Lustre High Availability on a “Shine” cluster

olivier.hargoaa@bull.fr
BULL Lustre HPC team (Grenoble - France)

Lustre User Group, spring 2009, Sausalito California
Content

Introduction
HA tool design
HA tool user guide
HA tool software architecture
Content

Introduction

HA tool design
HA tool user guide
HA tool software architecture
Introduction: Hardware at BULL

- Two failover pairs architecture:
  - 2005 = Tera 10 CEA + 50 OSSs & 50 DDN 9550
  - 2008 = Cardiff + 2 OSSs & 2 EMC Cx340F
  - 2009 = Genci CCRT + 20 OSSs & 10 DDN 9550
  - And others...
Introduction: Future hardware design (1)

- N node failover group
  - 4 IO nodes
  - 2 storage devices

Today demand from some partners
Introduction: Future hardware design (2)

- N node failover group
  - 4 IO nodes
  - 4 storage devices

- Minimize impact of IO node lose
Introduction: Software design

- Today
  - Collection of script
  - Based on “lustre_util” open source management tool
  - Requires specific software and distribution

- HA tool project
  - Support open hardware design
    • How many storage devices per IO node?
    • How many IO node per failover group?
    • Storage infrastructure: IB, FC
    • Storage design: point to point connections, switch?
  - Support open software design
    • Offer a unique interface for any base HA software (Heartbeat – Cluster Suite, other?)
    • Administrator will choose software depending on hardware configuration
Content

Introduction

HA tool design

HA tool user guide

HA tool software architecture
HA tool design: specifications

- Use of shine as interface with Lustre
  - Shine: open source Lustre management framework
  - Shine allow HA configuration at mkfs.lustre time.
  - End to end HA Lustre management is out of the scope of shine.

- Allow individual target management

- Allow “n” nodes failover groups (n>=2)

- Maintain coherency on cluster:
  - Keep location of started targets and place where they are allowed to run
    - Full cluster restart
    - Restart at same place after a file system stop
    - Allow administrator to deactivate an IO node
  - Keep good Lustre start order (MGS, OSS, MDS)
HA tool design: target management on failover group

- Old BULL design on failover pairs
- Node crash → targets restarted on the failover node:

Active - Active
- OSS-1
  - OST1-fs1
  - OST2-fs2
- OSS-2
  - OST1-fs2
  - OST2-fs1

Passive - Active
- OSS-1
- OSS-2
  - OST1-fs1
  - OST2-fs1
  - OST1-fs2
  - OST2-fs2

- One HA service per node
HA tool design: target management on failover group

- Old BULL design on 4 IO nodes group

- 50% slower
HA tool design: target management on failover group

- New approach:
  - 1 service <=> 1 target
  - Plus 1 service per node (generics Lustre tests: network, health check)

- No impact on a two nodes cluster.
- Generic tests done one time only.
HA tool design: target management on failover group

- 4 nodes cluster example + target management

- 25% slower
- Target per node = m*(n-1)
  - n = number of nodes in failover group; m unsigned integer
  - Optimal (load balanced) if n or n-1 nodes are up
HA tool design: target management on failover group

- for fun!

- 50% slower
HA tool design: “n” node failover group configuration

- Sample configuration with RHCS5

```xml
<rm>
  <failoverdomains>
    <failoverdomain name="domain1">
      <failoverdomainnode name="node1" priority="1"/>
      ...
    </failoverdomain>
    ...
  </failoverdomains>
  <service domain="domain1" ... />
</rm>
```

<table>
<thead>
<tr>
<th>Domain</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain 1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>Domain 2</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Domain 3</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Domain 4</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Domain 5</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Domain 6</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>
Content

- Introduction
- HA tool design
- HA tool user guide
- HA tool software architecture
- HA framework management
  - Usage

# generate HA framework configuration files
hatool framework [--framework rhcs5|heartbeatv2] [--failovergroup ...] install

# manage HA framework daemons
hatool framework [--failovergroup ...] [--node ...] start
hatool framework [--failovergroup ...] [--node ...] stop
hatool framework [--failovergroup ...] [--node ...] status

- Configuration

failovergroup:
  Node: <node list> eg. node[2-6]
  target: <label list>
  quorum : <label>
failovergroup:
  Node: <node list> eg. node[7-8]
  target: <label list>
fenceplugin : fence
frameworkplugin : rhcs5
hatool: wrapper for shine

- HA shine management
  - Usage

  # Use shine for installation
  hatool lustre install --fsname <file system>

  # use shine to obtain target list
  # use HA framework to manage it
  hatool lustre --fsname <fsname> start
  hatool lustre --fsname <fsname> stop
  hatool lustre --fsname <fsname> status

  - Configuration

  Configuration is only “Shine” configuration
hatool: wrapper for HA resources

- HA service management
  - Usage

  # allow direct management of Lustre and other targets
  hatool resource --name <service name> [--node <node>] start
  hatool resource --name <service name> stop
  hatool resource --name <service name> status

- HA node management
  - Usage

  # stop all Lustre targets on the given node and,
  # restart it on other nodes in the failover group
  hatool node --name <node name> export

  # start all Lustre targets on the given node
  # if their primary location is on it.
  # Stop targets on the other nodes before.
  hatool node --name <node name> relocate

  # deactivate a node for maintenance purpose
  hatool node --name <node name> activate
  hatool node --name <node name> deactivate
Hatool software architecture: global

Lustre HA and administration core components
Hatool software architecture: plugin

- Example for ClusterNetwork module
Hatool software architecture: plugin example

- Cluster network example

```
$ clusterNetwork show

<table>
<thead>
<tr>
<th>ascii</th>
<th>local</th>
<th>remote</th>
<th>module</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes-network</td>
<td>YesLocal</td>
<td>YesRemote</td>
<td>YesModule</td>
</tr>
<tr>
<td>tcp</td>
<td>LocalDefaultTCP</td>
<td>RemoteDefaultTCP</td>
<td>ModuleDefaultTCP</td>
</tr>
</tbody>
</table>

$ clusterNetwork local -t yes-network-error --nid node@ib0
No plugin found for network yes-network-error in local test

$ echo $? 220

$ clusterNetwork retcode 220
PLUGIN_NOT_FOUND

$ clusterNetwork local -t yes-network --nid node@ib0
True

$ echo $? 0

$ clusterNetwork retcode 0
SUCCESS
```
Hatool software architecture: Open source

- Bull is evaluating to release it under an Open Source license
  - After a first stable release!
  - Allow community to add their own plugins
  - Make use of Shine under high availability user friendly
  - Get benefits of ideas of the community
Thanks!!