Bull Lustre Management Tools for Large Cluster

Bull Lustre Team (Grenoble, France)
Johann.Lombardi@bull.net

Lustre User Group Spring 2006, Hilton Head Island

- Introduction
- Administration Tools Overview
- Key Issues
- Conclusion

Motivation

- Need to administrate several filesystems from a single point of management
- Scalability: to manage Lustre installation such as Tera10 (1PB, 54 OSSs with at least 864 OSTs, ...)
- Command line interface for filesystem management
- Efficient monitoring

CFS's LMT

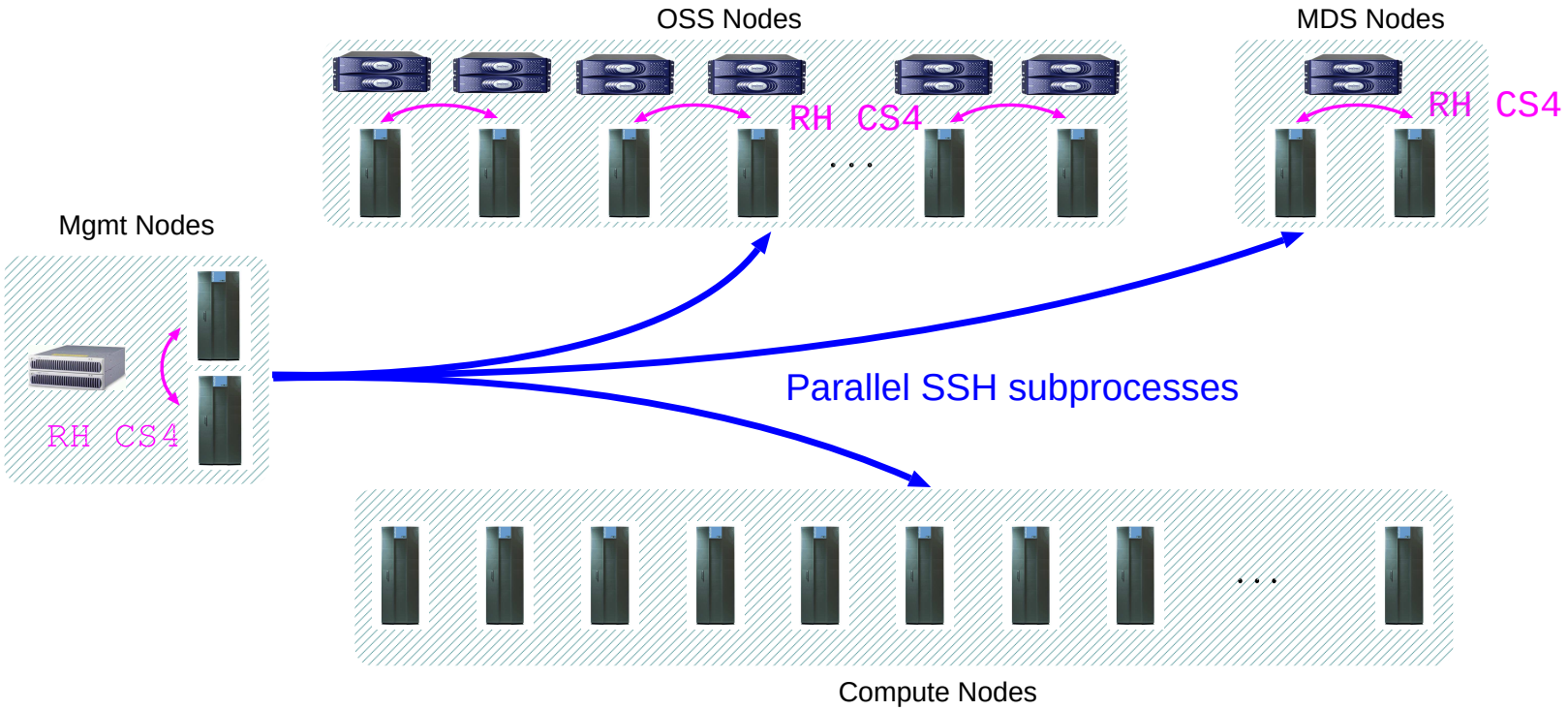
At that time (∼2004), we gave a shot to CFS's LMT:

- GUI, no command line interface
- Not suitable for large clusters
- Did not support HA

