Exploiting SCI in the MultiOS management system

Ronan Gunniffe
Brian Coghlan
The Problem

Those OS researchers crashed our cluster!
The Desire

Crashes may be illuminating!

OS research environment:
• may not be stable
• may be missing features

Separate cluster per project:
• is inefficient
• is expensive

Mmmm ....

As clusters become more common, problem gets more acute.
Existing solution

Partition the local disk

i.e. Dual/Multiple Boot

- all candidate environments must be there
- no easy way to add another environment
- assumes environments will not over-write others
- only the current image is accessible
MultiOS solution - import & export every time

Import the environment each time (from remote disk)

- can support any number of environments
- no assumptions about stability or good behaviour
- environments are accessible when off-line
- places great demands on network and remote disk

Time

OS #1 out  OS #2 in
MultiOS

How is it implemented?

• standard mechanism for diskless workstations
  BOOTP: Query remote server for file to download
  TFTP: Download indicated file
  Pass control to downloaded file
• alternate between two types of session

Management sessions:

• node runs management software obtained over network
• this loads user image to local disk

User sessions:

• node runs whatever environment is on the local disk
• MultiOS considers this a black box
Use a full OS as management environment

Management software is Linux

• can run from a RAM disk
• can use network-mounted filesystems

Advantages of a full OS

• SCI drivers
• raw disk I/O support
• standard UNIX tools: `dd`, `diff`, `gzip`, `rcp`, etc.
• new tools can be written as necessary
Overall MultiOS architecture

Architecture:
- standard servers: BOOTP, TFTP, HTTP
- isolated web console
Disk images are big - hard disks are slow

Our cluster: 16 nodes x 2GB disks

2GB raw over 100Mbps Ethernet:

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Network traffic</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>8GB@ 8MB/s</td>
<td>17.1 mins</td>
</tr>
<tr>
<td>8</td>
<td>16GB@ 8MB/s</td>
<td>34.1 mins</td>
</tr>
<tr>
<td>16</td>
<td>32GB@ 8MB/s</td>
<td>68.3 mins</td>
</tr>
<tr>
<td>32</td>
<td>64GB@ 8MB/s</td>
<td>136.5 mins</td>
</tr>
</tbody>
</table>

2GB raw over SCI:

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Network traffic</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>8GB@ 40MB/s</td>
<td>3.3 mins</td>
</tr>
<tr>
<td>8</td>
<td>16GB@ 50MB/s</td>
<td>5.5 mins</td>
</tr>
<tr>
<td>16</td>
<td>32GB@ 50MB/s</td>
<td>10.7 mins</td>
</tr>
<tr>
<td>32</td>
<td>64GB@ 50MB/s</td>
<td>21.3 mins</td>
</tr>
</tbody>
</table>
Making the numbers smaller

3 ways to do it:
- Compression
- Loading less than a disk
- Multicast reference image + patching

For 100MB differential information for each compute node:

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Network traffic</th>
<th>Storage</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2GB@10MB/s + 0.4GB@40MB/s</td>
<td>2.4GB</td>
<td>3.5 mins</td>
</tr>
<tr>
<td>8</td>
<td>2GB@10MB/s + 0.8GB@50MB/s</td>
<td>2.8GB</td>
<td>3.6 mins</td>
</tr>
<tr>
<td>16</td>
<td>2GB@10MB/s + 1.6GB@50MB/s</td>
<td>3.6GB</td>
<td>4.0 mins</td>
</tr>
<tr>
<td>32</td>
<td>2GB@10MB/s + 3.2GB@50MB/s</td>
<td>5.2GB</td>
<td>4.4 mins</td>
</tr>
</tbody>
</table>
If disks were faster

Assuming 50MB/s disks:

For 100MB differential information for each compute node:

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Network traffic</th>
<th>Storage</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2GB@50MB/s + 0.4GB@50MB/s</td>
<td>2.4GB</td>
<td>48 secs</td>
</tr>
<tr>
<td>8</td>
<td>2GB@50MB/s + 0.8GB@50MB/s</td>
<td>2.8GB</td>
<td>56 secs</td>
</tr>
<tr>
<td>16</td>
<td>2GB@50MB/s + 1.6GB@50MB/s</td>
<td>3.6GB</td>
<td>72 secs</td>
</tr>
<tr>
<td>32</td>
<td>2GB@50MB/s + 3.2GB@50MB/s</td>
<td>5.2GB</td>
<td>104 secs</td>
</tr>
</tbody>
</table>
SCI Multicast vs. Ethernet Multicast

IP Multicast
- multicast = broadcast
- image traffic disturbs everyone

SCI Multicast-by-propagation
- end-to-end latency small compared to total time
- little extraneous traffic

Difference becomes important in partitioned clusters
Partitionable clusters

- MultiOS traffic can easily saturate the image server
- fewer points of entry than partitions
- transport scripts can provide traffic shaping
Summary

MultiOS allows a cluster to be shared
• for any number of different environments
• for any type of research

MultiOS via SCI exploits:
• high bandwidth ceiling
• low protocol overheads
Conclusion

Our vision of the future

NB: OS research is an equal opportunity employer!

Normal user  OS researcher

http://www.cs.tcd.ie/multios/