End-to-End Framework for Fault Management for Open Source Clusters: Ranger

John L. Hammond

ICES/TACC

The University of Texas at Austin

jhammond@ices.utexas.edu
• **Co-authors**
  – Tommy Minyard, TACC
  – James C. Browne, U.T. C.S.

• **Collaborators**
  – Bill Barth, Karl Schulz, TACC
  – Edward Chuah, Howie Kuo, Gary Khoon, IHPC
Motivation

• Scale of systems, together with complexity of hardware/software stack means faults/failure are:
  – inevitable
  – hard to diagnose
  – hard to fix

• But no comprehensive fault management systems exist (for FOSS systems)
Ranger
High Performance Logging

• 3,936 compute nodes (63K cores), 72 I/O servers, 3 metadata servers
• 30K—2M msgs per day, (5K—200K Lustre)
• Only 5—10% have severity >= error.
• > 500 message types, and growing.

Traditional approaches to system logging are not up to the task.
Log analysis is for the birds.

-Kent B.

System logs are a kind of cognitive black hole. It's easy to pour vast amounts of time and energy in to staring at them, but then have nothing ever seem to come out.

-Anon
Non-Diagnostic Messages

- LustreError: 7424:0:
  (quota_master.c:514:mds_quota_adjust())
  mds adjust qunit failed! (opc:4 rc:-122)

- LustreError: 9247:0:
  (ldlmLib.c:1643:target_send_reply_msg())
  @@ processing error (-107)
  req@000001018ac30c00 x152525432/t0 o400-
  >&<?>@<?:0/0 lens 128/0 e 0 to 0 dl
  1267778275 ref 1 fl Interpret:H/0/0 rc
  -107/0
Obscure Messages

- Understanding Lustre (Linux Kernel, [insert software name],...) trace requires careful reading of source

Jan 19 16:58:45 i110-301 kernel: LustreError: 5647:0:(events.c:66:request_out_callback()) @@ type 4, status -5 req@ffff81060142b000 x70401815/t0 o8->work-OST003c_UUID@129.114.97.37@o2ib:28/4 lens 304/456 e 0 to 1 dl 1263941647 ref 1 fl Complete:EXN/0/0 rc -110/0
Example: Lustre OSS hangs

- Diagnosing causes of hangs requires trace not usually recorded (when generated).
- So we open the fire hydrant:

```plaintext
LustreError: 0:0:(ldlm_lockd.c:305:waiting_locks_callback()) ###
lock callback timer expired after 51s: evicting client at 129.114.113.8@o2ib
ns: filter-scratch-OST005f_UUID lock:
000001025a566b80/0xb9819aee4c67ae lrc: 3/0,0 mode: PW/PW res: 39069491/0 rrc: 54 type: EXT [71843840->105398271](req 71843840->72138751) flags: 0x20 remote: 0xf4b0b6ab42890c11 expref: 8 pid: 9671 timeout 4363742923
...
Call Trace:<ffffffffa0050cf6>{:raid5:get_active_stripe+1277}
<ffffffffa0052b75>{:raid5:make_request+1340}
...
[3900 messages omitted]
```
Log difficult (impossible) to parse

```c
int main ( ... )
{
    char *s1, *s2;
    /* ... */

    printf("%s%s", s1, s2);
}
```

Assume the output is "1234", then find s1 and s2.
Real examples

Jan 19 16:58:45 i110-301 kernel: [3038184.371463] LustreError: 5647:0:
(events.c:66:request_out_callback()) @@ type 4,
status -5 req@ffff81060142b000 x70401815/t0 o8-
>work-OST003c_UUID@129.114.97.37@o2ib:28/4 lens
304/456 e 0 to 1 dl 1263941647 ref 1 fl
Complete:EXN/0/0 rc -110/0

Jan 26 21:26:12 mds3 kernel: LustreError: 0:0:
(ldlm_lockd.c:305:waiting_locks_callback()) ### lock
callback timer expired after 50s: evicting client at
129.114.105.107@o2ib ns: mds-scratch-MDT0000_UUID
lock: 000001016ace8940/0xf9157c8032284b99 lrc: 3/0,0
mode: CR/CR res: 269333475/1604567023 bits 0x3 rrc: 2
type: IBT flags: 0x4000020 remote: 0x2312e9da3f6a73c3
expref: 55 pid: 13656 timeout: 4479804771
System Log Rationalization

• Make logs more amenable to automated analysis, data mining
  – Leverage existing logging interfaces
  – Encode in readily parsable format
  – Parallel stream of messages

• Improve human readability, rationalize message priorities, delete redundant messages
Modify printk() by inserting call to a new function, rat_printk().

Every call to printk(),

```
printk(KERN_ERR "tx_abrt_resched failed: rc %d: %s\n", rc, errmsg);
```

would produce the normal message:

```
Mar 10 15:09:30 dbs7 kernel: \
tx_abrt_resched failed: rc -5: Input/output error
```

along with a rationalized version of the same:

```
time:1268255370
host:dbs7
prog:kernel
0:tx_abrt_resched failed: rc %d: %s
 1:-5
 2:Input/output error
```
B&A: Segfault Message

Jul 8 12:27:52 i115-312 kernel: test_sine.x[27673]: segfault \
at 00002aaa9e094c88 rip 00002b92854db19e rsp 00007fff25bc0b40 \\
error 4

time:1278610053
host:i115-312
jobid:1461336
prog:kernel
0:%s%s[%d]: segfault at %016lx rip %016lx rsp %016lx error %lx
1:<6>
2:test_sine.x
3:27673
4:46912284216456
5:47908301681054
6:140733826468672
7:4
LustreError: 11-0: an error occurred while communicating with \ 
129.114.97.41@o2ib. The ost_write operation failed with -107

0:%sLustre: %x-%x: an error occurred while communicating with \ 
%s. The %s operation failed with %d
1:<3>
2:17
3:0
4:129.114.97.41@o2ib
5:ost_write
6:-107
Rationalized encoding stack
Part 2: Automated Diagnosis and Recovery

• Rule based approach to identify faults
• Make use of rationalized message stream
• Identify culpable nodes or servers
• Take corrective action, or refer to administrator

Work in progress
Methodology: FDiag

Edward Chuah (IHPC) et al.
Diagnosing the Root-Causes of Failures from Cluster Log Files
Uses "statistical correlation analysis to establish probable cause and effect relationships for the fault being analyzed."
Applied to compute node soft lockups.
To appear in HiPC 2010.
Part 3: Monitoring TACC Outage

- Compute node monitor, 2\textsuperscript{nd} generation
- Prevent jobs on bad node (SGE)
- Identify bad nodes to admin

- Lightweight, low jitter, sleeps during jobs
- Short UDP update messages, no XML
- Process all nodes in < 1 second.
Outage Monitoring

- InfiniBand hardware check
- Lustre connectivity
- /tmp and /dev/shm usage
- Out-of-memory
- SSD I/O errors
- Specific kernel messages
- More as needed
Outage Design

**login1**

- **outage-master (process)**
  - port 9192 (UDP)

**/dev/shm/outage-table** (shared binary table)

- **outage-sched (process)**
- **outage-report (process)**

- `msg` to `ack`
- `qconf -Mhgrp ...`
- `@outage2`
- `/var/outage-report` (ascii table)

outage2 i101-101

SGE hostgroup file