- Introduction
- Administration Tools Overview
- Key Issues
- Conclusion

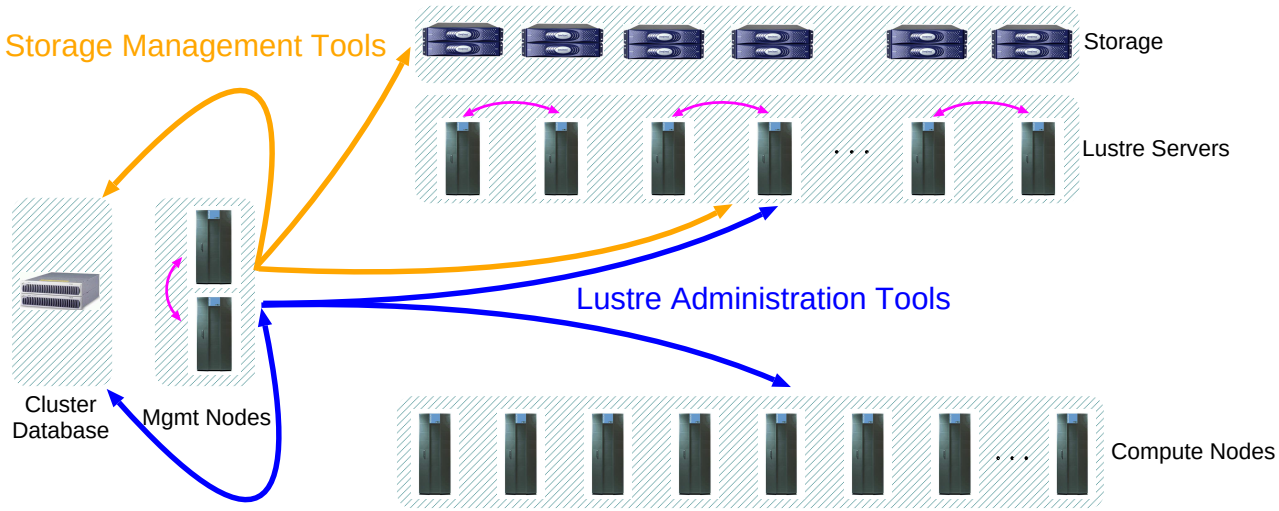
Implementation

- Leverage existing tools (RH CS4, Nagios, Ganglia, . . .)
- Everything on the management node
  - Management node failover
  - Commands are launched in parallel using ssh subprocesses



Framework

- For TERA10, tight coupling with Bull's cluster mgmt solution
  - Cluster database (Postgres)
  - Dedicated storage mgmt tools for:
    - Storage system low level formatting
    - Grant persistent links to block devices in /dev
    - Populate the cluster database
- Work in progress to allow standalone usage

