

Last Updated 2010-02-13

ARSC Comparison of NFS vs. LNET for esLogin on Cray XT5

Objective

Compare IO performance of Lustre filesystem on Cray XT system with esLogin node using LNET and NFS clients. Cray deployed the esLogin nodes to ARSC as NFS. ARSC professional instincts believed LNET would be a more appropriate solution. The esLogin nodes at ARSC will be used both as primary login nodes for users and users will be doing significant pre- and post-processing of data.

Configuration

Main system (pingo)	TDS system (ognip)
<ul style="list-style-type: none"> • CLE 2.2 with 432 compute nodes (5 cabinets) • /lustre/large stripe=4 46*OST (workdir) • /lustre/small stripe=1 1*OST (home, /usr/local) • native login 2 core 8gb 2.6GHz • esLogin 16 core (4x4) 128gb 2.7GHz • 10GigE LNET and NFS (single node) • /usr/local 120k files 7.5gb 	<ul style="list-style-type: none"> • CLE 2.2 with 20 compute nodes • /lustre/large stripe=1 3*OST (workdir) • /lustre/small stripe=1 1*OST (home, /usr/local) • native login 2 core 8gb 2.6GHz • esLogin 16 core (4x4) 128gb 2.7GHz • GigE LNET and NFS • /usr/local 160k files 13.6gb

Tests/Methodology

Four different tests were run:

1. Write (using iotest binary) to workdir.
Typically used 1mb block size, some runs at 2m, 4m, and 16m.
Used file size of 128gb to exceed available memory.
2. Read (using iotest binary) from workdir.
Typically used 1mb block size, some runs at 2m, 4m, and 16m.
Used file size of 128gb to exceed available memory.
3. Copy (using cp binary) 128gb file from and to workdir.
4. Stat and sum files in /usr/local using uals binary so single process for all files.

The pacct files (uakpacct binary) was used for elapsed and CPU time checks. For all tests, sync and drop_caches were run before and after each test. At least one of each test was run on a quiesced system and some tests were repeated on an active system. The tests were repeated enough to have confidence in observations. Additional repeats would raise confidence comparing LNET 1.6.7 and 1.8.1, but time (especially quiesced system time) was a significant constraint. No tuning was attempted, systems were as delivered by Cray.

Used Lustre on native login node as baseline and tested:

- Cray XT Lustre 1.6.5
- LNET 1.6.7 client (pingo only)
- LNET 1.8.1 client
- NFS

Cray supports both the 1.8 and 1.6 clients for esLogin. ARSC used the "patchless Lustre clients".

Significant Problems

- **LNET:**
If the LNET router or Cray XT system "disappears" abruptly, the esLogin nodes may hang and require reboot. With the user home filesystem coming from the Cray XT this is a non-issue for ARSC, but this may be an issue for a site wishing to continue user operation on esLogin while XT is down.
- **NFS:**
During 128gb NFS write operation, stat() operations (ls -l) in the directory hung for the duration of the write, one was measured as 14 minutes. This was not seen for native login Lustre or LNET. Delays like this will be intolerable for users on login nodes. There may be tuning opportunities to address this. The iotest binary issues fsync() only prior to fclose() not after each write operation For NFS the stat() operations may be on the tail of a long IO queue.

Results

Where Type	iotest read	iotest write	copy (cp)	stat/sum (uals)
pingo LNET 1.6.7	+6%	+28%	+19%	-15%
pingo LNET 1.8.1	+3%	+30%	+26%	-8%
pingo NFS	-71%	-39%	-61%	-16%
ognip LNET 1.8.1	-57%	-22%	-5%	-12%
ognip NFS	-73%	-44%	-52%	-15%

*Positive percentage is faster than native Lustre.
Negative is slower.*

Observations:

- For stat and sum, NFS is slightly worse than LNET which is worse than native.
The 1.8.1 client may be better.
- For large file IO (read, write, copy) NFS is significantly worse than LNET, for 10GigE:
 - LNET is 2.1 times faster than NFS for write
 - LNET is 3.5 times faster than NFS for read
 - LNET is 3.2 times faster than NFS for copy
- For native login nodes, read/write/copy effectively ran at CPU speed (CPU == elapsed).
- For esLogin nodes copy ran at CPU speed but read used ~60% CPU speed and write used ~75%.
- With 10GigE LNET on esLogin is faster than native login Lustre, especially for write. This is unexpected (drop_caches performed) and not yet understood.
- The drop_caches typically took 2 seconds for native login (8gb), 20+ seconds for NFS (128gb), and 50+ seconds for LNET (128gb).
- The large IO typically used all available memory. For HPC systems, this may not be optimal as IO buffers are not generally reusable for caching. There may be tuning opportunities to enhance IO throughput and memory utilization on a busy system by reducing IO caching.

Additional Information:

- See <http://www.arsc.edu/~kcarlson/Work/ARSCesLoginCharts.pdf> for additional charts of the results.
- See <http://www.arsc.edu/~kcarlson/Work/arscioesl.xls> for spreadsheet of results described above.
- See <http://www.arsc.edu/~kcarlson/software/Software.html> for [iotest](#), and [uals](#), and [uakpacct](#).

Summary

For ARSC, LNET clients on esLogin is an overwhelmingly obvious choice over NFS for performance and providing lustre utilities directly on esLogin. ARSC chose 1.8.1 client as possibly better performing (need more tests), likely more bug fixes, and compatibility with future external systems.

