ECE 6930-004
HPC Fault Tolerance

RESILIENCE AT PETASCALE AND BEYOND

DR. JON C. CALHOUN

Schadenfreude! 20 July 2008

8:40 am PDT: error rates in all Amazon S3 datacenters begin to climb

8:50 am PDT: error rates so high very few requests were completely successfully

9:41 am PDT: Amazon engineering determined S3 servers were having problems communicating with each other

10:32 am PDT: After exploring several options, it was determined that all communication between Amazon S3 servers must halt.

11:05 am PDT: All server-to-server traffic is stopped

2:20 pm PDT: internal communication restored

2:57 pm PDT: Amazon S3's EU location begins correctly servicing requests

Schadenfreude! 20 July 2008

3:10 pm PDT: request and error rates in EU have returned to normal

4:02 pm PDT: US locations began successfully completing customer requests

4:58 pm PDT: US locations operation has returned to normal

Schadenfreude! 20 July 2008

So what happened?

- Amazon determined that message corruption was the cause of the server-to-server communication problems
- Several messages had a single bit corrupted such that the message was still intelligible, but the system state information was incorrect.
- Did not have protection in place to detect whether this particular internal state information had been corrupted
- Corruption spread throughout the system causing other issues

Reminder

Date	Paper/Topic	Presenter
8/23	Introduction/Syllabus/What is HPC	Calhoun
8/28	Basic Fault Tolerance Concepts	Calhoun
8/30	Toward Exascale Resilience	Calhoun
9/4	Lessons Learned From the Analysis of System Failures at Petascale: The Case of Blue Waters	
9/6	Basics of Checkpoint-restart	Calhoun
9/11	Basics of Checkpoint-restart	Calhoun
9/13	Evaluation of Simple Causal Message Logging for Large-Scale Fault Tolerant HPC Systems	
9/28	Design, Modeling, and Evaluation of a Scalable Multi-level Checkpointing System	
9/20	MCRENGINE: A Scalable Checkpointing System Using Data-Aware Aggregation and Compression	
9/25	What is a soft error?	Calhoun

What is the problem?

Because of their massive scale and complexity, current HPC systems have frequent failures and run for only a few days before some part of the system requires rebooting

- Large HPC systems recommend checkpointing roughly every 4-6 hours

Current approaches for HPC resilience which relies on automatic or application level checkpoint-restart will not work because the time for checkpointing and restating will exceed the meant time to failure of a full system

Strawman

NCSA/UIUC Blue Waters:
- **Memory size:** 1.634 PB
- **Filesystem bandwidth:** 1 TB/s
- **Time to checkpoint:** 27 minutes

ORNL Summit:
- **Memory size:** 10+ PB
- **Filesystem bandwidth:** 2.5 TB/s
- **Time to checkpoint:** 1.11 hours

System level checkpointing at exascale combined with an increase in the rate of faults results in all the time spent checkpointing or restarting!!!!

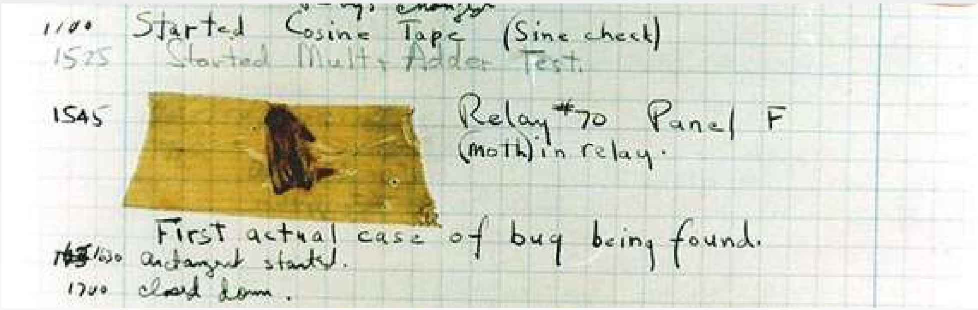
Hello failure, my old friend

Reliability in computing is not a new concept

Failures were common on early computer systems

- John von Neumann talks about the problem of building a reliability machine from unreliably components in 1956

The reliability problem is/will always be with us!!!



Solutions

Do you prefer solution in software or hardware?

What are some solutions to solve computer reliability that you have heard of?

Hardware based solutions

Hardware solutions to reliability issues offer several key advantages

- Application agnostic
- Generally more efficient than software
  - Time and Power

Techniques:

- Coding
  - lecture later on about this
- M of N systems
- Residue arithmetic