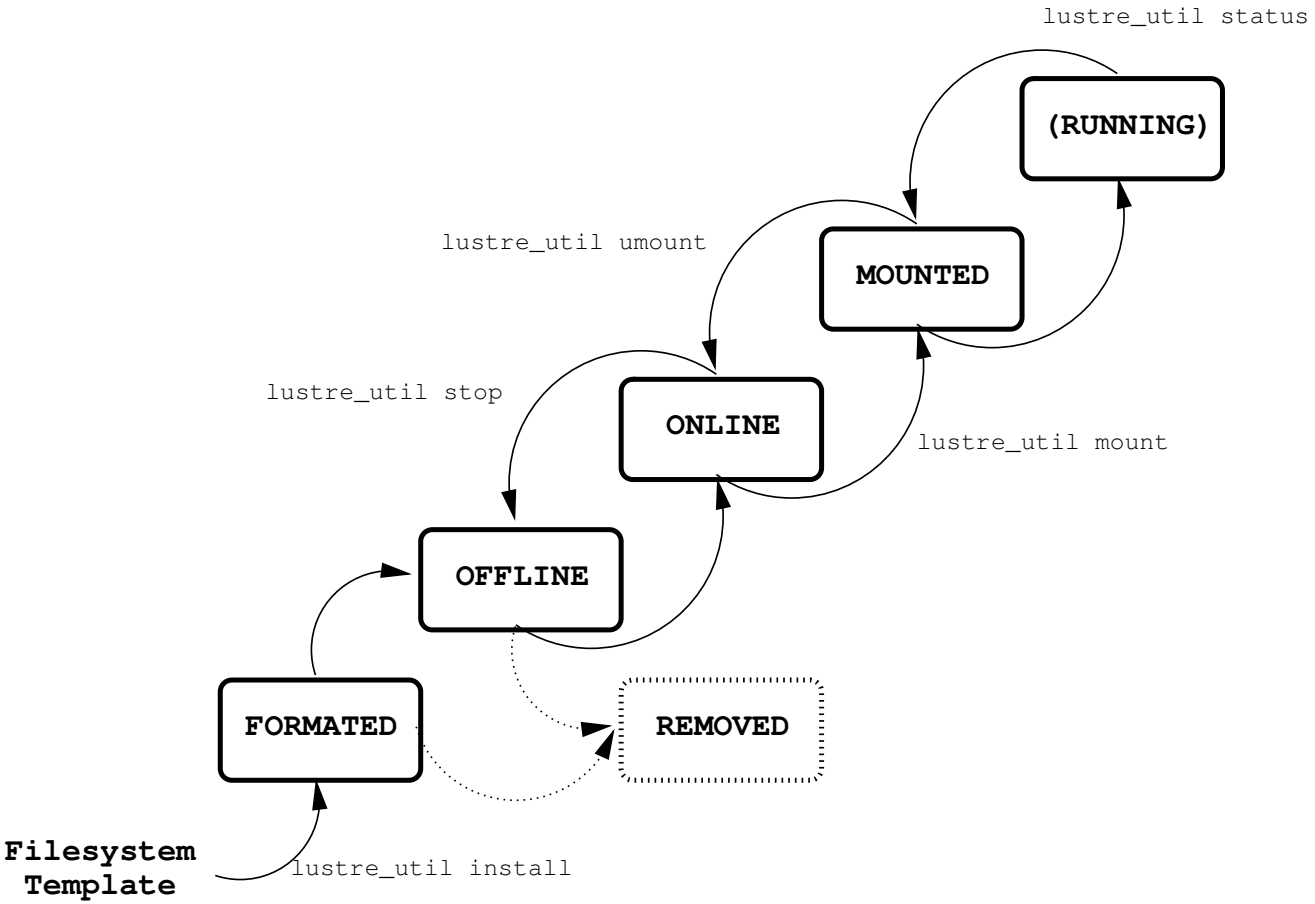


Features

- HA configuration[1](RH CS4)
- Offline filesystem extension (adding OSTs)
- SNMP setup
- Health monitoring via Nagios services (status, alarms, ...)
- Performance monitoring through Ganglia metrics
- Fast formatting
- Specific tuning for each filesystem
- But also external journal support, I/O scheduler setting, ...

[1]Cf. "Lustre management for 1PB of disks" Ramangalahy SGPFS'06

A Lustre Filesystem Lifecycle



Filesystem Template Motivations

- descriptive rather than exhaustive (do not name 800+ OSTs explicitely)
- reasonable defaults to minimize editing
- all options are kept (easy to add some)
- largely commented, can be prepared for customer before deployment

```
# Our filesystem will be called fs1, so the XML will be fs1.xml
# and put in the directory defined by LUSTRE_CONFIG_DIR in lustre.cfg
fs_name: fs1
    ################################################################
    # mount_path:
    mount_path: /mnt/lustre
    ################################################################
    # ost: [name=<RegExp>] [node_name=<RegExp>]
    # [dev=<RegExp>] [size=<RegExp>]
    # [jdev=<RegExp>] [jsize=<RegExp>]
    # [cfg_status=available|formated]
    # Specify OSTs to use with this filesystem, using regular
    # expressions matching their name, node_name,  device, size,
    # journal  device, journal size. At least one field must be
    # specified. If several fields are specified, only OSTs matching
    # every fields of the lines will be choosen. You can use as many
    # ost lines as  you  need.At least one ost line must be defined
    # for each filesystem.
    # We want to use all available osts

    ost: cfg_status=available
    ################################################################
    # mdt: [name=<RegExp>] [node_name=<RegExp>] [dev=<RegExp>]
    # [size=<RegExp>] [ jdev=<RegExp>] [jsize=<RegExp>]
    # [cfg_status=available|formated]

    mdt: cfg_status=available
```

Filesystem Declarative Template

- Simple readable text file describing the attribute of a filesystem such as:
  - mount options (`extents, mballoc, EA, ACL, quota options`)
  - filesystem description
  - ...(everything relevant to filesystem configuration)
- One file per filesystem, or one global file for all filesystems of a cluster.
- Expanded into .xml for use in standard Lustre workflow (xml, lconf, lmc...).
  - But, our framework is not tight to xml configuration file
- Handle potential conflicts between filesystems for backing storage

Command Line Interface for Filesystem Management

Swiss army knife to manipulate Lustre filesystems

```
lustre_util <install|update> -f <template>
                             [--kfeof] [--lconf <option>]
lustre_util <start|stop>     -f <fs_name>
                             [--lconf <option>]
lustre_util <mount|umount>   -f <fs_name>
                             -n <nodes> | -p <partition>
                             --mount <[+]opt1,opt2,...>
lustre_util status           [-f <fs_name>]
                             [-n <nodes> | -p <partition>]
lustre_util info             -f <template|fs_name>
lustre_util <fsck|rescue>    -f <fs_name>
lustre_util lfsck            -f <fs_name> -n <node>
...
```

