ✓

Thank you for your time!

Any questions?

Jeff Keopp
Harold Longley