Example: Status

```
# lustre_util status
Loading fs1 information from db...
Getting mount information from nova[12-66]
Getting devices information from nova[1-11]
---
FILESYSTEMS STATUS
+-----------+----------+--------+--------+----------------+----------+
| filesystem|  config  |running | number |   migration    |Available |
|           |  status  | status |of clts |                |  space   |
+-----------+----------+--------+--------+----------------+----------+
|fs1        |installed |online  |55      |0 OSTs migrated |142.6 TB  |
+-----------+----------+--------+--------+----------------+----------+
---
CLIENTS STATUS
+-----------+-----------+-----------+
| filesystem| correctly | correctly |
|           |  mounted  | unmounted |
+-----------+-----------+-----------+
|fs1        |nova[12-66]|nova[1-11] |
+-----------+-----------+-----------+
```

Keep an Eye on Underlaying Tools

Use `-V` option (verbose)

This displays the tools that are used (lconf, lustre mkfs, I/O scheduler positionning,...) with their options.

```
+Work on ns11:  +Formatting /dev/ldn.ddn18.15:
 + lustre_util makedev -c /dev/ldn.ddn18.14:1024000
+Work on ns11:  +Formatting /dev/ldn.ddn5.25:
 + /usr/lib/lustre/tune2fs -O dir_index /dev/ldn.ddn5.25
+Work on ns11:  +Formatting /dev/ldn.ddn18.13:
 + /usr/lib/lustre/mke2fs -q -F -O journal_dev /dev/ldn.ddn18.12
+Initialising MDS on ns11:
 + lconf --write_conf /etc/lustre/conf/fs_test_ddn22_23.xml
+Initialising MDS on ns11:  unloading module: lquota
```

- Introduction
- Administration Tools Overview
- Key Issues
- Conclusion

Formatting

- 'lconf –reformat' sequentially formats underlaying ext3 filesystems
- Does not matter when you have 1 or 2 small OSTs per OSS
- For Tera10, we have at least 16 2TB OSTs per OSS.

Formatting: Our Approach

- That's why our administration tools don't rely on lconf for formatting
- ext3 filesystems are formatted in parallel
- As a comparison, it takes 15 mins to format a 15TB filesystem (with one single OSS) against 3 hours with lconf

Per Filesystem Tuning

- Need to set specific parameters available through procfs for
  - Tuning, e.g. "This filesystem is expected to be used interactively, we would like to increase the DLM LRU size to improve responsiveness" (example #1)
  - Debuging, e.g. "We need to grab `D_NET` logs when a timeout occurs on the OSTs of this filesystem." (example #2)
  - ...
- We must grant that this setup will always be applied, regardless of:
  - new client nodes mount the filesystem
  - the filesystem is stopped/restarted
  - OSTs fail over another node
  - ...

Filesystem Tuning: Our Approach

One file on the mgmt node containing tunable settings

- tunables set up when the filesystem is started/mounted
- rely on python's globbing and support aliases

```
####  ALIAS DECLARATION ###################################

alias panic_on_lbug=/proc/sys/lnet/panic_on_lbug
alias max_pages_per_rpc=/proc/fs/lustre/osc/*${ost}*/max_pages_per_rpc
alias lru_size=/proc/fs/lustre/ldlm/namespaces/*{${ost},${mdt}}*/lru_size
alias dump_on_timeout=/proc/sys/lustre/dump_on_timeout
alias debug=/proc/sys/lnet/debug

#### TUNING PARAMETER ###################################

"1"       panic_on_lbug                                    CLT;OSS;MDS

"32"      max_pages_per_rpc                                CLT

# example #1
"41000"   lru_size                                         CLT            fs1

# example #2
"1"       dump_on_timeout                                  OSS            fs2
"512"     debug                                            OSS            fs2
```

A solution for Lustre management:

- **Simple**: but not simplistic
- **Powerful**: do not restrict/hide possibilities (tools are used for production as well as internal use)
- **Easily updated and extended**: example of recent work: "easy Lustre tuning" (dynamic variation of Lustre parameters)
- **Standard Cluster tools**: less code, faster to develop
- **Scales up, scales down**:
  - full featured (TERA10 oriented, with cluster DB)
  - partial set of features (standalone)
- Bull's **evaluating** to release it under an **Open Source** license

Architect of an Open World™

Thanks!!!

COPYRIGHT NOTICE